

## Lecture 01: Fundamentals of Kernel methods

**Gustavo Camps-Valls**

Image Processing Laboratory (IPL) – Universitat de València. Spain.  
gustavo.camps@uv.es, <http://www.uv.es/gcamps>



The organization of the course:

- ① **Fundamentals of kernel methods** <<<
- ② Supervised and unsupervised kernel-based classification
- ③ Kernel methods for regression and time series analysis
- ④ Nonlinear feature extraction with kernels

- Many methods in machine learning **classification**:
  - Linear discriminant analysis (LDA)
  - Decision trees
  - Neural networks
- Many methods in machine learning **regression**:
  - Regularized linear regression
  - Decision trees
  - Splines
  - Neural networks
- Many methods in machine learning **feature extraction**:
  - Principal component analysis (PCA)
  - Independent component analysis (ICA)
  - Partial least squares (PLS)
- Many methods in machine learning **clustering problems**:
  - $k$ -means, fuzzy  $k$ -means
  - Hierarchical clustering
  - Gaussian mixture models
- Many methods in machine learning **density estimation**:
  - Mixture of Gaussians
  - Parzen windows
  - RBIG

## Observations

- ✓ All methods based on estimating distances/similarities between samples
- × A toolbox of powerful yet unrelated methods
  - Different complexities
  - Different and unintuitive free parameters to tune
  - Not at all clear how regularization is included
  - Not clear how overfitting is avoided

## Observations

- ✓ All methods based on estimating distances/similarities between samples
- × A toolbox of powerful yet unrelated methods
  - Different complexities
  - Different and unintuitive free parameters to tune
  - Not at all clear how regularization is included
  - Not clear how overfitting is avoided

## Kernel methods

- A formalization of all machine learning problems
- They provide a way to develop new methods quite easily
- They exploit the notion of similarity between samples

## What we need for learning ...

## Learning from data means ...

- Suppose we are given empirical data

$$(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n) \in \mathcal{X} \times \mathcal{Y}$$

where  $\mathbf{x}_i$  are the *inputs* taken from the *set*  $\mathcal{X}$  and  $y_i \in \mathcal{Y}$  are the *targets*.

- **Learning** means to use these data to make statements about unseen elements  $\mathbf{x} \in \mathcal{X}$ .

## What we need for learning ...

## Learning from data means ...

- Suppose we are given empirical data

$$(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n) \in \mathcal{X} \times \mathcal{Y}$$

where  $\mathbf{x}_i$  are the *inputs* taken from the *set*  $\mathcal{X}$  and  $y_i \in \mathcal{Y}$  are the *targets*.

- **Learning** means to use these data to make statements about unseen elements  $\mathbf{x} \in \mathcal{X}$ .

Example: binary classification  $\mathcal{X} = \{-1, +1\}$ 

We want to construct a function  $f : \mathcal{X} \rightarrow \mathcal{Y}$  which assigns to each element of  $\mathcal{X}$  a class label.

- The function should not be arbitrary but one which *generalizes* well, i.e. making few errors on unseen data from the same problem.
- We will need to exploit structure of the training examples and to impose a *similarity* between data points that describes well the test set

## What we need for learning ...

Q1: What is a set  $\mathcal{X}$ ?

- A set is a collection of distinct objects
- A set is an object in its own right



*"By a 'set' we mean any collection  $M$  into a whole of definite, distinct objects  $m$  (which are called the 'elements' of  $M$ ) of our perception or of our thought. "*

— Georg Cantor, 1880

- The elements or members of a set can be anything: numbers, people, letters of the alphabet, other sets, ...
- Sets are conventionally denoted with capital mathematical letters
- Sets  $\mathcal{X} = \mathcal{X}'$  iff they have precisely the same elements

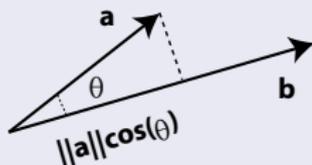
## What we need for learning ...

Q2: How to measure similarities between elements (vectors)  $\mathbf{a}$  and  $\mathbf{b}$ ?

- **Definition:** The dot (or scalar, or inner) product between vectors  $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$ :

$$\mathbf{a} \cdot \mathbf{b} = \langle \mathbf{a}, \mathbf{b} \rangle = \sum_{f=1}^d \mathbf{a}^{(f)} \mathbf{b}^{(f)}$$

- **Geometrical interpretation:**



$$\langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a} \cdot \mathbf{b} = \mathbf{a}^\top \mathbf{b} = \|\mathbf{a}\| \|\mathbf{b}\| \cos(\theta)$$

- **Intuitions:**

- The dot product measures how much of a vector  $\mathbf{a}$  is contained in  $\mathbf{b}$
- ... and how much the two vectors point in the same direction

