

# Solving Challenging Non-linear Regression Problems by Manipulating a Gaussian Distribution

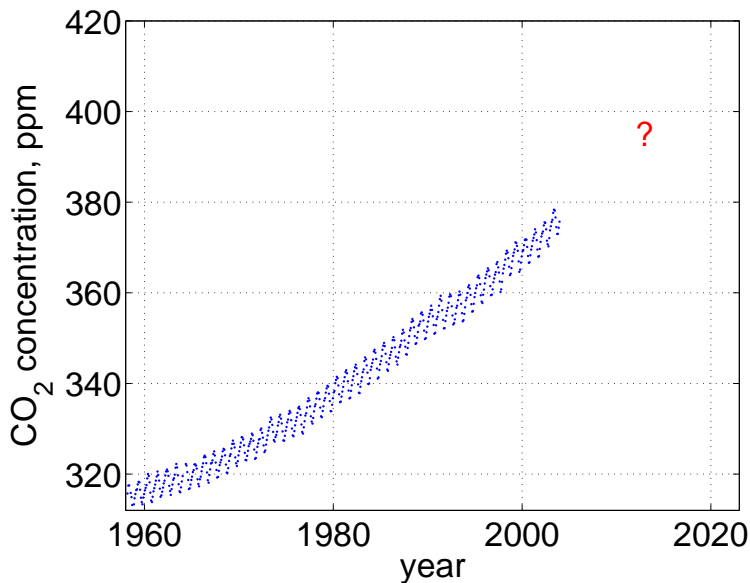
Machine Learning Summer School, Cambridge 2009

Carl Edward Rasmussen

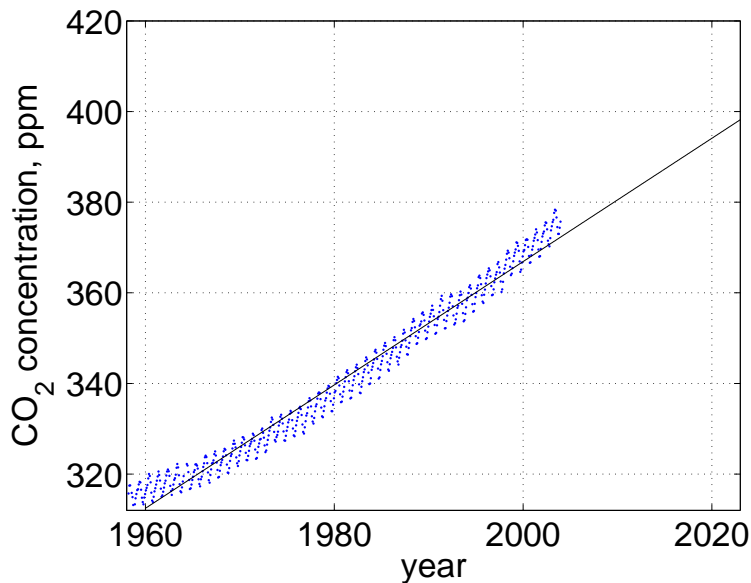
Department of Engineering, University of Cambridge

