Lanczos Approximations for the Speedup of Kernel Partial Least Squares Regression

Nicole Krämer[a], Masashi Sugiyama[b], Mikio L. Braun[a]
nkraemer@cs.tu-berlin.de, sugi@cs.titech.ac.jp, mikio@cs.tu-berlin.de

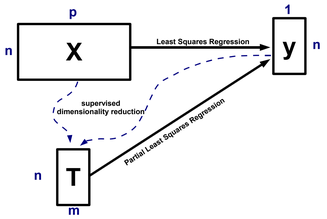
[a]Berlin Institute of Technology, Machine Learning Group [b]Tokyo Institute of Technology, Department of Computer Science

Summary

- The Degrees of Freedom of Kernel Partial Least Squares (KPLS) require all eigenvalues of the kernel matrix $\boldsymbol{K}$, hence the computation is cubic in the number of observations n.
- We use Kernel PLS *itself* to approximate the eigenvalues of the kernel matrix.
  $\longrightarrow$ We can compute approximate Degrees of Freedom of KPS in $\mathcal{O}\left(n^2\right)$!
- We can also compute approximate confidence intervals for KPLS in $\mathcal{O}\left(n^2\right)$!

Main Results

Kernel Partial Least Squares (KPLS)



- Find m orthogonal latent components $\boldsymbol{t}_i = \boldsymbol{X}\widetilde{\boldsymbol{w}}_i$ with maximal covariance to the response $\boldsymbol{y}$.

$$\widetilde{\boldsymbol{w}}_i = \arg\max_{\|\widetilde{\boldsymbol{w}}\|=1} \operatorname{cov}\left(\boldsymbol{X}\widetilde{\boldsymbol{w}}, \boldsymbol{y}\right) \quad \text{s.t. } \boldsymbol{X}\widetilde{\boldsymbol{w}} \perp \boldsymbol{X}\widetilde{\boldsymbol{w}}_l = 0 \quad \text{for } l < i\,.$$

$\rightarrow$ The latent components $\boldsymbol{T} = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_m)$ depend on the output $\boldsymbol{y}$ as well.

- $\boldsymbol{T}$ is used instead of $\boldsymbol{X}$ in a least-squares fit
  $\widehat{\boldsymbol{y}}_m = \boldsymbol{T}\left(\boldsymbol{T}^\top\boldsymbol{T}\right)^{-1}\boldsymbol{T}^\top\boldsymbol{y} = \mathcal{P}_{\boldsymbol{T}}\boldsymbol{y}$
  ($\mathcal{P} =$ projection operator)

Degrees of Freedom of KPLS

Unbiased estimate for the Degrees of Freedom of KPLS [2]

$$\widehat{\mathrm{DoF}}(m) = \operatorname{trace}\left(\frac{\partial \widehat{\boldsymbol{y}}_m}{\partial \boldsymbol{y}}\right).$$

Bad News

$$\widehat{\mathrm{DoF}}(m) = \sum_{j=1}^{m} c_j \operatorname{trace}\left(\mathbf{K^j}\right) + \mathcal{O}\left(n^2\right)$$

We need the eigenvalues of the kernel matrix $\boldsymbol{K}$ for the computation of the Degrees of Freedom of KPLS. The computation of the degrees of freedom of KPLS is cubic in the number of observations.

Lanczos Approximations

Partial Least Squares is equivalent to the Lanczos decomposition of $\boldsymbol{X}$

$$\boldsymbol{T}\boldsymbol{L} = \boldsymbol{X}\boldsymbol{W}$$

with $\boldsymbol{T}$ and $\boldsymbol{W}$ orthogonal and $\boldsymbol{L}$ upper bidiagonal. The eigenvalues of the $m \times m$ tridiagonal matrix $\boldsymbol{D} = \boldsymbol{L}^\top\boldsymbol{L}$ are good approximations of the eigenvalues of $\boldsymbol{K}$ [4].

Good News

The eigenvalues of the kernel matrix can be approximated by KPLS itself. Replace $\boldsymbol{K}^j$ by $\boldsymbol{D}^j$ in the formula for the degrees of freedom. The approximate Degrees of Freedom can be computed in quadratic runtime.

Approximate Confidence Intervals for KPLS

First order Taylor approximation of the kernel coefficients

$$\boldsymbol{H}_m = \left(\partial\widehat{\boldsymbol{\alpha}}_m/\partial\boldsymbol{y}\right).$$

leads to an approximate distribution of the predictions $\widehat{\boldsymbol{y}}(\boldsymbol{x})$.

More Good News

The product of $\boldsymbol{H}_m$ with a vector is a sufficient statistic for the confidence intervals of KPLS. It can be computed in $\mathcal{O}(n^2)$.

