Lanczos Approximations for the Speedup of Kernel Partial Least Squares Regression*

Nicole Krämer
Machine Learning Group
Berlin Institute of Technology
nkraemer@cs.tu-berlin.de

Masashi Sugiyama
Department of Computer Science
Tokyo Institute of Technology
sugi@cs.titech.ac.jp

Mikio L. Braun
Machine Learning Group
Berlin Institute of Technology
mikio@cs.tu-berlin.de

February 19, 2009

Abstract

The runtime for Kernel Partial Least Squares (KPLS) to compute the fit is quadratic in the number of examples. However, the necessity of obtaining sensitivity measures as degrees of freedom for model selection or confidence intervals for more detailed analysis requires cubic runtime, and thus constitutes a computational bottleneck in real-world data analysis. We propose a novel algorithm for KPLS which not only computes (a) the fit, but also (b) its approximate degrees of freedom and (c) error bars in quadratic runtime. The algorithm exploits a close connection between Kernel PLS and the Lanczos algorithm for approximating the eigenvalues of symmetric matrices, and uses this approximation to compute the trace of powers of the kernel matrix in quadratic runtime.

1 INTRODUCTION

Partial Least Squares (PLS) [28, 29] is a supervised dimensionality reduction technique. Given n observations $(\boldsymbol{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, it iteratively constructs an orthogonal set $\boldsymbol{T} = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_m) \in \mathbb{R}^{n \times m}$ of m latent features which have maximal covariance with the target variable $\boldsymbol{y} = (y_1, \ldots, y_n)$. For regression, these latent components are used as predictors in a least squares fit instead of the original data leading to fitted values

$$\widehat{\boldsymbol{y}}_m \;=\; \boldsymbol{T}\left(\boldsymbol{T}^\top \boldsymbol{T}\right)^{-1} \boldsymbol{T}^\top \boldsymbol{y} = \mathcal{P}_{\boldsymbol{T}} \boldsymbol{y}\,, \tag{1}$$

where $\mathcal{P}$ denotes the orthogonal projection operator. PLS is the standard tool e.g. in chemometrics [16], and has been successfully applied in various other scientific fields such as chemoinformatics, physiology or bioinformatics [25, 23, 1]. In combination with the kernel trick [20, 22], Kernel Partial Least Squares (KPLS) performs dimensionality reduction and

*to appear in Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS 09)

regression in a non-linear fashion. KPLS has some appealing properties over existing kernel methods. Due to its iterative nature, it only relies on matrix-vector multiplications. Hence its runtime is quadratic in the number of training examples , as opposed to – for example – Kernel Ridge Regression, which requires the inversion of a large symmetric matrix, having time complexity $\mathcal{O}(n^3)$. Furthermore, it is possible to compute the derivative of the KPLS solution with respect to $\boldsymbol{y}$ by differentiating the iterative formulation itself. Taking the trace of the derivative of the fitted values, one obtains an estimate of the degrees of freedom for KPLS, which can be used, for example, for effective model selection based on information criteria like AIC, BIC, or gMDL [13]. The first order Taylor approximation can also be used to construct confidence intervals for PLS [5, 19]. However, since we take the derivative of a vector (1), the derivative is a matrix, and the computation of the derivative involves a number of matrix-matrix multiplications which have time complexity $\mathcal{O}(n^3)$ for all practical considerations.

In this work, we propose an algorithm which computes (a) the fit of KPLS as well as (b) its approximate degrees of freedom and (c) confidence intervals for the KPLS solutions, all in quadratic runtime. These results are based on the fact that PLS is equivalent to the Lanczos method for approximating the eigenvalues of the kernel matrix $\boldsymbol{K}$ by the eigenvalues of a tridiagonal $m \times m$ matrix $\boldsymbol{D}$. The main contribution is to compute these approximate eigenvalues using KPLS *itself*. Then, using a different formulation of the derivative of the fit in KPLS, one can approximate the trace of powers $\boldsymbol{K}^j$ of the kernel matrix using the matrix $\boldsymbol{D}$. Since $\boldsymbol{D}$ is typically small (as it scales with the number of components), the runtime for computing the eigenvalues is cubic in m, and therefore, unproblematic. Since the powers of the Kernel matrices $\boldsymbol{K}^j$ are the only matrix-matrix multiplications of order n in the formula for the degrees of freedom, the approximation leads to quadratic runtime. Hence, we use the KPLS fit to approximate its degrees of freedom. In addition, using the alternative formulation of the derivative, one can perform a sensitivity analysis of KPLS resulting in confidence intervals on the estimates, also in quadratic runtime.

This paper is structured as follows. In Section 2, we review the connection between KPLS and Lanczos approximations, and summarize the state-of-the-art for computing the derivative of Kernel PLS. In Sections 3 and 4, we propose our novel formulation of the derivative together with the quadratic runtime algorithms for the degrees of freedom and the confidence intervals. We conclude with some practical examples.

PLS is closely related to Krylov methods. Therefore, we briefly recall the definition of Krylov subspaces. For a matrix $\boldsymbol{C} \in \mathbb{R}^{c \times c}$ and $\boldsymbol{c} \in \mathbb{R}^c$, we call the set of vectors $\boldsymbol{c}, \boldsymbol{C}\boldsymbol{c}, \ldots, \boldsymbol{C}^{m-1}\boldsymbol{c}$ the Krylov sequence of length m. The space spanned by these vectors is called a Krylov space and is denoted by $\mathcal{K}_m(\boldsymbol{C}, \boldsymbol{c})$.

