Reproducing kernel Hilbert spaces in Machine Learning

Arthur Gretton

Gatsby Computational Neuroscience Unit,
University College London

Advanced topics in Machine Learning

Difference in feature means

Distance between means (1)

Sample $(x_i)_{i=1}^m$ from p and $(y_i)_{i=1}^m$ from q. What is the distance between their means in feature space?

$$\left\| \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j) \right\|_{\mathcal{H}}^2$$

$$= \left\langle \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j), \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j) \right\rangle_{\mathcal{H}}$$

$$= \frac{1}{m^2}\left\langle \sum_{i=1}^m \phi(x_i), \sum_{i=1}^m \phi(x_i) \right\rangle + \ldots$$

$$= \frac{1}{m^2}\sum_{i=1}^m\sum_{j=1}^m k(x_i, x_j) + \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n k(y_i, y_j) - \frac{2}{mn}\sum_{i=1}^m\sum_{j=1}^m k(x_i, y_j).$$

Distance between means (1)

Sample $(x_i)_{i=1}^m$ from p and $(y_i)_{i=1}^m$ from q. What is the distance between their means in feature space?

$$\left\| \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j) \right\|_{\mathcal{H}}^2$$

$$= \left\langle \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j), \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j) \right\rangle_{\mathcal{H}}$$

$$= \frac{1}{m^2}\left\langle \sum_{i=1}^m \phi(x_i), \sum_{i=1}^m \phi(x_i) \right\rangle + \dots$$

$$= \frac{1}{m^2}\sum_{i=1}^m\sum_{j=1}^m k(x_i,x_j) + \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n k(y_i,y_j) - \frac{2}{mn}\sum_{i=1}^m\sum_{j=1}^m k(x_i,y_j).$$

Distance between means (1)

Sample $(x_i)_{i=1}^m$ from p and $(y_i)_{i=1}^m$ from q. What is the distance between their means in feature space?

$$
\begin{aligned}
&\left\| \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j) \right\|_{\mathcal{H}}^2 \\
&= \left\langle \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j), \frac{1}{m}\sum_{i=1}^m \phi(x_i) - \frac{1}{n}\sum_{j=1}^n \phi(y_j) \right\rangle_{\mathcal{H}} \\
&= \frac{1}{m^2}\left\langle \sum_{i=1}^m \phi(x_i), \sum_{i=1}^m \phi(x_i) \right\rangle + \ldots \\
&= \frac{1}{m^2}\sum_{i=1}^m\sum_{j=1}^m k(x_i, x_j) + \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n k(y_i, y_j) - \frac{2}{mn}\sum_{i=1}^m\sum_{j=1}^m k(x_i, y_j).
\end{aligned}
$$

Distance between means (1)

Sample $(x_i)_{i=1}^m$ from p and $(y_i)_{i=1}^m$ from q. What is the distance between their means in feature space?

$$
\begin{aligned}
&\left\| \frac{1}{m} \sum_{i=1}^m \phi(x_i) - \frac{1}{n} \sum_{j=1}^n \phi(y_j) \right\|_{\mathcal{H}}^2 \\
&= \left\langle \frac{1}{m} \sum_{i=1}^m \phi(x_i) - \frac{1}{n} \sum_{j=1}^n \phi(y_j), \frac{1}{m} \sum_{i=1}^m \phi(x_i) - \frac{1}{n} \sum_{j=1}^n \phi(y_j) \right\rangle_{\mathcal{H}} \\
&= \frac{1}{m^2} \left\langle \sum_{i=1}^m \phi(x_i), \sum_{i=1}^m \phi(x_i) \right\rangle + \ldots \\
&= \frac{1}{m^2} \sum_{i=1}^m \sum_{j=1}^m k(x_i, x_j) + \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n k(y_i, y_j) - \frac{2}{mn} \sum_{i=1}^m \sum_{j=1}^m k(x_i, y_j).
\end{aligned}
$$

Distance between means (2)

Sample $(x_i)_{i=1}^m$ from p and $(y_i)_{i=1}^m$ from q. What is the distance between their means in feature space?

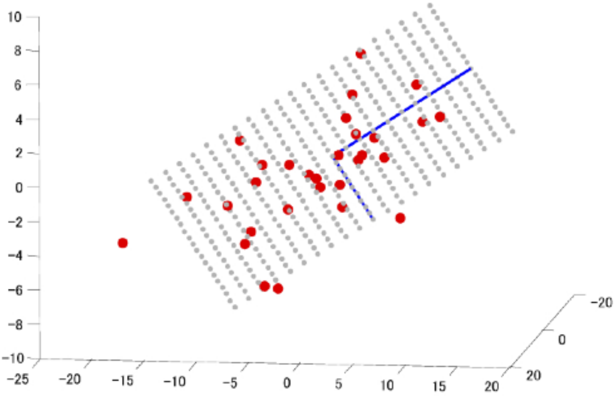
$$\left\| \frac{1}{m} \sum_{i=1}^{m} \phi(x_i) - \frac{1}{n} \sum_{j=1}^{n} \phi(y_j) \right\|_{\mathcal{H}}^2$$

- When $\phi(x) = x$, distinguish means. When $\phi(x) = [x \; x^2]$, distinguish means and variances.
- There are kernels that can distinguish any two distributions

Kernel Principal Component Analysis

PCA (1)

Goal of classical PCA: to find a d-dimensional subspace of a higher dimensional space (D-dimensional, $\mathbb{R}^D$) containing the directions of maximum variance.