## What we need for learning ...

## Properties of dot products (print and forget!)

- ① The size of the angle (similarity):  $\theta = \arccos\left(\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|}\right)$
- ② Convert vectors to unit vectors (unit distance):  
 $\tilde{\mathbf{a}} = \mathbf{a} / \|\mathbf{a}\| \rightarrow \theta = \arccos(\tilde{\mathbf{a}} \tilde{\mathbf{b}})$
- ③ The dot product is commutative:  $\mathbf{a} \cdot \mathbf{b} = \mathbf{b} \cdot \mathbf{a}$
- ④ The dot product is distributive:  $\mathbf{a} \cdot (\mathbf{b} + \mathbf{c}) = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \cdot \mathbf{c}$
- ⑤ The dot product is bilinear:  $\mathbf{a} \cdot (r\mathbf{b} + \mathbf{c}) = r\mathbf{a} \cdot \mathbf{b} + \mathbf{a} \cdot \mathbf{c}$
- ⑥ The dot product satisfies:  $(c_1\mathbf{a}) \cdot (c_2\mathbf{b}) = (c_1c_2)(\mathbf{a} \cdot \mathbf{b})$
- ⑦ Dot product on stacked vectors:  $\langle [\mathbf{a}, \mathbf{b}], [\mathbf{c}, \mathbf{d}] \rangle = \mathbf{ac} + \mathbf{bd}$
- ⑧ Two non-zero vectors  $\mathbf{a}$  and  $\mathbf{b}$  are perpendicular (orthogonal) iff  $\mathbf{a} \cdot \mathbf{b} = \langle \mathbf{a}, \mathbf{b} \rangle = 0$
- ⑨ If  $\mathbf{a} \cdot \mathbf{b} = \mathbf{a} \cdot \mathbf{c}$  and  $\mathbf{a} \neq \mathbf{0} \rightarrow \mathbf{a} \cdot (\mathbf{b} - \mathbf{c}) = 0 \rightarrow \mathbf{a} \perp (\mathbf{b} - \mathbf{c})$  but  $\mathbf{b} \neq \mathbf{c}$ .
- ⑩ Derivative of a dot product:  $\frac{d}{dt}(\mathbf{a} \cdot \mathbf{b}) = \frac{d\mathbf{a}}{dt} \cdot \mathbf{b} + \mathbf{a} \cdot \frac{d\mathbf{b}}{dt}$  is a vector
- ⑪ Frobenius inner product:  $\mathbf{A} : \mathbf{B} = \sum_{ij} \mathbf{A}_{ij} \mathbf{B}_{ij} = \text{trace}(\mathbf{A}^T \mathbf{B})$

## What we need for learning ...

## Hilbert spaces: a generalization of dot product spaces



- A Hilbert space is an abstract vector space that has the structure of an inner product that allows computing lengths and angles
- A Hilbert space is a space endowed with a dot product defined on possibly infinite-dimensional points  
— Hilbert, 1909

## What we need for learning ...

## Hilbert spaces: a generalization of dot product spaces



- A Hilbert space is an abstract vector space that has the structure of an inner product that allows computing lengths and angles
- A Hilbert space is a space endorsed with a dot product defined on possibly infinite-dimensional points  
— Hilbert, 1909

Example:  $\mathbb{R}^3$  Euclidean space with dot product is a Hilbert space

The dot product is:

- Symmetric:  $\mathbf{x} \cdot \mathbf{y} = \mathbf{y} \cdot \mathbf{x}$
- Linear:  $(a\mathbf{x}_1 + b\mathbf{x}_2) \cdot \mathbf{y} = a\mathbf{x}_1 \cdot \mathbf{y} + b\mathbf{x}_2 \cdot \mathbf{y}$
- Positive definite:  $\mathbf{x} \cdot \mathbf{x} \geq 0$  (equality only for  $\mathbf{x} = 0$ )

Other characteristics:

- Norm:  $\|\mathbf{x}\| = \sqrt{\mathbf{x} \cdot \mathbf{x}} = \sqrt{\langle \mathbf{x}, \mathbf{y} \rangle}$
- Distance between points:  $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\| = \sqrt{\langle \mathbf{x} - \mathbf{y}, \mathbf{x} - \mathbf{y} \rangle}$
- Cauchy-Schwarz ineq.:  $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z}) \rightarrow |\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\| \|\mathbf{y}\|$

**Check this out!**

- No assumption made on  $\mathcal{X}$  (we just said it is a 'set')
- No assumption about the similarity (any similarity function could serve)
- Therefore, let's do it general enough in Hilbert spaces.

## Check this out!

- No assumption made on  $\mathcal{X}$  (we just said it is a 'set')
- No assumption about the similarity (any similarity function could serve)
- Therefore, let's do it general enough in Hilbert spaces.

## Important definitions: Mapping function and kernel function

- Map the data into a space where we have a notion of similarity, namely a dot product space  $\mathcal{H}$  (**feature space**), using the **feature mapping**

$$\phi : \mathcal{X} \rightarrow \mathcal{H}, \quad \mathbf{x} \mapsto \phi(\mathbf{x})$$

- The similarity between the elements in  $\mathcal{H}$  can now be measured using its associated dot product  $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ .
- The **kernel function** measures similarity in  $\mathcal{H}$ :

$$K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}, \quad (\mathbf{x}, \mathbf{x}') \mapsto K(\mathbf{x}, \mathbf{x}')$$

which we require to satisfy for all  $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$

$$K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}}$$

## Positive Definite Kernels

Wait, wait, wait, is this magic!?

