Linear Algebra and its Applications 473 (2015) 316–338

Contents lists available at ScienceDirect

Linear Algebra and its Applications

www.elsevier.com/locate/laa

Computing Frechet derivatives in partial least squares regression

Lars Eldén [1]

Department of Mathematics, Linköping University, SE-58183 Linköping, Sweden

ARTICLE INFO

Article history:
Received 30 September 2013
Accepted 8 September 2014
Available online 17 September 2014
Submitted by Lek-Heng Lim

MSC:
62J05
65F10

Keywords:
Partial least squares
PLS
Regression
Least squares
Prediction
Golub–Kahan bidiagonalization
Krylov method
Frechet derivative
Recursion
Perturbation theory
Degrees of freedom

ABSTRACT

Partial least squares is a common technique for multivariate regression. The procedure is recursive and in each step basis vectors are computed for the explaining variables and the solution vectors. A linear model is fitted by projection onto the span of the basis vectors. The procedure is mathematically equivalent to Golub–Kahan bidiagonalization, which is a Krylov method, and which is equivalent to a pair of matrix factorizations. The vectors of regression coefficients and prediction are non-linear functions of the right hand side. An algorithm for computing the Frechet derivatives of these functions is derived, based on perturbation theory for the matrix factorizations. From the Frechet derivative of the prediction vector one can compute the number of degrees of freedom, which can be used as a stopping criterion for the recursion. A few numerical examples are given.

E-mail address: lars.elden@liu.se.
[1] Tel.: +46 13 282183.

http://dx.doi.org/10.1016/j.laa.2014.09.017
0024-3795/

1. Introduction

Partial least squares regression (PLSR) [1,2] is a frequently applied technique for multivariate regression in the case when the explaining variables (predictor variables) are highly correlated. It iteratively constructs an orthonormal sequence of latent components (basis vectors) from the explaining variables, which have maximal covariance with the response variable. In each step of the procedure, the data and the solution vectors are projected onto subspaces of low dimension, where a linear model is fitted. PLSR can be used as an alternative to principal components regression (PCR), and often a good fit is obtained with a model of considerably smaller dimension than with PCR, see, e.g., [3].

The PLS procedure is mathematically equivalent to a Krylov method, Golub–Kahan bidiagonalization [4,5]. While the so-called NIPALS variant of PLS [5] constructs the basis vectors by successively deflating the data matrix (the predictor variables) and the right hand side (the response variable), the Krylov method generates them by a recursion without modifying the data matrix, see e.g. [3]. The Krylov recursion is equivalent to a pair of matrix factorizations.

A basic problem in PLSR is to determine the "optimal" number of components, i.e. to derive a stopping criterion for the recursion. There are two alternatives, essentially. The standard approach is to use cross validation. Alternatively, in [6] an information criterion is applied and the complexity of the fitted model is defined as the number of degrees of freedom (DOF).

Let $y \in \mathbb{R}^m$ be a vector of observations of the response variable, and $X \in \mathbb{R}^{m\times n}$ be a matrix, whose columns are the observations of the explaining variables. Consider the least squares problem

$$\min_{\beta} \|X\beta - y\|, \tag{1}$$

to which an approximate solution is computed by PLS. Denote the solution after k steps of PLS by β_k, and the prediction by $y_k = X\beta_k$. It turns out that y_k and β_k are non-linear functions of y; we write $y_k = F_k(y)$ and $\beta_k = H_k(y)$. The number of degrees of freedom of the model, D_k, is defined

$$D_k = 1 + \operatorname{tr}\left(\frac{\partial F_k}{\partial y}\right) = 1 + \operatorname{tr}\left(X\frac{\partial \beta_k}{\partial y}\right), \tag{2}$$

where $\partial F_k/\partial y$ is the Frechet derivative of the function. Note that, with $\bar{y} = y + \epsilon\delta y$ a perturbed data vector,

$$\|y_k - \bar{y}_k\| \le \epsilon \left\|\frac{\partial F}{\partial y}\right\| \|\delta y\| + O\big(\epsilon^2\big).$$

Thus the Frechet derivative defines a condition number of the function, which is a measure of the sensitivity to perturbations in the data.

The quantity D_k and the Frechet derivative of F_k can be computed by differentiation of the PLS recursion, which gives a recursion for partial derivatives [6]. For m large this is very costly in terms of computation, because it involves matrix multiplications of $m \times m$ matrices in every step of the recursion. In addition, storage of derivative matrices of dimension $m \times m$ can be prohibitive. Also, numerical experiments show that this method is unstable, especially for large problems.

The main result if this paper is the derivation of a computable expression for the Frechet derivative $\partial F_k/\partial y$, based on perturbation analysis of the matrix decompositions obtained from the Krylov recursion. For large problems this new method is much faster than that based on differentiation of the recursion, and our preliminary numerical tests indicate that it does not suffer from stability problems.

We will also consider the function $\beta_k = H_k(y)$, and derive its Frechet derivative.

The paper is organized as follows. We start by reviewing the Krylov formulation of PLS in Section 2. Then in Section 3 we give computable expressions for the Frechet derivatives of the prediction vector (Section 3.2) and the regression coefficients (Section 3.3). Those expressions are based on perturbation theory for the Krylov factorization, which we derive in Section 3.1. Numerical examples illustrating the derivatives are given in Section 4. In an appendix we first give the NIPALS version of PLS, and then briefly review the method for computing Frechet derivatives by differentiation of the Krylov recursion. Finally we give pseudo codes for the computation of certain quantities in the perturbation theory.

1.1. Notation

The Euclidean vector norm is denoted $\|y\| = (y^T y)^{1/2}$. The same notation is used for the spectral matrix norm $\|A\| = \sup_{\|x\|=1} \|Ax\|$. We use I_k to denote the $k \times k$ identity matrix. Standard unit vectors are denoted e_i, where all components are zero except the i'th, which is equal to 1.

2. Partial least squares regression

PLSR was originally formulated in terms of the NIPALS algorithm, which deflates the data matrix and the right hand side [5], see Appendix A. This method has good stability properties [7] and can easily be adapted for problems with missing data [2]. However, it does not display very well the structure of the algorithm, and therefore we find it unsuitable for our analysis of the procedure. Instead we will use the equivalent Golub–Kahan (GK) bidiagonalization [4], [8, Section 7.6], [3], which is related to the Lanczos tridiagonalization procedure.

2.1. PLS: GK bidiagonalization

The GK bidiagonalization algorithm was originally designed as a first step in the algorithm for computing the singular value decomposition of a matrix [4]. Later it has become a method of choice for solving large and sparse least squares problems. There is a variant, LSQR [9], that avoids storing all the basis vectors and computes recursively also the least squares solution.[2]

GK bidiagonalization

1. $v_1 = \frac{1}{\|X^T y\|} X^T y$; $\alpha_1 u_1 = X v_1$
2. **for** $i = 2 : k$
   (a) $\gamma_{i-1} v_i = X^T u_{i-1} - \alpha_{i-1} v_{i-1}$
   (b) $\alpha_i u_i = X v_i - \gamma_{i-1} u_{i-1}$

The coefficients γ_{i-1} and α_i are determined so that $\|v_i\| = \|u_i\| = 1$.

It is easy to show that the u_i vectors are orthogonal, but in floating point arithmetic they should be reorthogonalized for better accuracy [7]. The same applies to the v_i vectors.

Define $V_k = (v_1, \ldots, v_k)$ and $U_k = (u_1, \ldots, u_k)$, and

$$B_k = \begin{pmatrix} \alpha_1 & \gamma_1 & & & \\ & \alpha_2 & \gamma_2 & & \\ & & \ddots & \ddots & \\ & & & \alpha_{k-1} & \gamma_{k-1} \\ & & & & \alpha_k \end{pmatrix}. \tag{3}$$

After k steps, we can write the recursion in matrix form,

$$\begin{aligned} XV_k &= U_k B_k. \\ X^T U_k &= V_k B_k^T + \gamma_k v_{k+1} e_k^T. \end{aligned} \tag{4}$$

We refer to this as the *GK factorization*.

The approximate least squares solution after k steps of the recursion is

$$\beta_k = V_k B_k^{-1} U_k^T y,$$

and the prediction is

$$y_k = X V_k B_k^{-1} U_k^T y = U_k U_k^T y, \tag{5}$$

due to (4).

[2] LSQR is related to GK bidiagonalization in much the same way as the Conjugate Gradient algorithm is related to Lanczos tridiagonalization for a symmetric matrix.

The Frechet derivatives of β_k and y_k as functions of y can be computed by differentiation of the recursion, giving a recursion for the derivatives of the computed quantities. Note that all quantities computed in the recursion depend on y. We outline this algorithm in Appendix B. It has the drawback that in each step it involves the multiplication of matrices potentially of large dimension.