(Figure by K. Fukumizu)

Applicationof kPCA: image denoising

What is the purpose of kernel PCA?

We consider the problem of denoising hand-written digits.

We are given a noisy digit x^*.

$$P_d\phi(x^*) = P_{f_1}\phi(x^*) + \ldots + P_{f_d}\phi(x^*)$$

is the projection of $\phi(x^*)$ onto one of the first d eigenvectors $\{f_\ell\}_{\ell=1}^d$ from kernel PCA (these are orthogonal).

Define the nearest point $y^* \in \mathcal{X}$ to this feature space projection as

$$y^* = \arg\min_{y \in \mathcal{X}} \|\phi(y) - P_d\phi(x^*)\|_{\mathcal{H}}^2 .$$

In many cases, not possible to reduce the squared error to zero, as no single y^* corresponds to exact solution.

Applicationof kPCA: image denoising

What is the purpose of kernel PCA?

We consider the problem of denoising hand-written digits.

We are given a noisy digit x^*.

$$P_d\phi(x^*) = \quad P_{f_1}\phi(x^*) \quad + \ldots + \quad P_{f_d}\phi(x^*)$$

is the projection of $\phi(x^*)$ onto one of the first d eigenvectors $\{f_\ell\}_{\ell=1}^d$ from kernel PCA (these are orthogonal).

Define the nearest point $y^* \in \mathcal{X}$ to this feature space projection as

$$y^* = \arg\min_{y \in \mathcal{X}} \|\phi(y) - P_d\phi(x^*)\|_{\mathcal{H}}^2 .$$

In many cases, not possible to reduce the squared error to zero, as no single y^* corresponds to exact solution.

Applicationof kPCA: image denoising

What is the purpose of kernel PCA?

We consider the problem of denoising hand-written digits.

We are given a noisy digit x^*.

$$P_d\phi(x^*) = \quad P_{f_1}\phi(x^*) \quad + \ldots + \quad P_{f_d}\phi(x^*)$$

is the projection of $\phi(x^*)$ onto one of the first d eigenvectors $\{f_\ell\}_{\ell=1}^d$ from kernel PCA (these are orthogonal).

Define the nearest point $y^* \in \mathcal{X}$ to this feature space projection as

$$y^* = \arg\min_{y \in \mathcal{X}} \|\phi(y) - P_d\phi(x^*)\|_{\mathcal{H}}^2 .$$

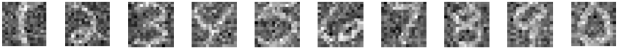
In many cases, not possible to reduce the squared error to zero, as no single y^* corresponds to exact solution.

Applicationof kPCA: image denoising

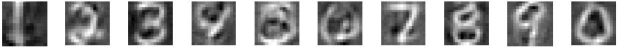
Projection onto PCA subspace for denoising. kPCA: data may not be Gaussian distributed, but can lie in a submanifold in input space.

USPS hand-written digits data:
7191 images of hand-written digits of 16×16 pixels.

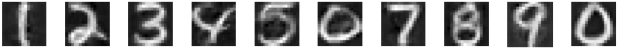
Sample of original images (not used for experiments)



Sample of noisy images



Sample of denoised images (linear PCA)



Sample of denoised images (kernel PCA, Gaussian kernel)

Generated by Matlab Stprtool (by V. Franc). (Figure: K. Fukumizu)

What is PCA? (reminder)

First principal component (max. variance)

$$
\begin{aligned}
u_1 &= \arg\max_{\|u\|\leq 1} \frac{1}{n}\sum_{i=1}^{n}\left(u^\top\left(x_i - \frac{1}{n}\sum_{j=1}^{n} x_j\right)\right)^2 \\
&= \arg\max_{\|u\|\leq 1} u^\top C u
\end{aligned}
$$

where

$$
C = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \frac{1}{n}\sum_{j=1}^{n} x_j\right)\left(x_i - \frac{1}{n}\sum_{j=1}^{n} x_j\right)^\top = \frac{1}{n} X H X^\top,
$$

$X = \begin{bmatrix} x_1 & \ldots & x_n \end{bmatrix}$, $H = I - n^{-1} 1_{n\times n}$, $1_{n\times n}$ a matrix of ones.

Definition (Principal components)

The pairs (λ_i, u_i) are the eigensystem of $\lambda_i u_i = C u_i$.

