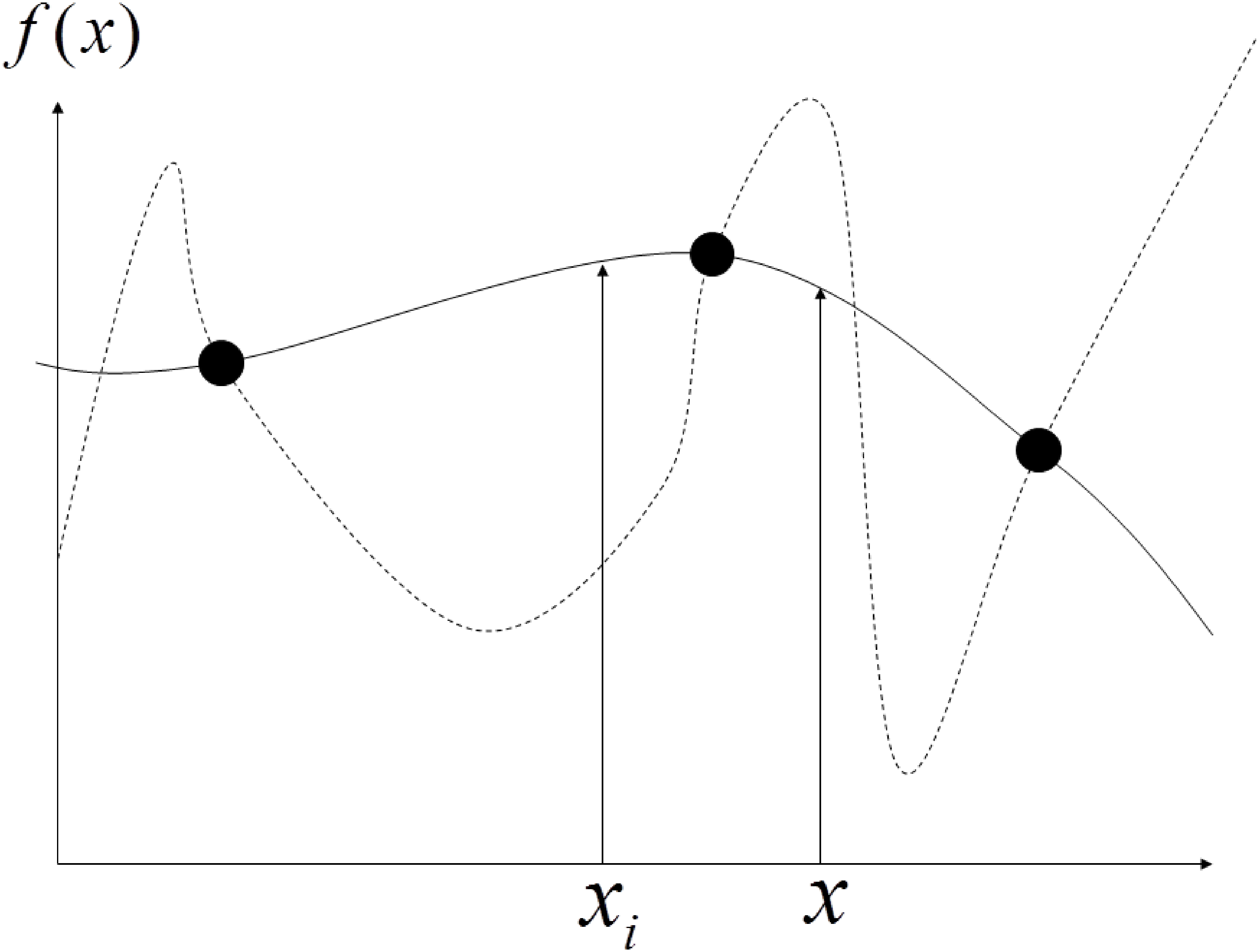
Kernels

Volker Tresp
Summer 2014

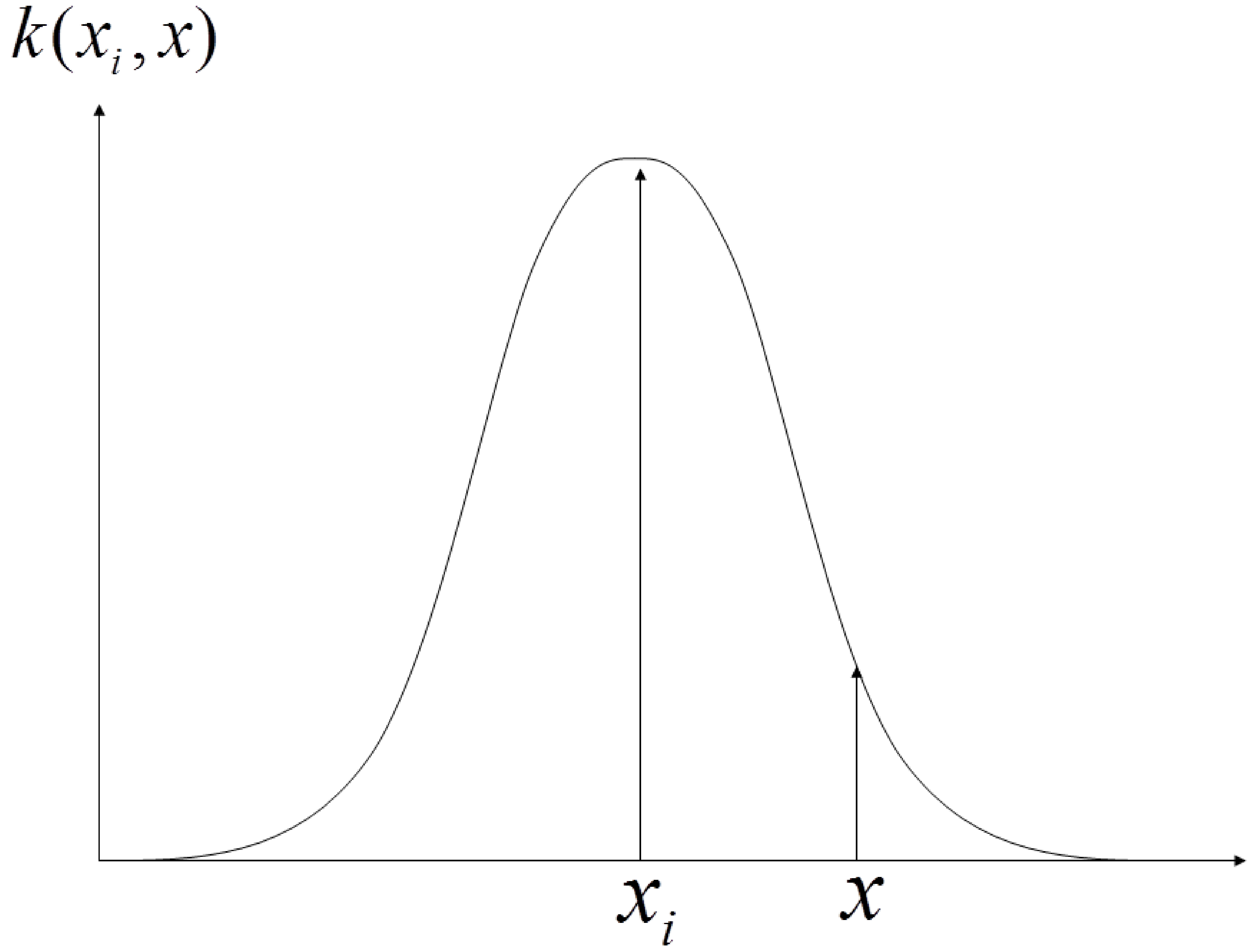
Smoothness Assumption

- So far we used prior knowledge to define the right basis functions: the assumption is that $f(x)$ can be approximated by a weighted sum of basis functions
- Alternatively, it might make sense to have a preference for smooth functions: functional values close in input space should have similar functional values
- In the figure it might make sense that the functional values at $\mathbf{x}_i$ and $\mathbf{x}$ are similar (smoothness assumption)
- Thus, one might prefer the smooth (continuous) function in favor of the dashed function