One can show that GK bidiagonalization is mathematically equivalent to applying Lanczos tridiagonalization to XX^T and X^TX simultaneously. In [6] an algorithm for computing the Frechet derivative is given, based on differentiation of the recursion for the Lanczos tridiagonalization algorithm for X^TX. For cases when $m \gg n$ it has the advantage that it avoids computing and multiplying $m \times m$ matrices, but it suffers from stability problems.

3. Computing Frechet derivatives

We will now derive an algorithm to compute the Frechet derivatives by performing a perturbation analysis of the GK algorithm. Let $\bar{y} = y(\epsilon) = y + \epsilon\delta y$, where $\|\delta y\| = 1$, be a perturbation of y; we will first use a bar[3] to denote all the quantities that are computed in the GK recursion for $\bar{y}$ and X. The matrix $\bar{B}_k$ is analogous to B_k in (3) with elements $\bar{\alpha}_i$ and $\bar{\gamma}_i$. The perturbed quantities satisfy

$$\begin{aligned} X\bar{V}_k &= \bar{U}_k\bar{B}_k. \\ X^T\bar{U}_k &= \bar{V}_k\bar{B}_k^T + \bar{\gamma}_k\bar{v}_{k+1}e_k^T, \end{aligned} \tag{6}$$

for $k = 1, 2, \ldots, t$.

Assume that altogether we have performed t steps of the GK bidiagonalization recursion, giving the matrices U_t, V_t, and B_t. The reason why we have stopped may be that the regression residual has been reduced so much that we are sure that we do not want to pursue the recursion any longer. If $\min(m, n)$ is not very large we may run the recursion to completion and have $t = \min(m, n)$. We will investigate numerically the case when $t < \min(m, n)$ in Section 4.

Having taken a decision that we are content with t steps of GK, then we are, in fact, considering a regression problem, where we have projected the data to a t-dimensional subspace of $\mathbb{R}^m$ spanned by U_t, and the solution to a t-dimensional subspace of $\mathbb{R}^n$ spanned by V_t. We assume that

$$\alpha_i \neq 0, \quad i = 1, 2, \ldots, t \tag{7}$$

$$\gamma_i \neq 0, \quad i = 1, 2, \ldots, t-1. \tag{8}$$

[3] We will use $\bar{y}$ and $y(\epsilon)$ as synonyms; the first is preferred in some places because it makes the equations look somewhat less complicated. The same convention applies to the other perturbed quantities.

We now want to compute the sensitivity of β_k and y_k, for $1 \le k \le t$, with respect to perturbations in the data vector y in the subspace spanned by the columns of U_t. If we let $\bar{y} = y(\epsilon) = y + \epsilon\delta y$, with $\|\delta y\| = 1$, then $\bar{U}_k$ and $\bar{V}_k$ will be functions of ϵ. Partition

$$U_t = (U_k \mathcal{U}_k), \quad \mathcal{U}_k \in \mathbb{R}^{m\times(t-k)}, \qquad V_t = (V_k \mathcal{V}_k), \quad \mathcal{V}_k \in \mathbb{R}^{n\times(t-k)}. \tag{9}$$

It is convenient to parameterize $\bar{U}_k$ and $\bar{V}_k$ as follows,

$$\begin{aligned} \bar{U}_k &= U_t Q_k(\epsilon) = (U_k \mathcal{U}_k) Q_k(\epsilon), \quad Q_k(\epsilon) \in \mathbb{R}^{t\times k}, \\ \bar{V}_k &= V_t P_k(\epsilon) = (V_k \mathcal{V}_k) P_k(\epsilon), \quad P_k(\epsilon) \in \mathbb{R}^{t\times k}, \end{aligned} \tag{10}$$

where $Q_k(\epsilon)$, and $P_k(\epsilon)$ have orthonormal columns, and then perform the perturbation analysis in terms of a GK factorization for Q_k and P_k given in Lemma 1 below.

Since the starting vector for the perturbed problem is

$$v_1(\epsilon) = 1/\big\|X^T y(\epsilon)\big\| X^T y(\epsilon),$$

and since $v_1(\epsilon) = V_t p_1(\epsilon)$, the projected starting vector becomes

$$p_1(\epsilon) = \frac{1}{\|B_t^T U_t^T y(\epsilon)\|} B_t^T U_t^T y(\epsilon), \tag{11}$$

where we have normalized to length 1. Note that $p_1(0) = V_t^T v_1 = e_1$.

With these definitions (6) translates into an equivalent GK factorization for $P_k(\epsilon)$, $B_k(\epsilon)$, and $Q_k(\epsilon)$.

Lemma 1. *Given the perturbed starting vector $p_1(\epsilon)$, the perturbed bidiagonal matrix $B_k(\epsilon)$ and the quantities $Q_k(\epsilon)$ and $P_k(\epsilon)$ defined in (10) satisfy the GK factorization*

$$\begin{aligned} B_t P_k(\epsilon) &= Q_k(\epsilon) B_k(\epsilon), \\ B_t^T Q_k(\epsilon) &= P_k(\epsilon) B_k^T(\epsilon) + \gamma_k(\epsilon) p_{k+1}(\epsilon) e_k^T, \end{aligned} \tag{12}$$

for $1 \le k \le t-1$.

Proof. Since $U_t = (U_k \mathcal{U}_k)$ and $V_t = (V_k \mathcal{V}_k)$ are the matrices of basis vectors from the GK recursion run t steps we have $U_t^T X V_t = B_t$. Writing the first equation in (6) using the parameterization (10), and multiplying by U_t^T we get $B_t P_k = Q_k \bar{B}_k$.

The derivation of the second equation in (12) is analogous. □

From the lemma we see that the reparametrization is equivalent to replacing the perturbed least squares problem $\min_\beta \|X\beta - \bar{y}\|$ by

$$\min_z \big\|B_t z - U_t^T \bar{y}\big\|, \quad \beta = V_t z. \tag{13}$$

If $m \geq n = t$, then the two least squares problems are completely equivalent. If $t < n$, (13) gives an approximate solution.

Thus, if the starting value $p_1(\epsilon)$ were known, along with B_t, we could compute the matrices $Q_k(\epsilon)$, $P_k(\epsilon)$, and $R_k(\epsilon)$ by a GK recursion. However, without that knowledge we can estimate the sensitivity by performing a perturbation analysis of the GK factorization (12).

3.1. Perturbation theory

Consider the GK factorization (12) of the perturbed quantities. We will differentiate these equations to estimate $\dot{Q}_k(0)$, where the dot denotes differentiation with respect to ϵ. From the Taylor expansion $Q_k(\epsilon) = Q_k(0) + \epsilon\dot{Q}_k(0) + O(\epsilon^2)$, we see that $\epsilon\dot{Q}_k(0)$ is a first order estimate of the perturbation of Q_k. Analogous statements hold for $P_k(\epsilon)$ and $B_k(\epsilon)$.

Before differentiating (12), we introduce some notation. We define $Q_k = Q_k(0)$, and similarly for the other ϵ-dependent quantities. Note that

$$Q_k = P_k = \begin{pmatrix} I_k \\ 0 \end{pmatrix}. \tag{14}$$

Define the partitioning

$$\dot{Q}_k = \begin{pmatrix} \dot{Q}_k^{(1)} \\ \dot{Q}_k^{(2)} \end{pmatrix} \in \mathbb{R}^{t\times k},$$

and similarly for $\dot{P}_k$.

Differentiating the identity $Q_k^T(\epsilon)Q_k(\epsilon) = I_k$, we get

$$\dot{Q}_k^T(\epsilon)Q_k(\epsilon) + Q_k^T(\epsilon)\dot{Q}_k(\epsilon) = 0. \tag{15}$$

Evaluating (15) for $\epsilon = 0$, and using the expression (14) for Q_k, we get

$$\dot{Q}_k^{(1)T} + \dot{Q}_k^{(1)} = 0,$$

i.e., $\dot{Q}_k^{(1)}$ is skew-symmetric. Similarly, we see that $\dot{P}_k^{(1)}$ is skew-symmetric.