PCA in feature space

Kernel version, first principal component:

$$f_1 = \arg\max_{\|f\|_{\mathcal{H}} \le 1} \frac{1}{n}\sum_{i=1}^{n}\left(\left\langle f, \phi(x_i) - \frac{1}{n}\sum_{j=1}^{n}\phi(x_j)\right\rangle_{\mathcal{H}}\right)^2$$

$$= \arg\max_{\|f\|_{\mathcal{H}} \le 1} \frac{1}{n}\sum_{i=1}^{n}\left(f(x_i) - \widehat{E}(f)\right)^2$$

$$= \arg\max_{\|f\|_{\mathcal{H}} \le 1} \widehat{\mathrm{var}}(f).$$

We can write

$$f = \sum_{i=1}^{n}\alpha_i\left(\phi(x_i) - \frac{1}{n}\sum_{j=1}^{n}\phi(x_j)\right) = \sum_{i=1}^{n}\alpha_i\tilde{\phi}(x_i),$$

since any component orthogonal to the span of $\tilde{\phi}(x_i) := \phi(x_i) - \frac{1}{n}\sum_{i=1}^{n}\phi(x_i)$ vanishes.

PCA in feature space

Kernel version, first principal component:

$$f_1 = \arg\max_{\|f\|_{\mathcal{H}} \le 1} \frac{1}{n}\sum_{i=1}^{n}\left(\left\langle f, \phi(x_i) - \frac{1}{n}\sum_{j=1}^{n}\phi(x_j)\right\rangle_{\mathcal{H}}\right)^2$$

$$= \arg\max_{\|f\|_{\mathcal{H}} \le 1} \frac{1}{n}\sum_{i=1}^{n}\left(f(x_i) - \widehat{E}(f)\right)^2$$

$$= \arg\max_{\|f\|_{\mathcal{H}} \le 1} \widehat{\operatorname{var}}(f).$$

We can write

$$f = \sum_{i=1}^{n}\alpha_i\left(\phi(x_i) - \frac{1}{n}\sum_{j=1}^{n}\phi(x_j)\right) = \sum_{i=1}^{n}\alpha_i\tilde{\phi}(x_i),$$

since any component orthogonal to the span of $\tilde{\phi}(x_i) := \phi(x_i) - \frac{1}{n}\sum_{i=1}^{n}\phi(x_i)$ vanishes.

PCA in feature space

Kernel version, first principal component:

$$
\begin{aligned}
f_1 &= \arg\max_{\|f\|_{\mathcal{H}} \leq 1} \frac{1}{n} \sum_{i=1}^{n} \left(\left\langle f, \phi(x_i) - \frac{1}{n} \sum_{j=1}^{n} \phi(x_j) \right\rangle_{\mathcal{H}} \right)^2 \\
&= \arg\max_{\|f\|_{\mathcal{H}} \leq 1} \frac{1}{n} \sum_{i=1}^{n} \left(f(x_i) - \widehat{E}(f) \right)^2 \\
&= \arg\max_{\|f\|_{\mathcal{H}} \leq 1} \widehat{\operatorname{var}}(f).
\end{aligned}
$$

We can write

$$
f = \sum_{i=1}^{n} \alpha_i \left(\phi(x_i) - \frac{1}{n} \sum_{j=1}^{n} \phi(x_j) \right) = \sum_{i=1}^{n} \alpha_i \tilde{\phi}(x_i),
$$

since any component orthogonal to the span of $\tilde{\phi}(x_i) := \phi(x_i) - \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)$ vanishes.

PCA in feature space

Kernel version, first principal component:

$$
\begin{aligned}
f_1 &= \arg\max_{\|f\|_{\mathcal{H}} \leq 1} \frac{1}{n} \sum_{i=1}^{n} \left(\left\langle f, \phi(x_i) - \frac{1}{n} \sum_{j=1}^{n} \phi(x_j) \right\rangle_{\mathcal{H}} \right)^2 \\
&= \arg\max_{\|f\|_{\mathcal{H}} \leq 1} \frac{1}{n} \sum_{i=1}^{n} \left(f(x_i) - \widehat{E}(f) \right)^2 \\
&= \arg\max_{\|f\|_{\mathcal{H}} \leq 1} \widehat{\mathrm{var}}(f).
\end{aligned}
$$

We can write

$$
f \quad = \quad \sum_{i=1}^{n} \alpha_i \left(\phi(x_i) - \frac{1}{n} \sum_{j=1}^{n} \phi(x_j) \right) = \sum_{i=1}^{n} \alpha_i \tilde{\phi}(x_i),
$$

since any component orthogonal to the span of $\tilde{\phi}(x_i) := \phi(x_i) - \frac{1}{n} \sum_{i=1}^{n} \phi(x_i)$ vanishes.

How to solve kernel PCA

We can also define an infinite dimensional analog of the covariance:

$$
\begin{aligned}
C &= \frac{1}{n}\sum_{i=1}^{n}\left(\phi(x_i) - \frac{1}{n}\sum_{j=1}^{n}\phi(x_j)\right) \otimes \left(\phi(x_i) - \frac{1}{n}\sum_{j=1}^{n}\phi(x_j)\right), \\
&= \frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i) \otimes \tilde{\phi}(x_i)
\end{aligned}
$$

where we use the definition

$$
(a \otimes b)c := \langle b, c\rangle_{\mathcal{H}}\, a \tag{1}
$$

this is analogous to the case of finite dimensional vectors, $(ab^\top)c = (b^\top c)a$.