2 BACKGROUND: PLS, LANCZOS METHODS, AND SENSITIVITY ANALYSIS

In this paper, we focus on the NIPALS algorithm [28] for PLS. For different forms of PLS, see [21]. The n centered observations $(\boldsymbol{x}_i, y_i)$ are pooled into a $n \times d$ data matrix $\boldsymbol{X}$ and a vector $\boldsymbol{y} \in \mathbb{R}^n$. PLS constructs m orthogonal latent components $\boldsymbol{T} = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_m) \in \mathbb{R}^{n \times m}$ in a greedy fashion. The first component $\boldsymbol{t}_1 = \boldsymbol{X}\boldsymbol{w}_1$ fulfills

$$\boldsymbol{w}_1 = \arg\max_{\|\boldsymbol{w}\|=1} \operatorname{cov}(\boldsymbol{X}\boldsymbol{w}, \boldsymbol{y})^2 = \frac{1}{\|\boldsymbol{X}^\top \boldsymbol{y}\|} \boldsymbol{X}^\top \boldsymbol{y}. \tag{2}$$

Subsequent components $\boldsymbol{t}_2, \boldsymbol{t}_3, \ldots$ are chosen such that they maximize the squared covariance to $\boldsymbol{y}$ and that all components are mutually orthogonal. Orthogonality can be ensured by deflating the original variables $\boldsymbol{X}$

$$\boldsymbol{X}_i \;=\; \boldsymbol{X} - \mathcal{P}_{\boldsymbol{t}_1,\ldots,\boldsymbol{t}_{i-1}}\boldsymbol{X}\,,$$

and then replacing $\boldsymbol{X}$ by $\boldsymbol{X}_i$ in (2). The matrix $\boldsymbol{W} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m) \in \mathbb{R}^{d\times m}$ can be shown to be orthogonal as well (e.g. [9]). Note furthermore that the latent components are usually scaled to unit norm. Kernel PLS [20, 22] can be derived by noting that $\boldsymbol{w}_i = \boldsymbol{X}^\top \boldsymbol{r}_i$ with

$$\boldsymbol{r}_i \;=\; (\boldsymbol{y} - \widehat{\boldsymbol{y}}_{i-1}) \,/\|\boldsymbol{K}^{1/2}\,(\boldsymbol{y} - \widehat{\boldsymbol{y}}_{i-1})\,\| \tag{3}$$

denoting the normalized residuals, and by deflating the kernel matrix $\boldsymbol{K}$ instead of $\boldsymbol{X}$,

$$\boldsymbol{K}_i \;=\; \left(\boldsymbol{I}_n - \mathcal{P}_{\boldsymbol{t}_{i-1}}\right)\boldsymbol{K}_{i-1}\left(\boldsymbol{I}_n - \mathcal{P}_{\boldsymbol{t}_{i-1}}\right)\,.$$

In contrast to e.g. Principal Component Analysis, the latent components $\boldsymbol{T}$ depend on the response, and hence the fitted values (1) are a nonlinear function of $\boldsymbol{y}$.

Recall that in the nonlinear case, KPLS depends on the kernel parameters (e.g. the width of an rbf-kernel) and the optimal number m of latent components. Thus, for model selection, one has to select the optimal combination on a grid of possible kernel parameters and components from 1 to a maximal amount m^* of components.

2.1 KRYLOV METHODS AND LANCZOS APPROXIMATION

To predict the output for a new observation, we have to derive the regression coefficients $\widehat{\boldsymbol{\beta}}_m$ (in the linear case) and kernel coefficients $\widehat{\boldsymbol{\alpha}}_m$ (in the nonlinear case), which are defined via $\widehat{\boldsymbol{y}}_m = \boldsymbol{X}\widehat{\boldsymbol{\beta}}_m = \boldsymbol{K}\widehat{\boldsymbol{\alpha}}_m$. This can be done by using the fact that PLS is equivalent to the Lanczos bidiagonalization of $\boldsymbol{X}$ [14]: The orthogonal matrices $\boldsymbol{T}$ and $\boldsymbol{W}$ represent a decomposition of $\boldsymbol{X}$ into a bidiagonal matrix $\boldsymbol{L} \in \mathbb{R}^{m\times m}$ via

$$\boldsymbol{X}\boldsymbol{W} \;=\; \boldsymbol{T}\boldsymbol{L} \tag{4}$$

with $l_{ij} = 0$ for $i > j$ or $i < j - 1$. This matrix is defined as $\boldsymbol{L} = \boldsymbol{T}^\top\boldsymbol{X}\boldsymbol{W}$. This implies [15, 9, 22]

$$\widehat{\boldsymbol{\beta}}_m = \boldsymbol{W}\boldsymbol{L}^{-1}\boldsymbol{T}^\top\boldsymbol{y} \quad \text{and} \quad \widehat{\boldsymbol{\alpha}}_m = \boldsymbol{R}\boldsymbol{L}^{-1}\boldsymbol{T}^\top\boldsymbol{y}$$

with $\boldsymbol{R} = (\boldsymbol{r}_1, \ldots, \boldsymbol{r}_m) \in \mathbb{R}^{n\times m}$. Furthermore, it can be shown [18] that PLS is equivalent to the conjugate gradient (CG) algorithm [8]. The latter is a procedure that iteratively computes approximate solutions of the normal equation $\boldsymbol{A}\boldsymbol{\beta} = \boldsymbol{b}$ (with $\boldsymbol{A} = \boldsymbol{X}^\top\boldsymbol{X}$ and $\boldsymbol{b} = \boldsymbol{X}^\top\boldsymbol{y}$) by minimizing the quadratic function $1/2\boldsymbol{\beta}^\top\boldsymbol{A}\boldsymbol{\beta} - \boldsymbol{\beta}^\top\boldsymbol{b}$ along directions that are $\mathbf{A}$-orthogonal. These search directions span the Krylov space defined by $\boldsymbol{A}$ and $\boldsymbol{b}$. The approximate solution of CG obtained after m steps is equal to the PLS estimate $\widehat{\boldsymbol{\beta}}_m$ with m components. Moreover, the weight vectors $\boldsymbol{W}$ are an orthogonal basis of $\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{b})$.