- How do you know there is such function  $K$  reproducing a similarity measure in a given (in principle unknown) space?
- We need to demonstrate there exists a kernel function satisfying that:

$$K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}}$$

- This equivalence would be really nice!
  - **Intuitive!** Just work with similarity functions  $K$ , forget about the mapping.
  - **General!** If we can replace dot products with a kernel function then we can generalize lots of algorithms

## Positive Definite Kernels

## A toy demo: Understanding the 'kernel trick' ...

- An example of (non-linear) transformation to a higher dimensional space is the following polynomial transformation:

$$x \in \mathbb{R} \text{ and } \phi(x) = \{x^2, \sqrt{2}x, 1\}^T \in \mathbb{R}^3$$

## Positive Definite Kernels

## A toy demo: Understanding the 'kernel trick' ...

- An example of (non-linear) transformation to a higher dimensional space is the following polynomial transformation:

$$x \in \mathbb{R} \text{ and } \phi(x) = \{x^2, \sqrt{2}x, 1\}^T \in \mathbb{R}^3$$

- **The question is:** *can we obtain a dot product (a similarity measure) in this space which can be expressed only in terms of input data  $x$ ?*

## Positive Definite Kernels

## A toy demo: Understanding the 'kernel trick' ...

- An example of (non-linear) transformation to a higher dimensional space is the following polynomial transformation:

$$x \in \mathbb{R} \text{ and } \phi(x) = \{x^2, \sqrt{2}x, 1\}^\top \in \mathbb{R}^3$$

- **The question is:** *can we obtain a dot product (a similarity measure) in this space which can be expressed only in terms of input data  $x$ ?*
- **The answer is:** *Yes, the explicit dot-product can be re-written as:*

$$\begin{aligned} \langle \phi(x_1), \phi(x_2) \rangle &\equiv \phi(x_1)^\top \phi(x_2) = \{x_1^2, \sqrt{2}x_1, 1\} \{x_2^2, \sqrt{2}x_2, 1\}^\top = \\ &= x_1^2 x_2^2 + 2x_1 x_2 + 1 = (x_1 x_2 + 1)^2 = (\langle x_1, x_2 \rangle + 1)^2 \equiv K_{1,2} \end{aligned}$$

## Positive Definite Kernels

## A toy demo: Understanding the 'kernel trick' ...

- An example of (non-linear) transformation to a higher dimensional space is the following polynomial transformation:

$$x \in \mathbb{R} \text{ and } \phi(x) = \{x^2, \sqrt{2}x, 1\}^\top \in \mathbb{R}^3$$

- **The question is:** *can we obtain a dot product (a similarity measure) in this space which can be expressed only in terms of input data  $x$ ?*
- **The answer is:** *Yes, the explicit dot-product can be re-written as:*

$$\begin{aligned} \langle \phi(x_1), \phi(x_2) \rangle &\equiv \phi(x_1)^\top \phi(x_2) = \{x_1^2, \sqrt{2}x_1, 1\} \{x_2^2, \sqrt{2}x_2, 1\}^\top = \\ &= x_1^2 x_2^2 + 2x_1 x_2 + 1 = (x_1 x_2 + 1)^2 = (\langle x_1, x_2 \rangle + 1)^2 \equiv K_{1,2} \end{aligned}$$

- You can do the same for higher dimensions very easily!

## Positive Definite Kernels

## A toy demo: Understanding the 'kernel trick' ...

- An example of (non-linear) transformation to a higher dimensional space is the following polynomial transformation:

$$x \in \mathbb{R} \text{ and } \phi(x) = \{x^2, \sqrt{2}x, 1\}^\top \in \mathbb{R}^3$$

- **The question is:** *can we obtain a dot product (a similarity measure) in this space which can be expressed only in terms of input data  $x$ ?*
- **The answer is:** *Yes, the explicit dot-product can be re-written as:*

$$\begin{aligned} \langle \phi(x_1), \phi(x_2) \rangle &\equiv \phi(x_1)^\top \phi(x_2) = \{x_1^2, \sqrt{2}x_1, 1\} \{x_2^2, \sqrt{2}x_2, 1\}^\top = \\ &= x_1^2 x_2^2 + 2x_1 x_2 + 1 = (x_1 x_2 + 1)^2 = (\langle x_1, x_2 \rangle + 1)^2 \equiv K_{1,2} \end{aligned}$$

- You can do the same for higher dimensions very easily!
- The dot product is a *kernel function*
- The higher dimensional space is a *Reproducing Kernel in Hilbert Spaces*.

## Positive Definite Kernels

## Definition: Gram matrix

Given a kernel  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  and inputs  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$ . We call the  $n \times n$  matrix  $\mathbf{K}$  with entries

$$\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) \quad (1)$$

the *Gram matrix* or the *kernel matrix* of  $K$  with respect to  $\mathbf{x}_1, \dots, \mathbf{x}_n$ .

## Positive Definite Kernels

## Definition: Gram matrix

Given a kernel  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  and inputs  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$ . We call the  $n \times n$  matrix  $\mathbf{K}$  with entries

$$\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) \quad (1)$$

the *Gram matrix* or the *kernel matrix* of  $K$  with respect to  $\mathbf{x}_1, \dots, \mathbf{x}_n$ .

