

# Kernel regression and gaussian processes

Course of Machine Learning  
Master Degree in Computer Science  
University of Rome "Tor Vergata"  
a.a. 2019-2020

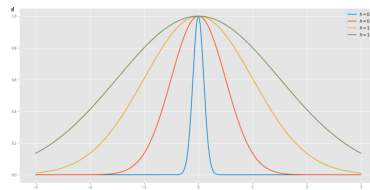
Giorgio Gambosi

## Kernel regression

- In kernel regression methods, the target value corresponding to any item  $\mathbf{x}$  is predicted by referring to items in the training set, and in particular to the items which are closer to  $\mathbf{x}$ .
- This is controlled by referring to a *kernel function*  $\kappa_h(\mathbf{x})$ , which is non zero only in an interval around 0
- $h$  is the *bandwidth* of the kernel, which controls the width of  $\kappa_h(\mathbf{x})$

A possible, common kernel, is the gaussian (or RBF) kernel

$$g(\mathbf{x}) = e^{-\frac{\|\mathbf{x}\|^2}{2h^2}}$$



## Kernel regression

In regression, we are interested in estimating the conditional expectation

$$f(\mathbf{x}) = E[t|\mathbf{x}] = \int t p(t|\mathbf{x}) dt = \int t \frac{p(\mathbf{x}, t)}{p(\mathbf{x})} dt = \frac{\int t p(\mathbf{x}, t) dt}{p(\mathbf{x})} = \frac{\int t p(\mathbf{x}, t) dt}{\int p(\mathbf{x}, t) dt}$$

The joint distribution  $p(\mathbf{x}, t)$  is approximated by means of a kernel function as

$$p(\mathbf{x}, t) \approx \frac{1}{n} \sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) \kappa_h(t - t_i)$$

## Kernel regression

This results into

$$f(\mathbf{x}) = \frac{\int t \frac{1}{n} \sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) \kappa_h(t - t_i) dt}{\int \frac{1}{n} \sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) \kappa_h(t - t_i) dt} = \frac{\sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) \int t \kappa_h(t - t_i) dy}{\sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) \int \kappa_h(t - t_i) dt}$$

and, since  $\int \kappa_h(t - t_i) dt = 1$  and  $\int t \kappa_h(t - t_i) dt = t_i$ , we get

$$f(\mathbf{x}) = \frac{\sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) t_i}{\sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i)}$$

### Kernel regression

By setting

$$w_i(\mathbf{x}) = \frac{\kappa_h(\mathbf{x} - \mathbf{x}_i)}{\sum_{j=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_j)}$$

we can write

$$f(\mathbf{x}) = \sum_{i=1}^n w_i(\mathbf{x}) t_i$$

that is, the predicted value is computed as a normalized linear combination of all target values, weighted by kernels (Nadaraya-Watson)

### Locally weighted regression

In Nadaraya-Watson model, the prediction is performed by means of a normalized weighted combination of constant values (target values in the training set).

Locally weighted regression (LOESS) improves that approach by referring to a weighted version of the sum of squared differences loss function used in regression.

If a value  $t$  has to be predicted for an item  $\mathbf{x}$ , a "local" version of the loss function is considered, with weight  $\mathbf{w}_i$  dependent from the "distance" between  $\mathbf{x}$  and  $\mathbf{x}_i$ .

$$L(\mathbf{x}) = \sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) (\mathbf{w}^T \bar{\mathbf{x}}_i - t_i)^2$$

### Locally weighted regression

The minimization of this loss function

$$\hat{\mathbf{w}}(\mathbf{x}) = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) (\mathbf{w}^T \bar{\mathbf{x}}_i - t_i)^2$$

has solution

$$\hat{\mathbf{w}}(\mathbf{x}) = (\bar{\mathbf{X}}^T \Psi(\mathbf{x}) \bar{\mathbf{X}})^{-1} \bar{\mathbf{X}}^T \Psi(\mathbf{x}) \mathbf{t}$$

where  $\Psi(\mathbf{x})$  is a diagonal  $n \times n$  matrix with  $\Psi(\mathbf{x})_{ii} = \kappa_h(\mathbf{x} - \mathbf{x}_i)$ .