Details

PLS Implementation

Partial Least Squares

Input: $\boldsymbol{X}_1 = \boldsymbol{X}$, $\boldsymbol{y}$, m
for i=1,...,m **do**
  $\boldsymbol{w}_i = \boldsymbol{X}_i^\top\boldsymbol{y}$
  $\boldsymbol{t}_i = \boldsymbol{X}_i\boldsymbol{w}_i$ (component)
  $\boldsymbol{X}_{i+1} = \boldsymbol{X}_i - \mathcal{P}_{\boldsymbol{t}_i}\boldsymbol{X}_i$ (deflation)
end for
$\boldsymbol{L} = \boldsymbol{T}^\top\boldsymbol{X}\boldsymbol{W}$ (upper diagonal $m \times m$ matrix)
$\widehat{\boldsymbol{\beta}}_m = \boldsymbol{W}\boldsymbol{L}^{-1}\boldsymbol{T}^\top\boldsymbol{y}$ (regression vector)

Kernel Partial Least Squares

Input: $\boldsymbol{K}$, $\boldsymbol{y}$, m, $\widehat{\boldsymbol{y}}_0 = \boldsymbol{t}_0 = \boldsymbol{0}$
for i=1,...,m **do**
  $\widetilde{\boldsymbol{r}}_i = \boldsymbol{y} - \widehat{\boldsymbol{y}}_{i-1}$ (residuals)
  $\boldsymbol{t}_i = \left(\boldsymbol{I}_n - \mathcal{P}_{t_{i-1}}\right)\boldsymbol{K}\widetilde{\boldsymbol{r}}_i$ (component)
  $\widehat{\boldsymbol{y}}_i = \widehat{\boldsymbol{y}}_{i-1} + \mathcal{P}_{\boldsymbol{t}_i}\boldsymbol{y}$ (fitted values)
end for
$\boldsymbol{L} = \boldsymbol{T}^\top\boldsymbol{K}\widetilde{\boldsymbol{R}}$ (upper diagonal $m \times m$ matrix)
$\widehat{\boldsymbol{\alpha}}_m = \widetilde{\boldsymbol{R}}\boldsymbol{L}^{-1}\boldsymbol{T}^\top\boldsymbol{y}$ (kernel coefficients)

The columns of $\boldsymbol{T}$ span the same space as the *Krylov* sequence $\boldsymbol{K}\boldsymbol{y}, \boldsymbol{K}^2\boldsymbol{y}, \ldots, \boldsymbol{K}^m\boldsymbol{y}$. Hence

$$\widehat{\boldsymbol{y}}_m = \mathcal{P}_{(\boldsymbol{K}\boldsymbol{y}, \boldsymbol{K}^2\boldsymbol{y}, \ldots, \boldsymbol{K}^m\boldsymbol{y})}\boldsymbol{y}\,. \tag{1}$$

The first derivative of KPLS is computed either via an iterative algorithm [2] or via formula (1) [3]. The computation time is cubic in the number of observations (as it involves matrix-matrix multiplications). Unfortunately, this is also true for its trace:

$$\widehat{\mathrm{DoF}}(m) = \sum_{j=1}^{m} c_j \operatorname{trace}\left(\mathbf{K^j}\right) + m - \sum_{j=1}^{m}\left(\sum_{l=1}^{m} \boldsymbol{t}_l{}^\top\boldsymbol{K}^j\boldsymbol{t}_l\right) + (\boldsymbol{y} - \widehat{\boldsymbol{y}}_m)^\top\sum_{j=1}^{m}\boldsymbol{K}^j\boldsymbol{v}_j \tag{2}$$

with $b_{ij} = \left\langle \boldsymbol{t}_i, \boldsymbol{K}^j\boldsymbol{y}\right\rangle$, $\boldsymbol{c} = \boldsymbol{B}^{-1}\boldsymbol{T}^\top\boldsymbol{y}$ and $\boldsymbol{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) = \boldsymbol{T}\boldsymbol{B}^{-\top}$.

KPLS and Lanczos Decompositions

$$\boldsymbol{L} = \boldsymbol{T}^\top\boldsymbol{X}\boldsymbol{W} = \begin{pmatrix} * & * & 0 & 0 & \ldots & 0 \\ 0 & * & * & 0 & \ldots & 0 \\ & & \vdots & \vdots & & \\ 0 & \ldots & 0 & 0 & * & * \\ 0 & \ldots & 0 & 0 & 0 & * \end{pmatrix} \in \mathbb{R}^{m\times m} \quad \text{and} \quad \boldsymbol{D} = \boldsymbol{L}^\top\boldsymbol{L} \in \mathbb{R}^{m\times m}$$

The eigenvalues of $\boldsymbol{D}$ are good approximations of the eigenvalues of $\boldsymbol{K}$ (a) for the leading eigenvalues of $\boldsymbol{K}$, (b) if $m \gg 0$, (c) if the eigenvalues of $\boldsymbol{K}$ decay fast [4].

