

# **STA 414/2104: Machine Learning**

Russ Salakhutdinov

Department of Computer Science

Department of Statistics

[rsalaku@cs.toronto.edu](mailto:rsalaku@cs.toronto.edu)

<http://www.cs.toronto.edu/~rsalaku/>

## Lecture 6

# Gaussian Processes

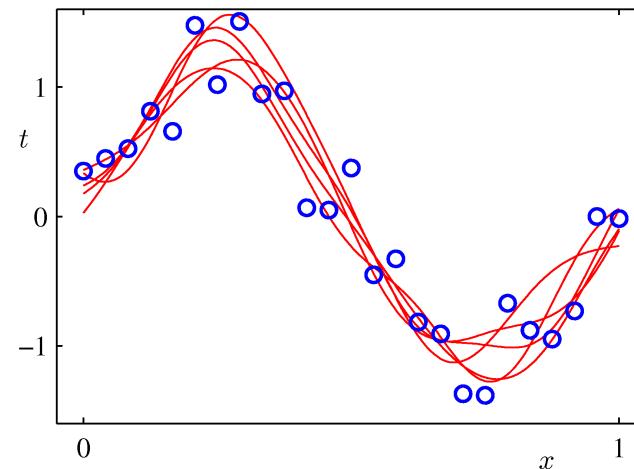
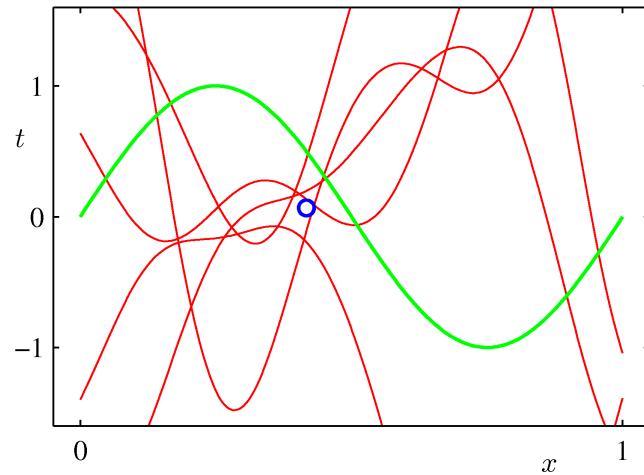
- So far, we have considered linear regression models of the form:

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \phi(\mathbf{x})$$

where  $\mathbf{w}$  is a vector of parameters and  $\phi(\mathbf{x})$  is a vector of fixed nonlinear basis functions.

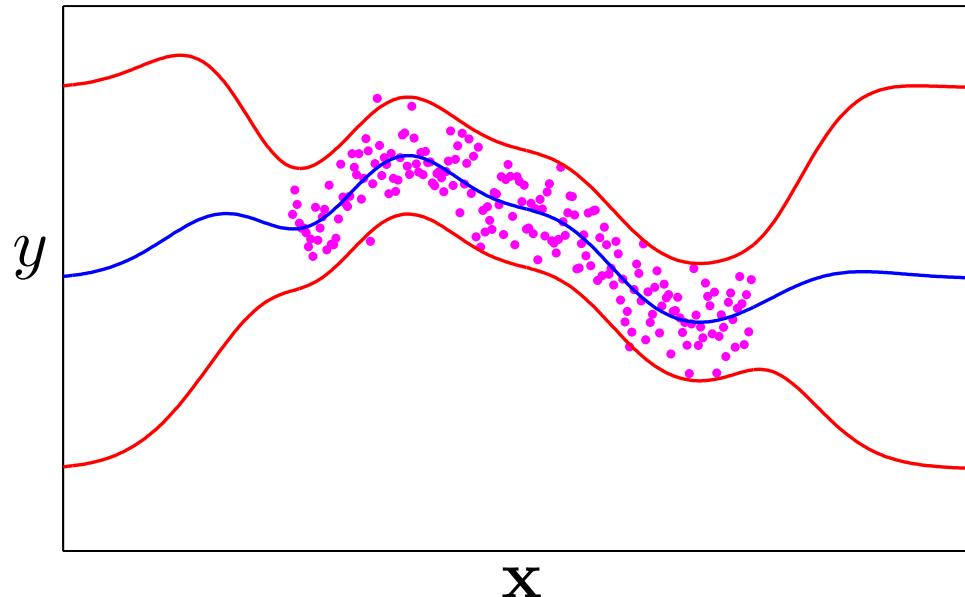
- A prior distribution over  $\mathbf{w}$  induces a prior distribution over functions  $f(\mathbf{x}, \mathbf{w})$ .
- Given a training dataset, we compute the posterior distribution over  $\mathbf{w}$ , which induces a posterior distribution over functions  $f(\mathbf{x}, \mathbf{w})$ .

Samples from the posterior



# Gaussian Processes

- You want to learn a function  $f$  with error bars from data  $D$ .



- A Gaussian process defines a distribution over functions  $p(f)$  which can be used for Bayesian regression:

$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})}$$

# Gaussian Processes

- In the Gaussian process viewpoint, we define a prior probability distributions over functions directly.
- May seem difficult: How can we define a distribution over the uncountably infinite space of functions?
- **Insight:** for a finite training set, we only need to consider the values of the functions at discrete set of input values  $x_n$ .
- Hence in practice, we work in a **finite space**.
- Many related models: In geostatistics literature, GP regression is known as kriging. See also a recent book on GPs by Rasmussen & Williams (2006).

# Linear Regression Revisited

- Consider the following linear model, defined in terms of **M** linear combinations of fixed basis functions:

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$$

- We place a Gaussian prior over model parameters:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w} | \mathbf{0}, \alpha^{-1} \mathbf{I})$$

- For any given fixed value of  $\mathbf{w}$ , we have a corresponding linear function. A probability distribution over  $\mathbf{w}$  defines a **probability distribution over functions**.

- Given a dataset  $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ , we will denote the values of the function as  $\mathbf{f} = [f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_N)]^T$ .

- Hence:

$$\mathbf{f} = \Phi \mathbf{w}.$$

N by M Design matrix

M by 1 vector of model parameters

# Linear Regression Revisited

$$\mathbf{f} = \Phi \mathbf{w}, \quad p(\mathbf{w}) = \mathcal{N}(\mathbf{w} | \mathbf{0}, \alpha^{-1} \mathbf{I})$$

- Observe that  $\mathbf{y}$  is a linear combination of Gaussian random variables, and hence is **itself Gaussian**:

$$\mathbb{E}[\mathbf{f}] = \Phi \mathbb{E}[\mathbf{w}] = 0$$

$$\text{cov}(\mathbf{f}) = \mathbb{E}[\mathbf{f}\mathbf{f}^T] = \Phi \mathbb{E}[\mathbf{w}\mathbf{w}^T] \Phi^T = \frac{1}{\alpha} \Phi \Phi^T = \mathbf{K}$$

Here,  $\mathbf{K}$  is known as the **Gramm matrix** with elements:

$$\mathbf{K}_{nm} = k(\mathbf{x}_n, \mathbf{x}_m) = \frac{1}{\alpha} \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m),$$

where  $k(\mathbf{x}, \mathbf{x}')$  is the **kernel function**.

- This model provides a **particular example of a Gaussian process**.

# Gaussian Process

- A Gaussian process (GP) is a **random function**  $\mathbf{f}: \mathbf{X} \rightarrow \mathbb{R}$ , such that **for any finite set** of input points  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ ,

$$\begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_N) \end{bmatrix} \sim \mathcal{N}\left( \begin{bmatrix} m(\mathbf{x}_1) \\ \vdots \\ m(\mathbf{x}_N) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \right)$$

where the parameters are the **mean function**  $m(\mathbf{x})$  and **covariance kernel**  $k(\mathbf{x}, \mathbf{x}')$ .

- Note that a **random function is a stochastic process**. It is a collection of random variables  $\{f(\mathbf{x})\}_{\mathbf{x} \in \mathcal{X}}$ , one for each possible value  $\mathbf{x}$  (see Rasmussen and Williams, 2006).
- **Key point about Gaussian Processes:** Given a dataset  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ , the marginal distribution over  $[f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_N)]$  is completely specified by the second-order statistics: the mean and covariance.