$f(x)$
x_i
x

Introduction Kernels

- One can implement smoothness assumptions over kernel functions
- A kernel function $k(\mathbf{x}_i, \mathbf{x})$ determines, how neighboring functional values are influenced when $f(\mathbf{x}_i)$ is given
- Example: Gaussian kernel

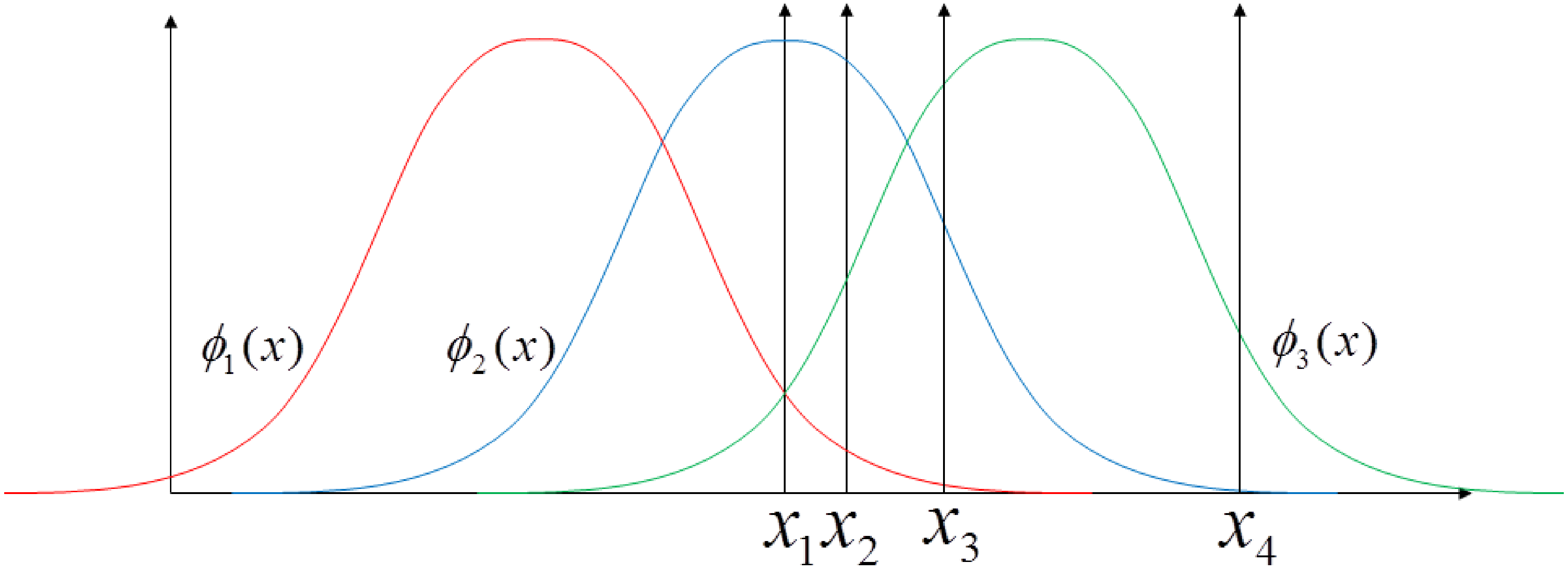

$k(x_i, x)$
x_i
x

Kernels and Basis Functions

- It turns out that there is a close relationship between kernels and basis functions:

$$k(\mathbf{x}_i, \mathbf{x}) = \sum_{j=1}^{M_\phi} \phi_j(\mathbf{x}_i)\phi_j(\mathbf{x})$$

