Introduction to Machine Learning: Kernels

Part 1: Kernels and feature space, ridge regression

Arthur Gretton

Gatsby Unit, CSML, UCL

May 26, 2016

Course overview

Part 1:

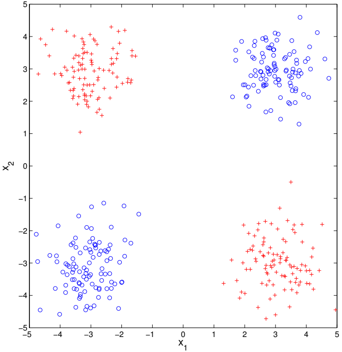
- What is a feature map, what is a kernel, and how do they relate?
- Applications: difference in means, kernel ridge regression (extra: kernel PCA)

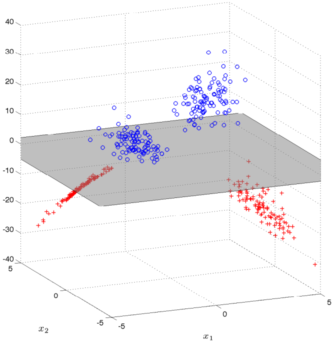
Part 2:

- Basics of convex optimization
- The support vector machine

Lecture notes will be put online at:
http://www.gatsby.ucl.ac.uk/~gretton/rkhsAdaptModel.html

Why kernel methods (1): XOR example



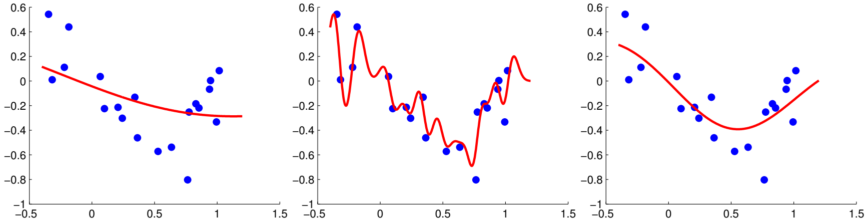


- No linear classifier separates red from blue
- Map points to **higher dimensional feature space**: $\phi(x) = \begin{bmatrix} x_1 & x_2 & x_1 x_2 \end{bmatrix} \in \mathbb{R}^3$

Why kernel methods (2): document classification

Kernels let us compare **objects** on the basis of **features**

Why kernel methods (3): smoothing



Kernel methods can control **smoothness** and **avoid overfitting/underfitting**.

Basics of reproducing kernel Hilbert spaces

Outline: reproducing kernel Hilbert space

We will describe in order:

1. Hilbert space (very simple)
2. Kernel (lots of examples: e.g. you can build kernels from simpler kernels)
3. Reproducing property

Hilbert space

Definition (Inner product)

Let $\mathcal{H}$ be a vector space over $\mathbb{R}$. A function $\langle \cdot, \cdot \rangle_{\mathcal{H}} : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ is an inner product on $\mathcal{H}$ if

1. $\langle \alpha_1 f_1 + \alpha_2 f_2, g \rangle_{\mathcal{H}} = \alpha_1 \langle f_1, g \rangle_{\mathcal{H}} + \alpha_2 \langle f_2, g \rangle_{\mathcal{H}}$
2. $\langle f, g \rangle_{\mathcal{H}} = \langle g, f \rangle_{\mathcal{H}}$
3. $\langle f, f \rangle_{\mathcal{H}} \geq 0$ and $\langle f, f \rangle_{\mathcal{H}} = 0$ if and only if $f = 0$.

Norm induced by the inner product: $\|f\|_{\mathcal{H}} := \sqrt{\langle f, f \rangle_{\mathcal{H}}}$

Definition (Hilbert space)

"Well behaved" (complete) inner product space.

Hilbert space

Definition (Inner product)

Let $\mathcal{H}$ be a vector space over $\mathbb{R}$. A function $\langle \cdot, \cdot \rangle_{\mathcal{H}} : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ is an inner product on $\mathcal{H}$ if

1. $\langle \alpha_1 f_1 + \alpha_2 f_2, g \rangle_{\mathcal{H}} = \alpha_1 \langle f_1, g \rangle_{\mathcal{H}} + \alpha_2 \langle f_2, g \rangle_{\mathcal{H}}$
2. $\langle f, g \rangle_{\mathcal{H}} = \langle g, f \rangle_{\mathcal{H}}$
3. $\langle f, f \rangle_{\mathcal{H}} \geq 0$ and $\langle f, f \rangle_{\mathcal{H}} = 0$ if and only if $f = 0$.

Norm induced by the inner product: $\|f\|_{\mathcal{H}} := \sqrt{\langle f, f \rangle_{\mathcal{H}}}$

Definition (Hilbert space)

"Well behaved" (complete) inner product space.