We now differentiate (12) in a neighborhood of $\epsilon = 0$, and put $\epsilon = 0$:

$$\begin{aligned} B_t\dot{P}_k &= \dot{Q}_k B_k + Q_k\dot{B}_k, \\ B_t^T\dot{Q}_k &= \dot{P}_k B_k^T + P_k\dot{B}_k^T + \dot{\gamma}_k p_{k+1}e_k^T + \gamma_k\dot{p}_{k+1}e_k^T. \end{aligned} \tag{16}$$

To derive a recursion from (16), we define the partitionings

$$B_t = \begin{pmatrix} B_k & 0 \\ 0 & C_k \end{pmatrix} + \gamma_k e_k e_{k+1}^T \tag{17}$$

$$= \begin{pmatrix} B_k & 0 & 0 \\ 0 & \alpha_{k+1} & 0 \\ 0 & 0 & C_{k+1} \end{pmatrix} + \gamma_k e_k e_{k+1}^T + \gamma_{k+1} e_{k+1} e_{k+2}^T. \tag{18}$$

It follows that

$$B_t Q_k = B_t \begin{pmatrix} I_k \\ 0 \end{pmatrix} = \begin{pmatrix} B_k \\ 0 \end{pmatrix}, \qquad B_t^T Q_k = \begin{pmatrix} B_k^T \\ 0 \end{pmatrix} + \gamma_k e_{k+1} e_k^T. \tag{19}$$

We introduce the following partitioning of $\dot{P}_k$,

$$\mathbb{R}^{n\times k} \ni \dot{P}_k = \begin{pmatrix} \dot{P}_k^{(1)} \\ \dot{r}_k^T \\ \dot{Y}_k \end{pmatrix}, \quad \dot{P}_k^{(1)} \in \mathbb{R}^{k\times k},\ \dot{Y}_k \in \mathbb{R}^{(n-k-1)\times k}, \tag{20}$$

and of $\dot{Q}_k$,

$$\mathbb{R}^{n\times k} \ni \dot{Q}_k = \begin{pmatrix} \dot{Q}_k^{(1)} \\ \dot{s}_k^T \\ \dot{Z}_k \end{pmatrix}, \quad \dot{Q}_k^{(1)} \in \mathbb{R}^{k\times k},\ \dot{Z}_k \in \mathbb{R}^{(n-k-1)\times k}. \tag{21}$$

We will use the following notation for $\dot{P}_{k+1}$,

$$\mathbb{R}^{n\times(k+1)} \ni \dot{P}_{k+1} = \begin{pmatrix} \dot{P}_k^{(1)} & \dot{p}_{k+1}^{(1)} \\ \dot{r}_k^T & 0 \\ \dot{Y}_k & \dot{p}_{k+1}^{(2)} \end{pmatrix}. \tag{22}$$

From skew-symmetry we see that $\dot{p}_{k+1}^{(1)} = -\dot{r}_k$; thus when $\dot{P}_k$ is known we immediately have $\dot{P}_{k+1}^{(1)}$. The same applies to $\dot{Q}_k$ and $\dot{Q}_{k+1}^{(1)}$. We next introduce the partitionings into (16) and consider the top equations (rows 1 through k):

$$B_k \dot{P}_k^{(1)} + \gamma_k e_k \dot{r}_k^T = \dot{Q}_k^{(1)} B_k + \dot{B}_k, \tag{23}$$

$$B_k^T \dot{Q}_k^{(1)} = \dot{P}_k^{(1)} B_k^T + \dot{B}_k^T + \gamma_k \dot{p}_{k+1}^{(1)} e_k^T, \tag{24}$$

the middle equations (row $k+1$),

$$\alpha_{k+1} \dot{r}_k^T + \gamma_{k+1} e_1^T \dot{Y}_k = \dot{s}_k^T B_k, \tag{25}$$

$$\alpha_{k+1} \dot{s}_k^T = \dot{r}_k^T B_k^T + \dot{\gamma}_k e_k^T, \tag{26}$$

and the bottom equations (rows $k+1$ through p),

$$C_{k+1}\dot{Y}_k = \dot{Z}_k B_k, \tag{27}$$

$$C_{k+1}^T \dot{Z}_k + \gamma_{k+1} e_1 \dot{s}_k^T = \dot{Y}_k B_k^T + \gamma_k \dot{p}_{k+1}^{(2)} e_k^T. \tag{28}$$

Consider the first step of the algorithm, where we shall compute $\dot{q}_1$, $\dot{\alpha}_1$, and $\dot{\gamma}_1$, given $\dot{p}_1$. We will consider each of Eqs. (23)–(28) for $k = 1$. The first top Eq. (23) reads

$$\alpha_1 \dot{p}_1^{(1)} + \gamma_1 \dot{r}_1 = \alpha_1 \dot{q}_1^{(1)} + \dot{\alpha}_1.$$

Since both $\dot{p}_1^{(1)}$ and $\dot{q}_1^{(1)}$ are equal to zero, this gives $\dot{\alpha}_1 = \gamma_1 \dot{r}_1$. The second top Eq. (24) reads

$$\alpha_1 \dot{q}_1^{(1)} = \alpha_1 \dot{p}_1^{(1)} + \dot{\alpha}_1 + \gamma_1 \dot{p}_2^{(1)},$$

which gives $\gamma_1 \dot{p}_2^{(1)} = -\dot{\alpha}_1$; this is the same as we get using the skew-symmetry requirement. For $k = 1$ (25) becomes

$$\alpha_2 \dot{r}_1 + \gamma_2 e_1^T \dot{Y}_1 = \alpha_1 \dot{s}_1,$$

which gives $\dot{s}_1$, since the quantities on the left side are all known. The second middle Eq. (26) is

$$\alpha_2 \dot{s}_1 = \alpha_1 \dot{r}_1 + \dot{\gamma}_1,$$

which gives $\dot{\gamma}_1$. Eq. (27) reads $C_2 \dot{Y}_1 = \alpha_1 \dot{Z}_1$, which implies that the whole vector $\dot{q}_1 = (0\ \dot{s}_1\ \dot{Z}_1^T)^T$ is computed. Finally, the lower part of the vector $\dot{p}_2$ is obtained from (28).

We then assume that $\dot{P}_k$, $\dot{Q}_{k-1}$, $\dot{B}_{k-1}$, and $\dot{\gamma}_{k-1}$ are known.[4] In an analogous manner, by considering columns k of each of Eqs. (23)–(28), we can now compute $\dot{\alpha}_k$, $\dot{\gamma}_k$, $\dot{q}_k$, and $\dot{p}_{k+1}$.

It is clear that, provided that, provided that (7)–(8) hold, i.e., the α_i's and the γ_i's are nonzero, (23)–(28) define a linear recursion for computing $\dot{B}_k$, $\dot{Q}_k$, and $\dot{P}_{k+1}$ from a starting vector $\dot{p}_1$. The algorithm outlined above is given in full detail in Appendix C. We will refer to is as Algorithm D.

If we regard the quantities $\dot{P}_k$ and $\dot{B}_k$ as intermediate and consider only the elements of $\dot{Q}_k$ as unknowns, then it is clear that the recursion is equivalent to a block lower triangular linear system, since the first column of $\dot{Q}_k$, $\dot{q}_1$, is computed from $\dot{p}_1$, and then the rest of the columns are computed one by one. Due to skew-symmetry, only the elements below the main diagonal in $\dot{Q}_k$ need be computed. If we organize those elements column-wise from left to right in a vector

[4] As well as $\dot{P}_{k+1}^{(1)}$ and $\dot{Q}_k^{(1)}$, by skew-symmetry.

$$\mathbb{R}^{k(t-(k+1)/2)} \ni \dot{\Phi}_k = \begin{pmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \\ \vdots \\ \dot{\phi}_k \end{pmatrix}, \quad \dot{\phi}_j \in \mathbb{R}^{t-j},$$

we have a linear system of equations

$$\mathcal{K}_k \dot{\Phi}_k = \dot{\Psi}_k, \quad \dot{\Psi}_k = \begin{pmatrix} \dot{\psi}_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{29}$$

where $\dot{\Psi}_k \in \mathbb{R}^{k(t-(k+1)/2)}$ and $\dot{\psi}_1$ consists of the last $t-1$ components of $\dot{p}_1$. Partition $\mathcal{K}_k$ and its inverse K_k conformally with the partitioning of $\dot{\Phi}_k$,

$$\mathcal{K}_k = \begin{pmatrix} \mathcal{K}_{11} & & & \\ \mathcal{K}_{21} & \mathcal{K}_{22} & & \\ \vdots & \cdots & \ddots & \\ \mathcal{K}_{k1} & \cdots & \cdots & \mathcal{K}_{kk} \end{pmatrix}, \qquad K_k = \begin{pmatrix} K_{11} & & & \\ K_{21} & K_{22} & & \\ \vdots & \cdots & \ddots & \\ K_{k1} & \cdots & \cdots & K_{kk} \end{pmatrix},$$

where the blocks $\mathcal{K}_{ij}$ and K_{ij} have dimensions $(t-i) \times (t-j)$. Then the strictly lower triangular part of $\dot{Q}_k$ can be computed from

$$\dot{\phi}_i = K_{i1} \dot{\psi}_1, \quad i = 1, 2, \ldots, k. \tag{30}$$

The first block column of the inverse can be computed by solving

$$\begin{pmatrix} \mathcal{K}_{11} & & & \\ \mathcal{K}_{21} & \mathcal{K}_{22} & & \\ \vdots & \cdots & \ddots & \\ \mathcal{K}_{k1} & \cdots & \cdots & \mathcal{K}_{kk} \end{pmatrix} \begin{pmatrix} K_{11} \\ K_{21} \\ \vdots \\ K_{k1} \end{pmatrix} = \begin{pmatrix} I_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{31}$$

i.e. applying Algorithm D $t-1$ times with unit vectors as starting vectors.