## Definition: Positive definite matrix

A real symmetric  $n \times n$  matrix  $\mathbf{K}$  is called *positive definite* if  $\forall \mathbf{c}_i$

$$\sum_{i,j=1}^n c_i c_j \mathbf{K}_{ij} \geq 0 \rightarrow \mathbf{c}^\top \mathbf{K} \mathbf{c} \geq 0$$

## Positive Definite Kernels

## Definition: Gram matrix

Given a kernel  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  and inputs  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$ . We call the  $n \times n$  matrix  $\mathbf{K}$  with entries

$$\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) \quad (1)$$

the *Gram matrix* or the *kernel matrix* of  $K$  with respect to  $\mathbf{x}_1, \dots, \mathbf{x}_n$ .

## Definition: Positive definite matrix

A real symmetric  $n \times n$  matrix  $\mathbf{K}$  is called *positive definite* if  $\forall \mathbf{c}$ ,

$$\sum_{i,j=1}^n c_i c_j \mathbf{K}_{ij} \geq 0 \rightarrow \mathbf{c}^\top \mathbf{K} \mathbf{c} \geq 0$$

## Definition: Positive definite kernel (p.d.)

- If for all  $n \in \mathbb{N}$  and for all  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{X}$  the Gram matrix  $\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$  is positive definite we call the kernel a *positive definite kernel* (p.d.),  $\mathbf{K} \succeq 0$ .
- If the kernel  $K$  gives rise to a strictly positive definite Gram matrix we will call it a *strictly positive definite kernel*,  $\mathbf{K} \succ 0$

## Positive Definite Kernels

## Proposition

A function  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  is a positive definite kernel *if and only if* there exists a Hilbert space  $\mathcal{H}$  and a feature map  $\phi : \mathcal{X} \rightarrow \mathcal{H}$  such that for all  $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$  we have  $K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}}$ .

“ $\Leftarrow$ ” Assume the kernel can be written as  $K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}}$ . Is  $K$  positive definite?

$$\sum_{i,j=1}^n c_i c_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}} = \left\langle \sum_{i=1}^n c_i \phi(\mathbf{x}_i), \sum_{j=1}^n c_j \phi(\mathbf{x}_j) \right\rangle_{\mathcal{H}} = \left\| \sum_{i=1}^n c_i \phi(\mathbf{x}_i) \right\|_{\mathcal{H}}^2 \geq 0.$$

“ $\Rightarrow$ ” Given a positive definite kernel, how to construct a Hilbert space and the feature map  $\phi$ ? [see Schölkopf02]  $\square$

## Representer

## Representer theorem [Kimeldorf71, Cox90]

Let  $\Omega : [0, \infty) \rightarrow \mathbb{R}$  be a strictly monotonic increasing function; let  $V : (\mathcal{X} \times \mathbb{R}^2)^n \rightarrow \mathbb{R} \cup \{\infty\}$  be an arbitrary loss function; and let  $\mathcal{H}$  be a RKHS with reproducing kernel  $K$ . Then:

$$f^* = \min_{f \in \mathcal{H}} \left\{ V((\mathbf{x}_1, y_1, f(\mathbf{x}_1)), \dots, (\mathbf{x}_n, y_n, f(\mathbf{x}_n))) + \Omega(\|f\|_{\mathcal{H}}^2) \right\}$$

admits a representation  $f^*(\mathbf{z}) = \sum_{i=1}^n \alpha_i \mathbf{K}(\mathbf{z}, \mathbf{x}_i)$ ,  $\alpha_i \in \mathbb{R}$ ,  $\boldsymbol{\alpha} \in \mathbb{R}^{n \times 1}$

## Representer

## Representer theorem [Kimeldorf71, Cox90]

Let  $\Omega : [0, \infty) \rightarrow \mathbb{R}$  be a strictly monotonic increasing function; let  $V : (\mathcal{X} \times \mathbb{R}^2)^n \rightarrow \mathbb{R} \cup \{\infty\}$  be an arbitrary loss function; and let  $\mathcal{H}$  be a RKHS with reproducing kernel  $K$ . Then:

$$f^* = \min_{f \in \mathcal{H}} \left\{ V((\mathbf{x}_1, y_1, f(\mathbf{x}_1)), \dots, (\mathbf{x}_n, y_n, f(\mathbf{x}_n))) + \Omega(\|f\|_{\mathcal{H}}^2) \right\}$$

admits a representation  $f^*(\mathbf{z}) = \sum_{i=1}^n \alpha_i \mathbf{K}(\mathbf{z}, \mathbf{x}_i)$ ,  $\alpha_i \in \mathbb{R}$ ,  $\boldsymbol{\alpha} \in \mathbb{R}^{n \times 1}$

## How to use it?

- 1 'Kernel functions  $f$  are defined as a lin. comb. of similarities'

## Representer

## Representer theorem [Kimeldorf71, Cox90]

Let  $\Omega : [0, \infty) \rightarrow \mathbb{R}$  be a strictly monotonic increasing function; let  $V : (\mathcal{X} \times \mathbb{R}^2)^n \rightarrow \mathbb{R} \cup \{\infty\}$  be an arbitrary loss function; and let  $\mathcal{H}$  be a RKHS with reproducing kernel  $K$ . Then:

$$f^* = \min_{f \in \mathcal{H}} \left\{ V((\mathbf{x}_1, y_1, f(\mathbf{x}_1)), \dots, (\mathbf{x}_n, y_n, f(\mathbf{x}_n))) + \Omega(\|f\|_{\mathcal{H}}^2) \right\}$$

admits a representation  $f^*(\mathbf{z}) = \sum_{i=1}^n \alpha_i \mathbf{K}(\mathbf{z}, \mathbf{x}_i)$ ,  $\alpha_i \in \mathbb{R}$ ,  $\boldsymbol{\alpha} \in \mathbb{R}^{n \times 1}$

