Machine Intelligence 2

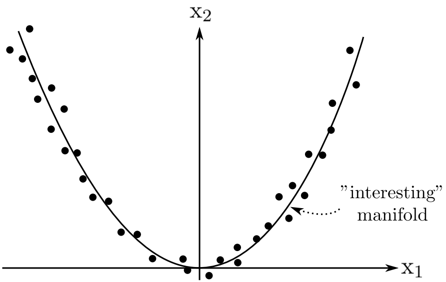
1.3 Kernel PCA

Prof. Dr. Klaus Obermayer

Fachgebiet Neuronale Informationsverarbeitung (NI)

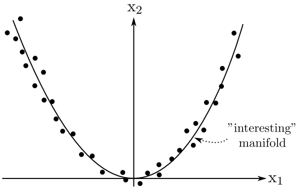
SS 2018

Kernel Principal Component Analysis: motivation

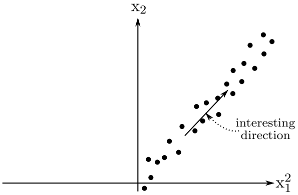


- standard PCA: two directions with high variance
- but: only one "interesting" manifold (nonlinear combination of the elementary features)

Kernel Principal Component Analysis: intuition



original (data) space



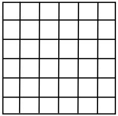
transformed (feature) space

Agenda

1. data preprocessing:
   nonlinear transformation into an "appropriate" feature space
   $\underline{\phi} : \underline{\mathbf{x}} \mapsto \underline{\phi}_{(\underline{\mathbf{x}})}$
2. application of standard (linear) PCA

Projections & kernels

relevant feature spaces may be extremely high-dimensional



pixel image

- interesting structure in correlations (of high order) between pixel values
- suitable feature space: space spanned by all d^{th}-order monomials

  example: $d = 2$

  $$\underline{\phi}_{(\underline{\mathrm{x}})} = (1,\ \mathrm{x}_1,\ \mathrm{x}_2,\ \ldots\ \mathrm{x}_N,\ \mathrm{x}_1^2,\ \mathrm{x}_1\mathrm{x}_2,\ \mathrm{x}_2^2,\ \mathrm{x}_1\mathrm{x}_3,\ \mathrm{x}_2\mathrm{x}_3,\ \mathrm{x}_3^2,\ \ldots \mathrm{x}_N^2)^T$$

- dimensionality $O(N^d)$ prohibits "direct" application of this idea

$\rightsquigarrow$ application of the **kernel trick** to avoid this problem (cf. *MI I*)

PCA & scalar products

eigenvalue problem of PCA:

$$\underline{\mathbf{C}}\,\underline{\mathbf{e}} = \lambda \underline{\mathbf{e}}$$

expansion of the eigenvectors:

$$\underline{\mathbf{e}} = \sum_{\beta=1}^{p} a^{(\beta)} \underline{\mathbf{x}}^{(\beta)}$$

PCs always lie in the subspace spanned by the (centered) data.

eigenvectors $\underline{\mathbf{e}} \in \mathbb{R}^N$, coefficients $\underline{\mathbf{a}} \in \mathbb{R}^p$: potential problem: $p \gg N$

PCA & scalar products

eigenvalue problem:

$$\underline{\mathbf{C}}\,\underline{\mathbf{e}} = \lambda \underline{\mathbf{e}}$$

ansatz:

$$\underline{\mathbf{e}} = \sum_{\beta=1}^{p} a^{(\beta)} \underline{\mathbf{x}}^{(\beta)} \qquad \underbrace{\underline{\mathbf{C}} = \frac{1}{p} \sum_{\alpha=1}^{p} \underline{\mathbf{x}}^{(\alpha)} \left(\underline{\mathbf{x}}^{(\alpha)} \right)^T}_{\text{centered data}}$$

$$\frac{1}{p} \sum_{\alpha,\beta=1}^{p} a^{(\beta)} \overbrace{\left[\left(\underline{\mathbf{x}}^{(\alpha)} \right)^T \underline{\mathbf{x}}^{(\beta)} \right]}^{\text{scalar product}} \underline{\mathbf{x}}^{(\alpha)} = \lambda \sum_{\beta=1}^{p} a^{(\beta)} \underline{\mathbf{x}}^{(\beta)}$$