Krylov methods are also used to approximate eigenvalues of $\boldsymbol{A}$ by "restricting" $\boldsymbol{A}$ onto Krylov subspaces: In terms of the orthogonal basis $\boldsymbol{W}$ of $\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{b})$, the map

$$\boldsymbol{D} \;=\; \mathcal{P}_{\mathcal{K}_m(\boldsymbol{A},\boldsymbol{b})}\boldsymbol{A}\mathcal{P}_{\mathcal{K}_m(\boldsymbol{A},\boldsymbol{b})}$$

is represented by

$$\boldsymbol{D} \;=\; \boldsymbol{W}^\top \boldsymbol{A}\boldsymbol{W}\,. \tag{5}$$

$\boldsymbol{D}$ is shown to be tridiagonal, and the m distinct eigenvalues $\mu_1 > \mu_2 > \ldots > \mu_m$ of $\boldsymbol{D}$ – called Ritz values – are good approximations of the eigenvalues of $\boldsymbol{A}$ [24]. One immediate consequence of the connection between PLS and Krylov spaces is the fact that the latent components span the Krylov space defined by $\boldsymbol{K}$ and $\boldsymbol{K}\boldsymbol{y}$. This implies that

$$\widehat{\boldsymbol{y}}_m \;=\; \mathcal{P}_{\mathcal{K}_m(\boldsymbol{K},\boldsymbol{K}\boldsymbol{y})}\boldsymbol{y}\,. \tag{6}$$

2.2 SENSITIVITY ANALYSIS FOR KPLS

Sensitivity measures are crucial in at least two important scenarios. On the one hand, they are needed to select the correct model (in terms of a suitable kernel and the number of components) when using information criteria. On the other hand, to assess the stability of the solution, one needs to measure the influence of small noise in the training points on the learned function. For example, areas with a high sensitivity require further data points to stabilize the solution in an ambiguous area. Furthermore, if for some regions, the prediction does not depend on the training points at all, this indicates that further data points are necessary.

Both of these questions – model selection and stability analysis – can be addressed by computing the derivatives of the KPLS solution with respect to $\boldsymbol{y}$, either of the fitted labels $\widehat{\boldsymbol{y}}_m$, or of the learned kernel coefficients $\widehat{\boldsymbol{\alpha}}_m$. Let us consider the regression model

$$Y_i \;=\; f(\boldsymbol{x}_i) + \varepsilon_i\,,\; \varepsilon_i \sim \mathcal{N}(0,\sigma^2)\,. \tag{7}$$

For a general regression method with fitted values $\widehat{\boldsymbol{y}}$, the degrees of freedom are defined as [30, 6]

$$\mathrm{DoF} \;=\; E\left[\mathrm{trace}\left(\partial\widehat{\boldsymbol{y}}/\partial\boldsymbol{y}\right)\right]$$

with the expectation E taken with respect to $Y_1,\ldots,Y_n$. An unbiased plug-in estimate of the degrees of freedom is therefore given by

$$\widehat{\mathrm{DoF}} \;=\; \mathrm{trace}\left(\partial\widehat{\boldsymbol{y}}/\partial\boldsymbol{y}\right). \tag{8}$$

Degrees of freedom in combination with information criteria can be used for model selection. As the KPLS solution depends nonlinearly on $\boldsymbol{y}$, the computation of the derivative is necessary. Krämer & Braun [13] derive an algorithm for the derivative of $\widehat{\boldsymbol{y}}_m$ by transforming the Lanczos decomposition (4) into a Kernel representation and by exploiting its sparsity. The resulting iterative algorithm for (8) is then used successfully for model selection. This method scales cubically in the number of examples.

For the construction of confidence intervals for a fitted kernel function

$$\widehat{f}(\boldsymbol{x}) \;=\; \sum_{i=1}^{n} \widehat{\alpha}_i k(\boldsymbol{x},\boldsymbol{x}_i)\,.$$

one needs to study the influence of an infinitesimal perturbation in the values of $\boldsymbol{y}$. If the kernel coefficients depended linearly on $\boldsymbol{y}$ via $\widehat{\boldsymbol{\alpha}} = \boldsymbol{H}\boldsymbol{y}$, the distribution of the prediction $\widehat{f}(\boldsymbol{x})$ at any point $\boldsymbol{x}$ would be given by

$$\widehat{f}(\boldsymbol{x}) \;\sim\; \mathcal{N}\left(\boldsymbol{k}(\boldsymbol{x})^\top E\left[\widehat{\boldsymbol{\alpha}}\right], \sigma^2\boldsymbol{k}(\boldsymbol{x})^\top \boldsymbol{H}\boldsymbol{H}^\top \boldsymbol{k}(\boldsymbol{x})\right) \tag{9}$$

with $\boldsymbol{k}(\boldsymbol{x}) = (k(\boldsymbol{x}, \boldsymbol{x}_1), \ldots, k(\boldsymbol{x}, \boldsymbol{x}_n)) \in \mathbb{R}^n$. However, as KPLS depends nonlinearly on $\boldsymbol{y}$, the distribution of $\widehat{\boldsymbol{\alpha}}_m$ can only be determined approximately by using a first order Taylor expansion, i.e. one uses

$$\boldsymbol{H}_m \approx (\partial \widehat{\boldsymbol{\alpha}}_m / \partial \boldsymbol{y}) \, . \tag{10}$$

To the best of our knowledge, confidence intervals for PLS have only been constructed in the linear setting, but the results can easily been extended to the Kernel case. Phatak et. al. [19] use (6) to explicitly calculate the derivative of the PLS coefficients $\widehat{\boldsymbol{\beta}}_m$, and obtain an approximate distribution of $\widehat{\boldsymbol{\beta}}_m$. As the formula depends on matrix multiplications of order $(nm) \times (nm)$, this approach is computationally expensive. Furthermore, as the Krylov sequence $\boldsymbol{K}\boldsymbol{y}, \ldots, \boldsymbol{K}^m\boldsymbol{y}$ is nearly collinear, the formula is numerically unstable. In [5, 26], an iterative formulation of PLS is used to construct the derivative of $\widehat{\boldsymbol{\beta}}_m$. Finally, we remark that the approach by [13] using the Lanczos decomposition can be extended to the derivative of the kernel coefficients.