## How to use it?

- 1 'Kernel functions  $f$  are defined as a lin. comb. of similarities'
- 2 'A vector in a RKHS,  $\mathbf{w} \in \mathcal{H}$  can be expressed (spanned) as a linear combination of  $n$  points in that space':

$$\underbrace{\mathbf{w}}_{d_{\mathcal{H}} \times 1} = \sum_{i=1}^n \alpha_i \phi(\mathbf{x}_i) \rightarrow \underbrace{\mathbf{w}}_{d_{\mathcal{H}} \times 1} = \underbrace{\Phi^T}_{d_{\mathcal{H}} \times n} \underbrace{\boldsymbol{\alpha}}_{n \times 1}$$

## Representer

## Representer theorem [Kimeldorf71, Cox90]

Let  $\Omega : [0, \infty) \rightarrow \mathbb{R}$  be a strictly monotonic increasing function; let  $V : (\mathcal{X} \times \mathbb{R}^2)^n \rightarrow \mathbb{R} \cup \{\infty\}$  be an arbitrary loss function; and let  $\mathcal{H}$  be a RKHS with reproducing kernel  $K$ . Then:

$$f^* = \min_{f \in \mathcal{H}} \left\{ V((\mathbf{x}_1, y_1, f(\mathbf{x}_1)), \dots, (\mathbf{x}_n, y_n, f(\mathbf{x}_n))) + \Omega(\|f\|_{\mathcal{H}}^2) \right\}$$

admits a representation  $f^*(\mathbf{z}) = \sum_{i=1}^n \alpha_i \mathbf{K}(\mathbf{z}, \mathbf{x}_i)$ ,  $\alpha_i \in \mathbb{R}$ ,  $\alpha \in \mathbb{R}^{n \times 1}$

## How to use it?

- 1 'Kernel functions  $f$  are defined as a lin. comb. of similarities'
- 2 'A vector in a RKHS,  $\mathbf{w} \in \mathcal{H}$  can be expressed (**spanned**) as a linear combination of  $n$  points in that space':

$$\underbrace{\mathbf{w}}_{d_{\mathcal{H}} \times 1} = \sum_{i=1}^n \alpha_i \phi(\mathbf{x}_i) \rightarrow \underbrace{\mathbf{w}}_{d_{\mathcal{H}} \times 1} = \underbrace{\Phi^{\top}}_{d_{\mathcal{H}} \times n} \underbrace{\alpha}_{n \times 1}$$

- 3 The  $l_2$  regularizer:  $\|\mathbf{w}\|^2 = \mathbf{w}^{\top} \mathbf{w} = (\Phi^{\top} \alpha)^{\top} (\Phi^{\top} \alpha) = \alpha^{\top} \mathbf{K} \alpha$

## Representer

## Representer theorem [Kimeldorf71, Cox90]

Let  $\Omega : [0, \infty) \rightarrow \mathbb{R}$  be a strictly monotonic increasing function; let  $V : (\mathcal{X} \times \mathbb{R}^2)^n \rightarrow \mathbb{R} \cup \{\infty\}$  be an arbitrary loss function; and let  $\mathcal{H}$  be a RKHS with reproducing kernel  $K$ . Then:

$$f^* = \min_{f \in \mathcal{H}} \left\{ V((\mathbf{x}_1, y_1, f(\mathbf{x}_1)), \dots, (\mathbf{x}_n, y_n, f(\mathbf{x}_n))) + \Omega(\|f\|_{\mathcal{H}}^2) \right\}$$

admits a representation  $f^*(\mathbf{z}) = \sum_{i=1}^n \alpha_i \mathbf{K}(\mathbf{z}, \mathbf{x}_i)$ ,  $\alpha_i \in \mathbb{R}$ ,  $\alpha \in \mathbb{R}^{n \times 1}$

## How to use it?

- 1 'Kernel functions  $f$  are defined as a lin. comb. of similarities'
- 2 'A vector in a RKHS,  $\mathbf{w} \in \mathcal{H}$  can be expressed (**spanned**) as a linear combination of  $n$  points in that space':

$$\underbrace{\mathbf{w}}_{d_{\mathcal{H}} \times 1} = \sum_{i=1}^n \alpha_i \phi(\mathbf{x}_i) \rightarrow \underbrace{\mathbf{w}}_{d_{\mathcal{H}} \times 1} = \underbrace{\Phi^{\top}}_{d_{\mathcal{H}} \times n} \underbrace{\alpha}_{n \times 1}$$

- 3 The  $\ell_2$  regularizer:  $\|\mathbf{w}\|^2 = \mathbf{w}^{\top} \mathbf{w} = (\Phi^{\top} \alpha)^{\top} (\Phi^{\top} \alpha) = \alpha^{\top} \mathbf{K} \alpha$
- 4 If  $\Omega(\|f\|_{\mathcal{H}}^2) = \|\mathbf{K} \alpha\|^2 = \alpha^{\top} \mathbf{K}^{\top} \mathbf{K} \alpha = \alpha^{\top} \tilde{\mathbf{K}} \alpha$