Hilbert space

Definition (Inner product)

Let $\mathcal{H}$ be a vector space over $\mathbb{R}$. A function $\langle \cdot, \cdot \rangle_{\mathcal{H}} : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ is an inner product on $\mathcal{H}$ if

1. $\langle \alpha_1 f_1 + \alpha_2 f_2, g \rangle_{\mathcal{H}} = \alpha_1 \langle f_1, g \rangle_{\mathcal{H}} + \alpha_2 \langle f_2, g \rangle_{\mathcal{H}}$
2. $\langle f, g \rangle_{\mathcal{H}} = \langle g, f \rangle_{\mathcal{H}}$
3. $\langle f, f \rangle_{\mathcal{H}} \geq 0$ and $\langle f, f \rangle_{\mathcal{H}} = 0$ if and only if $f = 0$.

Norm induced by the inner product: $\|f\|_{\mathcal{H}} := \sqrt{\langle f, f \rangle_{\mathcal{H}}}$

Definition (Hilbert space)

"Well behaved" (complete) inner product space.

Kernel: inner product between feature maps

Definition

Let $\mathcal{X}$ be a non-empty set. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a **kernel** if there exists a Hilbert space and a map $\phi : \mathcal{X} \to \mathcal{H}$ such that $\forall x, x' \in \mathcal{X}$,

$$k(x, x') := \left\langle \phi(x), \phi(x') \right\rangle_{\mathcal{H}} .$$

- Almost no conditions on $\mathcal{X}$ (eg, $\mathcal{X}$ itself doesn't need an inner product, eg. documents).
- Think of kernel as similarity measure between features

What are some simple kernels? E.g for books? For images?

- A single kernel can correspond to multiple sets of underlying features.

$$\phi_1(x) = x \qquad \text{and} \qquad \phi_2(x) = \left[\, x/\sqrt{2} \quad x/\sqrt{2} \,\right]$$

Kernel: inner product between feature maps

Definition

Let $\mathcal{X}$ be a non-empty set. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a **kernel** if there exists a Hilbert space and a map $\phi : \mathcal{X} \to \mathcal{H}$ such that $\forall x, x' \in \mathcal{X}$,

$$k(x, x') := \langle \phi(x), \phi(x') \rangle_{\mathcal{H}} .$$

- Almost no conditions on $\mathcal{X}$ (eg, $\mathcal{X}$ itself doesn't need an inner product, eg. documents).
- Think of kernel as similarity measure between features

What are some simple kernels? E.g for books? For images?

- A single kernel can correspond to multiple sets of underlying features.

$$\phi_1(x) = x \qquad \text{and} \qquad \phi_2(x) = \left[\, x/\sqrt{2} \quad x/\sqrt{2} \,\right]$$

New kernels from old: sums, transformations

The great majority of useful kernels are built from simpler kernels.

Theorem (Sums of kernels are kernels)

Given $\alpha \geq 0$ and k, k_1 and k_2 all kernels on $\mathcal{X}$, then αk and $k_1 + k_2$ are kernels on $\mathcal{X}$.

Proof later! A difference of kernels may not be a kernel (**why?**)

Theorem (Mappings between spaces)

Let $\mathcal{X}$ and $\widetilde{\mathcal{X}}$ be sets, and define a map $A : \mathcal{X} \to \widetilde{\mathcal{X}}$. Define the kernel k on $\widetilde{\mathcal{X}}$. Then the kernel $k(A(x), A(x'))$ is a kernel on $\mathcal{X}$.

Example: $k(x, x') = x^2 \, (x')^2$.

New kernels from old: sums, transformations

The great majority of useful kernels are built from simpler kernels.

Theorem (Sums of kernels are kernels)

Given $\alpha \geq 0$ and k, k_1 and k_2 all kernels on $\mathcal{X}$, then αk and $k_1 + k_2$ are kernels on $\mathcal{X}$.

Proof later! A difference of kernels may not be a kernel (**why?**)

Theorem (Mappings between spaces)

Let $\mathcal{X}$ and $\widetilde{\mathcal{X}}$ be sets, and define a map $A : \mathcal{X} \to \widetilde{\mathcal{X}}$. Define the kernel k on $\widetilde{\mathcal{X}}$. Then the kernel $k(A(x), A(x'))$ is a kernel on $\mathcal{X}$.

Example: $k(x, x') = x^2 \, (x')^2$.

New kernels from old: sums, transformations

The great majority of useful kernels are built from simpler kernels.

Theorem (Sums of kernels are kernels)

Given $\alpha \geq 0$ and k, k_1 and k_2 all kernels on $\mathcal{X}$, then αk and $k_1 + k_2$ are kernels on $\mathcal{X}$.

Proof later! A difference of kernels may not be a kernel (**why?**)

Theorem (Mappings between spaces)

Let $\mathcal{X}$ and $\widetilde{\mathcal{X}}$ be sets, and define a map $A : \mathcal{X} \to \widetilde{\mathcal{X}}$. Define the kernel k on $\widetilde{\mathcal{X}}$. Then the kernel $k(A(x), A(x'))$ is a kernel on $\mathcal{X}$.

Example: $k(x, x') = x^2 (x')^2$.

New kernels from old: products

Theorem (Products of kernels are kernels)

Given k_1 on $\mathcal{X}_1$ and k_2 on $\mathcal{X}_2$, then $k_1 \times k_2$ is a kernel on $\mathcal{X}_1 \times \mathcal{X}_2$. If $\mathcal{X}_1 = \mathcal{X}_2 = \mathcal{X}$, then $k := k_1 \times k_2$ is a kernel on $\mathcal{X}$.