# Gaussian Process

- In many applications, we will have no prior knowledge about the mean function  $f(\mathbf{x})$ . By symmetry, **we take it be zero**.
- The specification of a Gaussian Process is then completed by **specifying the covariance function**, evaluated at any two input points  $\mathbf{x}_n$  and  $\mathbf{x}_m$ :

$$\mathbb{E}[f(\mathbf{x}_n)f(\mathbf{x}_m)] = k(\mathbf{x}_n, \mathbf{x}_m).$$

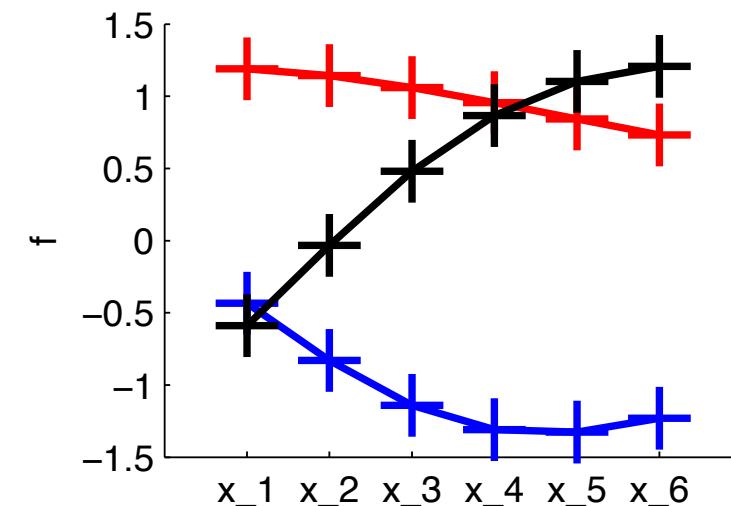
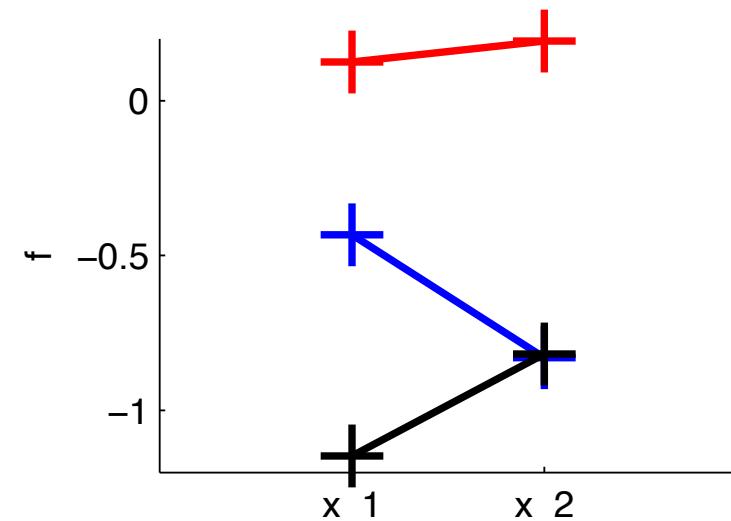
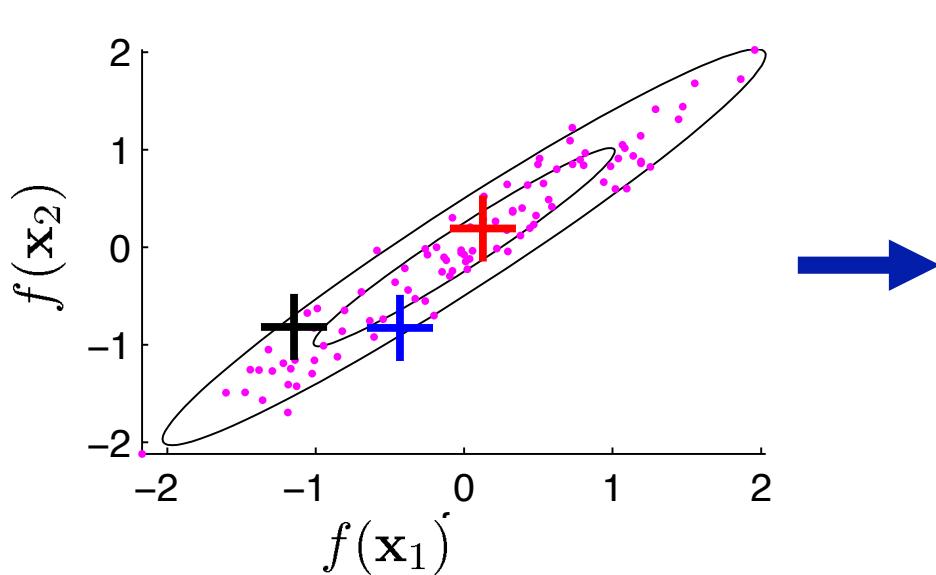
- One commonly used covariance function is squared exponential:

$$k(\mathbf{x}_n, \mathbf{x}_m) = \exp\left(-\frac{\theta}{2}||\mathbf{x}_n - \mathbf{x}_m||^2\right)$$

- Covariance (kernel) function is typically chosen to express the property that, for **inputs  $\mathbf{x}_n$  and  $\mathbf{x}_m$  that are similar**, the corresponding values  $f(\mathbf{x}_n)$  and  $f(\mathbf{x}_m)$  will **be more strongly correlated** than for dissimilar points.

# Visualizing Draws from GPs

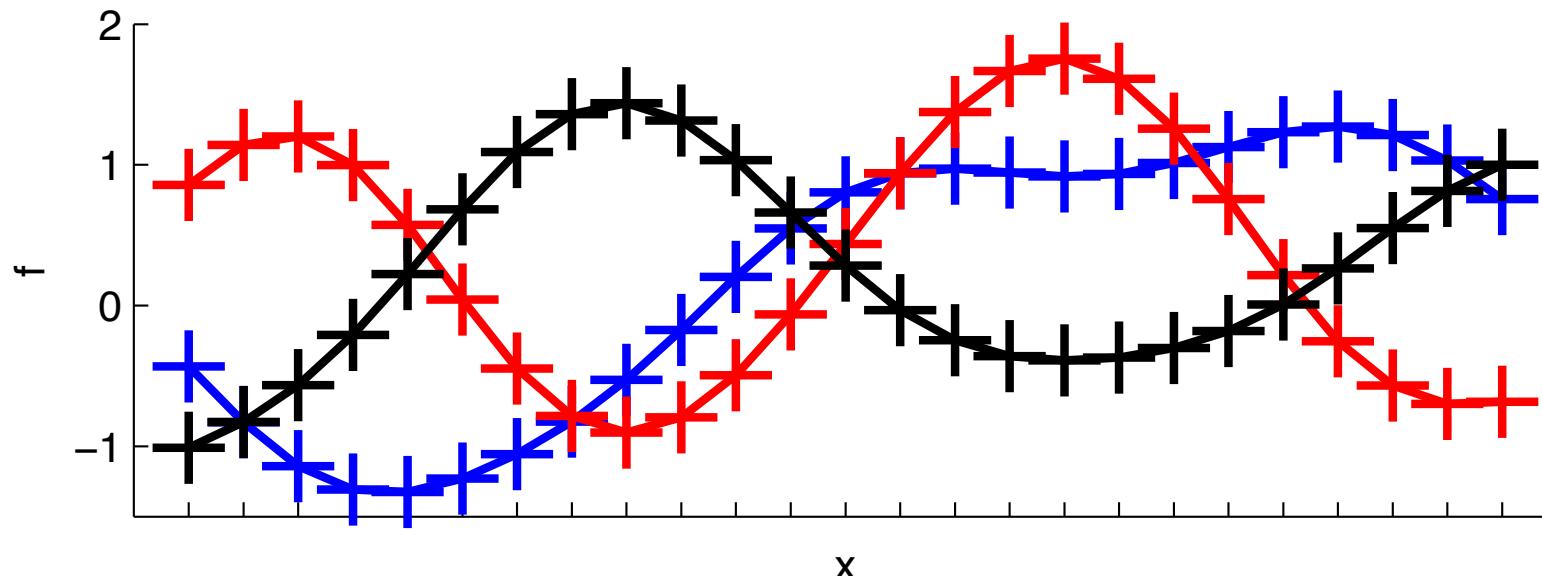
- Visualizing draws from 2-D Gaussian:



- Three draws from a 6-D Gaussian:

# Visualizing Draws from GPs

- Three draws from 25-D Gaussian

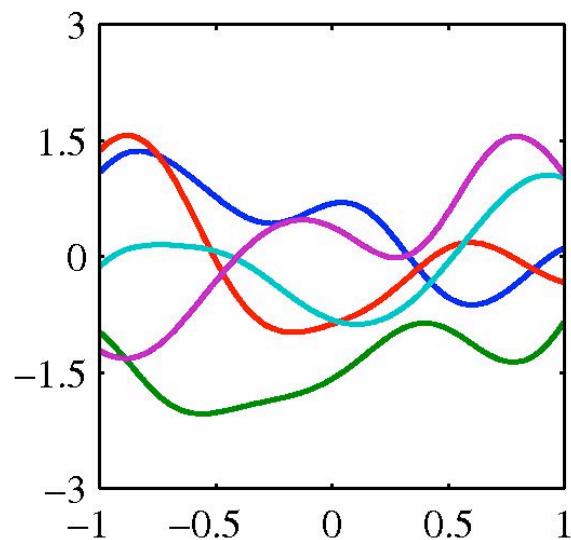


- To generate these, the mean was set to zero: `zeros(25,1)`
- The covariance was set using a covariance function:  $\Sigma_{nm} = k(\mathbf{x}_n, \mathbf{x}_m)$ .
- The x's are the positions that are planted the tics on the axis.