Similarly, the recursion for $\dot{P}_k$ can be written as a block lower triangular linear system of equations with matrix $\mathcal{M}_k$. Denoting the column vectors of the strictly lower triangular part of $\dot{P}_k$ by $\dot{\eta}_i \in \mathbb{R}^{t-i}$, we get

$$\dot{\eta}_i = M_{i1} \dot{\psi}_1, \quad i = 1, 2, \ldots, k, \tag{32}$$

where the matrices $M_{i1} \in \mathbb{R}^{(t-i)\times(t-1)}$ are from the first block column of the inverse M_k of $\mathcal{M}_k$.

The corresponding expression for the diagonals $\dot{\alpha} = (\dot{\alpha}_1 \ \dot{\alpha}_2 \ \cdots \ \dot{\alpha}_t)^T$, and $\dot{\gamma} = (\dot{\gamma}_1 \ \dot{\gamma}_2 \ \cdots \ \dot{\gamma}_{t-1})^T$ of $\dot{B}_t$ can also be computed using Algorithm D. We can write

$$\mathcal{N}^{(\alpha)}\dot{\alpha} = \begin{pmatrix} \dot{\psi}_1 \\ 0 \end{pmatrix}, \tag{33}$$

where $\mathcal{N}^{(\alpha)} \in \mathbb{R}^{t\times t}$ is a lower triangular matrix.[5] So we can write

$$\begin{pmatrix} \dot{\alpha}_1 \\ \vdots \\ \dot{\alpha}_k \end{pmatrix} = N_k^{(\alpha)}\dot{\psi}_1, \tag{34}$$

where $N_k^{(\alpha)} \in \mathbb{R}^{k\times(t-1)}$ consists of the first k rows and the first $t-1$ columns of the inverse of $\mathcal{N}^{(\alpha)}$. Similarly

$$\begin{pmatrix} \dot{\gamma}_1 \\ \vdots \\ \dot{\gamma}_{k-1} \end{pmatrix} = N_{k-1}^{(\gamma)}\dot{\psi}_1, \tag{35}$$

where $N_{k-1}^{(\gamma)} \in \mathbb{R}^{(k-1)\times(t-1)}$. Both $N_t^{(\alpha)}$ and $N_{t-1}^{(\gamma)}$ can be computed by applying Algorithm D to the unit matrix.

Since the dimension of $\mathcal{K}_t$ is $t(t-1)/2$ the work for applying Algorithm D once is $O(t^4)$ operations. Thus the computation of all first column blocks of the inverse matrices requires $O(t^5)$ operations.

3.2. The Frechet derivative of $F_k(y)$

The Frechet derivative of a function $F(y)$ is defined as the matrix $J(y)$ that satisfies

$$\lim_{h\to 0} \frac{1}{\|h\|}\big\|F(y+h) - F(y) - Jh\big\| = 0.$$

We will now derive an expression for the Frechet derivative of F_k, using the perturbation theory.

We have $y_k = F_k(y) = U_k U_k^T y$. We now define the perturbed function in terms of our reparametrization,

$$F_k(\bar{y}) = \bar{U}_k\bar{U}_k^T\bar{y} = U_t Q_k(\epsilon)\big(Q_k(\epsilon)\big)^T \begin{pmatrix} U_k^T\bar{y} \\ \mathcal{U}_k^T\bar{y} \end{pmatrix}.$$

[5] Considering (23) for $k = t$, we see that $\dot{\alpha}_t$ is determined from $\dot{\psi}_1$, see Appendix C; thus the zero in the right hand side of (33).

Then, since

$$Q_k(\epsilon) = Q_k(0) + \epsilon \dot{Q}_k(0) + O(\epsilon^2) = \begin{pmatrix} I_k \\ 0 \end{pmatrix} + \epsilon \begin{pmatrix} \dot{Q}_k^{(1)} \\ \dot{Q}_k^{(2)} \end{pmatrix} + O(\epsilon^2), \tag{36}$$

we get

$$\begin{aligned} Q_k(\epsilon)Q_k(\epsilon)^T &= \begin{pmatrix} I_k & 0 \\ 0 & 0 \end{pmatrix} + \epsilon \begin{pmatrix} \dot{Q}_k^{(1)} & 0 \\ \dot{Q}_k^{(2)} & 0 \end{pmatrix} + \epsilon \begin{pmatrix} (\dot{Q}_k^{(1)})^T & (\dot{Q}_k^{(2)})^T \\ 0 & 0 \end{pmatrix} + O(\epsilon^2) \\ &= \begin{pmatrix} I_k & 0 \\ 0 & 0 \end{pmatrix} + \epsilon \begin{pmatrix} 0 & (\dot{Q}_k^{(2)})^T \\ \dot{Q}_k^{(2)} & 0 \end{pmatrix} + O(\epsilon^2), \end{aligned}$$

where we have taken into account the skew-symmetry of $\dot{Q}_k^{(1)}$. It follows that

$$\begin{aligned} F_k(\bar{y}) - F_k(y) &= U_t \left(\begin{pmatrix} I_k & 0 \\ 0 & 0 \end{pmatrix} + \epsilon \begin{pmatrix} 0 & (\dot{Q}_k^{(2)})^T \\ \dot{Q}_k^{(2)} & 0 \end{pmatrix} + O(\epsilon^2) \right) \begin{pmatrix} U_k^T \bar{y} \\ \mathcal{U}_k^T \bar{y} \end{pmatrix} \\ &\quad - U_t \begin{pmatrix} U_k^T y \\ 0 \end{pmatrix} \\ &= \epsilon U_k U_k \delta y + \epsilon (U_k \mathcal{U}_k) \begin{pmatrix} 0 & (\dot{Q}_k^{(2)})^T \\ \dot{Q}_k^{(2)} & 0 \end{pmatrix} \begin{pmatrix} U_k^T y \\ \mathcal{U}_k^T y \end{pmatrix} + O(\epsilon^2) \\ &=: \epsilon J_k^F(\delta y) + O(\epsilon^2). \end{aligned} \tag{37}$$

Clearly, from (29), $\dot{Q}_k$ is a linear function of $\dot{p}_1$, and by (11), a linear function of δy. Therefore $J_k^F(\delta y)$ is a linear operator and the Frechet derivative of F. However, for our computations we need an explicit expression for the matrix of J_k^F.

For $i = 1, 2, \ldots, k$ we define $K_{i1}^{(k)} \in \mathbb{R}^{(t-k)\times(t-1)}$ to be the $t-k$ last rows of the corresponding matrix block K_{i1}, and note that by (30),

$$\dot{Q}_k^{(2)} = \begin{pmatrix} K_{11}^{(k)}\dot{\psi}_1 & K_{21}^{(k)}\dot{\psi}_1 & \cdots & K_{k1}^{(k)}\dot{\psi}_1 \end{pmatrix}. \tag{38}$$

Then we set

$$\check{K}_k = \check{K}_k(U_k^T y) \in \mathbb{R}^{(t-k)\times(t-1)}, \tag{39}$$

where $\check{K}_k(z) = \sum_{i=1}^k z_i K_{i1}^{(k)}$.

Further, we define the matrix

$$\widetilde{K}_k = \begin{pmatrix} y^T \mathcal{U}_k K_{11}^{(k)} \\ y^T \mathcal{U}_k K_{21}^{(k)} \\ \vdots \\ y^T \mathcal{U}_k K_{k1}^{(k)} \end{pmatrix} \in \mathbb{R}^{k\times t}. \tag{40}$$

We will need the following lemma.

Lemma 2. *The vector $\dot{\psi}_1$, consisting of the last $t-1$ components of the vector $\dot{p}_1$, is given by*

$$\dot{\psi}_1 = \Pi_1 \delta y, \quad \Pi_1 = \tau \widehat{B}_t^T U_t^T, \tag{41}$$

where $\tau = 1/\|B_t^T U_t^T y\|$, and $\widehat{B}_t \in \mathbb{R}^{t\times(t-1)}$ consists of columns 2 to t of B_t.

Proof. From (11) we have

$$p_1(\epsilon) = \frac{1}{\|B_t^T U_t^T y(\epsilon)\|} B_t^T U_t^T y(\epsilon),$$

with $y(\epsilon) = y + \epsilon\delta y$. Differentiating this and evaluating the derivative for $\epsilon = 0$, we get

$$\dot{p}_1 = \tau\big(I - \tau^2 B_t^T U_t^T y y^T U_t B_t\big) B_t^T U_t^T \delta y. \tag{42}$$

As is seen from (11) $\tau B_t^T U_t^T y = p_1(0) = e_1$, so we get the expression (41) by removing the first row of (42). □

We now have a computable expression for the Frechet derivative of F_k.

Theorem 3. *The Frechet derivative of the function $F_k(y)$ is*

$$J_k^F = U_k U_k^T + \tau U_t \begin{pmatrix} \widetilde{K}_k \\ \check{K}_k \end{pmatrix} \widehat{B}_t^T U_t^T, \tag{43}$$

where $\widehat{B}_t$ is given in Lemma 2, and $\widetilde{K}_k$ and $\check{K}_k$ are defined in (39)–(40).