Proof: Main idea only!

k_1 is a kernel between **shapes**,

$$\phi_1(x) = \begin{bmatrix} \mathbb{I}_\square \\ \mathbb{I}_\triangle \end{bmatrix} \qquad \phi_1(\square) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad k_1(\square, \triangle) = 0.$$

k_2 is a kernel between **colors**,

$$\phi_2(x) = \begin{bmatrix} \mathbb{I}_{\color{red}\bullet} \\ \mathbb{I}_{\color{blue}\bullet} \end{bmatrix} \qquad \phi_2({\color{blue}\bullet}) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad k_2({\color{red}\bullet}, {\color{red}\bullet}) = 1.$$

New kernels from old: products

"Natural" feature space for **colored shapes**:

$$\Phi(x) = \begin{bmatrix} \mathbb{I}_{\color{red}\square} & \mathbb{I}_{\color{red}\triangle} \\ \mathbb{I}_{\color{blue}\square} & \mathbb{I}_{\color{blue}\triangle} \end{bmatrix} = \begin{bmatrix} \mathbb{I}_{\color{red}\bullet} \\ \mathbb{I}_{\color{blue}\bullet} \end{bmatrix} \begin{bmatrix} \mathbb{I}_{\square} & \mathbb{I}_{\triangle} \end{bmatrix} = \phi_2(x)\phi_1^\top(x)$$

Kernel is:

$$k(x,x')$$

$$= \sum_{i\in\{\color{red}\bullet,\color{blue}\bullet\}} \sum_{j\in\{\square,\triangle\}} \Phi_{ij}(x)\Phi_{ij}(x') = \mathrm{trace}\left(\phi_1(x)\underbrace{\phi_2^\top(x)\phi_2(x')}_{k_2(x,x')}\phi_1^\top(x')\right)$$

$$= \mathrm{trace}\left(\underbrace{\phi_1^\top(x')\phi_1(x)}_{k_1(x,x')}\right) k_2(x,x') = k_1(x,x')k_2(x,x')$$

New kernels from old: products

"Natural" feature space for **colored shapes**:

$$\Phi(x) = \begin{bmatrix} \mathbb{I}_{\color{red}\square} & \mathbb{I}_{\color{red}\triangle} \\ \mathbb{I}_{\color{blue}\square} & \mathbb{I}_{\color{blue}\triangle} \end{bmatrix} = \begin{bmatrix} \mathbb{I}_{\color{red}\bullet} \\ \mathbb{I}_{\color{blue}\bullet} \end{bmatrix} \begin{bmatrix} \mathbb{I}_{\square} & \mathbb{I}_{\triangle} \end{bmatrix} = \phi_2(x)\phi_1^\top(x)$$

Kernel is:

$$k(x,x')$$

$$= \sum_{i\in\{{\color{red}\bullet},{\color{blue}\bullet}\}} \sum_{j\in\{\square,\triangle\}} \Phi_{ij}(x)\Phi_{ij}(x') = \mathrm{trace}\left(\phi_1(x)\underbrace{\phi_2^\top(x)\phi_2(x')}_{k_2(x,x')}\phi_1^\top(x')\right)$$

$$= \mathrm{trace}\left(\underbrace{\phi_1^\top(x')\phi_1(x)}_{k_1(x,x')}\right)k_2(x,x') = k_1(x,x')k_2(x,x')$$

Sums and products $\implies$ polynomials

Theorem (Polynomial kernels)

Let $x, x' \in \mathbb{R}^d$ for $d \geq 1$, and let $m \geq 1$ be an integer and $c \geq 0$ be a positive real. Then

$$k(x, x') := \left(\langle x, x' \rangle + c\right)^m$$

is a valid kernel.

To prove: expand into a sum (with non-negative scalars) of kernels $\langle x, x' \rangle$ raised to integer powers. These individual terms are valid kernels by the product rule.

Infinite sequences

The kernels we've seen so far are dot products between finitely many features. E.g.

$$k(x, y) = \left[\begin{array}{ccc} \sin(x) & x^3 & \log x \end{array} \right]^\top \left[\begin{array}{ccc} \sin(y) & y^3 & \log y \end{array} \right]$$

where $\phi(x) = \left[\begin{array}{ccc} \sin(x) & x^3 & \log x \end{array} \right]$

Can a kernel be a dot product between infinitely many features?

Infinite sequences

Definition

The space ℓ_2 of 2-summable sequences is defined as all sequences $(a_i)_{i\geq 1}$ for which

$$\|a\|_{\ell_2}^2 = \sum_{i=1}^{\infty} a_i^2 < \infty.$$

Kernels can be defined in terms of sequences in ℓ_2.

Theorem

Given sequence of functions $(\phi_i(x))_{i\geq 1}$ in ℓ_2 where $\phi_i : \mathcal{X} \to \mathbb{R}$. Then

$$k(x, x') := \sum_{i=1}^{\infty} \phi_i(x)\phi_i(x') \tag{1}$$

is a well defined kernel on $\mathcal{X}$.