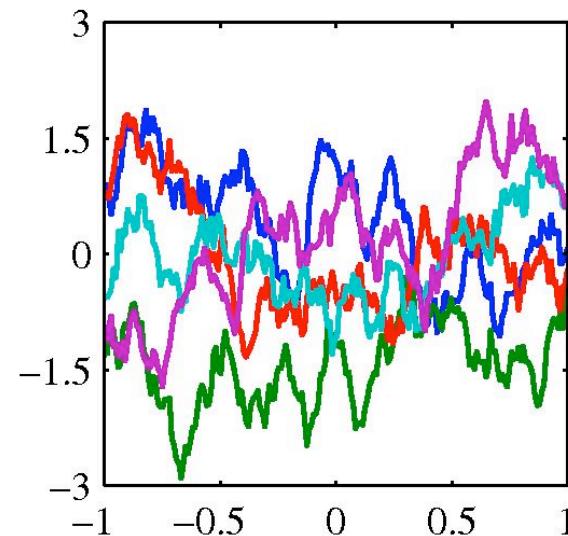
We can visualize draws from a GP iterative sampling  $f(x_n) | f(x_1), \dots, f(x_{n-1})$  on a sequence of input points  $x_1, x_2, \dots, x_n$ .

# Samples from GPs

Squared-exponential kernel



Exponential kernel



$$k(x_n, x_m) = \exp\left(-\frac{\theta}{2}(x_n - x_m)^2\right)$$

$$k(x_n, x_m) = \exp(-\theta|x_n - x_m|)$$

- Ornstein-Uhlenbeck process that describes Brownian motion.

# GPs for Regression

- We need to account for **noise on the observed target values**:

$$t_n = f_n + \epsilon_n,$$

where  $f_n = f(\mathbf{x}_n)$ , and  $\epsilon_n$  is an **independent random noise variable**. We will assume Gaussian noise:

$$p(t_n|f_n) = \mathcal{N}(t_n|f_n, \beta^{-1}).$$

- Given a dataset  $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ , and corresponding target values  $\mathbf{t} = \{t_1, t_2, \dots, t_N\}$ , the conditional takes form:

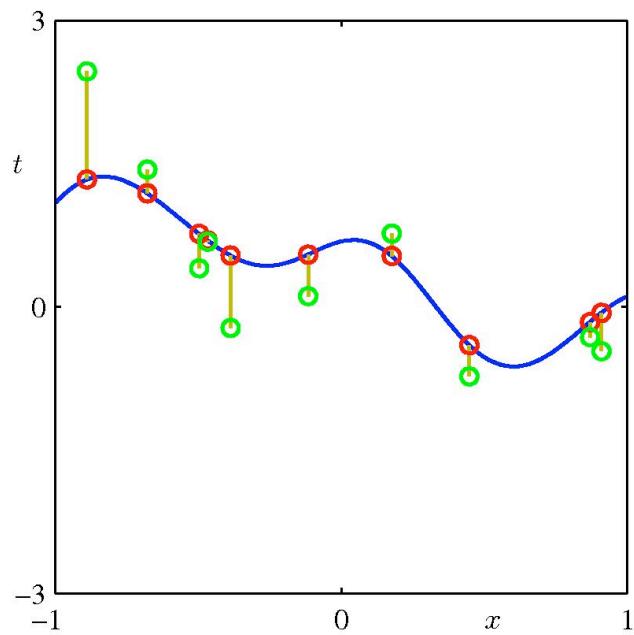
$$p(\mathbf{t}|\mathbf{f}) = \mathcal{N}(\mathbf{t}|\mathbf{f}, \beta^{-1} \mathbf{I}_N).$$

- From the definitions of a Gaussian process, **the marginal distribution**  $p(\mathbf{f})$  is given by the Gaussian:

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

# Illustration

- Illustration of sampling of targets  $\{t_n\}$  from a Gaussian process.

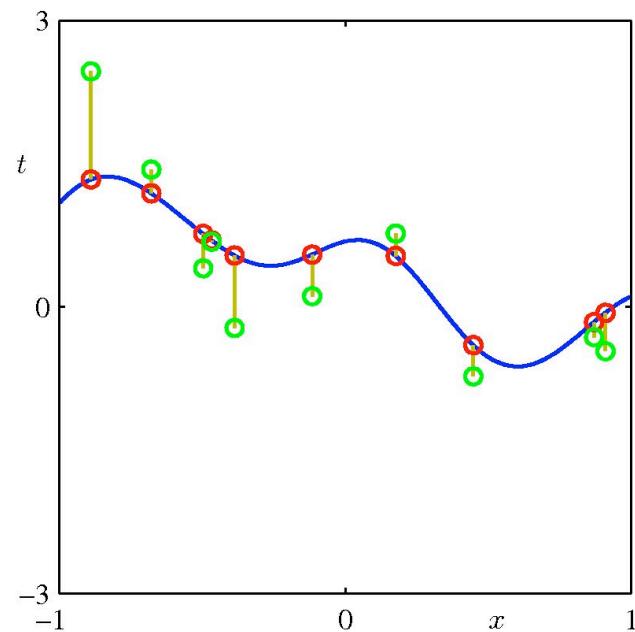


- The **blue curve** shows a sample from a GP prior:  
$$f \sim \mathcal{GP}$$
- The **red points** show the values of  $f_n$ , obtained by evaluating the function at a set of input values  $\{x_n\}$ .
- The **green points** show the corresponding values of  $\{t_n\}$ :

$$p(t_n | f_n) = \mathcal{N}(t_n | f_n, \beta^{-1}).$$

# Marginal Distribution

- The marginal distribution  $p(\mathbf{t})$ , conditioned on the set of inputs  $\mathbf{X}$ , can be obtained by integrating over  $\mathbf{f}$ :



$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{f})p(\mathbf{f})d\mathbf{f} = \mathcal{N}(\mathbf{t}|\mathbf{0}, \mathbf{C}),$$

where the covariance matrix is given by:

$$C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \beta^{-1}\delta_{nm}.$$

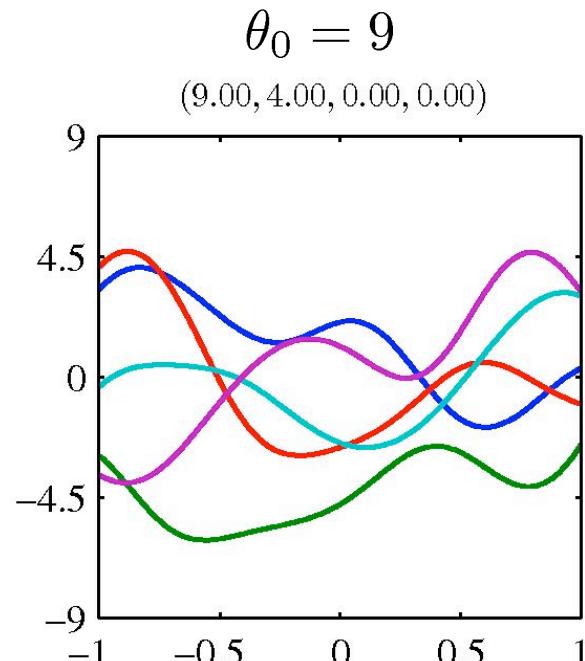
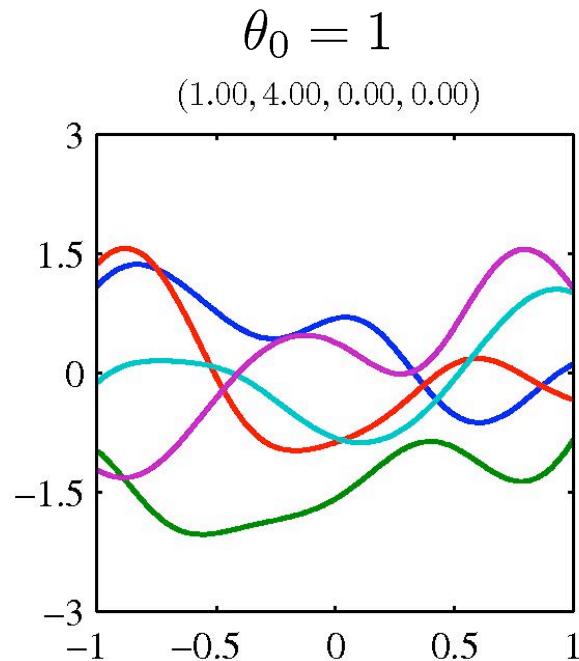
- The two Gaussian sources of randomness, one associated with  $f(\mathbf{x})$  and the other with noise, are independent, and so their covariances add.