Proof. Inserting (38) into the second term of (37) we have first, with $z = U_k^T y$,

$$\begin{aligned} \dot{Q}_k^{(2)} U_k^T y &= \begin{pmatrix} K_{11}^{(k)}\dot{\psi}_1 & K_{21}^{(k)}\dot{\psi}_1 & \cdots & K_{k1}^{(k)}\dot{\psi}_1 \end{pmatrix} z \\ &= \sum_{i=1}^{k} z_i K_{i1}^{(k)} \dot{\psi}_1 = \check{K}_k \dot{\psi}_1, \end{aligned}$$

and then, similarly,

$$\begin{aligned} \big(\dot{Q}_k^{(2)}\big)^T \mathcal{U}_k^T y &= \begin{pmatrix} y^T \mathcal{U}_k K_{11}^{(k)}\dot{\psi}_1 & K_{21}^{(k)}\dot{\psi}_1 & \cdots & K_{k1}^{(k)}\dot{\psi}_1 \end{pmatrix}^T \\ &= \begin{pmatrix} y^T \mathcal{U}_k K_{11}^{(k)}\dot{\psi}_1 \\ y^T \mathcal{U}_k K_{21}^{(k)}\dot{\psi}_1 \\ \vdots \\ y^T \mathcal{U}_k K_{k1}^{(k)}\dot{\psi}_1 \end{pmatrix} = \widetilde{K}_k \dot{\psi}_1, \end{aligned}$$

which, using (41), lead to (43). □

Using the identity $\operatorname{tr}(U_t A U_t^T) = \operatorname{tr}(A)$ in (43) we get

$$\operatorname{tr}\big(J_k^F\big) = k + \tau \operatorname{tr}\left(\begin{pmatrix} \widetilde{K}_k \\ \check{K}_k \end{pmatrix} \widehat{B}_t^T \right).$$

Obviously the second term in the expression (43) for J_k^F is due to the non-linearity of the function $F_k(y)$.

3.3. The Frechet derivative of $H_k(y)$

With our parameterization the perturbed vector of regression coefficients is

$$\bar{\beta}_k = \bar{V}_k \bar{B}_k^{-1} \bar{U}_k^T \bar{y} = V_t P_k(\epsilon) \big(B_k(\epsilon)\big)^{-1} \big(Q_k(\epsilon)\big)^T U_t^T \bar{y} =: H_k(\bar{y}).$$

For small enough ϵ (i.e. for $\epsilon < 1/\|\dot{B}_k B_k^{-1}\|$), the perturbed inverse is given by

$$\bar{B}_k^{-1} = \big(B_k + \epsilon \dot{B}_k + O\big(\epsilon^2\big)\big)^{-1} = B_k^{-1} - \epsilon B_k^{-1} \dot{B}_k B_k^{-1} + O\big(\epsilon^2\big)^{-1},$$

where we have used the Neumann series expansion. Using this, (36), and the analogous expression for $P_k(\epsilon)$, we get

$$\begin{aligned} \bar{\beta}_k &= V_t \left(\begin{pmatrix} I_k \\ 0 \end{pmatrix} + \epsilon \dot{P}_k + O\big(\epsilon^2\big) \right) \big(B_k^{-1} - \epsilon B_k^{-1} \dot{B}_k B_k^{-1} + O\big(\epsilon^2\big)^{-1} \big) \\ &\quad \big((\, I_k \quad 0\,) + \epsilon \dot{Q}_k^T + O\big(\epsilon^2\big) \big) U_t^T (y + \epsilon \delta y) \\ &= V_k B_k^{-1} U_k^T y + \epsilon \big(V_k B_k^{-1} U_k^T \delta y + V_t \dot{P}_k B_k^{-1} U_k^T y \\ &\quad - V_k B_k^{-1} \dot{B}_k B_k^{-1} U_k^T y + V_k B_k^{-1} \dot{Q}_k^T U_t^T y \big) + O\big(\epsilon^2\big). \end{aligned} \tag{44}$$

Obviously the Frechet derivative is inside the parentheses, and, in order to find its matrix, we must express the terms with dotted quantities as a matrix times δy, as in the previous section.

Recall the definition of the matrix inverse M_k that gives the column vectors of $\dot{P}_k$ (32), and define the matrix blocks

$$M_{i1}^{(0)} = \begin{pmatrix} 0 \\ M_{i1} \end{pmatrix} \in \mathbb{R}^{t \times (t-1)}, \quad i = 1, 2, \ldots, k, \tag{45}$$

where we have put i zero rows at the top to get a matrix with t rows. Now we define

$$\breve{M}_k = \breve{M}_k \big[B_k^{-1} U_k^T y \big] \in \mathbb{R}^{t \times (t-1)}, \tag{46}$$

where, for $z \in \mathbb{R}^k$, $\breve{M}_k[z] = \sum_1^k z_i M_{i1}^{(0)}$. Next we define

$$M_{i1}^{(1)} = E_k^T M_{i1}^{(0)} \in \mathbb{R}^{k\times(t-1)}, \quad i = 1, 2, \ldots, k,$$

where $E_k^T = (\, I_k \quad 0\,)$; thus $M_{i1}^{(1)}$ consists of the first k rows of $M_{i1}^{(0)}$. Then we put

$$\widehat{M}_k = \begin{pmatrix} y^T U_k^T B_k^{-T} M_{11}^{(1)} \\ y^T U_k^T B_k^{-T} M_{21}^{(1)} \\ \vdots \\ y^T U_k^T B_k^{-T} M_{k1}^{(1)} \end{pmatrix} \in \mathbb{R}^{k\times(t-1)}. \tag{47}$$

Further, define

$$\widehat{K}_k = \begin{pmatrix} y^T U_t^T K_{11}^{(0)} \\ y^T U_t^T K_{21}^{(0)} \\ \vdots \\ y^T U_t^T K_{k1}^{(0)} \end{pmatrix} \in \mathbb{R}^{k\times(t-1)}, \tag{48}$$

where the definition of the matrices $K_{i1}^{(0)} \in \mathbb{R}^{t\times(t-1)}$ is analogous to that of $M_{i1}^{(0)}$. In analogy to (46) we define

$$\breve{K}_k = \breve{K}_k\big[U_k^T y\big] \in \mathbb{R}^{k\times(t-1)}, \tag{49}$$

where, for $z \in \mathbb{R}^k$, $\breve{K}_k[z] = \sum_{i=1}^k z_i K_{i1}^{(1)}$, and the definition of $K_{i1}^{(1)} \in \mathbb{R}^{k\times(t-1)}$ is completely analogous to that of $M_{i1}^{(1)}$.

Finally, define

$$\widehat{N}_k = \Omega_k N_k^{(\alpha)} \in \mathbb{R}^{k\times(t-1)}, \tag{50}$$

where $\Omega_k = \operatorname{diag}(\omega_1, \omega_2, \ldots, \omega_k) := \operatorname{diag}(B_k^{-1} U_k^T y)$, and $N_k^{(\alpha)} \in \mathbb{R}^{k\times(t-1)}$, see (34). Similarly

$$\breve{N}_k = \begin{pmatrix} \bar{\Omega}_k N_{k-1}^{(\gamma)} \\ 0 \end{pmatrix} \in \mathbb{R}^{k\times(t-1)}, \tag{51}$$

where $N_{k-1}^{(\gamma)} \in \mathbb{R}^{(k-1)\times(t-1)}$, see (35), and $\bar{\Omega}_k = \operatorname{diag}(\omega_2, \ldots, \omega_k)$.

Theorem 4. *Let the matrices $\breve{M}_k$, $\widehat{M}_k$, $\widehat{K}_k$, $\breve{K}_k$, $\widehat{N}_k$, $\breve{N}_k$, be defined by (46)–(51). The Frechet derivative of the function $H_k(y)$ is*

$$J_k^H = V_k B_k^{-1} U_k^T + \tau\big(V_t \breve{M}_k - V_k \widehat{M}_k \tag{52}$$

$$- V_k B_k^{-1} (\breve{N}_k + \widehat{N}_k - \widehat{K}_k + \breve{K}_k)\big) \widehat{B}_t^T U_t^T, \tag{53}$$

where τ and $\widehat{B}_t^T$ are given in (41).