Infinite sequences (proof)

Proof: Cauchy-Schwarz:

$$\left|k(x,x')\right| = \left|\sum_{i=1}^{\infty} \phi_i(x)\phi_i(x')\right| \leq \left(\sum_{i=1}^{\infty} \phi_i^2(x)\right)^{1/2} \left(\sum_{i=1}^{\infty} \phi_i^2(x')\right)^{1/2}.$$

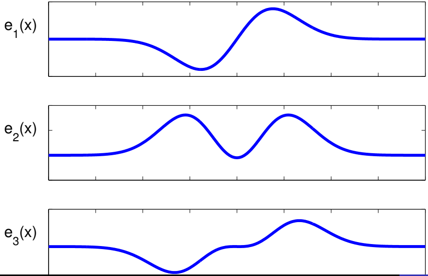
A famous infinite feature space kernel

Gaussian kernel,

$$k(x,x') = \exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right) = \sum_{i=1}^{\infty} \underbrace{\left(\sqrt{\lambda_i} e_i(x)\right)}_{\phi_i(x)} \underbrace{\left(\sqrt{\lambda_i} e_i(x')\right)}_{\phi_i(x')}$$

$$\lambda_k \propto b^k \qquad b < 1$$

$$e_k(x) \propto \exp(-(c-a)x^2) H_k(x\sqrt{2c}),$$



a, b, c are functions of σ, and H_k is kth order Hermite polynomial.

Positive definite functions

If we are given a "measure of similarity" with two arguments, $k(x, x')$, how can we determine if it is a valid kernel?

1. Find a feature map?
   1. Sometimes this is not obvious (eg if the feature vector is infinite dimensional)
   2. In any case, the feature map is not unique.
2. A direct property of the function: positive definiteness.

Positive definite functions

Definition (Positive definite functions)

A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is positive definite if $\forall n \geq 1$, $\forall (a_1, \ldots a_n) \in \mathbb{R}^n$, $\forall (x_1, \ldots, x_n) \in \mathcal{X}^n$,

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j k(x_i, x_j) \geq 0.$$

Why do we care? One good reason: it makes optimization *much* easier (e.g. when doing classification: Part II of the lecture!)

Kernels are positive definite

Theorem

The kernel $k(x,y) := \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$ for Hilbert space $\mathcal{H}$ is positive definite.

Proof.

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j k(x_i, x_j) &= \sum_{i=1}^{n}\sum_{j=1}^{n} \langle a_i \phi(x_i), a_j \phi(x_j) \rangle_{\mathcal{H}} \\ &= \left\| \sum_{i=1}^{n} a_i \phi(x_i) \right\|_{\mathcal{H}}^{2} \geq 0.\end{aligned}$$

Reverse also holds: positive definite $k(x,x')$ is inner product in $\mathcal{H}$ between $\phi(x)$ and $\phi(x')$. $\square$

Sum of kernels is a kernel

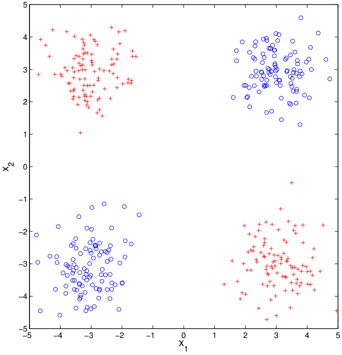
Consider two kernels $k_1(x,x')$ and $k_2(x,x')$. Then

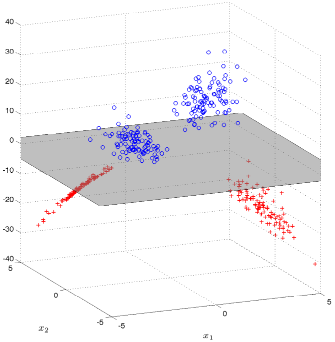
$$\begin{aligned}
&\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \left[k_1(x_i,x_j)+k_2(x_i,x_j)\right] \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j k_1(x_i,x_j) + \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j k_2(x_i,x_j) \\
&\geq 0
\end{aligned}$$

The reproducing kernel Hilbert space

First example: finite space, polynomial features

Reminder: XOR example:





First example: finite space, polynomial features

Reminder: Feature space from XOR motivating example:

$$\begin{aligned} \phi : \mathbb{R}^2 &\to \mathbb{R}^3 \\ x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} &\mapsto \phi(x) = \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \end{bmatrix}, \end{aligned}$$

with kernel

$$k(x, y) = \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \end{bmatrix}^\top \begin{bmatrix} y_1 \\ y_2 \\ y_1 y_2 \end{bmatrix}$$

(the standard inner product in $\mathbb{R}^3$ between features). Denote this feature space by $\mathcal{H}$.