# Covariance Function

- One widely used covariance (kernel) function for GP regression is given by the **squared-exponential** plus constant and linear terms:

$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left(-\frac{\theta_1}{2} \|\mathbf{x}_n - \mathbf{x}_m\|^2\right) + \theta_2 + \theta_3 \mathbf{x}_n^T \mathbf{x}_m$$

- Note that the last term corresponds to a parametric model that is a **linear function of the input variables**.



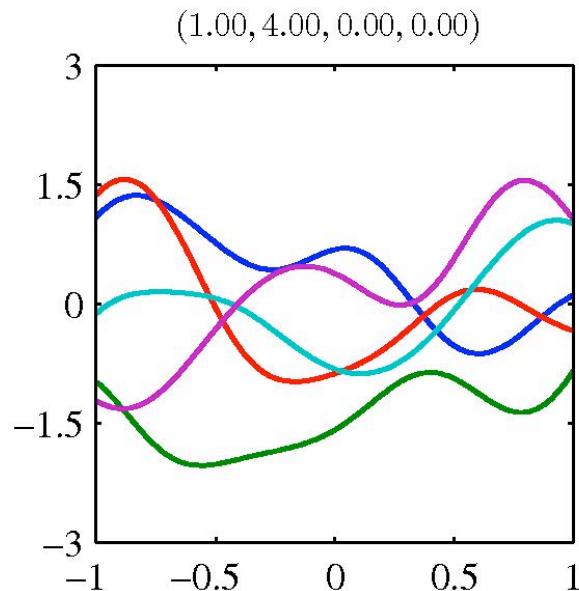
# Covariance Function

- One widely used covariance (kernel) function for GP regression is given by the **squared-exponential** plus constant and linear terms:

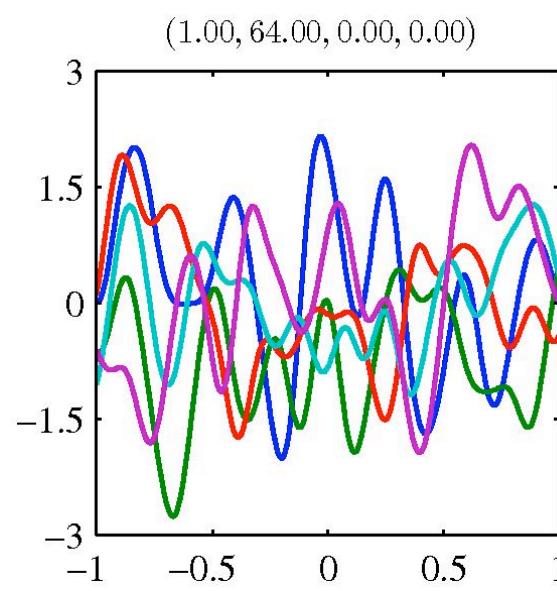
$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left(-\frac{\theta_1}{2} \|\mathbf{x}_n - \mathbf{x}_m\|^2\right) + \theta_2 + \theta_3 \mathbf{x}_n^T \mathbf{x}_m$$

- Note that the last term corresponds to a parametric model that is a **linear function of the input variables**.

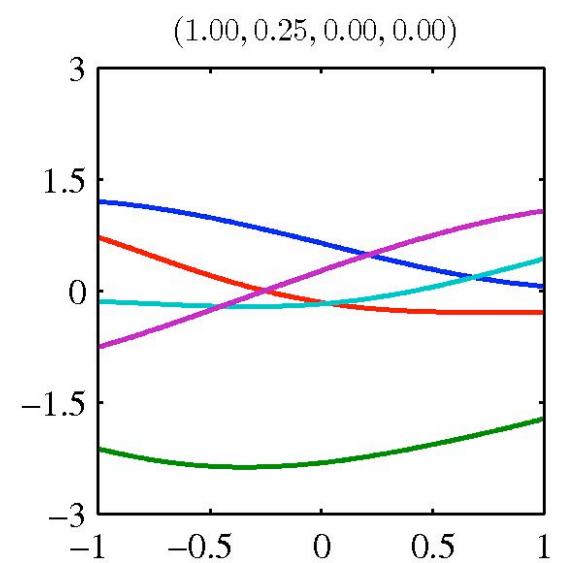
$$\theta_1 = 1$$



$$\theta_1 = 64$$



$$\theta_1 = 0.25$$



# Covariance Function

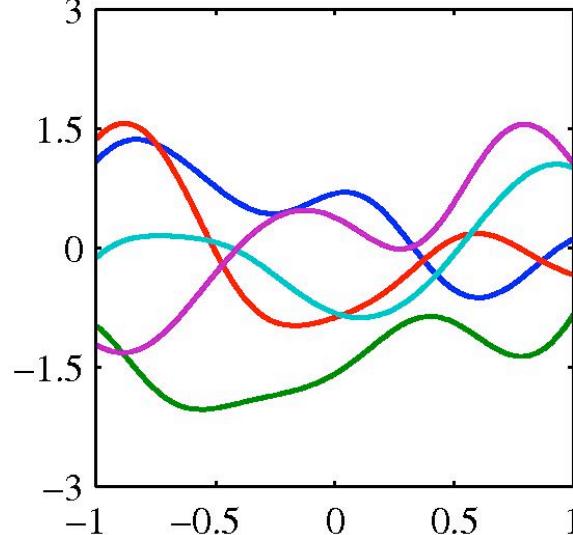
- One widely used covariance (kernel) function for GP regression is given by the **squared-exponential** plus constant and linear terms:

$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left(-\frac{\theta_1}{2} \|\mathbf{x}_n - \mathbf{x}_m\|^2\right) + \theta_2 + \theta_3 \mathbf{x}_n^T \mathbf{x}_m$$

- Note that the last term corresponds to a parametric model that is a **linear function of the input variables**.

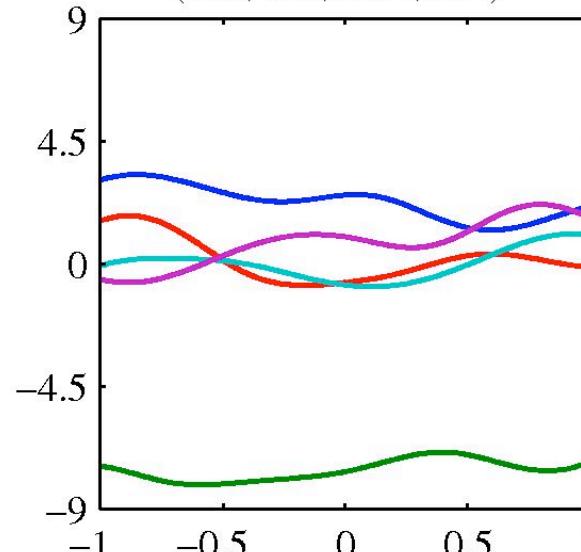
$$\theta_2 = 0, \theta_3 = 0$$

(1.00, 4.00, 0.00, 0.00)



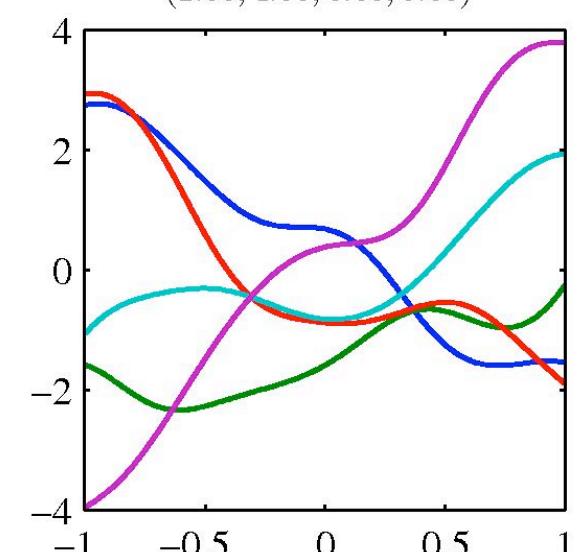
$$\theta_2 = 10, \theta_3 = 0$$

(1.00, 4.00, 10.00, 0.00)