How to solve kernel PCA (1)

Eigenfunctions of kernel covariance:

$$
\begin{aligned}
f\lambda &= Cf \\
&= \underbrace{\left(\frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\otimes\tilde{\phi}(x_i)\right)}_{C} f \\
&= \frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\left\langle\tilde{\phi}(x_i), \underbrace{\sum_{j=1}^{n}\alpha_j\tilde{\phi}(x_j)}_{f}\right\rangle_{\mathcal{H}} \\
&= \frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\left(\sum_{j=1}^{n}\alpha_j\tilde{k}(x_i,x_j)\right)
\end{aligned}
$$

$\tilde{k}(x_i, x_j)$ is the (i,j)th entry of the matrix $\tilde{K} := HKH$ (exercise!).

How to solve kernel PCA (1)

Eigenfunctions of kernel covariance:

$$
\begin{aligned}
f\lambda &= Cf \\
&= \underbrace{\left(\frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\otimes\tilde{\phi}(x_i)\right)}_{C} f \\
&= \frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\left\langle \tilde{\phi}(x_i), \underbrace{\sum_{j=1}^{n}\alpha_j\tilde{\phi}(x_j)}_{f}\right\rangle_{\mathcal{H}} \\
&= \frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\left(\sum_{j=1}^{n}\alpha_j\tilde{k}(x_i,x_j)\right)
\end{aligned}
$$

$\tilde{k}(x_i,x_j)$ is the (i,j)th entry of the matrix $\tilde{K} := HKH$ (exercise!).

How to solve kernel PCA (1)

Eigenfunctions of kernel covariance:

$$
\begin{aligned}
f\lambda &= Cf \\
&= \underbrace{\left(\frac{1}{n}\sum_{i=1}^{n} \tilde{\phi}(x_i) \otimes \tilde{\phi}(x_i)\right)}_{C} f \\
&= \frac{1}{n}\sum_{i=1}^{n} \tilde{\phi}(x_i) \Big\langle \tilde{\phi}(x_i), \underbrace{\sum_{j=1}^{n} \alpha_j \tilde{\phi}(x_j)}_{f} \Big\rangle_{\mathcal{H}} \\
&= \frac{1}{n}\sum_{i=1}^{n} \tilde{\phi}(x_i) \left(\sum_{j=1}^{n} \alpha_j \tilde{k}(x_i, x_j)\right)
\end{aligned}
$$

$\tilde{k}(x_i, x_j)$ is the (i, j)th entry of the matrix $\tilde{K} := HKH$ (exercise!).

How to solve kernel PCA (1)

Eigenfunctions of kernel covariance:

$$
\begin{aligned}
f\lambda &= Cf \\
&= \underbrace{\left(\frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\otimes\tilde{\phi}(x_i)\right)}_{C} f \\
&= \frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\left\langle\tilde{\phi}(x_i), \underbrace{\sum_{j=1}^{n}\alpha_j\tilde{\phi}(x_j)}_{f}\right\rangle_{\mathcal{H}} \\
&= \frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(x_i)\left(\sum_{j=1}^{n}\alpha_j\tilde{k}(x_i,x_j)\right)
\end{aligned}
$$

$\tilde{k}(x_i,x_j)$ is the (i,j)th entry of the matrix $\tilde{K} := HKH$ (exercise!).

How to solve kernel PCA (2)

We can now project both sides of

$$f_\ell \lambda_\ell = C f_\ell$$

onto all of the $\tilde{\phi}(x_q)$:

$$\left\langle \tilde{\phi}(x_q), \mathrm{LHS} \right\rangle_{\mathcal{H}} = \lambda_\ell \left\langle \tilde{\phi}(x_q), f_\ell \right\rangle_{\mathcal{H}} = \lambda_\ell \sum_{i=1}^{n} \alpha_{\ell i} \tilde{k}(x_q, x_i) \qquad \forall q \in \{1 \dots n\}$$

$$\left\langle \tilde{\phi}(x_q), \mathrm{RHS} \right\rangle_{\mathcal{H}} = \left\langle \tilde{\phi}(x_q), C f_\ell \right\rangle_{\mathcal{H}} = \frac{1}{n} \sum_{i=1}^{n} \tilde{k}(x_q, x_i) \left(\sum_{j=1}^{n} \alpha_{\ell j} \tilde{k}(x_i, x_j) \right)$$

Writing this as a matrix equation,

$$n \lambda_\ell \widetilde{K} \alpha_\ell = \widetilde{K}^2 \alpha_\ell \qquad n \lambda_\ell \alpha_\ell = \widetilde{K} \alpha_\ell.$$

How to solve kernel PCA (2)

We can now project both sides of

$$f_\ell \lambda_\ell = C f_\ell$$

onto all of the $\tilde{\phi}(x_q)$:

$$\left\langle \tilde{\phi}(x_q), \mathrm{LHS} \right\rangle_{\mathcal{H}} = \lambda_\ell \left\langle \tilde{\phi}(x_q), f_\ell \right\rangle_{\mathcal{H}} = \lambda_\ell \sum_{i=1}^{n} \alpha_{\ell i} \tilde{k}(x_q, x_i) \qquad \forall q \in \{1 \ldots n\}$$

$$\left\langle \tilde{\phi}(x_q), \mathrm{RHS} \right\rangle_{\mathcal{H}} = \left\langle \tilde{\phi}(x_q), C f_\ell \right\rangle_{\mathcal{H}} = \frac{1}{n} \sum_{i=1}^{n} \tilde{k}(x_q, x_i) \left(\sum_{j=1}^{n} \alpha_{\ell j} \tilde{k}(x_i, x_j) \right)$$