First example: finite space, polynomial features

Define a linear function of the inputs x_1, x_2, and their product $x_1 x_2$,

$$f(x) = f_1 x_1 + f_2 x_2 + f_3 x_1 x_2.$$

f in a space of functions mapping from $\mathcal{X} = \mathbb{R}^2$ to $\mathbb{R}$. Equivalent representation for f,

$$f(\cdot) = \begin{bmatrix} f_1 & f_2 & f_3 \end{bmatrix}^\top.$$

$f(\cdot)$ refers to the function as an object (here as a vector in $\mathbb{R}^3$)
$f(x) \in \mathbb{R}$ is function evaluated at a point (a real number).

$$f(x) = f(\cdot)^\top \phi(x) = \langle f(\cdot), \phi(x) \rangle_{\mathcal{H}}$$

Evaluation of f at x is an **inner product in feature space** (here standard inner product in $\mathbb{R}^3$)
$\mathcal{H}$ is a space of functions mapping $\mathbb{R}^2$ to $\mathbb{R}$.

First example: finite space, polynomial features

Define a linear function of the inputs x_1, x_2, and their product $x_1 x_2$,

$$f(x) = f_1 x_1 + f_2 x_2 + f_3 x_1 x_2.$$

f in a space of functions mapping from $\mathcal{X} = \mathbb{R}^2$ to $\mathbb{R}$. Equivalent representation for f,

$$f(\cdot) = \begin{bmatrix} f_1 & f_2 & f_3 \end{bmatrix}^\top.$$

$f(\cdot)$ refers to the function as an object (here as a vector in $\mathbb{R}^3$)
$f(x) \in \mathbb{R}$ is function evaluated at a point (a real number).

$$f(x) = f(\cdot)^\top \phi(x) = \langle f(\cdot), \phi(x) \rangle_{\mathcal{H}}$$

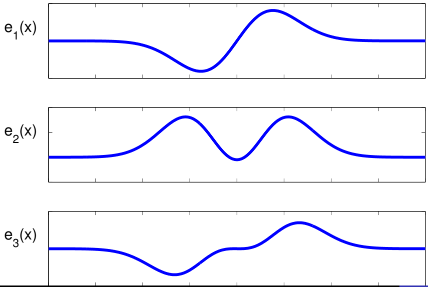
Evaluation of f at x is an **inner product in feature space** (here standard inner product in $\mathbb{R}^3$)
$\mathcal{H}$ is a space of functions mapping $\mathbb{R}^2$ to $\mathbb{R}$.

What if we have infinitely many features?

Gaussian kernel,

$$k(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right) = \sum_{i=1}^{\infty} \phi_i(x)\phi_i(x')$$

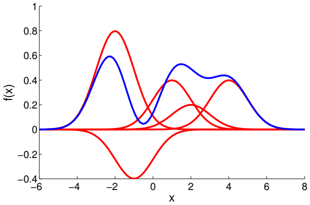
$$f(x) = \sum_{i=1}^{\infty} f_i \phi_i(x) \qquad \sum_{i=1}^{\infty} f_i^2 < \infty.$$



What if we have infinitely many features?

Function with Gaussian kernel:

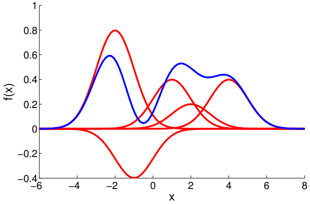
$$
\begin{aligned}
f(x) &:= \sum_{i=1}^{m} \alpha_i k(x_i, x) \\
&= \sum_{i=1}^{m} \alpha_i \left\langle \phi(x_i), \phi(x) \right\rangle_{\mathcal{H}} \\
&= \left\langle \sum_{i=1}^{m} \alpha_i \phi(x_i), \phi(x) \right\rangle_{\mathcal{H}}
\end{aligned}
$$



What if we have infinitely many features?

Function with Gaussian kernel:

$$
\begin{aligned}
f(x) :&= \sum_{i=1}^{m} \alpha_i k(x_i, x) \\
&= \sum_{i=1}^{m} \alpha_i \langle \phi(x_i), \phi(x) \rangle_{\mathcal{H}} \\
&= \left\langle \sum_{i=1}^{m} \alpha_i \phi(x_i), \phi(x) \right\rangle_{\mathcal{H}} \\
&= \sum_{\ell=1}^{\infty} f_\ell \phi_\ell(x) \\
&= \langle f(\cdot), \phi(x) \rangle_{\mathcal{H}}
\end{aligned}
$$



$f_\ell := \sum_{i=1}^{m} \alpha_i \phi_\ell(x_i)$

Much more convenient way to write functions of infinitely many features!