The drawback of all of these approaches is their poor scalability. All algorithms are cubic in the number of observations. In the following two sections we exploit that we do not need the derivative itself, but only the trace of the derivative for the degrees of freedom, and (9) for the construction of confidence intervals. The key advantage is that we can compute these approximation schemes in quadratic runtime.

3 APPROXIMATE DEGREES OF FREEDOM IN QUADRATIC RUNTIME

The key ingredients for the derivation of approximate degrees of freedom are (1) the identification of those terms that are cubic in n, and (2) the approximation of those terms using Lanczos approximations.

First, we extend the results of [19] to the computation of the derivative of $\widehat{\boldsymbol{y}}_m$. We define the $m \times m$ matrix $\boldsymbol{B}$ via $b_{ij} = \left\langle \boldsymbol{t}_i, \boldsymbol{K}^j \boldsymbol{y} \right\rangle$. The matrix is regular and upper triangular, as the latent components $\boldsymbol{T}$ are an orthogonal basis of the Krylov subspace $\mathcal{K}_m(\boldsymbol{K}, \boldsymbol{K}\boldsymbol{y})$.

Proposition 1. *Let* $\boldsymbol{c} = \boldsymbol{B}^{-1}\boldsymbol{T}^\top \boldsymbol{y}$ *and* $\boldsymbol{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) = \boldsymbol{T}\boldsymbol{B}^{-\top}$. *We have*

$$\begin{aligned} \frac{\partial \widehat{\boldsymbol{y}}_m}{\partial \boldsymbol{y}} &= \left[\boldsymbol{c}^\top \otimes (\boldsymbol{I}_n - \mathcal{P}_{\boldsymbol{T}})\right] \boldsymbol{Q}^\top + \left[\boldsymbol{V} \otimes (\boldsymbol{y} - \widehat{\boldsymbol{y}}_m)^\top\right] \boldsymbol{Q}^\top + \mathcal{P}_{\boldsymbol{T}} \\ &= \sum_{j=1}^{m} c_j \left(\boldsymbol{I}_n - \boldsymbol{T}\boldsymbol{T}^\top\right) \boldsymbol{K}^j + \sum_{j=1}^{m} \boldsymbol{v}_j \left(\boldsymbol{y} - \widehat{\boldsymbol{y}}_m\right)^\top \boldsymbol{K}^j + \boldsymbol{T}\boldsymbol{T}^\top \, . \end{aligned}$$

Here, $\otimes$ *is the Kronecker product and* $\boldsymbol{Q} = \left(\boldsymbol{K}, \boldsymbol{K}^2, \ldots, \boldsymbol{K}^m\right) \in \mathbb{R}^{n \times nm}$.

Proof. The first line follows by computing the derivative of the projection operator (6) and by applying a basis transformation from the Krylov sequence $\boldsymbol{K}\boldsymbol{y}, \ldots, \boldsymbol{K}^m\boldsymbol{y}$ to the orthogonal basis $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_m$. The latter ensures that the formula is numerically more stable. For the second line, we represent the Kronecker product as a sum. □

As a consequence, we yield a formula for the degrees of freedom of KPLS.

Corollary 2. *An unbiased estimated of the degrees of freedom of KPLS with m components is given by*

$$\begin{aligned}\widehat{\mathrm{DoF}}(m) &= \sum_{j=1}^{m} c_j \,\mathrm{trace}\left[\left(\boldsymbol{I}_n - \boldsymbol{T}\boldsymbol{T}^\top\right)\boldsymbol{K}^j\right] + \sum_{j=1}^{m} (\boldsymbol{y} - \widehat{\boldsymbol{y}}_m)^\top \boldsymbol{K}^j \boldsymbol{v}_j + m \\ &= \sum_{j=1}^{m} c_j \,\mathrm{trace}\left(\boldsymbol{K}^j\right) + m \\ &\quad - \sum_{j=1}^{m}\left(\sum_{l=1}^{m} \boldsymbol{t}_l{}^\top \boldsymbol{K}^j \boldsymbol{t}_l\right) + (\boldsymbol{y} - \widehat{\boldsymbol{y}}_m)^\top \sum_{j=1}^{m} \boldsymbol{K}^j \boldsymbol{v}_j\end{aligned}$$

This representation of the DoF of KPLS reveals an interesting feature. The computation of last line is quadratic in n, as it only involves matrix-vector multiplications. The first line however is cubic in n, as we need to compute the trace of powers of the kernel matrix $\boldsymbol{K}^j$ for $j = 1, \ldots, m$.

3.1 APPROXIMATE DEGREES OF FREEDOM VIA RITZ VALUES

As explained above, PLS is equivalent to Lanczos approximations, and can be used to approximate the eigenvalues of $\boldsymbol{X}^\top\boldsymbol{X}$ via the tridiagonal matrix $\boldsymbol{D}$ defined in (5). Note that $\boldsymbol{D}$ has a kernel representation

$$\boldsymbol{D} = \boldsymbol{R}^\top \boldsymbol{K}^2 \boldsymbol{R} = \boldsymbol{L}^\top \boldsymbol{L}\,. \tag{11}$$

with $\boldsymbol{R}$ the matrix of normalized residuals defined in (3). The eigenvalues of $\boldsymbol{D}$ are called Ritz values and constitute approximations of the eigenvalues of $\boldsymbol{X}^\top\boldsymbol{X}$ [24]. The quality of the approximation increases with the number m of latent components, and as the computation of the Ritz values scales cubically only in m, an efficient strategy is to allow a generous amount of components for the computation of $\boldsymbol{D}$.