Writing this as a matrix equation,

$$n \lambda_\ell \widetilde{K} \alpha_\ell = \widetilde{K}^2 \alpha_\ell \qquad n \lambda_\ell \alpha_\ell = \widetilde{K} \alpha_\ell.$$

Eigenfunctions f have unit norm in feature space?

$$
\begin{aligned}
&\|f\|_{\mathcal{H}}^2 \\
&= \left\langle \sum_{i=1}^n \alpha_i \tilde{\phi}(x_i), \sum_{i=1}^n \alpha_i \tilde{\phi}(x_i) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_i \left\langle \tilde{\phi}(x_i), \tilde{\phi}(x_j) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_i \tilde{k}(x_i, x_j) \\
&= \alpha^\top \widetilde{K} \alpha = n\lambda \alpha^\top \alpha = n\lambda \|\alpha\|^2.
\end{aligned}
$$

Thus $\alpha \leftarrow \alpha/\sqrt{n\lambda}$ (assumed: original eigenvector solution has $\|\alpha\| = 1$)

Eigenfunctions f have unit norm in feature space?

$$
\begin{aligned}
&\|f\|^2_{\mathcal{H}} \\
&= \left\langle \sum_{i=1}^n \alpha_i \tilde{\phi}(x_i), \sum_{i=1}^n \alpha_i \tilde{\phi}(x_i) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_i \left\langle \tilde{\phi}(x_i), \tilde{\phi}(x_j) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_i \tilde{k}(x_i, x_j) \\
&= \alpha^\top \widetilde{K} \alpha = n\lambda \alpha^\top \alpha = n\lambda \|\alpha\|^2.
\end{aligned}
$$

Thus $\alpha \leftarrow \alpha / \sqrt{n\lambda}$ (assumed: original eigenvector solution has $\|\alpha\| = 1$)

Eigenfunctions f have unit norm in feature space?

$$
\begin{aligned}
&\|f\|_{\mathcal{H}}^2 \\
&= \left\langle \sum_{i=1}^n \alpha_i \tilde{\phi}(x_i), \sum_{i=1}^n \alpha_i \tilde{\phi}(x_i) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_i \left\langle \tilde{\phi}(x_i), \tilde{\phi}(x_j) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_i \tilde{k}(x_i, x_j) \\
&= \alpha^\top \widetilde{K} \alpha = n\lambda \alpha^\top \alpha = n\lambda \|\alpha\|^2.
\end{aligned}
$$

Thus $\alpha \leftarrow \alpha / \sqrt{n\lambda}$ (assumed: original eigenvector solution has $\|\alpha\| = 1$)

Eigenfunctions f have unit norm in feature space?

$$
\begin{aligned}
&\|f\|_{\mathcal{H}}^2 \\
&= \left\langle \sum_{i=1}^{n} \alpha_i \tilde{\phi}(x_i), \sum_{i=1}^{n} \alpha_i \tilde{\phi}(x_i) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_i \left\langle \tilde{\phi}(x_i), \tilde{\phi}(x_j) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_i \tilde{k}(x_i, x_j) \\
&= \alpha^\top \widetilde{K} \alpha = n\lambda \alpha^\top \alpha = n\lambda \|\alpha\|^2.
\end{aligned}
$$

Thus $\alpha \leftarrow \alpha / \sqrt{n\lambda}$ (assumed: original eigenvector solution has $\|\alpha\| = 1$)

Eigenfunctions f have unit norm in feature space?

$$
\begin{aligned}
&\|f\|^2_{\mathcal{H}} \\
&= \left\langle \sum_{i=1}^{n} \alpha_i \tilde{\phi}(x_i), \sum_{i=1}^{n} \alpha_i \tilde{\phi}(x_i) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_i \left\langle \tilde{\phi}(x_i), \tilde{\phi}(x_j) \right\rangle_{\mathcal{H}} \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_i \tilde{k}(x_i, x_j) \\
&= \alpha^\top \widetilde{K} \alpha = n\lambda \alpha^\top \alpha = n\lambda \|\alpha\|^2.
\end{aligned}
$$

Thus $\alpha \leftarrow \alpha / \sqrt{n\lambda}$ (assumed: original eigenvector solution has $\|\alpha\| = 1$)

Projection onto kernel PC

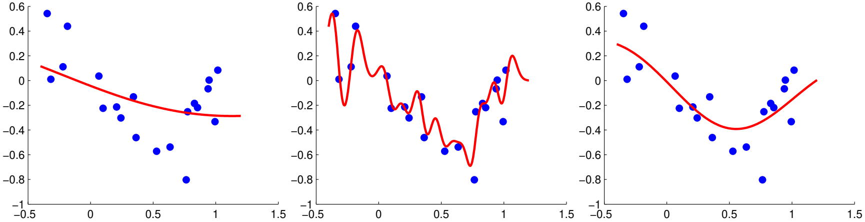
How do you project a new point x^* onto the principal component f?