The reproducing property

We can write without ambiguity

$$\phi(x) = k(x, \cdot).$$

The two defining features of an RKHS:

- **The reproducing property:**
  $\forall x \in \mathcal{X}, \forall f \in \mathcal{H}, \ \ \langle f(\cdot), k(\cdot, x) \rangle = \langle f(\cdot), \phi(x) \rangle = f(x)$
- $k(\cdot, x) = \phi(x) \in \mathcal{H}$ for any $x \in \mathcal{X}$, and

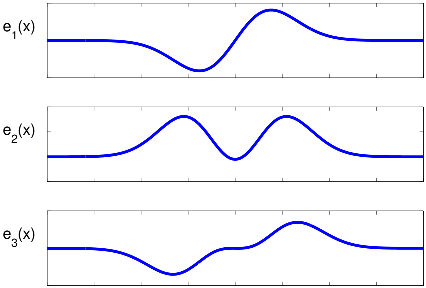
$$k(x, x') = \left\langle \phi(x), \phi(x') \right\rangle_{\mathcal{H}} = \left\langle k(\cdot, x), k(\cdot, x') \right\rangle_{\mathcal{H}}$$

A closer look: feature representation, Gaussian kernel

Reminder, Gaussian kernel,

$$k(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right) = \sum_{i=1}^{\infty} \underbrace{\left(\sqrt{\lambda_i} e_i(x)\right)}_{\phi_i(x)} \underbrace{\left(\sqrt{\lambda_i} e_i(x')\right)}_{\phi_i(x')}$$

$$\lambda_k \propto b^k \qquad b < 1$$

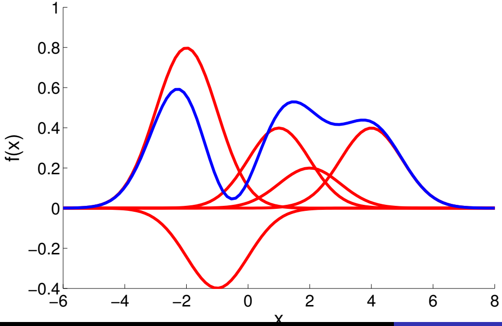


A closer look: feature representation, Gaussian kernel

RKHS function, Gaussian kernel:

$$f(x) := \sum_{i=1}^{m} \alpha_i k(x_i, x) = \sum_{\ell=1}^{\infty} f_\ell \underbrace{\left[\sqrt{\lambda_\ell} e_\ell(x)\right]}_{\phi_\ell(x)}$$

where $f_\ell = \sum_{i=1}^m \alpha_i \sqrt{\lambda_\ell} e_\ell(x_i)$.



NOTE that this enforces smoothing: λ_k decay as e_k become rougher, f_j decay since $\sum_j f_j^2 < \infty$.

Moore-Aronszajn

Theorem (Moore-Aronszajn)

Every positive definite kernel k uniquely associated with RKHS $\mathcal{H}$.

Recall feature map is *not* unique (as we saw earlier): only kernel is.

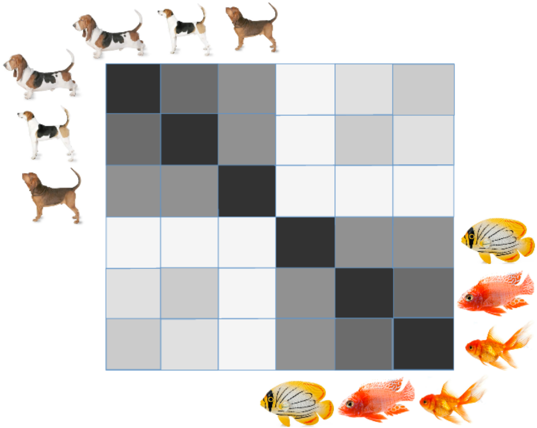
Simple Kernel Algorithms

Distance between feature means

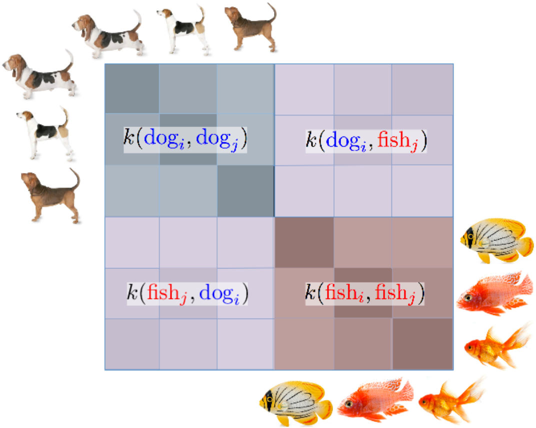
$$\sim P$$

$$\sim Q$$

Distance between feature means



Distance between feature means



$$\mathrm{MMD}^2 = \overline{K_{PP}} + \overline{K_{Q,Q}} - 2\overline{K_{P,Q}}$$

Distance between feature means

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

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

Distance between feature means

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

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

Distance between feature means

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

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

- 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 (e.g. the Gaussian kernel, where the feature space is infinite).

Distance between feature means

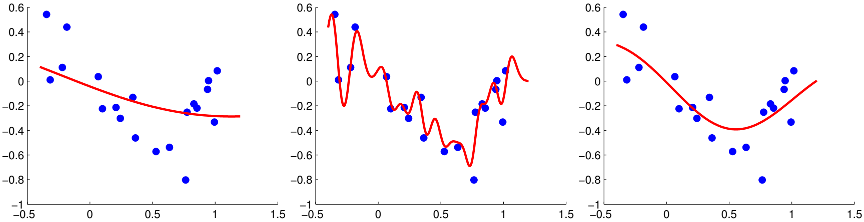
Sample $(x_i)_{i=1}^m$ from P and $(y_i)_{i=1}^n$ from Q. What is the distance between their means *in feature space*?