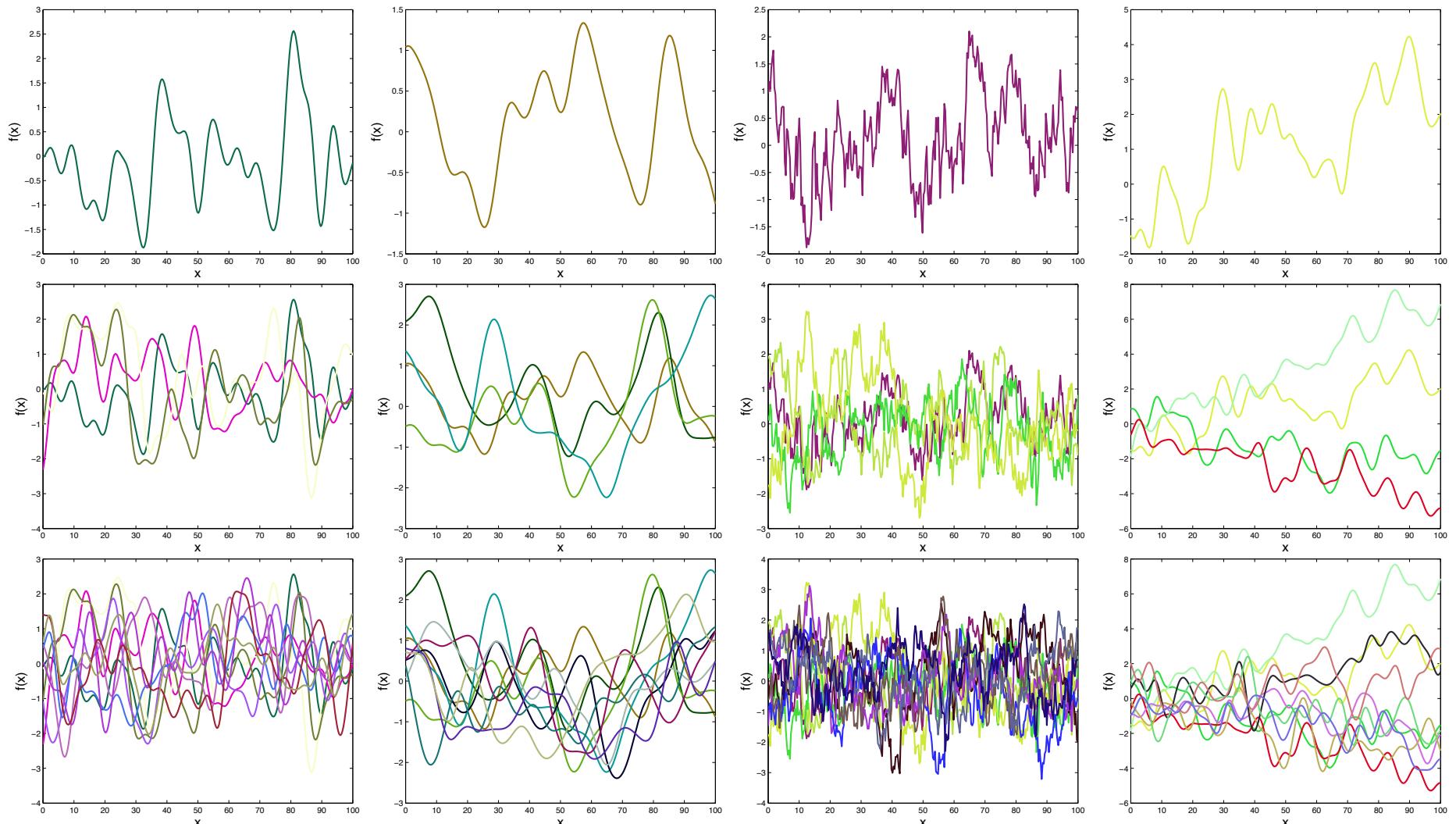
$$\theta_2 = 0, \theta_3 = 5$$

(1.00, 4.00, 0.00, 5.00)



# Samples from GPs

- Samples from GPs with different covariance functions:



# Prediction

- Suppose we are given a dataset  $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ , with target values  $\mathbf{t} = \{t_1, t_2, \dots, t_N\}$ .
- Our goal is predict  $t_{N+1}$  for a new input vector  $\mathbf{x}_{N+1}$ .
- Note that the joint distribution over  $\mathbf{t}$  and  $t_{N+1}$  is given by:

$$P\left(\begin{bmatrix} \mathbf{t} \\ t_{N+1} \end{bmatrix}\right) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{C}_N & \mathbf{k} \\ \mathbf{k}^T & c \end{bmatrix}\right)$$

where  $\mathbf{C}_N$  is the **N by N matrix** with elements:

$$C_N(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \beta^{-1} \delta_{nm}.$$

$c$  is the **scalar**:

$$c = k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \beta^{-1}$$

and  $\mathbf{k}$  is the **N by 1 vector** with elements  $k(\mathbf{x}_n, \mathbf{x}_{N+1})$ .

# Prediction

- Suppose we are given a dataset  $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ , with target values  $\mathbf{t} = \{t_1, t_2, \dots, t_N\}$ .
- Our goal is predict  $t_{N+1}$  for a new input vector  $\mathbf{x}_{N+1}$ .
- Note that the joint distribution over  $\mathbf{t}$  and  $t_{N+1}$  is given by:

$$P\left(\begin{bmatrix} \mathbf{t} \\ t_{N+1} \end{bmatrix}\right) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{C}_N & \mathbf{k} \\ \mathbf{k}^T & c \end{bmatrix}\right)$$

- Hence the conditional distribution is Gaussian:

$$P(t_{N+1}|\mathbf{t}) = \mathcal{N}(m(\mathbf{x}_{N+1}), \sigma^2(\mathbf{x}_{N+1}))$$

with

$$m(\mathbf{x}_{N+1}) = \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{t}$$

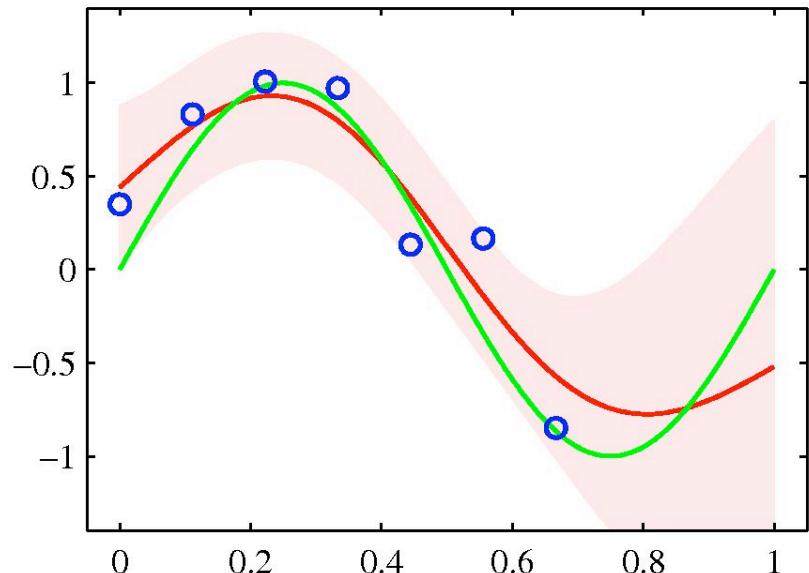
Key results that define  
GP regression

$$\sigma^2(\mathbf{x}_{N+1}) = c - \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{k}$$

Positive: hence the  
reduction in uncertainty

# Illustration

- Illustration of GP regression applied to the sinusoidal data set.



- The green curve shows the true function.
  - The blue data points are samples from the true function plus some additive Gaussian noise
  - The red curve shows the mean of the GP predictive distribution, with shaded region corresponding to +/- 2 standard deviations.
- Restriction on the kernel function: The covariance matrix:

$$C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \beta^{-1} \delta_{nm}.$$

must be positive definite.

# Mean of Predictive Distribution

- Note that the **mean of the predictive distribution**

$$m(\mathbf{x}_{N+1}) = \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{t}$$

can be written as a function of  $\mathbf{x}_{N+1}$ :

$$m(\mathbf{x}_{N+1}) = \sum_{n=1}^N a_n k(\mathbf{x}_n, \mathbf{x}_{N+1})$$

Linear combination

$a_n$  is the  $n^{\text{th}}$  component of  $\mathbf{C}_N^{-1} \mathbf{t}$

- Also, note that **the mean and variance** of the predictive distribution both depend on  $\mathbf{x}_{N+1}$ .