## Remember

- Kernels compute dot products in some space  $\mathcal{H}$
- A positive definite kernel always produces a symmetric positive definite Gram matrix for elements in  $\mathcal{X}$ .
- Matlab checks for p.d.: `all(eig(K)>0)? det(K)>0?`
- Matlab checks for s.p.d.: `all(eig(K)>=0)? det(K)>=0?`
- A Gram matrix contains similarities (dot products in a given space) between samples
- We will sometimes refer to a positive definite kernel simply as a *kernel*
- A vector in  $\mathcal{H}$  lies in the span of a subset of mapped points  $\{\phi(\mathbf{x}_i) | i = 1, \dots, n\}$

Exercise: demonstrate the Cauchy-Schwarz inequality in  $\mathcal{H}$

Demonstrate that if  $K$  is p.d. then  $K(\mathbf{x}_1, \mathbf{x}_2)^2 \leq K(\mathbf{x}_1, \mathbf{x}_1)K(\mathbf{x}_2, \mathbf{x}_2)$

Operations in  $\mathcal{H}$ 

- ④ **Translation:** A translation in feature space can be written as the modified feature map  $\tilde{\phi}(\mathbf{x}) = \phi(\mathbf{x}) + \Gamma$  with  $\Gamma \in \mathcal{H}$ . Then:

$$\langle \phi(\mathbf{x}) + \Gamma, \phi(\mathbf{x}') + \Gamma \rangle = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle + \langle \phi(\mathbf{x}), \Gamma \rangle + \langle \Gamma, \phi(\mathbf{x}') \rangle + \langle \Gamma, \Gamma \rangle$$

Restrict  $\Gamma$  to lie in the span of  $\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n) \in \mathcal{H}$ :  $\Gamma = \sum_i \alpha_i \phi(\mathbf{x}_i)$

$$\langle \tilde{\phi}(\mathbf{x}), \tilde{\phi}(\mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}') + \sum_{i=1}^n \alpha_i K(\mathbf{x}, \mathbf{x}_i) + \sum_{i=1}^n \alpha_i K(\mathbf{x}', \mathbf{x}_i) + \sum_{i,j=1}^n \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j).$$

Operations in  $\mathcal{H}$ 

- ① **Translation:** A translation in feature space can be written as the modified feature map  $\tilde{\phi}(\mathbf{x}) = \phi(\mathbf{x}) + \Gamma$  with  $\Gamma \in \mathcal{H}$ . Then:

$$\langle \phi(\mathbf{x}) + \Gamma, \phi(\mathbf{x}') + \Gamma \rangle = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle + \langle \phi(\mathbf{x}), \Gamma \rangle + \langle \Gamma, \phi(\mathbf{x}') \rangle + \langle \Gamma, \Gamma \rangle$$

Restrict  $\Gamma$  to lie in the span of  $\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n) \in \mathcal{H}$ :  $\Gamma = \sum_i \alpha_i \phi(\mathbf{x}_i)$

$$\langle \tilde{\phi}(\mathbf{x}), \tilde{\phi}(\mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}') + \sum_{i=1}^n \alpha_i K(\mathbf{x}, \mathbf{x}_i) + \sum_{i=1}^n \alpha_i K(\mathbf{x}', \mathbf{x}_i) + \sum_{i,j=1}^n \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j).$$

- ② **Centering:** The mean of the data is  $\phi_\mu = \frac{1}{n} \sum_{i=1}^n \phi(\mathbf{x}_i)$ , then

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) - \frac{2}{n} \sum_i K(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{n^2} \sum_{i,j} K(\mathbf{x}_i, \mathbf{x}_j)$$

Operations in  $\mathcal{H}$ 

- ① **Translation:** A translation in feature space can be written as the modified feature map  $\tilde{\phi}(\mathbf{x}) = \phi(\mathbf{x}) + \Gamma$  with  $\Gamma \in \mathcal{H}$ . Then:

$$\langle \phi(\mathbf{x}) + \Gamma, \phi(\mathbf{x}') + \Gamma \rangle = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle + \langle \phi(\mathbf{x}), \Gamma \rangle + \langle \Gamma, \phi(\mathbf{x}') \rangle + \langle \Gamma, \Gamma \rangle$$

Restrict  $\Gamma$  to lie in the span of  $\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n) \in \mathcal{H}$ :  $\Gamma = \sum_i \alpha_i \phi(\mathbf{x}_i)$

$$\langle \tilde{\phi}(\mathbf{x}), \tilde{\phi}(\mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}') + \sum_{i=1}^n \alpha_i K(\mathbf{x}, \mathbf{x}_i) + \sum_{i=1}^n \alpha_i K(\mathbf{x}', \mathbf{x}_i) + \sum_{i,j=1}^n \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j).$$

- ② **Centering:** The mean of the data is  $\phi_\mu = \frac{1}{n} \sum_{i=1}^n \phi(\mathbf{x}_i)$ , then

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) - \frac{2}{n} \sum_i K(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{n^2} \sum_{i,j} K(\mathbf{x}_i, \mathbf{x}_j)$$