The prediction is then performed as usual, as

$$y = \hat{\mathbf{w}}(\mathbf{x})^T \bar{\mathbf{x}}$$

### Local logistic regression

The same approach applied in the case of local regression can be applied for classification, by defining a weighted loss function to be minimized, with weights dependent from the item whose target must be predicted.

In this case, a weighted version of the cross entropy function is considered, which has to be maximized

$$L(\mathbf{x}) = \sum_{i=1}^n \kappa_h(\mathbf{x} - \mathbf{x}_i) (t_i \log p_i - (1 - t_i) \log(1 - p_i))$$

with  $p_i = \sigma(\mathbf{w}^T \bar{\mathbf{x}}_i)$ , as usual.

The loss function minimization can be performed, for example, by applying a suitable modification of the IRLS algorithm for logistic regression

### Recap: some properties of Gaussian distribution

In order to introduce Gaussian processes and how they can be exploited for regression, let us first provide a short reminder on some properties of multivariate gaussian distributions.

Let  $\mathbf{x} = (x_1, \dots, x_n)^T$  be a random vector with gaussian distribution  $p(\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$  and let  $\mathbf{x} = (\mathbf{x}_A, \mathbf{x}_B)$  be a partition of the components  $\mathbf{x}$  such that:

- $\mathbf{x}_A = (x_1, \dots, x_r)^T$
- $\mathbf{x}_B = (x_{r+1}, \dots, x_n)^T$

Then, the *marginal* densities  $p(\mathbf{x}_A)$  and  $p(\mathbf{x}_B)$  are both gaussian with means  $\boldsymbol{\mu}_A, \boldsymbol{\mu}_B$  and covariance matrices  $\Sigma_A, \Sigma_B$  which can be derived from  $\boldsymbol{\mu}, \Sigma$  by observing that

$$\boldsymbol{\mu} = (\boldsymbol{\mu}_A, \boldsymbol{\mu}_B)^T \quad \Sigma = \begin{pmatrix} \Sigma_A & \Sigma_{AB} \\ \Sigma_{AB}^T & \Sigma_B \end{pmatrix}$$

### Recap: some properties of the Gaussian distribution

In the same situation, the conditional densities  $p(\mathbf{x}_A|\mathbf{x}_B)$  and  $p(\mathbf{x}_B|\mathbf{x}_A)$  are also gaussian with means

$$\begin{aligned} \boldsymbol{\mu}_{A|B} &= \boldsymbol{\mu}_A + \Sigma_{AB}\Sigma_B^{-1}(\mathbf{x}_B - \boldsymbol{\mu}_B) \\ \boldsymbol{\mu}_{B|A} &= \boldsymbol{\mu}_B + \Sigma_{BA}\Sigma_A^{-1}(\mathbf{x}_A - \boldsymbol{\mu}_A) \end{aligned}$$

and covariance matrices

$$\begin{aligned} \Sigma_{A|B} &= \Sigma_A - \Sigma_{AB}\Sigma_B^{-1}\Sigma_{BA} \\ \Sigma_{B|A} &= \Sigma_B - \Sigma_{BA}\Sigma_A^{-1}\Sigma_{AB} \end{aligned}$$

### Gaussian processes

- Multivariate gaussians on random vectors are useful for modeling finite collections of real-valued variables. They have nice analytical properties (see previous slides).
- Gaussian processes: extension of multivariate gaussians to infinite-sized collections of real-valued variables.
- We may think of gaussian processes as distributions not just over random vectors but over random real functions.

### Probability distributions over functions with finite domains

Let us first consider the case of functions defined over finite vectors.