Multiply from left with $\left(\underline{\mathbf{x}}^{(\gamma)}\right)^T$, $\gamma = 1, \ldots, p$:

$$\frac{1}{p} \sum_{\alpha,\beta=1}^{p} a^{(\beta)} \overbrace{\left[\left(\underline{\mathbf{x}}^{(\alpha)}\right)^T \underline{\mathbf{x}}^{(\beta)}\right]}^{\text{scalar product}} \overbrace{\left[\left(\underline{\mathbf{x}}^{(\gamma)}\right)^T \underline{\mathbf{x}}^{(\alpha)}\right]}^{\text{scalar product}} = \lambda \sum_{\beta=1}^{p} a^{(\beta)} \overbrace{\left[\left(\underline{\mathbf{x}}^{(\gamma)}\right)^T \underline{\mathbf{x}}^{(\beta)}\right]}^{\text{scalar product}}$$

$$\left(\underline{\mathbf{x}}^{(\alpha)}\right)^T \underline{\mathbf{x}}^{(\beta)} =: K_{\alpha\beta}$$

in matrix notation:

$$\underline{\mathbf{K}}^2 \underline{\mathbf{a}} = p\lambda \underline{\mathbf{K}}\, \underline{\mathbf{a}}$$

$\underline{\mathbf{K}}$: $p \times p$ matrix of scalar products between data, $K_{\alpha\beta} = \left(\underline{\mathbf{x}}^{(\alpha)}\right)^T \underline{\mathbf{x}}^{(\beta)}$

λ_k : variance along Principal Component $\underline{\mathbf{e}}_k$

$\underline{\mathbf{a}}_k$: Principal Component, represented in the basis $\left\{\underline{\mathbf{x}}^{(\alpha)}\right\}, \alpha = 1, \ldots, p$

Remark

$\underline{\mathbf{K}}$ is symmetric and positive semidefinite.

For an arbitrary vector $\underline{\mathbf{y}}$:

$$
\begin{aligned}
\underline{\mathbf{y}}^T \underline{\mathbf{K}} \underline{\mathbf{y}} &= \sum_{\alpha,\beta=1}^{p} y^{(\alpha)} \left(\underline{\mathbf{x}}^{(\alpha)} \right)^T \underline{\mathbf{x}}^{(\beta)} y^{(\beta)} \\
&= \left(\sum_{\alpha=1}^{p} y^{(\alpha)} \underline{\mathbf{x}}^{(\alpha)} \right)^2 \\
&\geq 0
\end{aligned}
$$

Transformed eigenvalue problem

$$\underline{\mathbf{K}}^2\underline{\mathbf{a}} = p\lambda\underline{\mathbf{K}}\,\underline{\mathbf{a}}$$

$$\underline{\mathbf{K}}(\underline{\mathbf{K}}\,\underline{\mathbf{a}} - p\lambda\underline{\mathbf{a}}) = 0$$

- $\leadsto$ solution $\underline{\mathbf{a}}$ is eigenvector of $\underline{\mathbf{K}}$
- $\leadsto$ if $\underline{\mathbf{K}}$ has zero eigenvalues: solution $\underline{\mathbf{a}}$ is a linear combination of one eigenvector with non-zero λ and all eigenvectors with zero eigenvalues *(see blackboard)*
- $\leadsto$ transformed eigenvalue problem

$$\underline{\mathbf{K}}\,\underline{\mathbf{a}} = p\lambda\underline{\mathbf{a}}$$