- ③ **Normalizing:**

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = \left\langle \frac{\phi(\mathbf{x}_i)}{\|\phi(\mathbf{x}_i)\|}, \frac{\phi(\mathbf{x}_j)}{\|\phi(\mathbf{x}_j)\|} \right\rangle = \dots = \frac{K(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i)} \sqrt{K(\mathbf{x}_j, \mathbf{x}_j)}}$$

Operations in  $\mathcal{H}$ 

- ① **Translation:** A translation in feature space can be written as the modified feature map  $\tilde{\phi}(\mathbf{x}) = \phi(\mathbf{x}) + \Gamma$  with  $\Gamma \in \mathcal{H}$ . Then:

$$\langle \phi(\mathbf{x}) + \Gamma, \phi(\mathbf{x}') + \Gamma \rangle = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle + \langle \phi(\mathbf{x}), \Gamma \rangle + \langle \Gamma, \phi(\mathbf{x}') \rangle + \langle \Gamma, \Gamma \rangle$$

Restrict  $\Gamma$  to lie in the span of  $\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n) \in \mathcal{H}$ :  $\Gamma = \sum_i \alpha_i \phi(\mathbf{x}_i)$

$$\langle \tilde{\phi}(\mathbf{x}), \tilde{\phi}(\mathbf{x}') \rangle = K(\mathbf{x}, \mathbf{x}') + \sum_{i=1}^n \alpha_i K(\mathbf{x}, \mathbf{x}_i) + \sum_{i=1}^n \alpha_i K(\mathbf{x}', \mathbf{x}_i) + \sum_{i,j=1}^n \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j).$$

- ② **Centering:** The mean of the data is  $\phi_\mu = \frac{1}{n} \sum_{i=1}^n \phi(\mathbf{x}_i)$ , then

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_i, \mathbf{x}_j) - \frac{2}{n} \sum_i K(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{n^2} \sum_{i,j} K(\mathbf{x}_i, \mathbf{x}_j)$$

- ③ **Normalizing:**

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = \left\langle \frac{\phi(\mathbf{x}_i)}{\|\phi(\mathbf{x}_i)\|}, \frac{\phi(\mathbf{x}_j)}{\|\phi(\mathbf{x}_j)\|} \right\rangle = \dots = \frac{K(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i)} \sqrt{K(\mathbf{x}_j, \mathbf{x}_j)}}$$

- ④ **Computing Distances:**

$$d_{\mathcal{H}}(\mathbf{x}_i, \mathbf{x}_j) = \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|_{\mathcal{H}} = \sqrt{K(\mathbf{x}_i, \mathbf{x}_i) + K(\mathbf{x}_j, \mathbf{x}_j) - 2K(\mathbf{x}_i, \mathbf{x}_j)}$$

## Standard kernels and construction of kernels

OK, OK, but... what's the kernel  $K$ ?

**Valid kernels must be symmetric and positive definite similarity measures**

## Common kernels

- Linear:  $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j \rightarrow \phi(\mathbf{x}) = \mathbf{x}$
- Polynomial:  $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j + 1)^d \rightarrow \phi(\mathbf{x}) = \text{monomials}$
- Gaussian Function (RBF):  $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / (2\sigma^2)) \rightarrow \phi(\mathbf{x}) = ?$

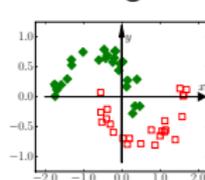
## Properties of Mercer's kernels

Let  $K_1$ ,  $K_2$  and  $K_3$  be valid Mercer's kernels over  $\mathcal{X} \times \mathcal{X}$ , with  $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^N$ , with  $\mathbf{A}$  being a symmetric positive semi-definite  $N \times N$  matrix, and  $\eta > 0$ . Then the following functions are valid kernels:

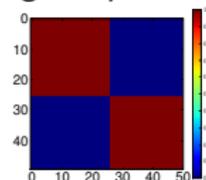
- ①  $K(\mathbf{x}_i, \mathbf{x}_j) = K_1(\mathbf{x}_i, \mathbf{x}_j) + K_2(\mathbf{x}_i, \mathbf{x}_j)$
- ②  $K(\mathbf{x}_i, \mathbf{x}_j) = K_1(\mathbf{x}_i, \mathbf{x}_j) \cdot K_2(\mathbf{x}_i, \mathbf{x}_j)$
- ③  $K(\mathbf{x}_i, \mathbf{x}_j) = \eta K_1(\mathbf{x}_i, \mathbf{x}_j)$

## Standard kernels and construction of kernels

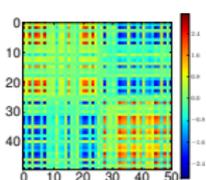
Choosing  $\sigma$  for the RBF kernel is critical, as it indicates the degree of shared information among training samples