- Let  $\chi = (\mathbf{x}_1, \dots, \mathbf{x}_m)$  be any finite vector, and let  $\mathcal{H}$  be the set of functions  $f : \chi \mapsto \mathbb{R}$ :  $f$  assigns a value  $f(\mathbf{x}_i)$  to each  $\mathbf{x}_i \in \chi$ 
  - A function  $f \in \mathcal{H}$  can be described by the vector  $(f(\mathbf{x}_1), \dots, f(\mathbf{x}_m))$
  - Any vector  $(y_1, \dots, y_m)$  can be seen as the description of a function  $f \in \mathcal{H}$  such that  $f(\mathbf{x}_i) = y_i$
  - The set  $\mathcal{H}$  is then in 1-to-1 correspondence with the set of vectors in  $\mathbb{R}^m$
- A probability distribution  $p(\mathbf{x}), \mathbf{x} \in \mathbb{R}^m$  over  $m$ -dimensional real vectors is also a distribution  $p(f), f \in \mathcal{H}$  over functions from  $\mathbb{R}^m$  to  $\mathbb{R}$

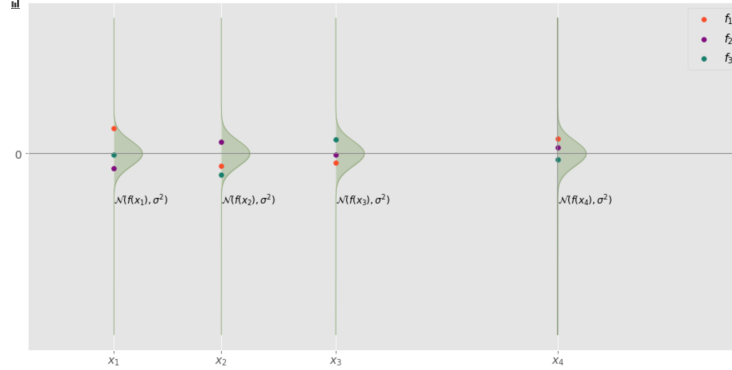
### Gaussian distributions over functions with finite domains

Assume that  $p(\mathbf{x})$  (or, equivalently,  $p(f)$ ) is a (multivariate,  $m$ -dimensional) Gaussian distribution centered on  $\mathbf{0}$  and with diagonal covariance  $\sigma^2 \mathbf{I}$ , that is

$$p(f|\sigma^2) = \mathcal{N}(f|\mathbf{0}, \sigma^2 \mathbf{I}) = \prod_{i=1}^m \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{f(\mathbf{x}_i)^2}{2\sigma^2}}$$

- This is equivalent to assuming that each function value  $f(\mathbf{x}_i)$  has normal distribution with mean 0 and variance  $\sigma^2$ , and that items are independent
- A dependence between function values at different points could be modeled through a non-diagonal covariance matrix

### Gaussian distributions over functions with finite domains



We may consider  $p(f|\sigma^2)$  as a *prior* distribution of functions, with respect to the observation of the value  $t_j$  actually taken by any variable  $\mathbf{x}_j$ ,  $1 \leq j \leq m$ .

### Gaussian distributions over functions with finite domains

- Assume now that for some subset  $\mathbf{X} = \{\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}\}$  of component indices, the corresponding targets  $\mathbf{t} = \{t_{i_1}, \dots, t_{i_k}\}$  are available.
- the posterior distribution  $p(f|\mathbf{X}, \mathbf{t})$  of functions (wrt to  $\mathbf{X}, \mathbf{t}$ ) can be defined and derived according to Bayes' rule, provided a likelihood model is defined

$$p(\mathbf{X}, \mathbf{t}|f) = \prod_{\mathbf{x}_i \in \mathbf{X}} p(\mathbf{x}_i, t_i|f) = \prod_{\mathbf{x}_i \in \mathbf{X}} p(t_i|\mathbf{x}_i, f)p(\mathbf{x}_i|f) \propto \prod_{\mathbf{x}_i \in \mathbf{X}} p(t_i|\mathbf{x}_i, f)$$

- for example, we could assume the usual likelihood  $p(t|\mathbf{x}, f, \beta) = \mathcal{N}(t|f(\mathbf{x}), \beta)$ , which implies

$$p(\mathbf{X}, \mathbf{t}|f, \beta) \propto \prod_{\mathbf{x}_i \in \mathbf{X}} \mathcal{N}(t_i|f(\mathbf{x}_i), \beta)$$

- the posterior distribution then would be

$$p(f|\mathbf{X}, \mathbf{t}, \beta, \sigma^2) \propto \prod_{\mathbf{x}_i \in \mathbf{X}} \mathcal{N}(t_i|f(\mathbf{x}_i), \beta)p(f|\sigma^2)$$

### Gaussian distributions over functions with finite domains

Since both the prior and the posterior distributions of  $f$  are gaussian, the predictive distribution

$$p(t|\mathbf{x}, \mathbf{X}, \mathbf{t}, \beta, \sigma^2) = \int p(t|\mathbf{x}, f, \beta)p(f|\mathbf{X}, \mathbf{t}, \beta, \sigma^2)df$$

is itself a gaussian.