As the eigenvalues of $\boldsymbol{X}^\top\boldsymbol{X}$ correspond to the eigenvalues of $\boldsymbol{K}$ in the kernel setting, we can use Ritz values to derive an approximation of the trace of $\boldsymbol{K}^j$.

Definition 3 (Approximate Degrees of Freedom)**.** *We define the approximate degrees of freedom of KPLS with m components as*

$$\begin{aligned}\widehat{\mathrm{DoF}}_{\mathrm{appr}}(m) &= \sum_{j=1}^{m} c_j \,\mathrm{trace}\left(\boldsymbol{D}^j_{m_{\max}}\right) + m \\ &\quad - \sum_{j=1}^{m}\left(\sum_{l=1}^{m} \boldsymbol{t}_l{}^\top \boldsymbol{K}^j \boldsymbol{t}_l\right) + (\boldsymbol{y} - \widehat{\boldsymbol{y}}_m)^\top \sum_{j=1}^{m} \boldsymbol{K}^j \boldsymbol{v}_j\,,\end{aligned}$$

where $\boldsymbol{D}_{m_{\max}}$ is the tridiagonal matrix defined in (11) computed with $m_{\max} \geq m$ latent components.

The computation of $\boldsymbol{D}$ only requires one additional $m_{\max} \times m_{\max}$ matrix multiplication $\boldsymbol{L}^\top\boldsymbol{L}$. As the matrix $\boldsymbol{D}$ is of size $m_{\max} \times m_{\max}$, the runtime for the computation is cubic in the number of maximal components $m_{\max}$ (which is typically small), and quadratic in the number n of examples.

3.2 QUALITY OF THE APPROXIMATION

Theoretically, the validity of this approximation can be justified in terms of a deviation bound.

Proposition 4 (Saad [24]). *Denote by $\mu_1, \ldots, \mu_m$ the eigenvalues of $\boldsymbol{D}$ and by $\lambda_1 \geq \ldots \geq \lambda_n$ the eigenvalues of the Kernel matrix $\boldsymbol{K}$. We have*

$$0 \leq \lambda_i - \mu_i \leq (\lambda_1 - \lambda_n) \left(\frac{\kappa_i \tan \theta_i}{C_{m-i}(1 + 2\gamma_i)} \right)^2$$

with $\boldsymbol{u}_i$ the ith eigenvector of $\boldsymbol{K}$,

$$\theta_i \quad = \quad \operatorname{acos} \frac{\left\langle \boldsymbol{y}, \sqrt{\lambda_i} \boldsymbol{u}_i \right\rangle}{\|y\|_K}$$

and

$$\kappa_i = \prod_{j=1}^{i-1} \frac{\mu_j - \lambda_n}{\mu_j - \lambda_i} \quad \gamma_i = \frac{\lambda_i - \lambda_{i-1}}{\lambda_{i+1} - \lambda_n} \,.$$

Here, C_l denotes the Chebychev polynomial of order l.

Note that θ_i is the angle between $\boldsymbol{b}$ and the ith eigenvector of $\boldsymbol{X}^\top \boldsymbol{X}$ - computed in feature space. This inequality implies that the approximation for the ith eigenvalue is good under two different scenarios. Either λ_i is already close to zero, so $\mu_i \leq \lambda_i$ is close to zero as well. For large eigenvalues λ_i, the approximation is good if (a) the eigenvalues of $\boldsymbol{K}$ decay fast, (b) the angle θ_i corresponding to the ith eigenvector is small, and (c) the index i is not too large compared to m. Property (a) is a feature of rbf-kernels, which we use throughout the rest of the paper. Condition (b) is typically fulfilled for the leading eigenvectors of $\boldsymbol{K}$ [2, 3], and condition (c) can be fulfilled by using a sufficient large amount $m_{\max}$ of components.

In practical applications, two important issues are the quality of the approximate degrees of freedom, and the quality of the model selection criteria based on these approximate degrees of freedom. In accordance with [13], we choose generalized minimum description length (gMDL) [7] as model selection criterion.

The simulation setting follows the regression model (7) with $f(x) = \operatorname{sinc}(x)$. We draw $n = 100$ inputs X_i uniformly from $[-\pi, \pi]$ and set the standard deviation to $\sigma = 0.1$. We fit KPLS with three different rbf-kernels of width $0.01, 0.1, 1$ and use different numbers $m_{\max}$ of maximal components. In addition, we compute the DoF, the approximate DoF, the gMDL criterion, and gMDL based on the approximate DoF.

Figure 1 displays the results for the different kernel widths 0.01 (left), 0.1 (center) and 1 (right). The first row shows the degrees of freedom of KPLS (blue line) and approximate degrees of freedom of KPLS depending on the number of maximal components (red dashed line). As indicated by Proposition 4, the approximation becomes more accurate if $m_{\max}$ is large. Furthermore, the approximation depends on the width of the rbf-kernel. For very small kernel widths (left), the eigenvalues of the kernel matrix decay very slowly, and more components are needed to compensate. In the second line of Figure 1, we display gMDL (blue line) and the approximate gMDL depending on the number of maximal components (red dashed line). The behavior of the approximation is qualitatively the same: It depends on the size of the kernel widths, and in general, it becomes more accurate if more components are used to compute $\boldsymbol{D}$.