Proof. We start from (44), and express $\dot{P}_k$ using (32), which gives

$$\dot{P}_k = \begin{pmatrix} M_{11}^{(0)}\dot{\psi}_1 & M_{21}^{(0)}\dot{\psi}_1 & \cdots & M_{k1}^{(0)}\dot{\psi}_1 \end{pmatrix} - E_k \begin{pmatrix} M_{11}^{(0)}\dot{\psi}_1 & M_{21}^{(0)}\dot{\psi}_1 & \cdots & M_{k1}^{(0)}\dot{\psi}_1 \end{pmatrix}^T E_k =: T_1 - T_2,$$

where

$$E_k = \begin{pmatrix} I_k \\ 0 \end{pmatrix} \in \mathbb{R}^{t\times k}.$$

For the third term in the right hand side of (44) we first get,

$$V_t T_1 B_k^{-1} U_k^T y = V_t \breve{M}_k \left[B_k^{-1} U_k^T y\right] \Pi_1 \delta y,$$

where we have used (46) and Lemma 2. Then, performing a few elementary matrix operations, we get

$$V_t T_2 B_k^{-1} U_k^T y = V_k \widehat{M_k} \Pi_1 \delta y,$$

This leads to the first two terms in (53). The derivation of the last two terms in (53) is analogous.

For the term in (44) involving $\dot{B}_k$ we get, due to bidiagonality,

$$\dot{B}_k B_k^{-1} U_k^T y = \begin{pmatrix} \alpha_1\omega_1 \\ \vdots \\ \alpha_{k-1}\omega_{k-1} \\ \alpha_k\omega_k \end{pmatrix} + \begin{pmatrix} \gamma_1\omega_2 \\ \vdots \\ \gamma_{k-1}\omega_k \\ 0 \end{pmatrix} = \Omega_k N_k^{(\alpha)}\dot{\psi}_1 + \begin{pmatrix} \hat{\Omega}_k N_k^{(\gamma)}\dot{\psi}_1 \\ 0 \end{pmatrix},$$

which, using Lemma 2, leads the first two terms in (53). □

In our numerical experiments we will compute the singular values of the Frechet derivative, and this is done by computing the singular values of $V_t^T J_k^H U_t$.

4. Numerical examples

In this section we first report the results of the computation of degrees of freedom for PLS using three algorithms: The one described in Appendix B (denoted D–L), the algorithm by Krämer and Sugiyama [6] (denoted D–K), mentioned at the end of Appendix B, and the one based on perturbation theory proposed in this paper (denoted Pert). We then give an example of the computation of the Frechet derivative for the regression coefficients.

The tests were performed on a four-core desktop computer with clock frequency 3.1 GHz using Matlab R2013a. Execution times were measured using Matlab's `tic-toc`

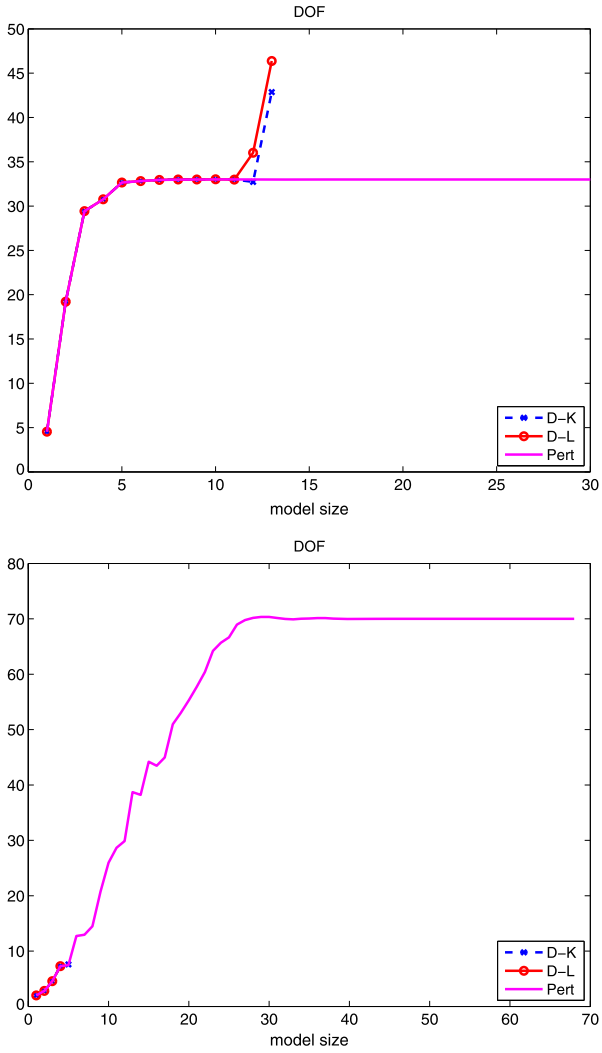

Fig. 1. The estimated number of degrees of freedom for the `kin32` problem (top) and the `cookie` problem (bottom). The model dimension is given on the horizontal axis.

functions. As is well-known, timing experiments in Matlab are difficult to evaluate, because there are many system- and coding-dependent factors that influence the execution time. The `cookie` problem is so small that all methods executed in less than a second (7 steps for D–K and D–L, 70 for Pert). The `kin32` is relatively large, and D–K and D–L executed 14 steps in 3 and 24 seconds, respectively. The Pert method required less than 0.1 second for 32 steps. Of course, only the order of magnitude is relevant in such a comparison.

We computed the degrees of freedom for two problems, the `kin32` and `cookie` problems, both described in [6] (originally published at http://www.cs.toronto.edu/~delve

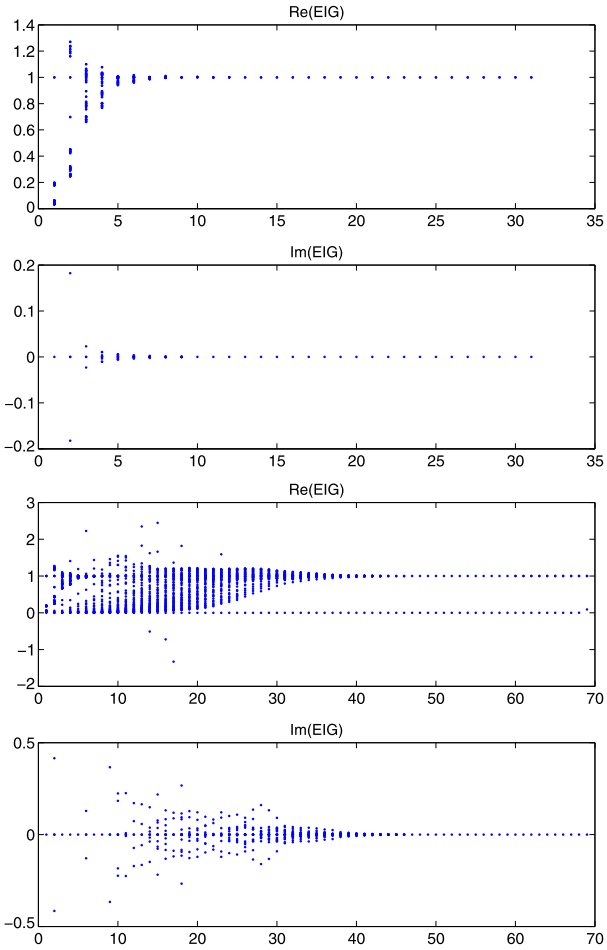

Fig. 2. Real and imaginary parts of the eigenvalues of the Frechet derivative J_k^F for the `kin32` problem (two top graphs) and the `cookie` problem (two bottom). The model dimension is given on the horizontal axis.

and in [10]). In the first problem X has dimension 8192×32 and in the second 72×700. However, in both cases the differentiation approaches D–L and D–K became completely unstable[6] quite early, so we had to stop after few steps, see the results illustrated in Fig. 1.

We also computed the eigenvalues of J_k^F for the two problems. For the `kin32` problem all the non-zero eigenvalues were very close to 1 already after 11 steps. This is about the same step when the degrees of freedom curve levels off. Similar behavior occurs for the `cookies` problem.

[6] The instability is *not* due to loss of orthogonality in the computed basis vectors: we used reorthogonalization in the GK process so that the computed matrices U_k and V_k had perfectly orthogonal column vectors in the floating point system, e.g., $\|U_t^T U_t - I\| \approx 10^{-15}$.

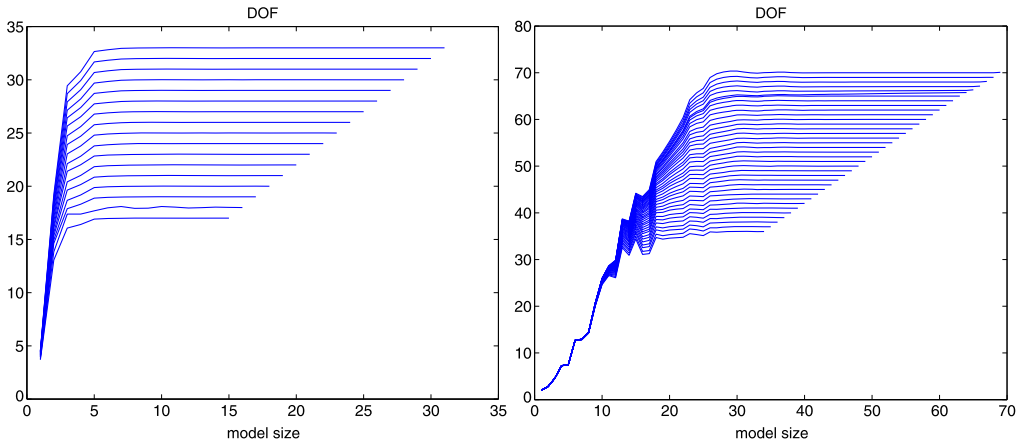

Fig. 3. Degrees of freedom curves for different values of t: `kin32` problem (left) and the `cookie` problem (right). The model dimension is given on the horizontal axis, and the value of t can be read off from the end of each curve.