August 30th - September 10th, 2009

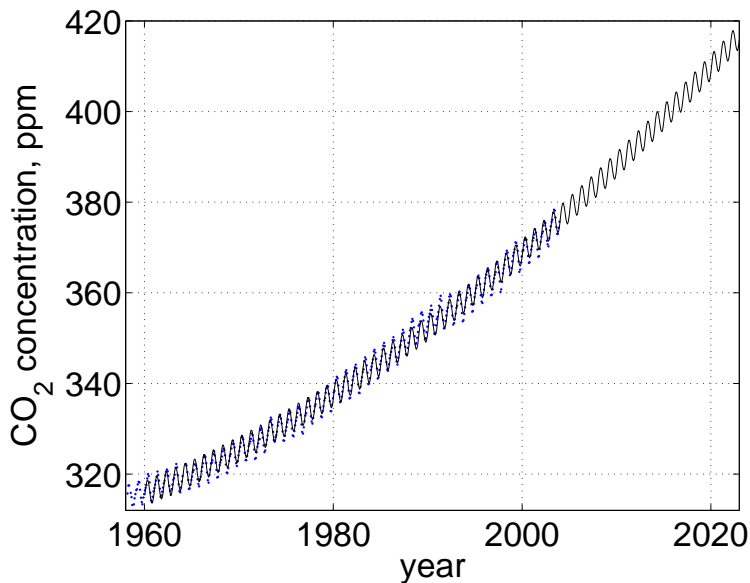
# The Prediction Problem



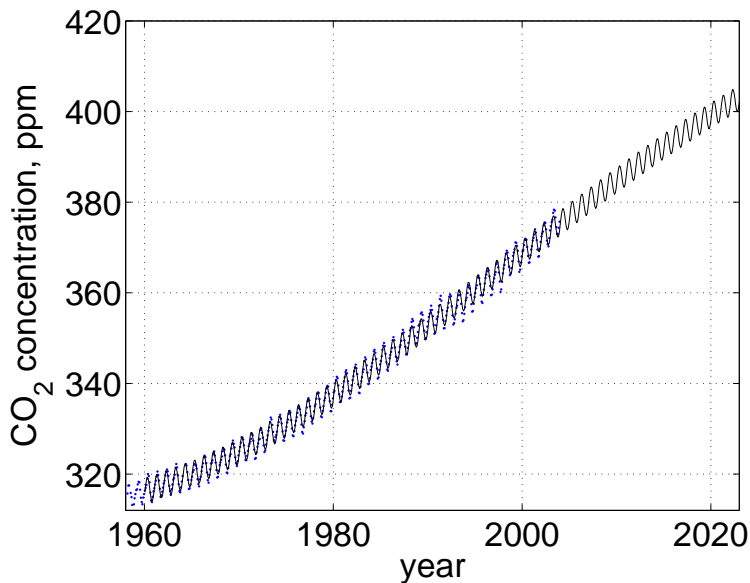
# The Prediction Problem



# The Prediction Problem



# The Prediction Problem



# Maximum likelihood, parametric model

Supervised parametric learning:

- data:  $\mathbf{x}, \mathbf{y}$
- model:  $y = f_{\mathbf{w}}(x) + \varepsilon$

Gaussian likelihood:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}) \propto \prod_c \exp(-\frac{1}{2}(y_c - f_{\mathbf{w}}(x_c))^2 / \sigma_{\text{noise}}^2).$$

Maximize the likelihood:

$$\mathbf{w}_{\text{ML}} = \underset{\mathbf{w}}{\operatorname{argmax}} p(\mathbf{y}|\mathbf{x}, \mathbf{w}).$$

Make predictions, by plugging in the ML estimate:

$$p(y^*|x^*, \mathbf{w}_{\text{ML}})$$

# Bayesian Inference, parametric model

Supervised parametric learning:

- data:  $\mathbf{x}, \mathbf{y}$
- model:  $y = f_{\mathbf{w}}(\mathbf{x}) + \varepsilon$

Gaussian likelihood:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}) \propto \prod_c \exp(-\frac{1}{2}(y_c - f_{\mathbf{w}}(\mathbf{x}_c))^2 / \sigma_{\text{noise}}^2).$$

Parameter prior:

$$p(\mathbf{w})$$

Posterior parameter distribution by Bayes rule  $p(a|b) = p(b|a)p(a)/p(b)$ :

$$p(\mathbf{w}|\mathbf{x}, \mathbf{y}) = \frac{p(\mathbf{w})p(\mathbf{y}|\mathbf{x}, \mathbf{w})}{p(\mathbf{y}|\mathbf{x})}$$

# Bayesian Inference, parametric model, cont.

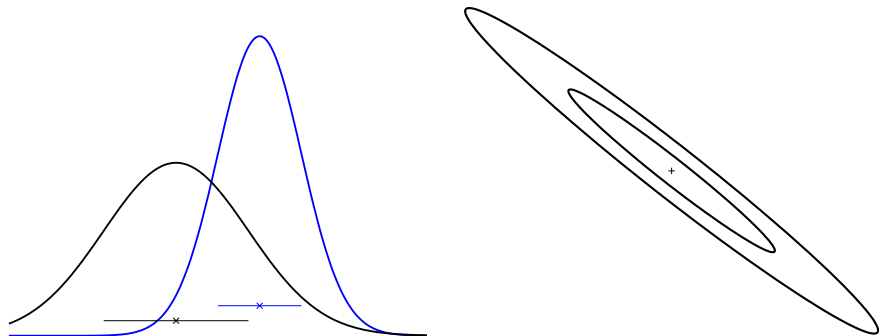
Making predictions:

$$p(y^*|x^*, \mathbf{x}, \mathbf{y}) = \int p(y^*|\mathbf{w}, x^*)p(\mathbf{w}|\mathbf{x}, \mathbf{y})d\mathbf{w}$$