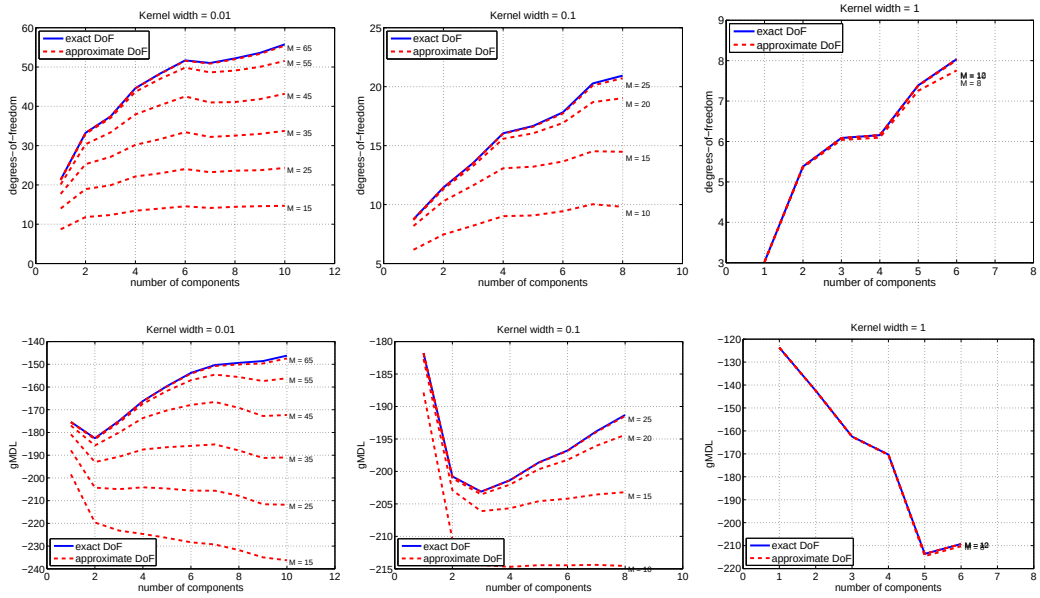

Figure 1: Quality of the approximate degrees of freedom. Results for kernel widths 0.01 (left), 0.1 (center), and 1 (right). Top row: DoF (blue line) and approximate DoF (red dashed line) for different numbers of maximal components. Bottom row: gMDL (blue line) and approximate gMDL (red dashed line) for different numbers of maximal components.

3.3 RUNTIME COMPARISON

As shown above, the approximation of the Degrees of Freedom of KPLS leads to reduction in runtime from cubic to quadratic. We now illustrate that this leads to a considerable speed up even for medium sized data. We used the kin regression data set from the delve repository[1]. This eight-dimensional synthetic data set is based on a model of a robotic arm, and the task consists in predicting the position of the arm based on the angles of its joints. It consists of 8192 data points. For sub-samples of size $100, 200, \ldots, 1000$, we compute (a) KPLS and its Degrees of Freedom for up to $m = 10$ components and (b) KPLS and its approximate Degrees of Freedom for up to $m = 10$ components. In both cases, we use a Gaussian Kernel. Note that for (b), we compute $m_{max} = 30$ components in order to obtain a close approximation, hence the number of KPLS iterations is three times higher for alternative (b). The runtime of both variants are displayed in Fig. 2. The gap between the two methods is clearly visible already for small sample sizes, and the two graphs show the expected quadratic versus cubic form. While the latter is an empirical illustration of the theoretical runtime analysis that we present above, it is important to stress that the improvement from $\mathcal{O}\left(n^3\right)$ to $\mathcal{O}\left(n^2\right)$ is not an asymptotic result but also leads to a significant improvement in runtime already for medium sized data.

[1] `http://www.cs.toronto.edu/~delve/`

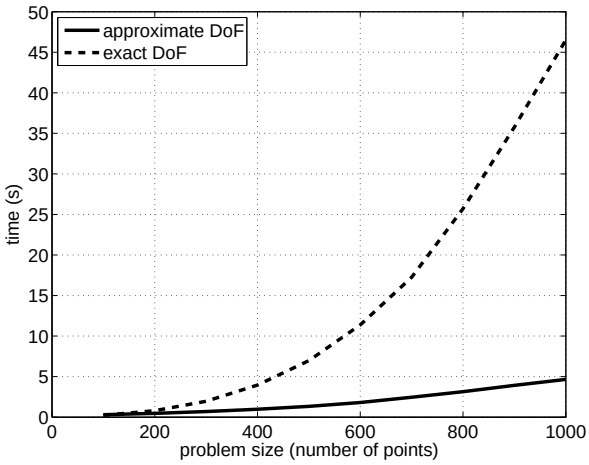

Figure 2: Comparison of runtime on the "kin" data set. Jagged line: KPLS with exact Degrees of Freedom for $m = 10$ components. Solid line: KPLS with approximate Degrees of Freedom for $m = 10$ components and $m_{max} = 30$ components for the approximation of the eigenvalues of the kernel matrix. Hence, for the approximation, the effective number of components is three times higher.

4 CONFIDENCE INTERVALS IN QUADRATIC RUNTIME

For the derivation of (approximate) confidence intervals (9), we need to compute the quantity $\boldsymbol{H}_m \boldsymbol{k}(\boldsymbol{x})$, where $\boldsymbol{H}_m \boldsymbol{y}$ is the first order Taylor approximation of the kernel coefficients $\widehat{\boldsymbol{\alpha}}_m$. Using the representation from proposition 1, we can directly compute this matrix-vector product, even without approximating the eigenvalues and thus compute the exact expression in quadratic runtime.

Note that Taylor expansions occur in both types of approximations, for the Degrees of Freedom as well as for the confidence intervals. However, there are essential conceptual differences. For the Degrees of Freedom, the representation in terms of derivatives (8) is in fact no approximation but due to the assumption of normally distributed errors in (7) which leads to Stein's Lemma [27]. In this case, the Degrees of Freedom are approximated using Lanczos methods and Ritz values. In contrast, for the confidence intervals, we have to use the Taylor expansion (10) to obtain an approximate distribution (9) for the KPLS parameters. The computation of the Taylor expansion $\boldsymbol{H}_m$ defined in (10) is cubic in n as it involves multiplications of matrices of size $n \times n$. Here, we reduce the computational cost to $\mathcal{O}\left(n^2\right)$ by cleverly exploiting the fact that the matrix-vector product $\boldsymbol{H}_m \boldsymbol{k}(\boldsymbol{x})$ is a sufficient statistic.