- Thus: given the basis functions, this equation gives you the corresponding kernel
- We have encountered the kernel already in the discussion on basis functions. Note the kernel is a function that is represented with basis functions $\phi_j(\mathbf{x})$, that have the weight $\phi_j(\mathbf{x}_i)$.
- For positive definite kernels, we can also go the other way: given the kernels I can give you a corresponding set of basis functions (not unique)



$$\phi(x_1) = (0.25, 1.00, 0.25)^T \qquad k(x_1, x_1) = \phi^T(x_1)\phi(x_1) = 1.12$$

$$\phi(x_2) = (0.10, 0.90, 0.50)^T \qquad k(x_1, x_2) = \phi^T(x_1)\phi(x_2) = 1.05$$

$$\phi(x_3) = (0.02, 0.60, 0.90)^T \qquad k(x_1, x_3) = \phi^T(x_1)\phi(x_3) = 0.83$$

$$\phi(x_4) = (0.00, 0.01, 0.30)^T \qquad k(x_1, x_4) = \phi^T(x_1)\phi(x_4) = 0.08$$

Kernel Prediction

- Regression

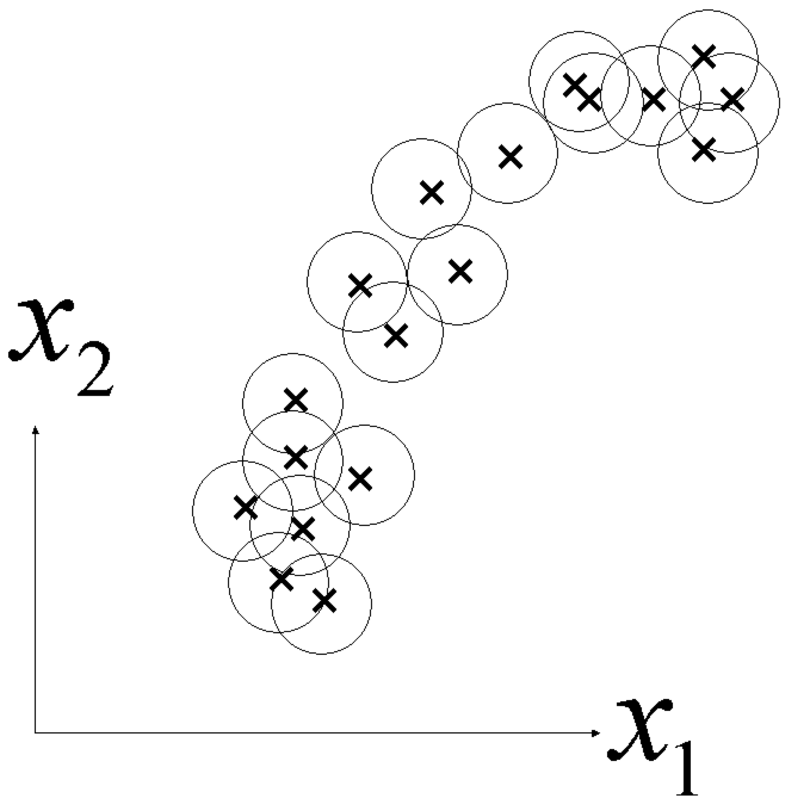
$$\hat{y}(\mathbf{z}) = \sum_{i=1}^{N} v_i k(\mathbf{z}, \mathbf{x}_i)$$

- Classification

$$\hat{y}(\mathbf{z}) = \text{sign}\left(\sum_{i=1}^{N} v_i k(\mathbf{z}, \mathbf{x}_i)\right)$$

- The solution contains as many kernels as there are data points N (independent on the number of underlying basis functions M)
- Thus: I can work with a finite number of kernels, instead of an infinite number of basis functions

One Kernel for Each Data Point



Different Forms of the Cost Function

- We start with the PLS cost function for models with basis functions
- Regularized cost function

$$\text{cost}^{pen}(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \sum_j w_j \phi_j(x_i))^2 + \lambda \sum_{i=0}^{M} w_i^2$$

$$= (\mathbf{y} - \mathbf{\Phi w})^T(\mathbf{y} - \mathbf{\Phi w}) + \lambda \mathbf{w}^T \mathbf{w}$$

where Φ is the design matrix design with $(\mathbf{\Phi})_{i,j} = \phi_j(\mathbf{x}_i)$.