If F_k had been linear, the degrees of freedom curve would have been a straight line of slope 1 (first term in (43)). The results in Figs. 1 and 2 indicate that after a small number of steps the non-linear procedure creates a model of the same complexity (in terms of the number of degrees of freedom) and with the same eigenvalue properties (of the Frechet derivative) as a linear projection of considerably higher dimension. Note that the eigenvalues of the Frechet derivative converge towards those of an orthogonal projection.

For efficiency and storage reasons it is not feasible to perform $t = \min(m, n)$ steps of the algorithm for very large problems. Therefore we must ask whether an approximation of the Frechet derivative for a smaller value of t gives the same information as with a large value. Our experiment illustrated in Fig. 3 indicates that the answer is affirmative. We computed degrees of freedoms for our two test sets with different values of t, between $\min(m, n)/2$, approximately, and $\min(m, n)$. Obviously, the maximum value for the number of degrees of freedom is $t+1$. In both cases all the curves level off to become constant at the same model dimension. This is not surprising, since the fact that PLS has captured almost all the structure of the problem at a certain step does not depend on how many extra steps are performed. On the other hand, the Frechet derivative (43) clearly depends on t. However, due to its property of choosing the basis vectors to maximize covariance, PLS orders the information so that the absolute values of the components of the vector $U_t^T y$ decrease almost monotonically, see Fig. 4, in contrast to the behavior of the coordinates of y in terms of the left singular vectors. Therefore, in these examples, the contributions from the second term in (43) to the degrees of freedom become smaller as t increases.

From the limited numerical experiments reported above, we believe that the Frechet derivative computed by our approach can be used for reliable computations of the degrees of freedom for the PLS model.

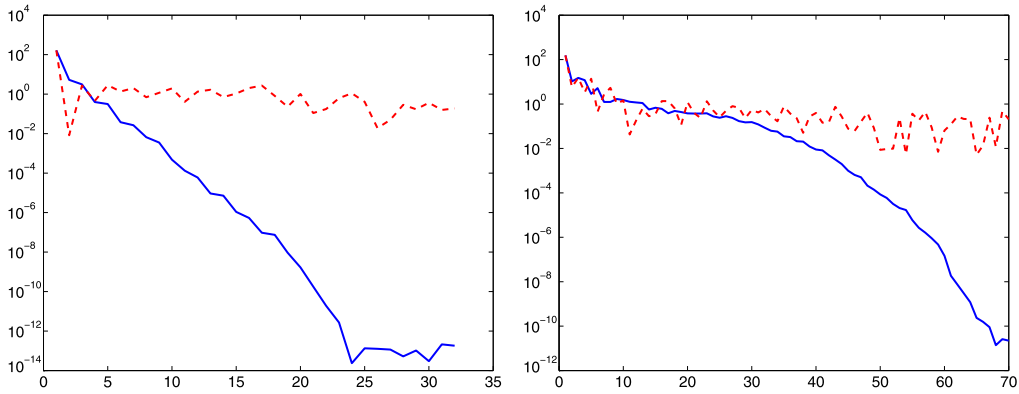

Fig. 4. Absolute values of the coordinates of the right hand side y in terms of the columns of U_t (solid curve) and the left singular vectors (dashed curve) for the `kin32` problem (left) and the `cookie` problem (right).

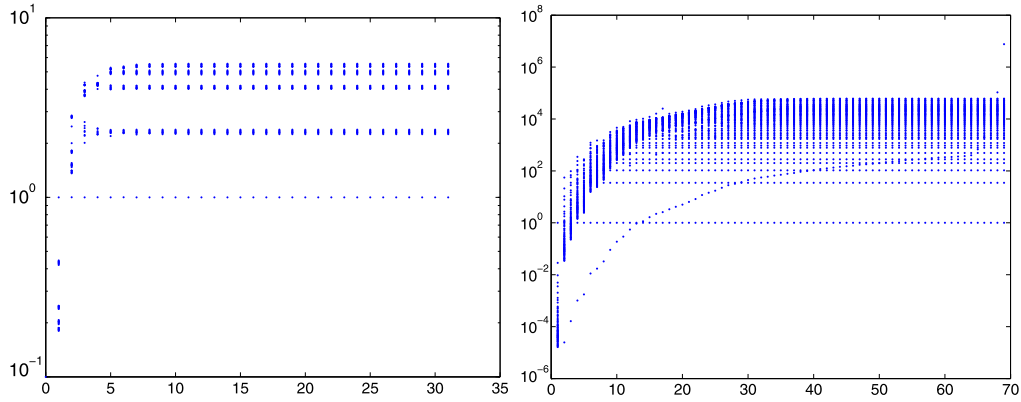

Fig. 5. The singular values of the Frechet derivative for the `kin32` problem (left) and the `cookie` problem (right). The model dimension is given on the horizontal axis.

We then computed the singular values of the Frechet derivative for different model dimensions. The results are illustrated in Fig. 5. It is seen that the singular values "stabilize" at about the same model dimension as the number of degrees of freedom curves level off.

Appendix A. PLS: The NIPALS formulation

NIPALS PLS

1. $X_0 = X$
2. **for** $i = 1, 2, \ldots, k$
   (a) $w_i = \frac{1}{\|X_{i-1}^T y\|} X_{i-1}^T y$
   (b) $t_i = \frac{1}{\|X_{i-1} w_i\|} X_{i-1} w_i$
   (c) $p_i = X_{i-1}^T t_i$
   (d) $X_i = X_{i-1} - t_i p_i^T$

In the statistics/chemometrics literature the vectors w_i, t_i, and p_i are called *weight, score, and loading vectors,* respectively. Notice that the t_i vectors are scaled to have norm 1. The deflation of the data matrix in item 2d, and the corresponding deflation of the right hand side y can be written

$$(X_i, y_i) = \big(I - t_i t_i^T\big)(X_{i-1}, y_{i-1}).$$

It is shown in [7] that deflation of the right hand side is essential for the stability of this version of PLS.

Appendix B. Computing the Frechet derivative: Differentiation approach

We here sketch the differentiation approach for computing the degrees of freedom. From (2) and (5) we get

$$D_k = 1 + \operatorname{tr}\left(\frac{\partial(U_k U_k^T y)}{\partial y}\right).$$

Since $U_k U_k^T y = \sum_{i=1}^{k} u_i u_i^T y$, DOF can be computed recursively,

$$D_k = D_{k-1} + \operatorname{tr}\left(\frac{\partial(u_k u_k^T y)}{\partial y}\right).$$

Thus we can compute the degrees of freedom D_i recursively along with the orthogonal vectors. Note that if a vector u is a function of y then

$$\frac{\partial(uu^T y)}{\partial y} = \big(u^T y I + u y^T\big)\frac{\partial u}{\partial y} + uu^T.$$

The initialization of the recursion can be rewritten

$$\begin{aligned} \tilde{v}_1 &= X^T y, \qquad \gamma_0 = \|\tilde{v}_1\|, \qquad v_1 = 1/\gamma_0 \tilde{v}_1, \\ \tilde{u}_1 &= X v_1, \qquad \alpha_1 = \|\tilde{u}_1\|, \qquad u_1 = 1/\alpha_1 \tilde{u}_1, \end{aligned}$$

with derivatives

$$\begin{aligned} \frac{\partial \tilde{v}_1}{\partial y} &= X^T, \\ \frac{\partial \gamma_0}{\partial y} &= 1/\|\tilde{v}_1\| \tilde{v}_1^T \frac{\partial \tilde{v}_1}{\partial y} = v_1^T \frac{\partial \tilde{v}_1}{\partial y}, \\ \frac{\partial v_1}{\partial y} &= -1/\gamma_0^2 \tilde{v}_1 \frac{\partial \gamma_0}{\partial y} + 1/\gamma_0 \frac{\partial \tilde{v}_1}{\partial y} = -\frac{1}{\gamma_0}\left(v_1 \frac{\partial \gamma_0}{\partial y} - \frac{\partial \tilde{v}_1}{\partial y}\right), \\ \frac{\partial \tilde{u}_1}{\partial y} &= X \frac{\partial v_1}{\partial y}, \end{aligned}$$

$$\frac{\partial \alpha_1}{\partial y} = 1/\|\tilde{u}_1\| \tilde{u}_1^T \frac{\partial \tilde{u}_1}{\partial y} = u_1^T \frac{\partial \tilde{u}_1}{\partial y},$$
$$\frac{\partial u_1}{\partial y} = -1/\alpha_1^2 \tilde{u}_i \frac{\partial \alpha_i}{\partial y} + 1/\alpha_1 \frac{\partial \tilde{u}_1}{\partial y} = -\frac{1}{\alpha_1}\left(u_1 \frac{\partial \alpha_1}{\partial y} - \frac{\partial \tilde{u}_1}{\partial y} \right).$$