That would be the case also in the more general case when some dependency between function points is assumed. In this case, a general covariance matrix is defined for the prior distribution

$$p(f|\Sigma) = \mathcal{N}(f|\mathbf{0}, \Sigma)$$

### Gaussian distributions over functions with infinite domains

- In the case of infinite  $\chi$ , we have to deal with an infinite collection of random variables.
- In this case, the role of multidimensional distributions is covered by stochastic processes.
  - A *stochastic process* is a collection of random variables,  $\{f(\mathbf{x}) : \mathbf{x} \in \chi\}$ , indexed by elements from some set  $\chi$ , known as the index set.
- A *Gaussian process* is a stochastic process such that for any finite subset  $\mathbf{x}_1, \dots, \mathbf{x}_n$  of  $\chi$ , the function values  $f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)$  have joint multivariate Gaussian distribution

### Gaussian processes

- In order to define a Gaussian process, both a mean and a covariance function must be defined.
  - a mean function  $m : \mathbf{R}^d \mapsto \mathbf{R}$  mapping each point  $\mathbf{x}_i \in \chi$  to the expectation

$$m(\mathbf{x}_i) = E_f[f(\mathbf{x}_i)]$$

of  $f(x)$  over all functions  $f$

- a covariance function  $\kappa : \mathbf{R}^d \times \mathbf{R}^d \mapsto \mathbf{R}$  mapping each pair of variables  $(\mathbf{x}_i, \mathbf{x}_j) \in \chi^2$  to the covariance

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = E_f[(f(\mathbf{x}_i) - m(\mathbf{x}_i))^2(f(\mathbf{x}_j) - m(\mathbf{x}_j))^2]$$

of  $f(x_j)$  and  $f(x_i)$  over all functions  $f$ .

### Kernels in gaussian processes

- The covariance function  $\kappa$  is assumed to be a positive definite (Mercer) kernel.
- This means that for any set of distinct points  $\mathbf{x}_1, \dots, \mathbf{x}_n$  it must be

$$\sum_{i=1}^n \sum_{j=1}^n c_i c_j \kappa(\mathbf{x}_i, \mathbf{x}_j) > 0$$

for any choice of the constants  $c_1, \dots, c_n$  such that not all  $c_i$  are equal to 0.

- Equivalently, the square *Gram* matrix  $G$  defined as

$$G = \begin{pmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \kappa(\mathbf{x}_1, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_n) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) & \kappa(\mathbf{x}_2, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_2, \mathbf{x}_n) \\ \cdots & \cdots & \cdots & \cdots \\ \kappa(\mathbf{x}_n, \mathbf{x}_1) & \kappa(\mathbf{x}_n, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_n, \mathbf{x}_n) \end{pmatrix}$$

must have positive eigenvalues.

- A collection of positive definite kernels is known in the literature and can be constructed by applying suitable rules.

### Gaussian processes

Given a gaussian process  $p(f) = \mathcal{GP}(m, \kappa)$ , then for any set of items  $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$ , the distribution of  $f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)$  is a gaussian

$$(f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)) \sim \mathcal{N}(\boldsymbol{\mu}(\mathbf{X}) | \Sigma(\mathbf{X}))$$

where

- $\boldsymbol{\mu}(\mathbf{X}) = (m(\mathbf{x}_1), \dots, m(\mathbf{x}_n))^T$
- $\Sigma(\mathbf{X})$  is the Gram matrix wrt  $\mathbf{x}_1, \dots, \mathbf{x}_n$  of a kernel function  $\kappa(\mathbf{x}, \mathbf{x}')$

The mean vector - at least initially, with no information from data - is usually assumed to be  $\mathbf{0}$ : different processes are then characterized only by their covariance kernel  $\kappa$ .