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

- 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 (e.g. the Gaussian kernel, where the feature space is infinite).

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 $\mathbb{R}^D$:

$$X = \left[\begin{array}{ccc} x_1 & \ldots & x_n \end{array}\right] \in \mathbb{R}^{D\times n} \qquad y := \left[\begin{array}{ccc} y_1 & \ldots & y_n \end{array}\right]^\top$$

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

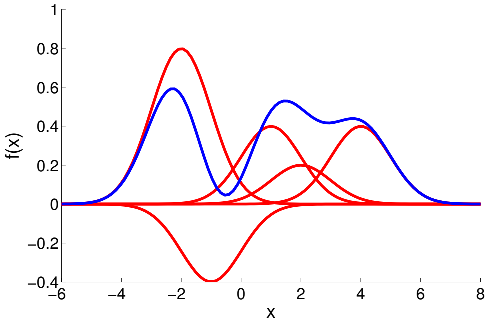
$$\begin{aligned} f^* &= \arg\min_{f\in\mathbb{R}^d} \left(\sum_{i=1}^n (y_i - x_i^\top f)^2 + \lambda \|f\|^2 \right) \\ &= \arg\min_{f\in\mathbb{R}^d} \left(\left\| y - X^\top f \right\|^2 + \lambda \|f\|^2 \right), \end{aligned}$$

The second term $\lambda\|f\|^2$ is chosen to avoid problems in high dimensional spaces (more soon).

Kernel ridge regression

We *begin* knowing f is a linear combination of feature space mappings of points (representer theorem)

$$f = \sum_{i=1}^{n} \alpha_i \phi(x_i) = \sum_{i=1}^{n} \alpha_i k(x_i, \cdot).$$



Kernel ridge regression

We *begin* knowing f is a linear combination of feature space mappings of points (representer theorem: second set of notes)

$$f = \sum_{i=1}^{n} \alpha_i \phi(x_i) = \sum_{i=1}^{n} \alpha_i k(x_i, \cdot).$$

Then

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

Differentiating wrt α and setting this to zero, we get

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

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

Kernel ridge regression

We *begin* knowing f is a linear combination of feature space mappings of points (representer theorem: second set of notes)

$$f = \sum_{i=1}^{n} \alpha_i \phi(x_i) = \sum_{i=1}^{n} \alpha_i k(x_i, \cdot).$$

Then

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

Differentiating wrt α and setting this to zero, we get

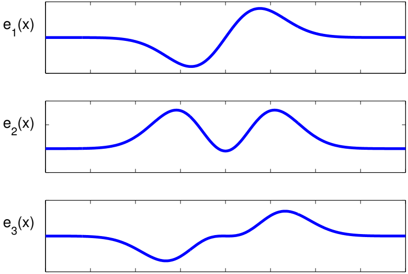
$$\alpha^* = (K + \lambda I_n)^{-1} y.$$

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

Smoothness

What does a small $\|f\|_{\mathcal{H}}$ achieve? Smoothness!
Recall for the Gaussian kernel:

$$f(x) = \sum_{i=1}^{\infty} f_i \sqrt{\lambda_i} e_i(x), \qquad \|f\|_{\mathcal{H}}^2 = \sum_{i=1}^{\infty} f_i^2.$$



Parameter selection for KRR

Given the objective

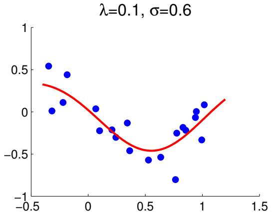
$$f^* = \arg\min_{f \in \mathcal{H}} \left(\sum_{i=1}^{n} (y_i - \langle f, \phi(x_i) \rangle_{\mathcal{H}})^2 + \lambda \|f\|_{\mathcal{H}}^2 \right).$$

How do we choose

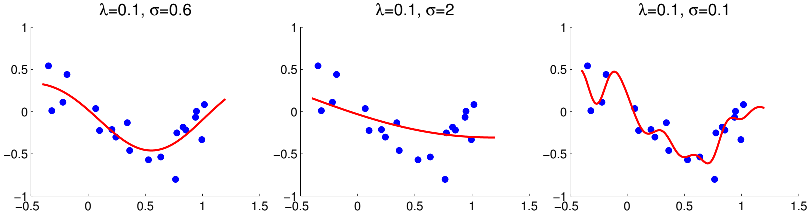
- The regularization parameter λ?
- The kernel parameter: for Gaussian kernel, σ in