Proposition 5. *We have*

$$\begin{aligned} \boldsymbol{H}_m^\top \boldsymbol{k}(\boldsymbol{x}) &= \sum_{j=1}^{m} \boldsymbol{K}^{j-1} \Big\{ c_j \left(\boldsymbol{I}_n - \boldsymbol{K}\boldsymbol{T}\boldsymbol{N}\boldsymbol{R}^\top \right) \\ &\quad + \boldsymbol{K} \left(\boldsymbol{y} - \widehat{\boldsymbol{y}}_m \right) \boldsymbol{u}_j^\top \Big\} \boldsymbol{k}(\boldsymbol{x}) + \boldsymbol{T}\boldsymbol{N}\boldsymbol{R}^\top \boldsymbol{k}(\boldsymbol{x}) \,. \end{aligned}$$

with $\boldsymbol{R}$ denoting the matrix of normalized residuals, $\boldsymbol{N}$ denoting the $m \times m$ diagonal matrix consisting of elements $n_{ii} = 1/\|\boldsymbol{K}\boldsymbol{r}_i\|$ and

$$\boldsymbol{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m) = \boldsymbol{R}\boldsymbol{N}\boldsymbol{B}^{-\top} \,.$$

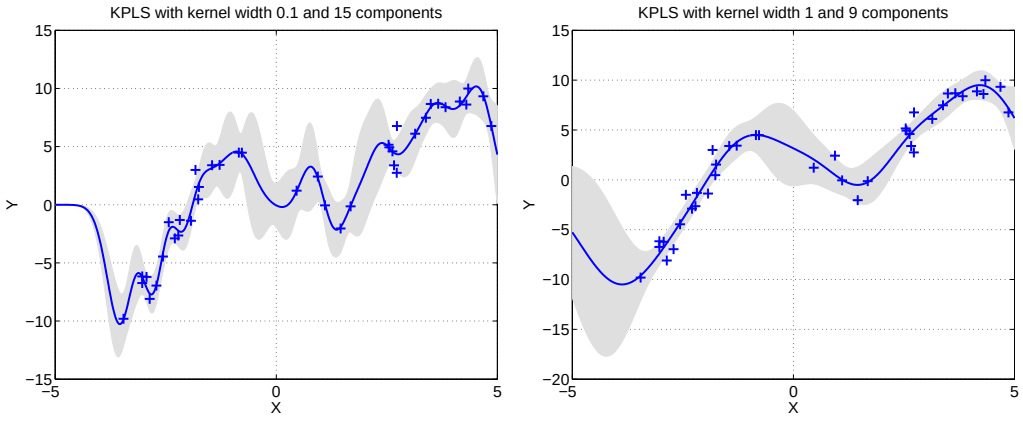

Figure 3: Confidence Intervals for KPLS. Left: KPLS with 15 components and an rbf-kernel of width 0.1 Right: KPLS with 9 components and an rbf kernel of width 1

Proof. As

$$\partial \widehat{\boldsymbol{y}}_m / \partial \boldsymbol{y} \quad = \quad \boldsymbol{K} \left(\partial \widehat{\boldsymbol{\alpha}}_m / \partial \boldsymbol{y} \right) ,$$

the formula can be shown by "canceling out" $\boldsymbol{K}$ in the formula of the derivative of $\widehat{y}_m$, and then multiplying the formula with $\boldsymbol{k}(\boldsymbol{x})$. □

Illustration Again, we use the regression model (7), with $f(x) = (x-1)(x+2)(x-1.5)\exp(-x^2/10)$ and $\sigma = 1$. We draw $n = 40$ points X_i from a mixture of two normal distribution with mean -2 and 3 a variance of 1 in both cases. We fit KPLS with for two different models, (1) KPLS with 15 components and an rbf-kernel of width 0.1 and (2) KPLS with 9 components and an rbf kernel of width 1. Figure 3 shows the KPLS fit and its confidence intervals (based on a level of 98%) for the two models.

In areas with high data density, the prediction is quite stable with small confidence intervals. Next to such high density areas, the predictions becomes unstable, as they can depend quite sensitively on the neighboring data. Finally, when one moves far away from the data points, their influence decreases to zero. This is much more apparent in the left plot with the small kernel widths.

5 CONCLUSION

We proposed an implementation of the Kernel PLS method which not only computes the fit in quadratic time, but a degree-of-freedom estimate and confidence intervals based on a sensitivity analysis, which formerly required cubic runtime. The latter estimates can be used, for example, for model selection, or to measure the local stability of the learned function. The approximation schemes exploit the fact that Kernel PLS can be extended to compute Lanczos type approximations of the eigenvalues as well. Together with a novel formula for computing the derivatives of the kernel parameters $\boldsymbol{\alpha}$, these approximations allow us to replace costly computation of powers of the kernel matrix. In summary, one obtains a Kernel PLS algorithm which also provides relevant additional information for model selection and provide further insight into the complexity and stability of the learned function.

Our results capitalize on the close connection between the dimensionality reduction technique PLS on the one hand and Krylov methods and Lanczos approximations on the other hand. While the latter two methods are commonly used in numerical linear algebra, their benefits for data analysis have not yet been exploited sufficiently. Only recently (e.g. [17, 4, 10]) they are utilized explicitly in a machine learning framework. Recent research results on the correspondence of penalization techniques and preconditioning of linear systems [11, 12] further underpin the strong potential of these methods. We strongly believe that the interplay between numerical linear algebra and machine learning will further stimulate the field of data analysis.