Assuming $\|f\|_{\mathcal{H}} = 1$, the projection is

$$
\begin{aligned}
P_f \phi(x^*) &= \langle \phi(x^*), f \rangle_{\mathcal{H}} f \\
&= \underbrace{\left(\sum_{j=1}^{n} \alpha_j \left\langle \phi(x^*), \tilde{\phi}(x_j) \right\rangle_{\mathcal{H}} \right)}_{\langle \phi(x^*), f \rangle_{\mathcal{H}}} \underbrace{\sum_{i=1}^{n} \alpha_i \tilde{\phi}(x_i)}_{f} \\
&= \left(\sum_{j=1}^{n} \alpha_j \left(k(x^*, x_j) - \frac{1}{n} \sum_{\ell=1}^{n} k(x^*, x_\ell) \right) \right) \sum_{i=1}^{n} \alpha_i \tilde{\phi}(x_i).
\end{aligned}
$$

Kernel Ridge Regression

Kernel ridge regression



Very simple to implement, works well when no outliers.

Ridge regression: case of $\mathbb{R}^D$

We are given n training points in $\mathbb{R}^D$:

$$X = \begin{bmatrix} x_1 & \dots & x_n \end{bmatrix} \in \mathbb{R}^{D \times n} \quad y := \begin{bmatrix} y_1 & \dots & y_n \end{bmatrix}^\top$$

Define some $\lambda > 0$. Our goal is:

$$\begin{aligned} a^* &= \arg\min_{a \in \mathbb{R}^D} \left(\sum_{i=1}^{n} (y_i - x_i^\top a)^2 + \lambda \|a\|^2 \right) \\ &= \arg\min_{a \in \mathbb{R}^D} \left(\left\| y - X^\top a \right\|^2 + \lambda \|a\|^2 \right), \end{aligned}$$

The second term $\lambda\|a\|^2$ is chosen to avoid problems in high dimensional spaces (see below).

Ridge regression: solution (1)

Expanding out the above term, we get

$$\begin{aligned}\left\|y - X^\top a\right\|^2 + \lambda\|a\|^2 &= y^\top y - 2y^\top X^\top a + a^\top X X^\top a + \lambda a^\top a \\ &= y^\top y - 2y^\top X^\top a + a^\top \left(XX^\top + \lambda I\right) a = (*)\end{aligned}$$

- Define $b = \left(XX^\top + \lambda I\right)^{1/2} a$
- Square root defined since matrix positive definite
- $XX^\top$ may not be invertible eg when $D > n$, adding λI means we can write $a = \left(XX^\top + \lambda I\right)^{-1/2} b$).

Ridge regression: solution (2)

Complete the square:

$$
\begin{aligned}
(*) &= y^\top y - 2y^\top X^\top \left(XX^\top + \lambda I\right)^{-1/2} b + b^\top b \\
&= y^\top y + \left\| \left(XX^\top + \lambda I\right)^{-1/2} Xy - b \right\|^2 - \left\| y^\top X^\top \left(XX^\top + \lambda I\right)^{-1/2} \right\|^2
\end{aligned}
$$

This is minimized when

$$
\begin{aligned}
b^* &= \left(XX^\top + \lambda I\right)^{-1/2} Xy \quad \text{or} \\
a^* &= \left(XX^\top + \lambda I\right)^{-1} Xy,
\end{aligned}
$$

which is the classic regularized least squares solution.

Ridge regression solution as sum of training points (1)

We may rewrite this expression in a way that is more informative, $a^* = \sum_{i=1}^{n} \alpha_i^* x_i$.

The solution is a linear combination of training points x_i.

Proof: Assume $D > n$ (in feature space case D can be very large or even infinite).

Perform an SVD on X, i.e.

$$X = USV^\top,$$

where

$$U = \begin{bmatrix} u_1 & \dots & u_D \end{bmatrix} \quad S = \begin{bmatrix} \tilde{S} & 0 \\ 0 & 0 \end{bmatrix} \quad V = \begin{bmatrix} \tilde{V} & 0 \end{bmatrix}.$$

Here U is $D \times D$ and $U^\top U = UU^\top = I_D$ (subscript denotes unit matrix size), S is $D \times D$, where $\tilde{S}$ has n non-zero entries, and V is $n \times D$, where $\tilde{V}^\top \tilde{V} = \tilde{V} \tilde{V}^\top = I_n$.

Ridge regression solution as sum of training points (1)

We may rewrite this expression in a way that is more informative, $a^* = \sum_{i=1}^{n} \alpha_i^* x_i$.

The solution is a linear combination of training points x_i.

Proof: Assume $D > n$ (in feature space case D can be very large or even infinite).