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

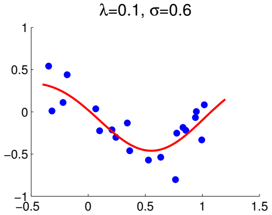
Choice of σ



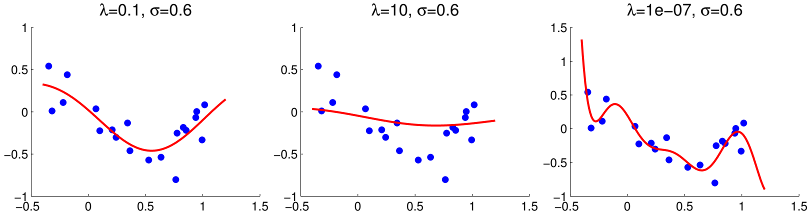
Choice of σ



Choice of λ



Choice of λ

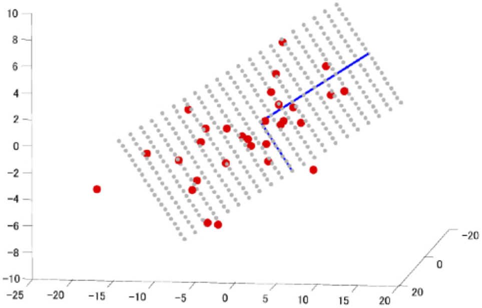


Cross validation

- Split n data into training set size n_{tr} and **test set** size $n_{\mathrm{te}} = n - n_{\mathrm{tr}}$.
- Split trainining set into m equal chunks of size $n_{\mathrm{val}} = n_{\mathrm{tr}}/m$. Call these $X_{\mathrm{val},i}, Y_{\mathrm{val},i}$ for $i \in \{1, \ldots, m\}$
- For each λ, σ pair
  - For each $X_{\mathrm{val},i}, Y_{\mathrm{val},i}$
    - Train ridge regression on remaining trainining set data $X_{\mathrm{tr}} \setminus X_{\mathrm{val},i}$ and $Y_{\mathrm{tr}} \setminus Y_{\mathrm{val},i}$,
    - Evaluate its error on the validation data $X_{\mathrm{val},i}, Y_{\mathrm{val},i}$
  - 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 $X_{\mathrm{te}}, Y_{\mathrm{te}}$.

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 from Kenji Fukumizu)

Application of 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 from kernel PCA (these are orthogonal).

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

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

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

Application of 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 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.

Application of 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 from kernel PCA (these are orthogonal).
Define the nearest point $y^* \in \mathcal{X}$ to this feature space projection as

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

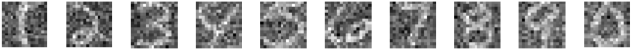
In many cases, not possible to reduce the squared error to zero, as no single y^* corresponds to exact solution.

Application of kPCA: image denoising

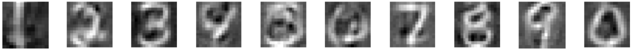
Projection onto PCA subspace for denoising. kPCA: data may not be Gaussian distributed, but can lie in a submanifold in input space.

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)

What is PCA?

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

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

$X = \left[\begin{array}{ccc} x_1 & \ldots & x_n \end{array}\right]$, $H = I_n - n^{-1}\mathbf{1}_{n\times n}$, $\mathbf{1}_{n\times n}$ a matrix of ones.

Definition (Principal components)

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

PCA in feature space

Kernel version, first principal component:

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

We can write

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

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

PCA in feature space

Kernel version, first principal component:

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

We can write

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

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

$$(a\otimes b)c := a\langle b, c\rangle_{\mathcal{H}} \tag{2}$$

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

How to solve kernel PCA (1)

Eigenfunctions of kernel covariance:

$$
\begin{aligned}
f_\ell \lambda_\ell &= C f_\ell \\
&= \left(\frac{1}{n} \sum_{i=1}^{n} \tilde{\phi}(x_i) \otimes \tilde{\phi}(x_i) \right) f_\ell \\
&= \frac{1}{n} \sum_{i=1}^{n} \tilde{\phi}(x_i) \left\langle \tilde{\phi}(x_i), \sum_{j=1}^{n} \alpha_{\ell j} \tilde{\phi}(x_j) \right\rangle_{\mathcal{H}} \\
&= \frac{1}{n} \sum_{i=1}^{n} \tilde{\phi}(x_i) \left(\sum_{j=1}^{n} \alpha_{\ell j} \tilde{k}(x_i, x_j) \right)
\end{aligned}
$$

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

How to solve kernel PCA (1)

Eigenfunctions of kernel covariance:

$$\begin{aligned} f_\ell \lambda_\ell &= C f_\ell \\ &= \left(\frac{1}{n} \sum_{i=1}^{n} \tilde{\phi}(x_i) \otimes \tilde{\phi}(x_i) \right) f_\ell \\ &= \frac{1}{n} \sum_{i=1}^{n} \tilde{\phi}(x_i) \left\langle \tilde{\phi}(x_i), \sum_{j=1}^{n} \alpha_{\ell j} \tilde{\phi}(x_j) \right\rangle_{\mathcal{H}} \\ &= \frac{1}{n} \sum_{i=1}^{n} \tilde{\phi}(x_i) \left(\sum_{j=1}^{n} \alpha_{\ell j} \tilde{k}(x_i, x_j) \right) \end{aligned}$$

$\tilde{k}(x_i, x_j)$ 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 $\tilde{\phi}(x_q)$:

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

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

Writing this as a matrix equation,

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

How to solve kernel PCA (2)

We can now project both sides of

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

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

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

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

Writing this as a matrix equation,

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

Projection onto kernel PC

How do you project a new point x^* onto the principal component f? Assuming f is properly normalised, the projection is

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