Normalization

$$\underline{\mathbf{e}}_k = \sum_{\beta=1}^{p} a_k^{(\beta)}$$

$$\underline{\mathbf{e}}_k^2 = \sum_{\alpha,\beta=1}^{p} a_k^{(\alpha)} \left(\underline{\mathbf{x}}^{(\alpha)}\right)^T \underline{\mathbf{x}}^{(\beta)} a_k^{(\beta)}$$

$$= \underline{\mathbf{a}}_k^T \underline{\mathbf{K}}\, \underline{\mathbf{a}}_k \quad = p\lambda_k \underline{\mathbf{a}}_k^2 \quad \stackrel{!}{=} 1$$

$$\boxed{\underline{\mathbf{a}}_k^{\text{norm.}} = \frac{1}{\sqrt{p\lambda_k}} \underline{\mathbf{a}}_k}$$

Projecting onto PCs

feature extraction:

$$\begin{aligned} u_k(\underline{\mathbf{x}}) &= \underline{\mathbf{e}}_k^T \cdot \underline{\mathbf{x}} \\ &= \sum_{\beta=1}^{p} a_k^{(\beta)} \underbrace{\left[\left(\underline{\mathbf{x}}^{(\beta)} \right)^T \cdot \underline{\mathbf{x}} \right]}_{\text{scalar product}} \end{aligned}$$

The kernel trick

$$\underline{\phi} : \underline{\mathbf{x}} \xrightarrow{\text{nonlinear transformation}} \underline{\phi}_{(\underline{\mathbf{x}})}$$

Kernel trick

$\Rightarrow$ formulate PCA in feature space (replace $\underline{\mathbf{x}}^{(\alpha)}$ by $\underline{\phi}_{(\underline{\mathbf{x}}^{(\alpha)})}$)

$\Rightarrow$ replace all scalar products by "kernel functions"

$$\underline{\phi}^T_{(\underline{\mathbf{x}})} \underline{\phi}_{(\underline{\mathbf{x}}')} \longleftrightarrow k(\underline{\mathbf{x}}, \underline{\mathbf{x}}')$$

Mercer's theorem

Every **positive semidefinite definite** kernel k corresponds to a scalar product in some metric feature space (*cf. MI I*).

If a linear method can be formulated solely in terms of scalar products, a nonlinear version can be derived without an explicit projection into the (high-dimensional) feature space!

Mercer's theorem

Statement of the theorem

Every **positive semidefinite** kernel k corresponds to a scalar product in some metric feature space.

Consider

- $D \subset \mathbb{R}^N$ compact *subset of data space*
- $k : D \times D \to \mathbb{R}$ is a continuous and symmetric function (''kernel'')
- T_k is the corresponding integral operator

$$T_k : L_{2(D)} \to L_{2(D)},$$
$$\left(T_k f\right)_{(\underline{\mathbf{x}})} := \int_D k_{(\underline{\mathbf{x}}, \underline{\mathbf{x}}')} f_{(\underline{\mathbf{x}}')} d\underline{\mathbf{x}}$$

with eigenvalues λ_j and normalized eigenfunctions $\psi_j \in L_{2(D)}$

Mercer's theorem: condition and its consequences

Essential part

If T_k is positive semidefinite, i.e.

$$\langle T_k f, f \rangle \;=\; \int\limits_{D \times D} k_{(\underline{\mathbf{x}}, \underline{\mathbf{x}}')} f_{(\underline{\mathbf{x}})} f_{(\underline{\mathbf{x}}')} d\underline{\mathbf{x}} d\underline{\mathbf{x}}' \;\geq\; 0 \qquad \forall f \in L_{2(D)}$$

then $\quad k_{(\underline{\mathbf{x}}, \underline{\mathbf{x}}')} = \sum_{j=1}^{M} \lambda_j \psi_{j(\underline{\mathbf{x}})} \psi_{j(\underline{\mathbf{x}}')}$ with $\lambda_j \geq 0$

k corresponds to a scalar product in an M-dimensional space:

$$\underline{\phi} : \underline{\mathbf{x}} \mapsto \left(\sqrt{\lambda_1} \psi_{1(\underline{\mathbf{x}})}, \sqrt{\lambda_2} \psi_{2(\underline{\mathbf{x}})}, \ldots, \sqrt{\lambda_M} \psi_{M(\underline{\mathbf{x}})}, \right)^T$$

$$\Longrightarrow \; k_{(\underline{\mathbf{x}}, \underline{\mathbf{x}}')} = \underline{\phi}^T_{(\underline{\mathbf{x}})} \underline{\phi}_{(\underline{\mathbf{x}}')} \qquad \text{(with } M \leq \infty\text{)}$$

Common kernel functions

$$k_{(\underline{\mathbf{x}},\underline{\mathbf{x}}')} = \left(\underline{\mathbf{x}}^T \underline{\mathbf{x}}' + 1\right)^d$$

polynomial kernel of degree d
image processing (pixel correlation)

$$k_{(\underline{\mathbf{x}},\underline{\mathbf{x}}')} = \exp\left\{ - \frac{\left(\underline{\mathbf{x}} - \underline{\mathbf{x}}'\right)^2}{2\sigma^2} \right\}$$

RBF-kernel with range σ
infinite dimensional feature space

$$k_{(\underline{\mathbf{x}},\underline{\mathbf{x}}')} = \tanh\left\{ K \underline{\mathbf{x}}^T \underline{\mathbf{x}}' + \theta \right\}$$

neural network kernel with parameters K and θ
not necessarily positive definite

Centering the kernel matrix

$$\frac{1}{p}\sum_{\alpha=1}^{p} \underline{\mathbf{x}}^{(\alpha)} \stackrel{!}{=} \underline{\mathbf{0}} \quad \not\rightarrow \quad \frac{1}{p}\sum_{\alpha=1}^{p} \underline{\phi}\Big(\underline{\mathbf{x}}^{(\alpha)}\Big) = \underline{\mathbf{0}}$$

"centered" feature vectors:

$$\underbrace{\underline{\phi}\Big(\underline{\mathbf{x}}^{(\alpha)}\Big)}_{\text{"centered" feature vectors}} = \underline{\widetilde{\phi}}\Big(\underline{\mathbf{x}}^{(\alpha)}\Big) - \frac{1}{p}\sum_{\gamma=1}^{p} \underbrace{\underline{\widetilde{\phi}}\Big(\underline{\mathbf{x}}^{(\gamma)}\Big)}_{\text{uncentered feature vectors}}$$