Perform an SVD on X, i.e.

$$X = USV^\top,$$

where

$$U = \begin{bmatrix} u_1 & \dots & u_D \end{bmatrix} \quad S = \begin{bmatrix} \tilde{S} & 0 \\ 0 & 0 \end{bmatrix} \quad V = \begin{bmatrix} \tilde{V} & 0 \end{bmatrix}.$$

Here U is $D \times D$ and $U^\top U = UU^\top = I_D$ (subscript denotes unit matrix size), S is $D \times D$, where $\tilde{S}$ has n non-zero entries, and V is $n \times D$, where $\tilde{V}^\top \tilde{V} = \tilde{V}\tilde{V}^\top = I_n$.

Ridge regression solution as sum of training points (2)

Proof (continued):

$$
\begin{aligned}
a^* &= \left(XX^\top + \lambda I_D\right)^{-1} Xy \\
&= \left(US^2U^\top + \lambda I_D\right)^{-1} USV^\top y \\
&= U\left(S^2 + \lambda I_D\right)^{-1} U^\top USV^\top y \\
&= U\left(S^2 + \lambda I_D\right)^{-1} SV^\top y \\
&= US\left(S^2 + \lambda I_D\right)^{-1} V^\top y \\
&= U\underbrace{SV^\top V}_{\text{(a)}}\left(S^2 + \lambda I_D\right)^{-1} V^\top y \\
&\underset{\text{(b)}}{=} X(X^\top X + \lambda I_n)^{-1} y \qquad (2)
\end{aligned}
$$

Ridge regression solution as sum of training points (3)

Proof (continued):

(a): both S and $V^\top V$ are non-zero in same sized top-left block, and $V^\top V$ is I_n in that block.

(b): since

$$
\begin{aligned}
& V\left(S^2+\lambda I_D\right)^{-1} V^\top \\
&= \begin{bmatrix} \tilde{V} & 0 \end{bmatrix} \begin{bmatrix} \left(\tilde{S}^2+\lambda I_n\right)^{-1} & 0 \\ 0 & (\lambda I_{D-n})^{-1} \end{bmatrix} \begin{bmatrix} \tilde{V}^\top \\ 0 \end{bmatrix} \\
&= \tilde{V}\left(\tilde{S}^2+\lambda I_n\right)^{-1} \tilde{V}^\top \\
&= \left(X^\top X+\lambda I_n\right)^{-1}.
\end{aligned}
$$

Kernel ridge regression

Use features of $\phi(x_i)$ in the place of x_i:

$$a^* = \arg\min_{a \in \mathcal{H}} \left(\sum_{i=1}^{n} \left(y_i - \langle a, \phi(x_i) \rangle_{\mathcal{H}}\right)^2 + \lambda \| a \|_{\mathcal{H}}^2 \right).$$

E.g. for finite dimensional feature spaces,

$$\phi_p(x) = \begin{bmatrix} x \\ x^2 \\ \vdots \\ x^\ell \end{bmatrix} \qquad \phi_s(x) = \begin{bmatrix} \sin x \\ \cos x \\ \sin 2x \\ \vdots \\ \cos \ell x \end{bmatrix}$$

a is a vector of length ℓ giving weight to each of these features so as to find the mapping between x and y. Feature vectors can also have infinite length (more soon).

Kernel ridge regression: proof

Use previous proof!

$$X = \left[\begin{array}{ccc} \phi(x_1) & \ldots & \phi(x_n) \end{array} \right].$$

All of the steps that led us to $a^* = X(X^\top X + \lambda I_n)^{-1} y$ follow.

$$XX^\top = \sum_{i=1}^{n} \phi(x_i) \otimes \phi(x_i)$$

(using tensor notation from kernel PCA), and

$$(X^\top X)_{ij} = \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}} = k(x_i, x_j).$$

Making these replacements, we get

$$\begin{aligned} a^* &= X(K + \lambda I_n)^{-1} y \\ &= \sum_{i=1}^{n} \alpha_i^* \phi(x_i) \qquad \alpha^* = (K + \lambda I_n)^{-1} y. \end{aligned}$$

Kernel ridge regression: easier proof

We begin knowing a is a linear combination of feature space mappings of points (representer theorem: later in course)

$$a = \sum_{i=1}^{n} \alpha_i \phi(x_i).$$

Then

$$\begin{aligned}\sum_{i=1}^{n}\left(y_i - \langle a, \phi(x_i)\rangle_{\mathcal{H}}\right)^2 + \lambda \|a\|_{\mathcal{H}}^2 &= \|y - K\alpha\|^2 + \lambda \alpha^\top K \alpha \\ &= y^\top y - 2y^\top K\alpha + \alpha^\top \left(K^2 + \lambda K\right)\alpha\end{aligned}$$

Differentiating wrt α and setting this to zero, we get

$$\alpha^* = (K + \lambda I_n)^{-1} y.$$

Recall: $\frac{\partial \alpha^\top U \alpha}{\partial \alpha} = (U + U^\top)\alpha, \qquad \frac{\partial v^\top \alpha}{\partial \alpha} = \frac{\partial \alpha^\top v}{\partial \alpha} = v$

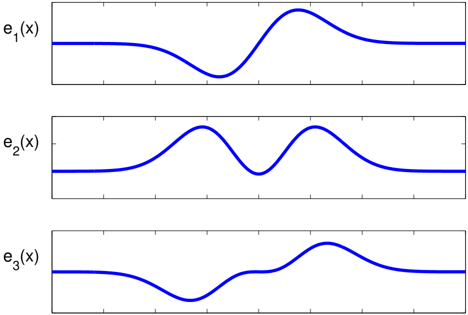
Reminder: smoothness

What does $\|a\|_{\mathcal{H}}$ have to do with smoothing?