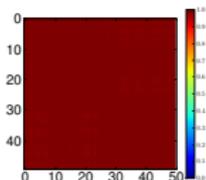
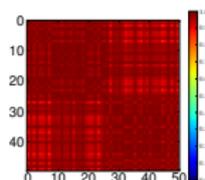
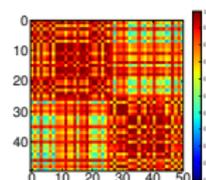
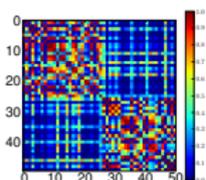
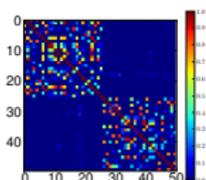
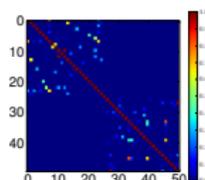
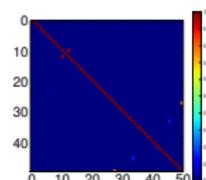
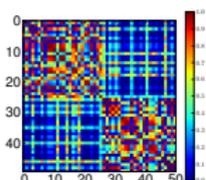
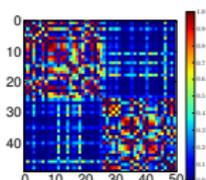
two moons



label "kernel"



linear

Gauss:  $\gamma = 0.001$  $\gamma = 0.01$  $\gamma = 0.1$  $\gamma = 1$  $\gamma = 10$  $\gamma = 100$  $\gamma = 1000$  $\gamma = 0.6$   
rule of thumb $\gamma = 1.6$   
5-fold CV

## The good, the bad, and the ideal kernel matrix

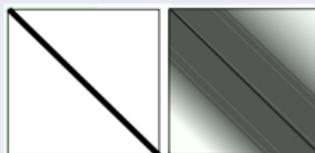
- **Bad kernel:** mostly diagonal, all points orthogonal to each other, no clusters, no structure.
- **Good kernel:** the kernel matrix should have clusters and structure.



- **Ideal kernel:**  $\mathbf{K}_{ideal} = \mathbf{y}\mathbf{y}^T$
- **Kernel alignment:**  $\min_{\theta} \{ \|\mathbf{K}(\mathbf{X}|\theta) - \mathbf{y}\mathbf{y}^T\|_F \}$

### The good, the bad, and the ideal kernel matrix

- **Bad kernel:** mostly diagonal, all points orthogonal to each other, no clusters, no structure.
- **Good kernel:** the kernel matrix should have clusters and structure.



- **Ideal kernel:**  $\mathbf{K}_{ideal} = \mathbf{y}\mathbf{y}^\top$
- **Kernel alignment:**  $\min_{\theta} \{ \|\mathbf{K}(\mathbf{X}|\theta) - \mathbf{y}\mathbf{y}^\top\|_F \}$

### Some ideas for designing your own kernel function

- 1 Put your favourite distance  $d$  in  $K = \exp(-d)$ . Done!
- 2 Take your favourite well-known nonlinear transform  $\Psi(\mathbf{x})$  and write:

$$K(\mathbf{x}, \mathbf{z}) = \Psi(\mathbf{x})^\top \Psi(\mathbf{z}) = \dots = f(\mathbf{x}^\top \mathbf{z})$$

- 3 Take millions of data, cluster them, and infer a metric
- 4 Combine all the previous kernels as you like: '×', '⊙', '+', ...

## Some facts...

- 1950: the theory of kernel functions is developed [Aronszajn50]
- 1960: linear functions used for classification
- 1970: proposed the representer theorem
- 1980: neural networks use it unconsciously
- 1990: Vapnik uses the theory to formalize nonlinear regularized machines
- 1995: Support vector machines excel in many applications
- 2000: Every single linear method is 'kernelized'
- 2010: How to adapt the kernel to your data characteristics

## Kernelization

Take your favorite linear algorithm expressed in dot products and do:

- 1 Replace  $\mathbf{X}$  by  $\Phi$
- 2 Exploit the reproducing property:  $\mathbf{W} = \Phi^\top \alpha$
- 3 Apply the kernel trick and replace:  $\mathbf{K} = \Phi\Phi^\top$
- 4 Your problem is a function of  $\mathbf{K}$  and your weights are now  $\alpha$
- 5 Use your favorite similarity measure to build  $\mathbf{K}$  ('kernelmatrix.m')

### Conclusions

- Given definition of kernel function, kernel mapping, and RKHS
- Analyzed kernel properties
- Studied how to build new kernels

## References

-  J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.
-  B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.
-  Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer, NY, 1995.
-  Vladimir Vapnik. *Statistical Learning Theory*. Wiley, NY, 1998.
-  Ralf Herbrich. *Learning Kernel Classifiers*. MIT Press, Cambridge, MA, 2002.
-  J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, *Least Squares Support Vector Machines*, World Scientific Pub. Co., Singapore, 2002 <http://www.esat.kuleuven.be/sista/lssvmlab/>
-  Tax, David M. J. *One-class classification*, Delft University of Technology, June, Delft, (2001). <http://prlab.tudelft.nl>
-  G. Camps-Valls, J. L. Rojo and M. Martinez, *Kernel Methods in Bioengineering, Signal and Image Processing*, Idea Inc., 2007.
-  Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. *Kernel methods in machine learning*. Ann. Statist. Volume 36, Number 3 (2008), 1171-1220.
-  Conferences: NIPS, ICML, ECML, COLT, ICANN, ESANN, MLSP
-  Webs: [videlectures.net](http://videlectures.net), <http://www.kernel-machines.org>