Centering the kernel matrix

$$
\begin{aligned}
K_{\alpha\beta} &= \underline{\phi}^T_{\left(\underline{\mathbf{x}}^{(\alpha)}\right)} \cdot \underline{\phi}_{\left(\underline{\mathbf{x}}^{(\beta)}\right)} \\
&= \left(\underline{\widetilde{\phi}}^T_{\left(\underline{\mathbf{x}}^{(\alpha)}\right)} - \frac{1}{p} \sum_{\gamma=1}^{p} \underline{\widetilde{\phi}}^T_{\left(\underline{\mathbf{x}}^{(\gamma)}\right)} \right) \left(\underline{\widetilde{\phi}}_{\left(\underline{\mathbf{x}}^{(\beta)}\right)} - \frac{1}{p} \sum_{\delta=1}^{p} \underline{\widetilde{\phi}}_{\left(\underline{\mathbf{x}}^{(\delta)}\right)} \right) \\
&= \underline{\widetilde{\phi}}^T_{\left(\underline{\mathbf{x}}^{(\alpha)}\right)} \underline{\widetilde{\phi}}_{\left(\underline{\mathbf{x}}^{(\beta)}\right)} - \frac{1}{p} \sum_{\delta=1}^{p} \underline{\widetilde{\phi}}^T_{\left(\underline{\mathbf{x}}^{(\alpha)}\right)} \underline{\widetilde{\phi}}_{\left(\underline{\mathbf{x}}^{(\delta)}\right)} \\
&\quad - \frac{1}{p} \sum_{\gamma=1}^{p} \underline{\widetilde{\phi}}^T_{\left(\underline{\mathbf{x}}^{(\gamma)}\right)} \underline{\widetilde{\phi}}_{\left(\underline{\mathbf{x}}^{(\beta)}\right)} + \frac{1}{p^2} \sum_{\gamma,\delta=1}^{p} \underline{\widetilde{\phi}}^T_{\left(\underline{\mathbf{x}}^{(\gamma)}\right)} \underline{\widetilde{\phi}}_{\left(\underline{\mathbf{x}}^{(\delta)}\right)} \\
&= \underbrace{\widetilde{K}_{\alpha\beta}}_{= k\left(\underline{\mathbf{x}}^{(\alpha)}, \underline{\mathbf{x}}^{(\beta)}\right)} - \underbrace{\frac{1}{p} \sum_{\delta=1}^{p} \widetilde{K}_{\alpha\delta}}_{\text{row avg.}} - \underbrace{\frac{1}{p} \sum_{\gamma=1}^{p} \widetilde{K}_{\gamma\beta}}_{\text{col. avg.}} + \underbrace{\frac{1}{p^2} \sum_{\gamma,\delta=1}^{p} \widetilde{K}_{\gamma\delta}}_{\text{matrix avg.}}
\end{aligned}
$$

Centering & projections

For data points $\underline{\mathbf{x}}^{(\alpha)}$ we have onto the k-th PC (in feature space):

$$u_k\left(\underline{\phi}_{(\underline{\mathbf{x}}^{(\alpha)})}\right) = \sum_{\beta=1}^{p} a_k^{(\beta)} K_{\beta\alpha} \quad \leftarrow \text{use centered kernel matrix and normalized eigenvector!}$$

More generally, for new/arbitrary $\underline{\mathbf{x}} \in \mathbb{R}^N$ the projection is computed as:

$$u_k\left(\underline{\phi}_{(\underline{\mathbf{x}})}\right) = \sum_{\beta=1}^{p} a_k^{(\beta)} \underline{\phi}^T_{(\underline{\mathbf{x}}^{(\beta)})} \underline{\phi}_{(\underline{\mathbf{x}})} \quad \leftarrow \text{centered feature vectors}$$

$$= \sum_{\beta=1}^{p} a_k^{(\beta)} \left(\left[\underline{\widetilde{\phi}}_{(\underline{\mathbf{x}}^{(\beta)})} - \frac{1}{p} \sum_{\gamma=1}^{p} \underline{\widetilde{\phi}}_{(\underline{\mathbf{x}}^{(\gamma)})} \right]^T \left[\underline{\widetilde{\phi}}_{(\underline{\mathbf{x}})} - \frac{1}{p} \sum_{\delta=1}^{p} \underline{\widetilde{\phi}}_{(\underline{\mathbf{x}}^{(\delta)})} \right] \right)$$

$$= \sum_{\beta=1}^{p} a_k^{(\beta)} \left(k(\underline{\mathbf{x}}^{(\beta)}, \underline{\mathbf{x}}) - \frac{1}{p} \sum_{\delta=1}^{p} \widetilde{K}_{\beta\delta} - \frac{1}{p} \sum_{\gamma=1}^{p} k(\underline{\mathbf{x}}^{(\gamma)}, \underline{\mathbf{x}}) + \frac{1}{p^2} \sum_{\gamma,\delta=1}^{p} \widetilde{K}_{\gamma\delta} \right)$$

1. calculation of the un-normalized kernel matrix

$$\widetilde{K}_{\alpha\beta} = k\big(\underline{\mathbf{x}}^{(\alpha)}, \underline{\mathbf{x}}^{(\beta)}\big), \qquad \alpha, \beta = 1, \ldots, p$$