### Sampling functions from gaussian processes

- For any finite subset  $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$  of  $\chi$  it is possible to sample from  $p(f)$  the values of  $f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)$  by sampling from  $\mathcal{N}(f | \mathbf{0}, \Sigma(\mathbf{X}))$ , where, as stated before

$$\begin{aligned}\boldsymbol{\mu}(\mathbf{X})_i &= m(\mathbf{x}_i) \\ \Sigma(\mathbf{X})_{ij} &= \kappa(\mathbf{x}_i, \mathbf{x}_j)\end{aligned}$$

### RBF kernel

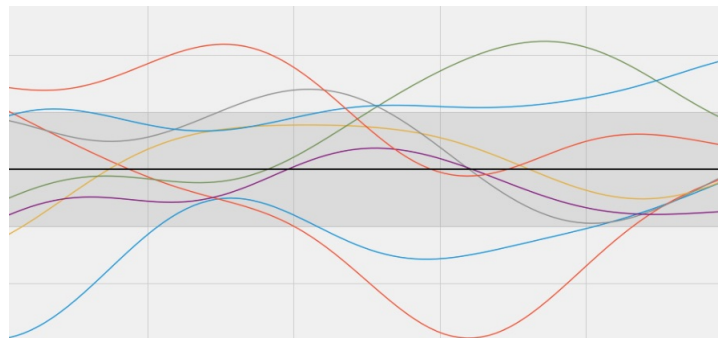
- One of the most applied kernel is the RBF kernel

$$\kappa(\mathbf{x}_1, \mathbf{x}_2) = \sigma^2 e^{-\frac{\|\mathbf{x}_1 - \mathbf{x}_2\|^2}{2\tau^2}}$$

which tends to assign higher covariance between  $f(\mathbf{x}_1)$  and  $f(\mathbf{x}_2)$  if  $\mathbf{x}_1$  and  $\mathbf{x}_2$  are nearby points.

- Functions drawn from a Gaussian process with RBF kernel tend to be smooth (values computed for nearby points tend to have similar values). Smoothing is larger for larger  $\tau$ .

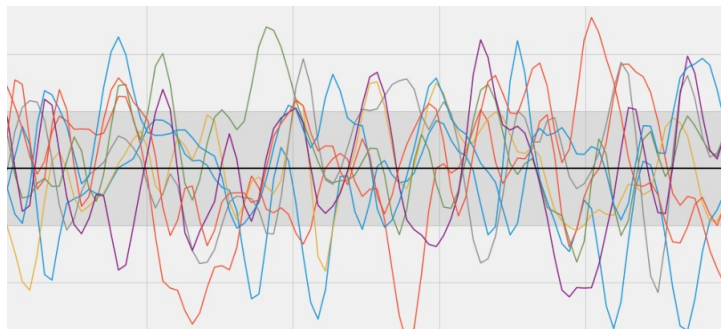
### RBF kernel



Larger smoothing

### RBF kernel

Smaller smoothing



### Gaussian process regression: no noise

Based on gaussian marginalization properties

- Given a gaussian process  $\mathcal{GP}(m, \kappa)$ , assume  $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T$  is a set of points such that  $\mathbf{f} = (f(\mathbf{x}_1), \dots, f(\mathbf{x}_n))^T$  are known
- that is, we assume that the target values in the training set correspond exactly to the function value  $t_i = f(\mathbf{x}_i)$ . Note that in the probabilistic model of regression this is not true, since a (gaussian) error is assumed