Implicit Solution

- We calculate the first derivatives and set them to zero,

$$\frac{\partial \text{cost}^{pen}(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{\Phi}^T(\mathbf{y} - \mathbf{\Phi}\mathbf{w}) + 2\lambda\mathbf{w} = 0$$

  It follows that one can write,

$$\mathbf{w}_{pen} = \frac{1}{\lambda}\mathbf{\Phi}^T(\mathbf{y} - \mathbf{\Phi}\mathbf{w}_{pen})$$

Approach

- This is not an explicit solution ($\mathbf{w}_{pen}$ appears on both sides of the equation). But we know now, that we can write the solution as a linear combination of the input vectors

$$\mathbf{w}_{pen} = \mathbf{\Phi}^T \mathbf{v} = \sum_{i=1}^{N} v_i \vec{\phi}(\mathbf{x}_i)$$

- Note that we have a sum over N data points (and not M basis functions)

Kernel Model

- We immediately get,

$$f(\mathbf{x}) = \sum_{j=1}^{M_\phi} w_{j,pen}\phi_j(\mathbf{x}_i) = \vec{\phi}(\mathbf{x})^T \mathbf{w}_{pen}$$

$$= \vec{\phi}(\mathbf{x})^T \mathbf{\Phi}^T \mathbf{v} = \sum_{i=1}^{N} v_i k(\mathbf{x}, \mathbf{x}_i)$$

with $\mathbf{v} = (v_1, \ldots, v_N)^T$ and

$$k(\mathbf{x}, \mathbf{x}_i) = \vec{\phi}(\mathbf{x})^T \vec{\phi}(\mathbf{x}_i) = \sum_{k=1}^{M_\phi} \phi_k(\mathbf{x})\phi_k(\mathbf{x}_i)$$

A New Form of the Cost Function

- We can substitute the constraints, and obtain

$$\text{cost}^{pen}(\mathbf{v}) = (\mathbf{y} - \mathbf{\Phi}\mathbf{\Phi}^T\mathbf{v})^T(\mathbf{y} - \mathbf{\Phi}\mathbf{\Phi}^T\mathbf{v}) + \lambda\mathbf{v}^T\mathbf{\Phi}\mathbf{\Phi}^T\mathbf{v}$$

$$= (\mathbf{y} - K\mathbf{v})^T(\mathbf{y} - K\mathbf{v}) + \lambda\mathbf{v}^T K\mathbf{v}$$

where K is an $N \times N$ matrix with elements

$$k_{i,j} = \vec{\phi}(\mathbf{x}_i)^T\vec{\phi}(\mathbf{x}_j) = \sum_{k=1}^{M_\phi} \phi_k(\mathbf{x}_i)\phi_k(\mathbf{x}_j)$$

- An important result: **We can write the cost function, such that only inner**

products of the basis functions appear (i.e., the kernels), but not the basis functions themselves!

Kernel Parameters

- Now we can take the derivative of the cost function with respect to $\mathbf{v}$ (note, that $K = K^T$)

$$\frac{\partial \mathbf{cost}^{pen}(\mathbf{v})}{\partial \mathbf{v}} = 2K(\mathbf{y} - K\mathbf{v}) + 2\lambda K\mathbf{v}$$