2. centering

$$K_{\alpha\beta} = \widetilde{K}_{\alpha\beta} - \frac{1}{p}\sum_{\delta=1}^{p} \widetilde{K}_{\alpha\delta} - \frac{1}{p}\sum_{\gamma=1}^{p} \widetilde{K}_{\gamma\beta} + \frac{1}{p^2}\sum_{\gamma,\delta=1}^{p} \widetilde{K}_{\gamma\delta}$$

3. solve the eigenvalue problem $\boxed{\frac{1}{p}\underline{\mathbf{K}}\,\widetilde{\underline{\mathbf{a}}}_k = \lambda_k \widetilde{\underline{\mathbf{a}}}_k}$

4. normalization of eigenvectors to unit length

$$\underline{\mathbf{a}}_k = \frac{1}{\sqrt{p\lambda_k}}\widetilde{\underline{\mathbf{a}}}_k$$

5. calculation of projections

$$u_k\left(\underline{\phi}_{(\underline{\mathbf{x}}^{(\alpha)})}\right) = \sum_{\beta=1}^{p} a_k^{(\beta)} K_{\beta\alpha}$$

$\leftarrow$ use centered kernel matrix and normalized eigenvector!

Comments

- kernel-PCA $\hat{=}$ PCA in feature space
  - $\rightsquigarrow$ projections onto PCs are uncorrelated
  - $\rightsquigarrow$ λ_k: variance of the data along PC k (in feature space)
- #PCs typically exceed #dimensions in the original space
- kernel PCA can be used for feature extraction & dimensionality reduction
  - e.g. solve classification problems in feature space
- optimal kernel parameters (σ, d, etc.) depend on data & task
  - selection via cross-validation possible for classification tasks
  - no general measure-of-goodness of the PC projections available
- custom kernels can be used (any positive definite kernel matrix)

Comments

- expansion of PCs into data points is <u>not</u> sparse
  - $\rightsquigarrow$ calculating projections can be computationally expensive
  - $\rightsquigarrow$ use expansions with less data points $q < p$

$$\underline{\mathbf{e}}_k = \sum_{\beta=1}^{p} a_k^{(\beta)} \underline{\phi}\big(\underline{\mathbf{x}}^{(\beta)}\big) \quad \rightsquigarrow \quad \widehat{\underline{\mathbf{e}}}_k = \sum_{\gamma=1}^{q} \widehat{a}_k^{(\gamma)} \underline{\phi}\Big(\overbrace{\underline{\mathbf{z}}^{(\gamma)}}^{\underline{\mathbf{x}}^{(i_\gamma)}}\Big)$$

$$\big(\underline{\mathbf{e}}_k - \widehat{\underline{\mathbf{e}}}_k\big)^2 =: \varphi \to \min_{\widehat{a}_k^{(\gamma)}, \underline{\mathbf{z}}^{(\gamma)}}$$

$$\varphi = 1 + \sum_{\gamma,\delta=1}^{q} \widehat{a}_k^{(\gamma)} \widehat{a}_k^{(\delta)} k_{\left(\underline{\mathbf{z}}^{(\gamma)}, \underline{\mathbf{z}}^{(\delta)}\right)} - 2 \sum_{\beta=1}^{p} \sum_{\gamma=1}^{q} a_k^{(\beta)} \widehat{a}_k^{(\gamma)} k_{\left(\underline{\mathbf{x}}^{(\beta)}, \underline{\mathbf{z}}^{(\gamma)}\right)}$$

- $p \gg N$: kernel matrices may be very large
  - $\rightsquigarrow$ only eigenvectors with largest eigenvalues are of interest
  - $\rightsquigarrow$ use specialized (iterative) routines (e.g. ARPACK via `eigs`)

$\Rightarrow$ analysis is performed in feature space (not data space)