Example 1: The exponentiated quadratic kernel. Recall

$$f(x) = \sum_{i=1}^{\infty} \hat{f}_\ell e_\ell(x), \qquad \langle e_i, e_j \rangle_{L_2(p)} = \int_{\mathcal{X}} e_i(x) e_j(x) p(x) dx = \begin{cases} 1 & i = j \\ 0 & i \neq j. \end{cases}$$

$$\|f\|_{\mathcal{H}}^2 = \sum_{\ell=1}^{\infty} \frac{\hat{f}_\ell^2}{\lambda_\ell}.$$



Reminder: smoothness

What does $\|a\|_{\mathcal{H}}$ have to do with smoothing?

Example 2: The Fourier series representation:

$$f(x) = \sum_{l=-\infty}^{\infty} \hat{f}_l \exp(\imath l x),$$

and

$$\langle f, g \rangle_{\mathcal{H}} = \sum_{l=-\infty}^{\infty} \frac{\hat{f}_l \overline{\hat{g}_l}}{\hat{k}_l}.$$

Thus,

$$\|f\|_{\mathcal{H}}^2 = \langle f, f \rangle_{\mathcal{H}} = \sum_{l=-\infty}^{\infty} \frac{\left|\hat{f}_l\right|^2}{\hat{k}_l}.$$

Parameter selection for KRR

Given the objective

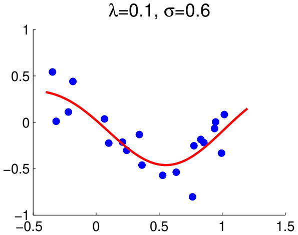
$$a^* \quad = \quad \arg\min_{a \in \mathcal{H}} \left(\sum_{i=1}^{n} \left(y_i - \langle a, \phi(x_i) \rangle_{\mathcal{H}} \right)^2 + \lambda \| a \|_{\mathcal{H}}^2 \right).$$

How do we choose

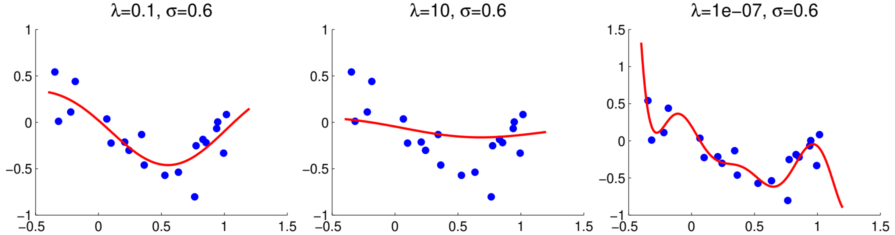
- The regularization parameter λ?
- The kernel parameter: for exponentiated quadratic kernel, σ in

$$k(x, y) = \exp\left(\frac{-\|x - y\|^2}{\sigma} \right).$$

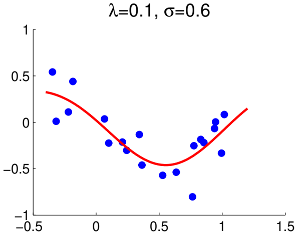
Choice of λ



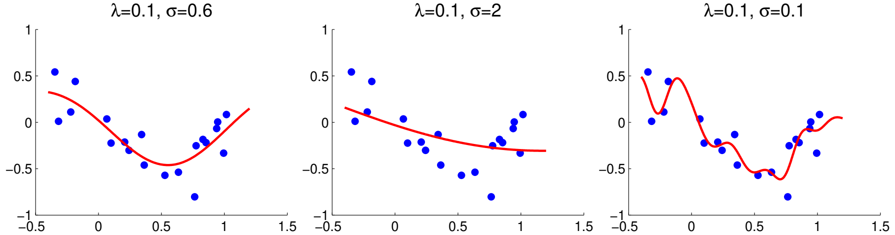
Choice of λ



Choice of σ



Choice of σ



Cross validation

- Split n data into training set size n_{tr} and test set size $n_{\text{te}} = n - n_{\text{tr}}$.
- Split training set into m equal chunks of size $n_{\text{val}} = n_{\text{tr}}/m$. Call these $X_{\text{val},i}$, $Y_{\text{val},i}$ for $i \in \{1, \ldots, m\}$
- For each λ, σ pair
  - For each $X_{\text{val},i}$, $Y_{\text{val},i}$
    - Train ridge regression on remaining trainining set data $X_{\text{tr}} \setminus X_{\text{val},i}$ and $Y_{\text{tr}} \setminus Y_{\text{val},i}$,
    - Evaluate its error on the validation data $X_{\text{val},i}$, $Y_{\text{val},i}$
  - Average the errors on the validation sets to get the average validation error for λ, σ.
- Choose λ^*, σ^* with the lowest average validation error
- Measure the performance on the test set X_{te}, Y_{te}.