Similarly, rewriting the main statements

$$\begin{aligned} \tilde{v}_i &= X^T u_{i-1} - \alpha_{i-1} v_{i-1}, \\ \gamma_{i-1} &= \|\tilde{v}_i\|, \\ v_i &= 1/\gamma_{i-1} \tilde{v}_i, \\ \tilde{u}_i &= X v_i - \gamma_{i-1} u_{i-1}, \\ \alpha_i &= \|\tilde{u}_i\|, \\ u_i &= 1/\alpha_i \tilde{u}_i, \end{aligned}$$

the corresponding statements differentiated are

$$\begin{aligned} \frac{\partial \tilde{v}_i}{\partial y} &= X^T \frac{\partial u_{i-1}}{\partial y} - v_{i-1} \frac{\partial \alpha_{i-1}}{\partial y} - \alpha_{i-1} \frac{\partial v_{i-1}}{\partial y}, \\ \frac{\partial \gamma_{i-1}}{\partial y} &= 1/\|\tilde{v}_i\| \tilde{v}_i^T \frac{\partial \tilde{v}_i}{\partial y} = v_i^T \frac{\partial \tilde{v}_i}{\partial y}, \\ \frac{\partial v_i}{\partial y} &= -1/\gamma_{i-1}^2 \tilde{v}_i \frac{\partial \gamma_{i-1}}{\partial y} + 1/\gamma_{i-1} \frac{\partial \tilde{v}_i}{\partial y} = -\frac{1}{\gamma_{i-1}}\left(v_i \frac{\partial \gamma_{i-1}}{\partial y} - \frac{\partial \tilde{v}_i}{\partial y} \right), \\ \frac{\partial \tilde{u}_i}{\partial y} &= X \frac{\partial v_i}{\partial y} - u_{i-1} \frac{\partial \gamma_{i-1}}{\partial y} - \gamma_{i-1} \frac{\partial u_{i-1}}{\partial y}, \\ \frac{\partial \alpha_i}{\partial y} &= 1/\|\tilde{u}_i\| \tilde{u}_i^T \frac{\partial \tilde{u}_i}{\partial y} = u_i^T \frac{\partial \tilde{u}_i}{\partial y}, \\ \frac{\partial u_i}{\partial y} &= -1/\alpha_i^2 \tilde{u}_i \frac{\partial \alpha_i}{\partial y} + 1/\alpha_i \frac{\partial \tilde{u}_i}{\partial y} = -\frac{1}{\alpha_i}\left(u_i \frac{\partial \alpha_i}{\partial y} - \frac{\partial \tilde{u}_i}{\partial y} \right). \end{aligned}$$

Note that the derivatives $\partial u_i/\partial y$ are matrices in $\mathbb{R}^{m\times m}$. Therefore, if $m \gg n$, the recursion is computationally very heavy, as we have to repeatedly multiply large, dense matrices by X^T. In [6, Algorithm 1] a scheme is described that avoids computing $\partial u_i/\partial y$ by applying the Lanczos recursion for $X^T X$.

Appendix C. Computation of $\dot{P}_k$, $\dot{Q}_k$, and $\dot{B}_k$

Here we give the algorithm for computing $\dot{Q}_k$, $\dot{B}_k$ and $\dot{P}_k$. For readability we use the notation $C_{k+1} = B_t(k+2:t, k+2:t)$.

Algorithm D Computation of $\dot{P}_t$, $\dot{Q}_t$, and $\dot{B}_t$.

Starting vector $\dot{P}_{1:t,1} = \dot{p}_1$

1. $\dot{Q}_{2,1} = \frac{1}{\alpha_1}(\alpha_2 \dot{P}_{2,1} + \gamma_2 \dot{P}_{3,1})$
2. $\dot{Q}_{3:t,1} = \frac{1}{\alpha_1} C_2 \dot{P}_{3:t,1}$
3. $\dot{P}_{3:t,2} = \frac{1}{\gamma_1}(C_2^T \dot{Q}_{3:t,1} + \gamma_2 \dot{Q}_{2,1} e_1 - \alpha_1 \dot{P}_{3:t,1})$
4. $\dot{\alpha}_1 = \gamma_1 \dot{P}_{2,1}$; $\dot{\gamma}_1 = \alpha_2 \dot{Q}_{2,1} - \alpha_1 \dot{P}_{2,1}$
5. **for** $k = 2, \ldots, t-2$
   (a) $\dot{Q}_{1:k-1,k} = -\dot{Q}_{k,1:k-1}^T$
   (b) $\dot{\alpha}_k = \gamma_k \dot{P}_{k+1,k} - \gamma_{k-1} \dot{Q}_{k,k-1}$
   (c) $\dot{Q}_{k+1,k} = \frac{1}{\alpha_k}(\alpha_{k+1} \dot{P}_{k+1,k} + \gamma_{k+1} \dot{P}_{k+2,k} - \gamma_{k-1} \dot{Q}_{k+1,k-1})$
   (d) $\dot{Q}_{k+2:t,k} = \frac{1}{\alpha_k}(C_{k+1} \dot{P}_{k+2:t,k} - \gamma_{k-1} \dot{Q}_{k+2:t,k-1})$
   (e) $\dot{\gamma}_k = \alpha_{k+1} \dot{Q}_{k+1,k} - \alpha_k \dot{P}_{k+1,k}$
   (f) $\dot{P}_{1:k,k+1} = -\dot{P}_{k+1,1:k}^T$
   (g) $\dot{P}_{k+2:t,k+1} = \frac{1}{\gamma_k}(C_{k+1}^T \dot{Q}_{k+2:t,k} + \gamma_{k+1} \dot{Q}_{k+1,k} e_1 - \alpha_k \dot{P}_{k+2:t,k})$
6. $\dot{Q}_{1:t-2,t-1} = -\dot{Q}_{t-1,1:t-2}^T$
7. $\dot{Q}_{t,t-1} = \frac{1}{\alpha_{t-1}}(\alpha_t \dot{P}_{t,t-1} - \gamma_{t-2} \dot{Q}_{t,t-2})$
8. $\dot{P}_{1:t-1,t} = -\dot{P}_{t,1:t-1}^T$
9. $\dot{\alpha}_{t-1} = \gamma_{t-1} \dot{P}_{t,t-1} - \gamma_{t-2} \dot{Q}_{t-1,t-2}$
10. $\dot{Q}_{1:t-1,t} = -\dot{Q}_{t,1:t-1}^T$
11. $\dot{\gamma}_{t-1} = \alpha_t \dot{Q}_{t,t-1} - \alpha_{t-1} \dot{P}_{t,t-1}$
12. $\dot{\alpha}_t = -\gamma_{t-1} \dot{Q}_{t,t-1}$

References

[1] H. Wold, Soft modeling by latent variables; the nonlinear iterative partial least squares approach, in: J. Gani (Ed.), Perspectives in Probability and Statistics, Academic Press, London, 1975, pp. 117–142. Papers in honour of M.S. Bartlett.

[2] S. Wold, M. Sjöström, L. Eriksson, PLS-regression: a basic tool of chemometrics, Chemom. Intell. Lab. Syst. 58 (2001) 109–130.

[3] L. Eldén, Partial least squares vs. Lanczos bidiagonalization I: analysis of a projection method for multiple regression, Comput. Statist. Data Anal. 46 (2004) 11–31.

[4] G.H. Golub, W. Kahan, Calculating the singular values and pseudo-inverse of a matrix, SIAM J. Numer. Anal. Ser. B 2 (1965) 205–224.

[5] S. Wold, A. Ruhe, H. Wold, W.J. Dunn, The collinearity problem in linear regression. The partial least squares (PLS) approach to generalized inverses, SIAM J. Sci. Stat. Comput. 5 (1984) 735–743.

[6] N. Krämer, M. Sugiyama, The degrees of freedom of partial least squares regression, J. Amer. Statist. Assoc. 106 (494) (2011) 697–705.

[7] A. Björck, Stability of two direct methods for bidiagonalization and partial least squares, SIMAX.

[8] Å. Björck, Numerical Methods for Least Squares Problems, SIAM, Philadelphia, PA, 1996.

[9] C.C. Paige, M. Saunders LSQR, An algorithm for sparse linear equations and sparse least squares, ACM Trans. Math. Software 8 (1982) 43–71.

[10] B. Osborne, A. Miller, T. Fearn, S. Douglas, Application of near infrared reflectance spectroscopy to the compositional analysis of biscuits and biscuit doughs, J. Sci. Food Agric. 35 (1984) 99–105.