### Gaussian process regression: no noise

- By the gaussian process definition,  $\mathbf{f}$  is distributed as a multivariate gaussian with known mean  $\boldsymbol{\mu}(\mathbf{X}) = (m(\mathbf{x}_1), \dots, m(\mathbf{x}_n))^T$  and covariance matrix  $\Sigma(\mathbf{X})$ , defined by  $\kappa$
- Given a test point  $\mathbf{x}$ , the set  $\mathbf{X} \cup \{\mathbf{x}\} = (\mathbf{x}_1, \dots, \mathbf{x}_n, \mathbf{x})^T$  is a set of points such that  $(f(\mathbf{x}_1), \dots, f(\mathbf{x}_n), f(\mathbf{x}))^T$  is again a multivariate gaussian with known mean and covariance. This implies that  $f(\mathbf{x}|\mathbf{X}, \mathbf{f})$  is gaussian, with

$$m_p(\mathbf{x}|\mathbf{X}, \mathbf{f}) = m(\mathbf{x}) + \Sigma(\mathbf{x}, \mathbf{X})\Sigma(\mathbf{X})^{-1}(\mathbf{f} - \boldsymbol{\mu}(\mathbf{X}))$$

$$\sigma^2 = \kappa_p(\mathbf{x}, \mathbf{x}) - \Sigma(\mathbf{x}, \mathbf{X})\Sigma(\mathbf{X})^{-1}\Sigma(\mathbf{x}, \mathbf{X})^T$$

where  $\Sigma(\mathbf{x}, \mathbf{X}) = (\kappa(\mathbf{x}, \mathbf{x}_1), \dots, \kappa(\mathbf{x}, \mathbf{x}_n))$

### Gaussian process regression: no noise

In this case, an *interpolation* of the given values is performed:  $f(\mathbf{x}_i) = t_i$  for all possible functions, sampled from  $f(\mathbf{x}|\mathbf{X}, \mathbf{f})$ .

In fact, for all  $\mathbf{x}_i \in \mathbf{X}$ ,

- $\Sigma(\mathbf{x}_i, \mathbf{X})$  is the  $i$ -th row of  $\Sigma(\mathbf{X})$ : since  $\Sigma(\mathbf{X})\Sigma(\mathbf{X})^{-1} = \mathbf{I}$  by definition, the product  $\Sigma(\mathbf{x}_i, \mathbf{X})\Sigma(\mathbf{X})^{-1}$  must be equal to the  $i$ -th row of  $\mathbf{I}$ , that is an array  $C_i$  such that  $C_i[i] = 1$  and  $C_i[j] = 0$  otherwise
- as a consequence,

$$m(\mathbf{x}_i|\mathbf{X}, \mathbf{f}) = m(\mathbf{x}_i) + (t_i - m(\mathbf{x}_i)) = t_i$$

$$\sigma^2 = \kappa(\mathbf{x}_i, \mathbf{x}_i) - \kappa(\mathbf{x}_i, \mathbf{x}_i) = 0$$

### Gaussian process regression: noise

In this case, we assume, as usual, that  $p(t_i|f, \mathbf{x}_i) = \mathcal{N}(f(\mathbf{x}_i), \sigma_f^2)$

That is, the value  $t_i$  observed for variable  $\mathbf{x}_i$  differs from the one obtained as  $f(\mathbf{x}_i)$  by a gaussian and independent noise

$$t_i = f(\mathbf{x}_i) + \varepsilon \quad p(\varepsilon) = \mathcal{N}(\varepsilon|0, \sigma_f^2)$$

### Gaussian process regression

By the assumption of gaussian noise

$$t_i = f(\mathbf{x}_i) + \varepsilon \quad i = 1, \dots, m$$

$$t = f(\mathbf{x}) + \varepsilon$$

with  $p(\varepsilon) = \mathcal{N}(\varepsilon|0, \sigma^2)$ , it derives that

$$p(t_1, \dots, t_m, t|\mathbf{x}, \mathbf{X}; \sigma^2) = p(\mathbf{t}, t|\mathbf{x}, \mathbf{X}; \sigma^2)$$

$$= p(f(\mathbf{x}_1) + \varepsilon, \dots, f(\mathbf{x}_m) + \varepsilon, f(\mathbf{x}) + \varepsilon)$$

$$= \mathcal{N}(\mathbf{0}, \Sigma(\mathbf{X} \cup \{\mathbf{x}\}) + \sigma^2 \mathbf{I})$$