Acknowledgement This work is funded in part by the FP7-ICT Programme of the European Community, under the PASCAL2 Network of Excellence, ICT-216886, by the BMBF grant FKZ 01-IS07007A (ReMind), by the MEXT Grant-in-Aid for Young Scientists (A), 20680007, and by the JFE 21st Century Foundation.

References

[1] A.-L. Boulesteix and K. Strimmer. Partial Least Squares: A Versatile Tool for the Analysis of High-Dimensional Genomic Data. *Briefings in Bioinformatics*, 8(1):32–44, 2007.

[2] M. L. Braun, J. M. Buhmann, and K.-R. Müller. Denoising and Dimension Reduction in Feature Space. *Advances in Neural Information Processing Systems*, 19:185–192, 2007.

[3] M.L. Braun, J.M. Buhmann, and K.R. Müller. On Relevant Dimensions in Kernel Feature Spaces. *Journal of Machine Learning Research*, 9:1875–1908, 2008.

[4] N. de Freitas, Y. Wang, M. Mahdaviani, and D. Lang. Fast Krylov Methods for N-body Learning. *Advances in Neural Information Processing Systems*, 18:251–258, 2006.

[5] M. C. Denham. Prediction Intervals in Partial Least Squares. *Journal of Chemometrics*, 11(1):39–52, 1997.

[6] B. Efron. The Estimation of Prediction Error: Covariance Penalties and Cross-Validation. *Journal of the American Statistical Association*, 99(467):619–633, 2004.

[7] M. Hansen and B. Yu. Model Selection and Minimum Descripion Length Principle. *Journal of the American Statistical Association*, 96:746–774, 2001.

[8] M. Hestenes and E. Stiefel. Methods for Conjugate Gradients for Solving Linear Systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

[9] A. Hoskuldsson. PLS Regression Methods. *Journal of Chemometrics*, 2(3):211–228, 1988.

[10] T. Ide and K. Tsuda. Change Point Detection Using Krylov Subspace Learning. In *Proceedings of the SIAM International Conference on Data Mining*, pages 515 – 520, 2007.

[11] A. Kondylis and J. Whittaker. Spectral Preconditioning of Krylov Spaces: Combining PLS and PC Regression. *Computational Statistics and Data Analysis*, 50(5):2588–2603, 2008.

[12] N. Krämer, A.-L. Boulesteix, and G. Tutz. Penalized Partial Least Squares with Applications to B-Spline Transformations and Functional Data. *Chemometrics and Intelligent Laboratory Systems*, 94:60–69, 2008.

[13] N. Krämer and M.L. Braun. Kernelizing PLS, Degrees of Freedom, and Efficient Model Selection. In *Proceedings of the 24th International Conference on Machine Learning*, pages 441 – 448, 2007.

[14] C. Lanczos. An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators. *Journal of Research of the National Bureau of Standards*, 45:225–280, 1950.

[15] R. Manne. Analysis of Two Partial-Least-Squares Algorithms for Multivariate Calibration. *Chemometrics and Intelligent Laboratory Systems*, 2:187–197, 1987.

[16] H. Martens and T. Naes. *Multivariate Calibration*. Wiley, New York, 1989.

[17] C.S. Ong, X. Mary, S. Canu, and A.J. Smola. Learning with non-positive kernels. In *Proceedings of the twenty-first international conference on Machine learning*, pages 639–646, 2004.

[18] A. Phatak and F. de Hoog. Exploiting the Connection between PLS, Lanczos, and Conjugate Gradients: Alternative Proofs of Some Properties of PLS. *Journal of Chemometrics*, 16:361–367, 2002.

[19] A. Phatak, P.M. Rilley, and A. Penlidis. The Asymptotic Variance of the Univariate PLS Estimator. *Linear Algebra and its Applications*, 354:245–253, 2002.

[20] S. Rännar, F. Lindgren, P. Geladi, and S. Wold. A PLS Kernel Algorithm for Data Sets with many Variables and Fewer Objects, Part I: Theory and Applications. *Journal of Chemometrics*, 8:111–125, 1994.

[21] R. Rosipal and N. Krämer. Overview and Recent Advances in Partial Least Squares. In *Subspace, Latent Structure and Feature Selection Techniques*, Lecture Notes in Computer Science, pages 34–51. 2006.

[22] R. Rosipal and L.J. Trejo. Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Spaces. *Journal of Machine Learning Research*, 2:97–123, 2001.

[23] R. Rosipal, L.J. Trejo, and B. Matthews. Kernel PLS-SVC for Linear and Nonlinear Classification. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 640–647, Washington, DC, 2003.

[24] Y. Saad. *Iterative methods for sparse linear systems*. PWS, 1st edition, 1996.

[25] H. Saigo, N.Krämer, and K. Tsuda. Partial Least Squares Regression for Graph Mining. In *14 International Conference on Knowledge Discovery and Data Mining*, pages 578–586, 2008.

[26] S. Serneels, P. Lemberge, and P.J. Van Espen. Calculation of PLS Prediction Intervals Using Efficient Recursive Relations for the Jacobian Matrix. *Journal of Chemometrics*, 18:76–80, 2004.

[27] C. Stein. Estimation of the mean of a multivariate normal distribution. *Annals of Statistics*, 9(6):1135–1151, 1981.

[28] H. Wold. Path models with Latent Variables: The NIPALS Approach. In H.M. Blalock et al., editor, *Quantitative Sociology: International Perspectives on Mathematical and Statistical Model Building*, pages 307–357. Academic Press, 1975.

[29] S. Wold, H. Ruhe, H. Wold, and W.J. Dunn III. The Collinearity Problem in Linear Regression. The Partial Least Squares (PLS) Approach to Generalized Inverses. *SIAM Journal of Scientific and Statistical Computations*, 5:735–743, 1984.

[30] J. Ye. On Measuring and Correcting the Effects of Data Mining and Model Selection. *Journal of the American Statistical Association*, 93(441):120–131, 1998.