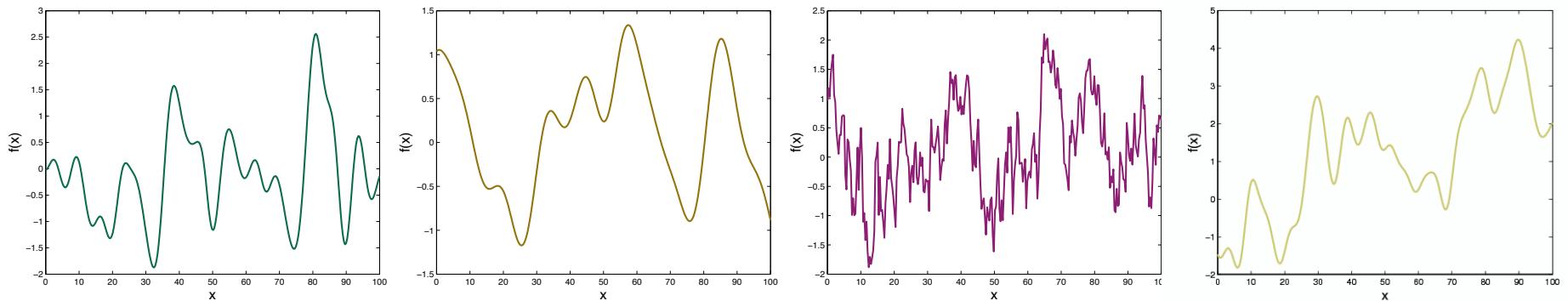
$$m(\mathbf{x}_{N+1}) = \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{t}$$

$$\sigma^2(\mathbf{x}_{N+1}) = c - \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{k}$$

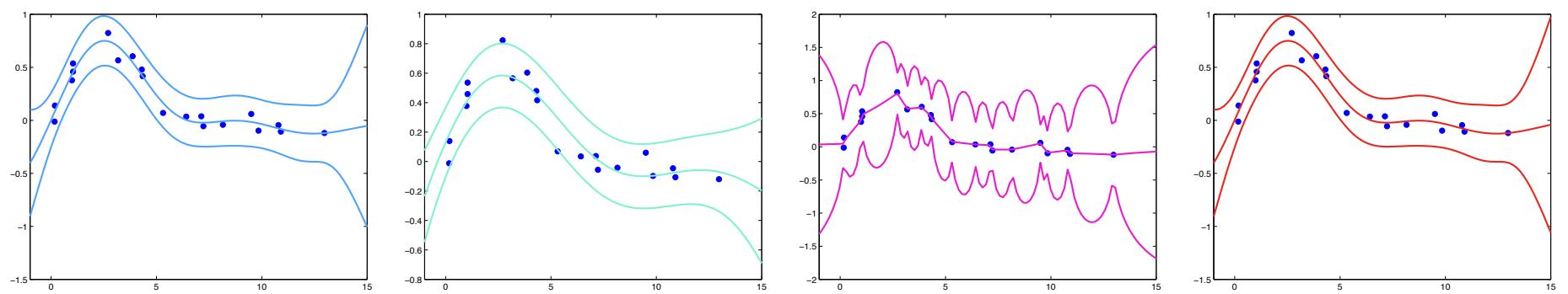
**Remember:**  $\mathbf{k}$  is the  $N$  by 1 vector with elements  $k(\mathbf{x}_n, \mathbf{x}_{N+1})$ .

# Prediction using GPs

- A sample from the prior using different covariance functions:



- Corresponding predictions: mean plus two standard deviations:



# Computational Complexity

- The central computation in using GPs will involve **the inversion of an N by N matrix  $\mathbf{C}_N$** , which is of order  $O(N^3)$ :

$$m(\mathbf{x}_{N+1}) = \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{t}$$

$$\sigma^2(\mathbf{x}_{N+1}) = c - \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{k}$$

- By contrast, in **the basis function model**, we have to invert a matrix  $\mathbf{S}_N$  of size M by M (where M is the number of basis functions).
- If the **number of M basis functions is smaller** than the number N of data points, then it will be computationally more efficient to work in the basis function framework (see the first few slides)
- The advantage of GPs is that we can consider covariance functions that can only **be expressed in terms of an infinite number of basis functions**.

# Learning the Hyperparameters

- The predictions of a GP regression model will depend on the **choice of the covariance function**.
- Instead of fixing the covariance function, we may prefer to use a parametric family of functions and **infer the parameter values from data**.
- These parameters may govern the length scale of the correlations or the precision of the noise model and **correspond to the hyperparameters in a standard parametric model**.

$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left(-\frac{\theta_1}{2} \|\mathbf{x}_n - \mathbf{x}_m\|^2\right) + \theta_2 + \theta_3 \mathbf{x}_n^T \mathbf{x}_m$$

hyperparameters

The diagram shows four blue arrows originating from the word "hyperparameters" at the top center and pointing downwards towards the parameters  $\theta_0$ ,  $\theta_1$ ,  $\theta_2$ , and  $\theta_3$  in the equation below.

- How can we infer the values of these parameters?

# Learning the Hyperparameters

- We can compute the marginal likelihood function:

$$p(\mathbf{t}|\theta) = \int p(\mathbf{t}|\mathbf{f}), \theta) p(\mathbf{f}|\theta) d\mathbf{f} = \mathcal{N}(\mathbf{t}|\mathbf{0}, \mathbf{C}_N),$$

 Hyperparameters of the GP model

- One option is to maximize the log of the marginal likelihood with respect to  $\theta$ .

$$\ln p(\mathbf{t}|\theta) = -\frac{1}{2} \ln |\mathbf{C}_N| - \frac{1}{2} \mathbf{t}^T \mathbf{C}_N^{-1} \mathbf{t} - \frac{N}{2} \ln(2\pi).$$

- This corresponds to the **type II maximum likelihood**, or **empirical Bayes**:
- The maximization can be performed using **gradient-based optimization techniques**, such as conjugate gradients. The gradients take form:

$$\frac{\partial}{\partial \theta_i} \ln p(\mathbf{t}|\theta) = -\frac{1}{2} \text{Tr} \left( \mathbf{C}_N^{-1} \frac{\partial \mathbf{C}_N}{\partial \theta_i} \right) + \frac{1}{2} \mathbf{t}^T \mathbf{C}_N^{-1} \frac{\partial \mathbf{C}_N}{\partial \theta_i} \mathbf{C}_N^{-1} \mathbf{t}.$$

hyperparameters: high-level tuning of the function

# Learning the Hyperparameters

$$\ln p(\mathbf{t}|\theta) = -\frac{1}{2} \ln |\mathbf{C}_N| - \frac{1}{2} \mathbf{t}^T \mathbf{C}_N^{-1} \mathbf{t} - \frac{N}{2} \ln(2\pi).$$

- Because  $\ln p(\mathbf{t}|\theta)$  will be a **nonconvex function**, it will have **multiple maxima**.
- In the **fully Bayesian approach**, we can introduce a prior  $p(\theta)$  and infer the posterior  $p(\theta | \mathbf{t})$ .
- In general, the posterior will not have a closed form solution, so we must resort of approximations (typically MCMC).
- **Noise:** We have assumed that the additive noise, governed by  $\beta$ , is **constant**.

$$p(t_n | f_n) = \mathcal{N}(t_n | f_n, \beta^{-1}).$$

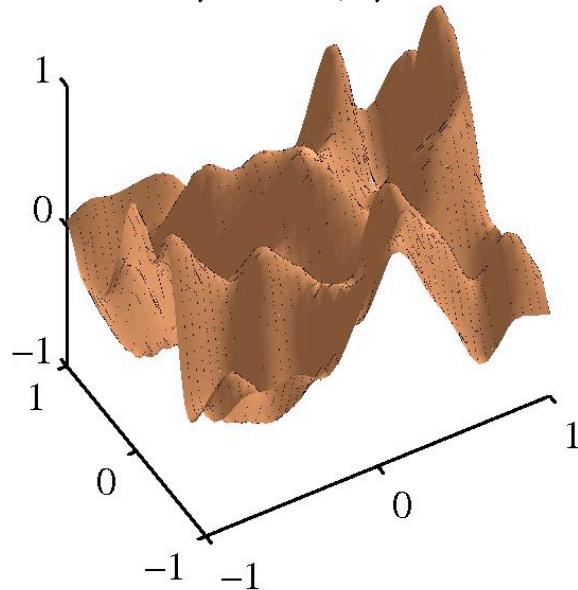
- For some models, known as **heteroscedastic**, the noise variance itself will depend on  $\mathbf{x}$  (e.g. by introducing another GP that will model  $\log \beta(\mathbf{x})$ ).