Application: feature extraction

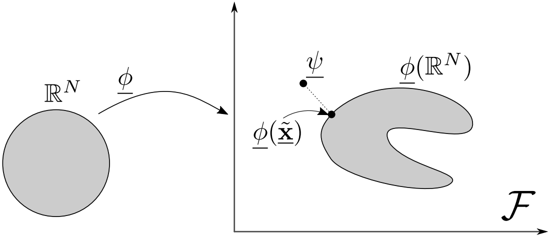
# of components	test error (%) for different polynomial kernels						
	1	2	3	4	5	6	7
32	9.6	8.8	8.1	8.5	9.1	9.3	10.8
64	8.8	7.3	6.8	6.7	6.7	7.2	7.5
128	8.6	5.8	5.9	6.1	5.8	6.0	6.8
256	8.7	5.5	5.3	5.2	5.2	5.4	5.4
512	n.a.	4.9	4.6	4.4	5.1	4.6	4.9
1024	n.a.	4.9	4.3	4.4	4.6	4.8	4.6
2048	n.a.	4.9	4.2	4.1	4.0	4.3	4.4

- Test error rates on the USPS handwritten digit database
- linear SVMs trained on nonlinear Principal Components
- nonlinear PCs extracted by PCA with a polynomial kernel (degrees 1 through 7)
- dimensionality of the space is 256 (16x16 pixel images)

Source: Schölkopf, 2002

Reconstruction

- reconstruction in data space non-trivial
  - data space is mapped to a low-dim. manifold in feature space



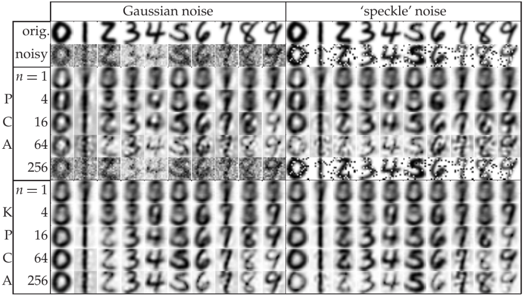
$$\underline{\psi} \in \mathcal{F} = \overline{\operatorname{span} \underline{\phi}(\mathbb{R}^N)}$$

- problem: in general there is no "pre-image" $\tilde{\underline{\mathbf{x}}}$ s.t. $\underline{\psi} = \underline{\phi}(\tilde{\underline{\mathbf{x}}})$
- solution: calculation of approximate "pre-images":

$$\tilde{\underline{\mathbf{x}}} = \operatorname*{argmin}_{\underline{\mathbf{x}}} \left\| \underline{\phi}(\underline{\mathbf{x}}) - \underline{\psi} \right\|^2$$

- algorithms: Schölkopf & Smola, ch. 18 (e.g. impl. in scikit-learn)

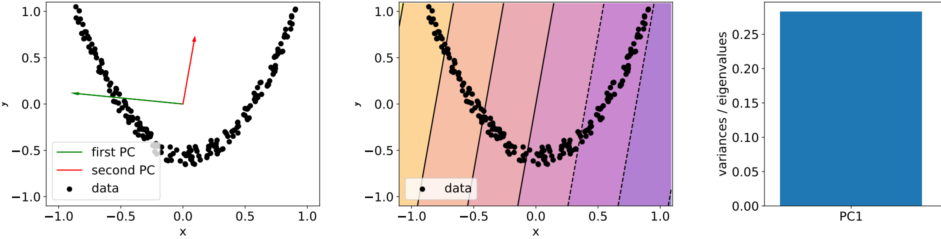
Application: denoising



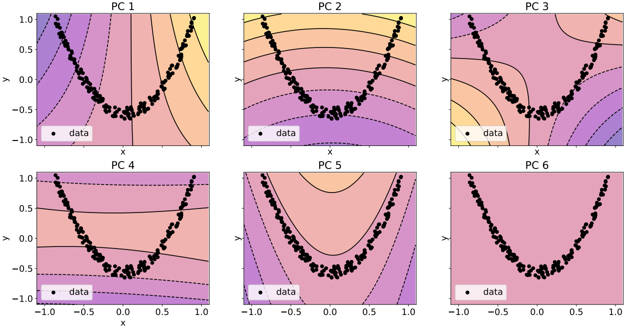
- Denoising of USPS data
- *First row*: original data (digits); *Second row*: noise added to original digits (Gaussian and "speckle")
- *Following five rows*: reconstruction of the noisy digits achieved with linear PCA using $n = 1,4,16,64,256$ components
- *Last five rows*: reconstruction of the noisy digits achieved with kernel PCA using the same number of components
- dimensionality of the space is 256 (16x16 pixel images)