such that

$$\mathbf{v}_{pen} = (K + \lambda I)^{-1}\mathbf{y}$$

Kernel Prediction

- A prediction can be written as

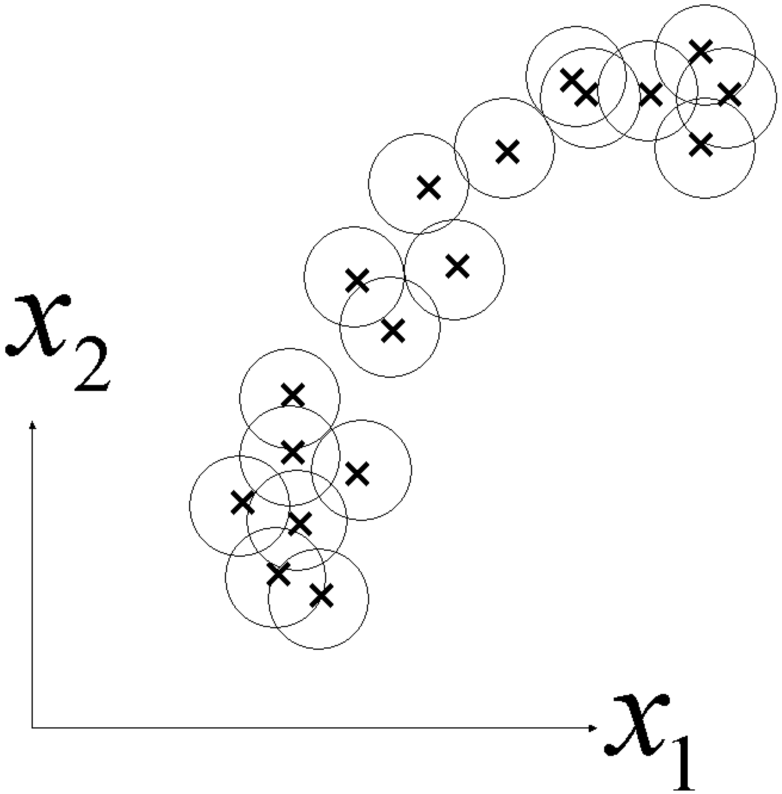
$$\hat{f}(\mathbf{z}) = \vec{\phi}(\mathbf{z})^T \mathbf{w} = \vec{\phi}(\mathbf{z})^T \mathbf{\Phi}^T \mathbf{v}_{pen} = \sum_{i=1}^{N} v_i k(\mathbf{z}, \mathbf{x}_i)$$

  with

$$k(\mathbf{z}, \mathbf{x}_i) = \vec{\phi}(\mathbf{z})^T \vec{\phi}(\mathbf{x_i})$$

- Another important result: we can write the solution such that only inner products are used; **the solution can be written as a weighted sum of N kernels**.

One Kernel for Each data Point

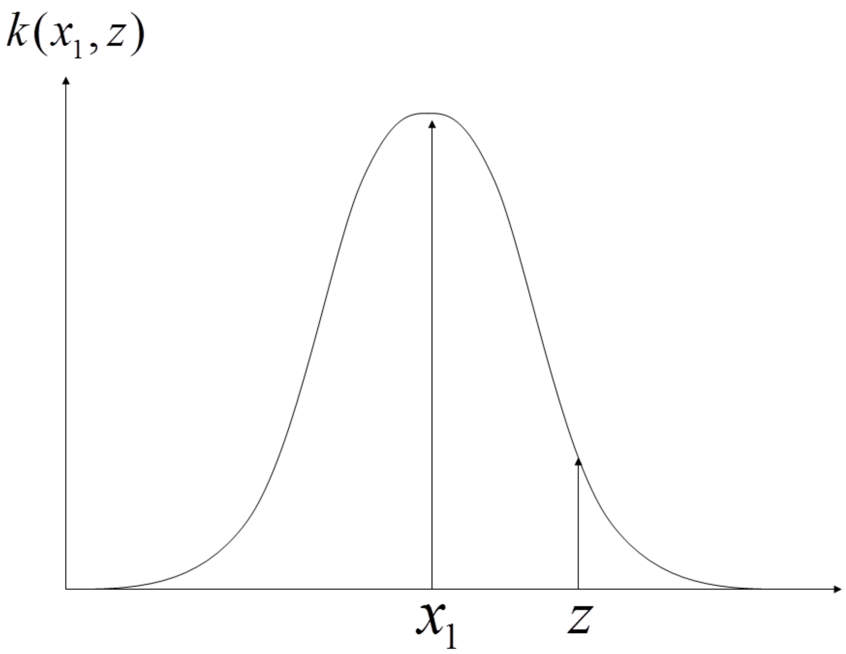


With only One Training Data Point

- With only one training data point we get

$$f(\mathbf{z}) = v_1 k(\mathbf{z}, \mathbf{x}_1)$$

- As discussed previously:



Comments and Interpretation of a Kernel

- This is interesting, since there can be more basis functions than data points; in particular this result is valid, even if we work with an **infinite number of basis functions**!
- It is even possible to start with the kernels, without knowing exactly, what the underlying basis functions are
- Different interpretations of the kernel
  - As inner product $k(\mathbf{x}_i, \mathbf{z}) = \vec{\phi}^T(\mathbf{x})\vec{\phi}(\mathbf{z})$
  - As covariance: how strong is the correlation of the functional values at different inputs $k(\mathbf{x}_i, \mathbf{z}) = cov(f(\mathbf{x}_i), f(\mathbf{z}))$
- When $N >> M$ it is computationally more efficient to work with basis functions (requiring $M^3 + M^2N$ operations). When $M >> N$, the kernel version is more efficient, requiring $N^3 + N^2M$ operations. If the kernels are known a priori (i.e., if they do not need to be calculates via inner product), the kernel solution requires N^3 operations.

- Still, not all functions are valid kernel functions. We need the following theorem ...

Mercer's Theorem

- (From Vapnik: The nature of statistical learning theory. Springer, 2000)
- *Mercer's Theorem:* To guarantee, that the symmetric functions $k(\mathbf{x}, \mathbf{z}) = k(\mathbf{z}, \mathbf{x})$ from L_2 permits an expansion as

$$k(\mathbf{x}, \mathbf{z}) = \sum_{h=1}^{\infty} \lambda_h \phi_h^T(\mathbf{x}) \phi_h(\mathbf{z})$$

with positive coefficients $\lambda_h > 0$, it is necessary and sufficient, that

$$\int \int k(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} > 0$$

for all $g \neq 0$, for which

$$\int g^2(\mathbf{x}) d\mathbf{x} < \infty$$

- The theorem says, that for so-called positive-definite kernels ("Mercer kernels"), a decomposition in basis functions is possible!

- Each kernel-matrix K is then also positive (semi-) definite

Kernel Design

- Linear Kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$$

Basis functions, as well as kernel functions, are linear

- Polynomial kernel (1)

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j)^d$$

The basis functions are all ordered polynomials of order d

- Polynomial kernel (2)

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + R)^d$$

The corresponding basis functions are all polynomials of order d **or smaller**

- Gauß-kernels (RBF-kernels)

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2s^2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$$

These kernels correspond to infinitely many Gaussian basis functions

- Sigmoid ("neural network") kernels

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathsf{sig}\left(\mathbf{x}_i^T \mathbf{x}_j\right)$$

Appendix: Detour on Function Spaces

Rewriting the Cost Function (see lecture on basis functions)

- The cost function

$$f(x) = \sum_i w_i \phi_i(x)$$

can be thought of as an inner product between the function $f(x') = \sum w_i \phi_i(x')$ and the function $k(x, x') = \sum \phi_i(x)\phi_i(x')$ in the space of the basis functions, thus

$$f(x) = \langle f, k_x \rangle_\phi$$

- $k(x, x')$ is our kernel and in this context is called is called a *reproducing kernel*.

- Also recall that $w^T w = \langle f, f \rangle_\Phi$

- With all of this, we can write our cost function as

$$\text{cost}^{pen}(\mathbf{w}) = \sum_{i=1}^{N} \left(y_i - \langle f, k_{x_i} \rangle_\Phi\right)^2 + \lambda \langle f, f \rangle_\Phi$$

- Note that only for the minimizer we can write also

$$\langle f, f \rangle_{\Phi} = v^T K v$$

Representer Theorem

- *Representer Theorem:* Let Ω be a strictly monotonously increasing function and let $\text{loss}()$ be an arbitrary loss function, then the minimizer of the loss function

$$\sum_{i=1}^{N} \text{loss}(y_i, f(\mathbf{x}_i)) + \Omega(\|f\|_\Phi)$$

  can be represented as

$$f(\mathbf{x}) = \sum_{i=1}^{N} v_i k(\mathbf{x}_i, \mathbf{x})$$

- $\|f\|_\Phi = \sqrt{\langle f, f \rangle_\Phi}$ is a norm in a *reproducing kernel Hilbert space* (RKHS)
- So kernel solutions are possible for all cost functions we are considering!