# Automatic Relevance Determination

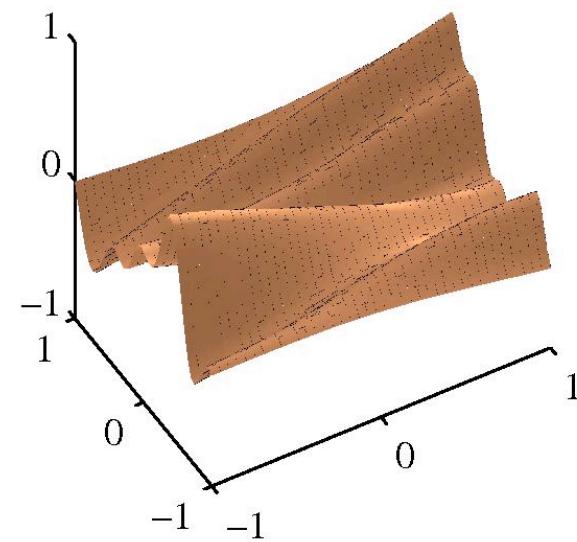
- How can we detect inputs variables that have **very little effect** on the **predictive distribution** (irrelevant inputs).
- Consider a GP with 2-D input space  $\mathbf{x} = (x_1, x_2)$  with the following covariance function:

$$k(\mathbf{x}, \mathbf{x}') = \theta_0 \exp \left( -\frac{\theta_1}{2} \sum_{i=1}^2 \eta_i (x_i - x'_i)^2 \right)$$

$$\eta_1 = 1, \eta_2 = 1$$



$$\eta_1 = 1, \eta_2 = 0.01$$



- As  $\eta_i$  becomes small, the function becomes insensitive to the corresponding value of  $x_i$  (input  $x_i$  becomes less relevant).

# Automatic Relevance Determination

- The ARD framework can be easily incorporated into exponential-quadratic kernel:

$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp \left( -\frac{1}{2} \sum_{i=1}^D \eta_i (x_{ni} - x_{mi})^2 \right) + \theta_2 + \theta_3 \mathbf{x}_n^T \mathbf{x}_m$$

Control relevance of input dimension i, where D is the dimensionality of the input space

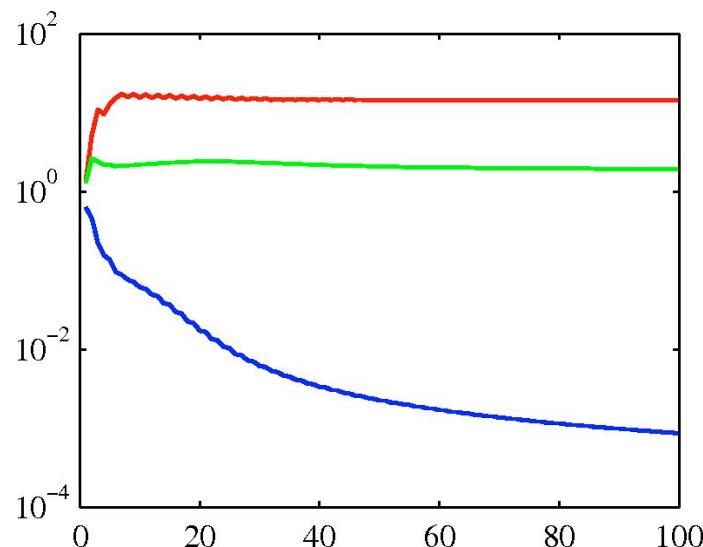
- We can optimize these parameters by performing type II maximum likelihood (by optimizing marginal log-likelihood)

$$\ln p(\mathbf{t}|\theta) = -\frac{1}{2} \ln |\mathbf{C}_N| - \frac{1}{2} \mathbf{t}^T \mathbf{C}_N^{-1} \mathbf{t} - \frac{N}{2} \ln(2\pi).$$

- The relative importance of different inputs can be inferred from data.

# Illustration

- **Example:** We have a dataset with 3-D inputs  $(x_1, x_2, x_3)$ . The target variables  $t_n$  are sampled as follows:
  - Sample 100 values of  $x_1$  from a Gaussian, **evaluate the function**  $\sin(2\pi x_1)$ , and add Gaussian noise.
  - Let  $x_2 = x_1$ , and **add Gaussian noise**.
  - Sample 100 values of  $x_3$  from an **independent Gaussian distribution**.
- Hence  $x_1$  is a good predictor of  $t$ ,  $x_2$  is a more noisy predictor of  $t$ , and  $x_3$  has only chance correlation with  $t$ .



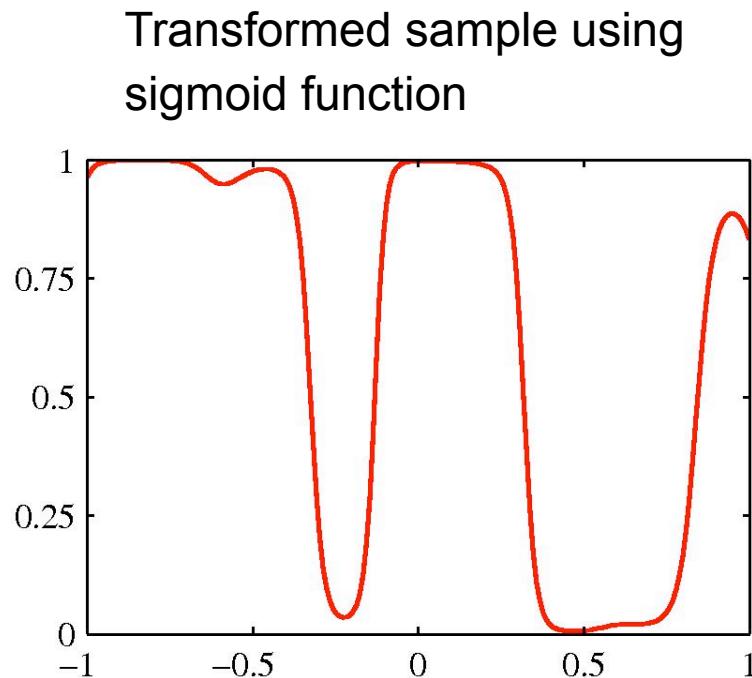
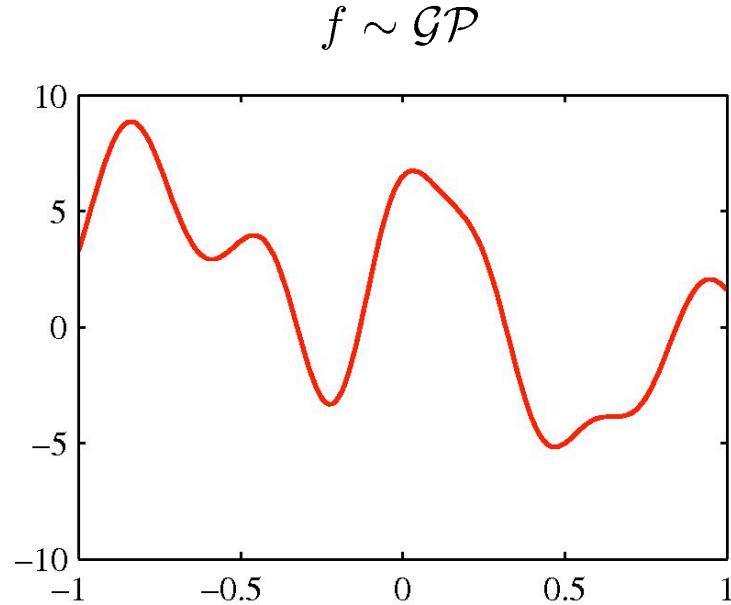
- Plot displays  $\eta_1$  (red),  $\eta_2$  (green), and  $\eta_3$  (blue) as a function of the number of iterations when optimizing the marginal likelihood.