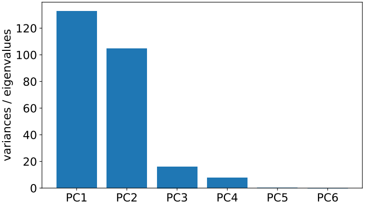
Source: Schölkopf, 2002

Parabola example: PCA

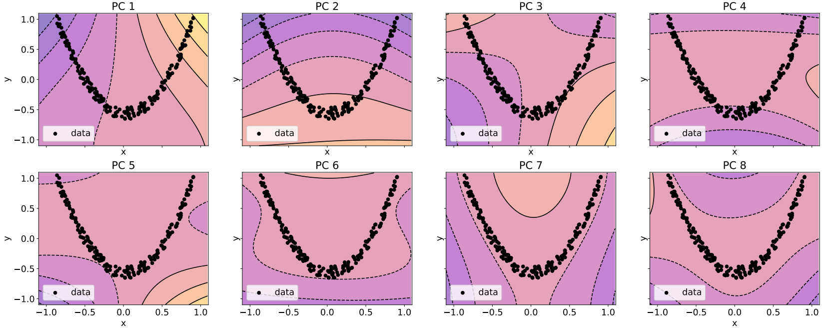


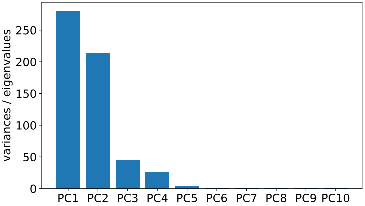
Parabola example: kPCA with polynomial kernel of degree 2



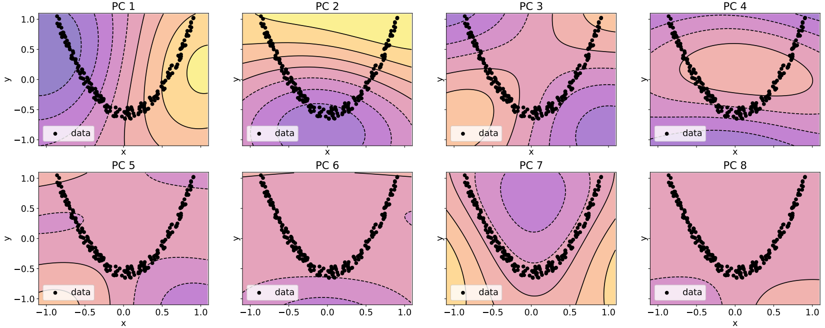


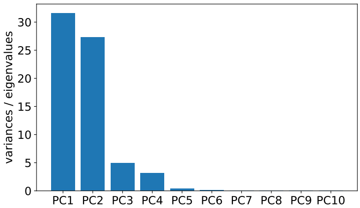
Parabola example: kPCA with polynomial kernel of degree 3



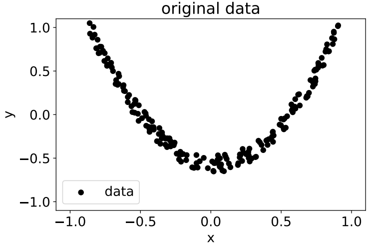


Parabola example: kPCA with RBF-kernel ($\sigma = 1.0$)





Parabola example: dimension reduction



Reconstruction errors for kPCA:

Euclidean distance between original data and pre-image after reduction.