Approximate Degrees of Freedom in $\mathcal{O}(n^2)$

Compute $\boldsymbol{D}$ for a large number of components $m_{\max} \geq m$ and replace $\boldsymbol{K}^j$ by $\boldsymbol{D}^j$ in formula (2).

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

Approximate Confidence Intervals in $\mathcal{O}(n^2)$

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

with $\boldsymbol{k}(\boldsymbol{x}) = (k(\boldsymbol{x}, \boldsymbol{x}_1), \ldots, k(\boldsymbol{x}, \boldsymbol{x}_n)) \in \mathbb{R}^n$ and σ the noise level.

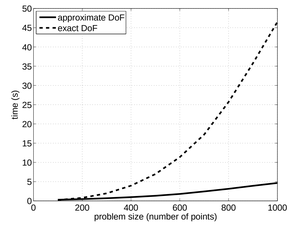
$$\boldsymbol{H}_m^\top\boldsymbol{k}(\boldsymbol{x}) = \sum_{j=1}^{m}\boldsymbol{K}^{j-1}\left\{c_j\left(\boldsymbol{I}_n - \boldsymbol{K}\boldsymbol{T}\boldsymbol{N}\boldsymbol{R}^\top\right) + \boldsymbol{K}\left(\boldsymbol{y} - \widehat{\boldsymbol{y}}_m\right)\boldsymbol{u}_j^\top\right\}\boldsymbol{k}(\boldsymbol{x})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

with $\boldsymbol{r}_i = \widetilde{\boldsymbol{r}}_i/\|\widetilde{\boldsymbol{r}}_i\|_{\boldsymbol{K}}$ (residuals) $\boldsymbol{N} = \operatorname{diag}\left(1/\|\boldsymbol{K}\widetilde{\boldsymbol{r}}_i\|\right)$ and $\boldsymbol{U} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m) = \boldsymbol{R}\boldsymbol{N}\boldsymbol{B}^{-\top}$.

Experiments

Runtime Comparison

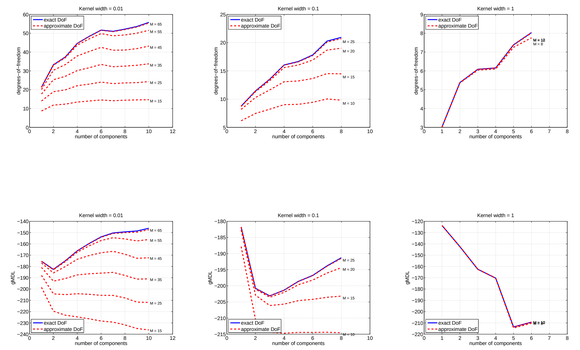
Comparison of runtime on the "kin" data set [1] ($p = 8$ dimensions and $n = 8192$ observations). Jagged line: KPLS with exact Degrees of Freedom for $m = 10$ components. Solid line: KPLS with approximate Degrees of Freedom for $m = 10$ components and $m_{max} = 30$ components for the approximation of the eigenvalues of the kernel matrix. Hence, for the approximation, the effective number of components is three times higher.



Quality of the Approximation

Computation of KPLS with rbf-kernels on simulated data

$$f(\boldsymbol{x}) = \operatorname{sinc}(\boldsymbol{x}) + \varepsilon,\ \varepsilon \sim \mathcal{N}(0, 0.1^2) \quad \boldsymbol{x}_i \sim \mathcal{U}[-\pi, \pi]\,, i = 1, \ldots, 100\,.$$



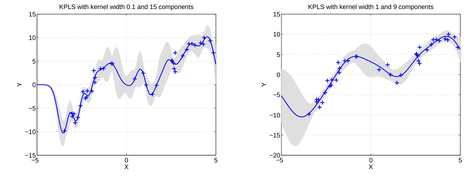
Results for kernel widths 0.01 (left), 0.1 (center), and 1 (right). Top row: DoF (blue line) and approximate DoF (red dashed line) for different numbers of maximal components. Bottom row: generalized Mimimum Description Length (gMDL) (blue line) and approximate gMDL (red dashed line) for different numbers of maximal components.

Approximate Confidence Intervals for KPLS

Computation of KPLS with rbf-kernels on simulated data

$$f(x) = (x-1)(x+2)(x-1.5)\exp(-x^2/10) + \varepsilon,\ \varepsilon \sim \mathcal{N}(0,1)$$

$n = 40$ observations x_i are drawn from a mixture of $\mathcal{N}(-2, 1)$ and $\mathcal{N}(3, 1)$.



Confidence intervals for two different kernel widths. Left: KPLS with 15 components and an rbf-kernel of width 0.1 and Right: KPLS with 9 components and an rbf kernel of width 1.

References

[1] *Delve repository*, `http://www.cs.toronto.edu/~delve/`.

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

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

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