### Gaussian process regression

- It is possible to show that the posterior distribution of functions is again a gaussian process, with different mean and covariance functions

$$p(f|\mathbf{X}, \mathbf{t}) = \mathcal{GP}(m_p, \kappa_p)$$

where

$$m_p(\mathbf{x}) = \Sigma(\mathbf{x}, \mathbf{X})(\Sigma(\mathbf{X}) + \sigma_f^2 \mathbf{I})^{-1} \mathbf{t}$$

$$\kappa_p(\mathbf{x}, \mathbf{x}') = \kappa(\mathbf{x}, \mathbf{x}') - \Sigma(\mathbf{x}, \mathbf{X})(\Sigma(\mathbf{X}) + \sigma_f^2 \mathbf{I})^{-1} \Sigma(\mathbf{x}', \mathbf{X})^T$$

### Gaussian process regression

By the properties of gaussian distributions wrt to conditioning, it results

$$p(t|\mathbf{x}, \mathbf{X}, \mathbf{t}) = \mathcal{N}(y|\bar{\mu}(\mathbf{x}), \bar{\sigma}^2(\mathbf{x}))$$

with

$$\bar{\mu}(\mathbf{x}) = k(\mathbf{x}, \mathbf{X})(\Sigma(\mathbf{X}) + \sigma_f^2 \mathbf{I})^{-1} \mathbf{t} = \sum_{i=1}^m \alpha(\mathbf{x}_i, \mathbf{x}) t_i$$

$$\bar{\sigma}^2(\mathbf{x}) = \kappa(\mathbf{x}, \mathbf{x}) + \sigma^2 - k(\mathbf{x}, \mathbf{X})(\Sigma(\mathbf{X}) + \sigma_f^2 \mathbf{I})^{-1} k(\mathbf{x}, \mathbf{X})^T$$

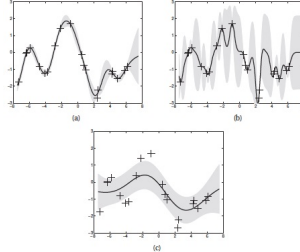
### Estimating kernel parameters

The predictive performance of gaussian processes depends exclusively on the suitability of the chosen kernel. Let us consider the case of an RBF kernel. Then,

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 e^{-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M}(\mathbf{x}_i - \mathbf{x}_j)} + \sigma_y^2 \delta_{ij}$$

$\mathbf{M}$  can be defined in several ways: the simplest one is  $\mathbf{M} = l^{-2} \mathbf{I}$ .

Even in this simple case, varying the values of  $\sigma_f, \sigma_y, l$  returns quite different results.



(figure from K.Murphy "Machine learning: a probabilistic perspective" p. 519, with  $(l, \sigma_f, \sigma_y)$  equal to  $(1, 1, 0.1)$ ,  $(0.3, 1.08, 0.00005)$ ,  $(3.0, 1.16, 0.89)$ )

### Estimating kernel parameters

Kernel parameters can be estimated, as usual, through grid search and (cross-)validation.

A different, more efficient approach relies on maximizing the marginal likelihood

$$p(\mathbf{t}|\mathbf{X}) = \int p(\mathbf{t}|\mathbf{f}, \mathbf{X}) p(\mathbf{f}|\mathbf{X}) d\mathbf{f} = \int \mathcal{N}(\mathbf{f}|\mathbf{0}, \Sigma(\mathbf{X})) \prod_{i=1}^m \mathcal{N}(t_i|f_i, \sigma_t^2) d\mathbf{f}$$

It can be shown that

$$\log p(\mathbf{t}|\mathbf{X}) = \log \mathcal{N}(\mathbf{t}|\mathbf{0}, \Sigma(\mathbf{X}) + \sigma_t^2 \mathbf{I})$$

$$= -\frac{1}{2} \mathbf{y}^T (\Sigma(\mathbf{X}) + \sigma_t^2 \mathbf{I})^{-1} \mathbf{t} - \frac{1}{2} \log |\Sigma(\mathbf{X})| + \sigma_t^2 \mathbf{I} - \frac{n}{2} \log(2\pi)$$

where the first term measures the fitting of the model to data, the second the complexity of the model, and the third + fourth ones are constants