# Classification with GPs

- Consider a two-class problem with targets  $t \in \{0,1\}$ .
- Define a Gaussian process over a function  $f(\mathbf{x})$ .
- Transform the function using sigmoid function:

$$y(\mathbf{x}) = \sigma(f(\mathbf{x})) = \frac{1}{1 + \exp(-f(\mathbf{x}))}$$

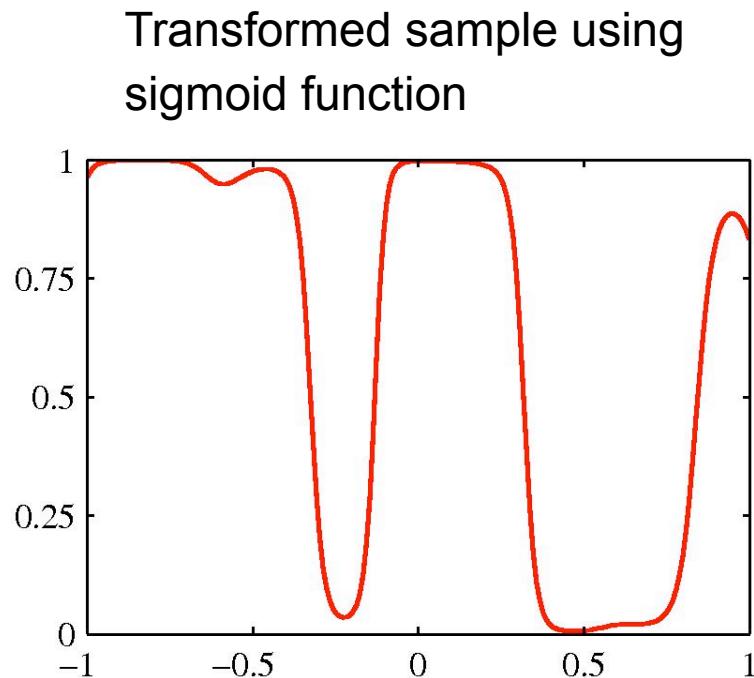
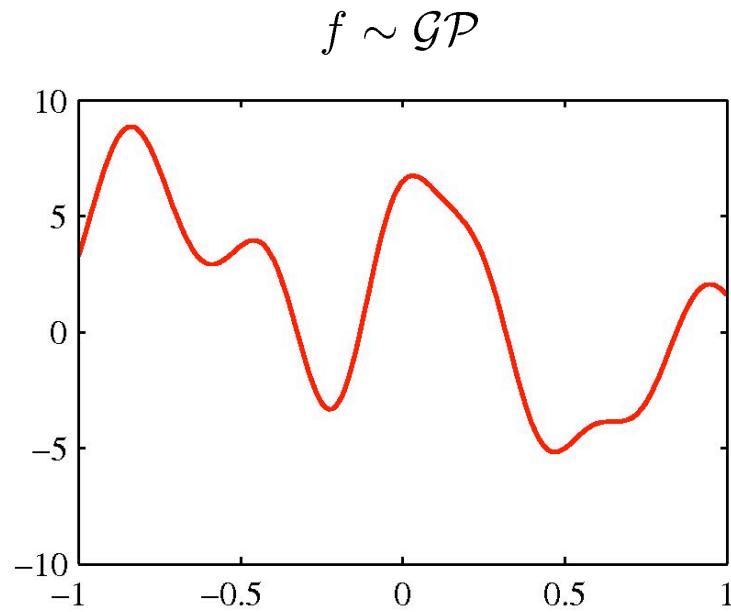
- Hence  $y(\mathbf{x}) \in (0,1)$ .



# Classification with GPs

- After transformation, we obtain a non-Gaussian stochastic process over functions  $y(\mathbf{x})$ .
- The probability distribution over  $t$  is given by the Bernoulli distribution:

$$p(t|f) = \sigma(f)^t(1 - \sigma(f))^{1-t}.$$



# Classification with GPs

- Suppose we are given a dataset  $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ , with target values  $\mathbf{t}_N = \{t_1, t_2, \dots, t_N\}$ .
- Our goal is predict  $t_{N+1}$  for a new input vector  $\mathbf{x}_{N+1}$
- Predictive distribution is given by:

$$p(t_{N+1} = 1 | \mathbf{t}_N) = \int p(t_{N+1} = 1 | f_{N+1}) p(f_{N+1} | \mathbf{t}_N) df_{N+1}$$

given by  $\sigma(f(\mathbf{x}_{N+1}))$       Posterior is also intractable.

- This integral is analytically intractable. Can resort to MCMC by approximately sampling from the posterior, and performing Monte Carlo integration:

$$p(t_{N+1} = 1 | \mathbf{t}) = \frac{1}{M} \sum_m p(t_{N+1} = 1 | f_{N+1}^{(m)})$$

where  $f_{N+1}^{(m)} \sim p(f_{N+1} | \mathbf{t}_N)$

# Approximations

- Another option:

$$p(t_{N+1} = 1 | \mathbf{t}_N) = \int p(t_{N+1} = 1 | f_{N+1}) p(f_{N+1} | \mathbf{t}_N) df_{N+1}$$

↗  
Gaussian approximation

- Use approximate formula for the convolution of a logistic sigmoid and a Gaussian distribution.
- Three different approaches to obtaining a Gaussian approximation:
  - Variational Inference
  - Expectation Propagation
  - Laplace Approximation

# Laplace Approximation

- We seek to obtain a **Gaussian approximation to the posterior**. Using Bayes rule we have:

$$p(f_{N+1}|\mathbf{t}_N) = \int p(f_{N+1}, \mathbf{f}_N|\mathbf{t}_N) d\mathbf{f}_N = \int p(f_{N+1}|\mathbf{f}_N)p(\mathbf{f}_N|\mathbf{t}_N) d\mathbf{f}_N$$

Easy to compute:  
Gaussian.

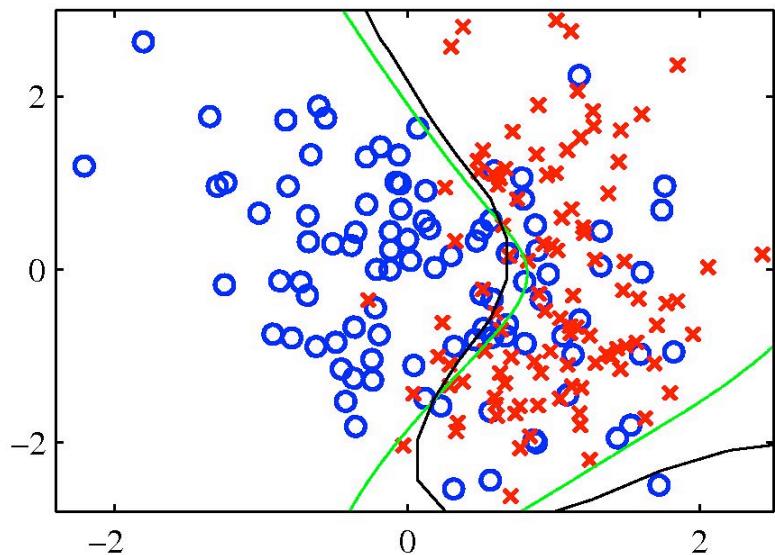
Laplace approximation

- Here  $p(\mathbf{f}_N)$  is given by a **zero-mean GP with covariance matrix  $\mathbf{C}_N$** , and the data term:

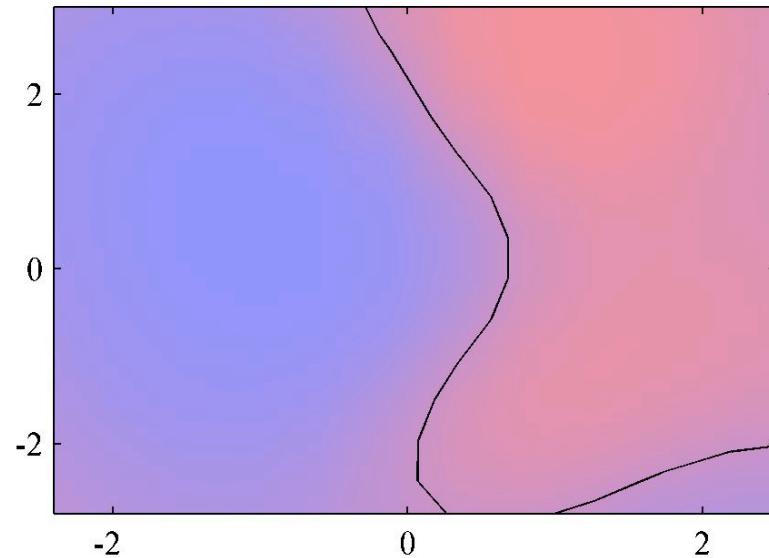
$$p(\mathbf{t}_N|\mathbf{f}_N) = \prod_{n=1}^N \sigma(f_n)^{t_n} (1 - \sigma(f_n))^{1-t_n}$$

- Obtain the **Laplace approximation** by Taylor expanding log of the posterior:  $\log p(\mathbf{f}_N | \mathbf{t}_N)$ .

# Classification Results



Optimal decision boundary from the true distribution (green) and the decision boundary from GP classifier (black)



Predictive posterior probability together with GP decision boundary.