Marginal likelihood:

$$p(\mathbf{y}|\mathbf{x}) = \int p(\mathbf{w})p(\mathbf{y}|\mathbf{x}, \mathbf{w})d\mathbf{w}.$$

# The Gaussian Distribution

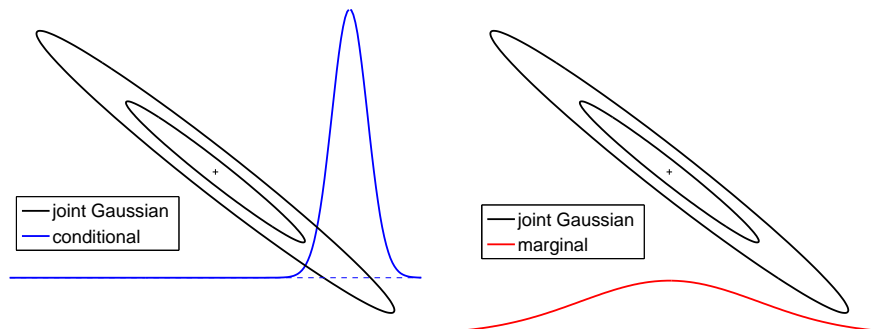


The Gaussian distribution is given by

$$p(\mathbf{x}|\mu, \Sigma) = \mathcal{N}(\mu, \Sigma) = (2\pi)^{-D/2} |\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu)^\top \Sigma^{-1}(\mathbf{x} - \mu)\right)$$

where  $\mu$  is the mean vector and  $\Sigma$  the covariance matrix.

# Conditionals and Marginals of a Gaussian



Both the **conditionals** and the **marginals** of a joint Gaussian are again Gaussian.

# Conditionals and Marginals of a Gaussian

In algebra, if  $\mathbf{x}$  and  $\mathbf{y}$  are jointly Gaussian

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}\right),$$

the marginal distribution of  $\mathbf{x}$  is

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}\right) \implies p(\mathbf{x}) = \mathcal{N}(\mathbf{a}, A),$$

and the conditional distribution of  $\mathbf{x}$  given  $\mathbf{y}$  is

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}\right) \implies p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{a} + BC^{-1}(\mathbf{y} - \mathbf{b}), A - BC^{-1}B^\top),$$

where  $\mathbf{x}$  and  $\mathbf{y}$  can be scalars or vectors.

# What is a Gaussian Process?

A *Gaussian process* is a generalization of a multivariate Gaussian distribution to **infinitely many variables**.

Informally: infinitely long vector  $\simeq$  function

**Definition:** *a Gaussian process is a collection of random variables, any finite number of which have (consistent) Gaussian distributions.*  $\square$

A Gaussian **distribution** is fully specified by a mean vector,  $\mu$ , and covariance matrix  $\Sigma$ :

$$\mathbf{f} = (f_1, \dots, f_n)^\top \sim \mathcal{N}(\mu, \Sigma), \quad \text{indexes } i = 1, \dots, n$$

A Gaussian **process** is fully specified by a mean function  $m(x)$  and covariance function  $k(x, x')$ :

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')), \quad \text{indexes: } x$$

# The marginalization property

Thinking of a GP as a Gaussian distribution with an infinitely long mean vector and an infinite by infinite covariance matrix may seem impractical...

...luckily we are saved by the *marginalization property*:

Recall:

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{y}) d\mathbf{y}.$$

For Gaussians:

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}\right) \implies p(\mathbf{x}) = \mathcal{N}(\mathbf{a}, A)$$

# Random functions from a Gaussian Process

Example one dimensional Gaussian process:

$$p(f(x)) \sim \mathcal{GP}(m(x) = 0, k(x, x') = \exp(-\frac{1}{2}(x - x')^2)).$$

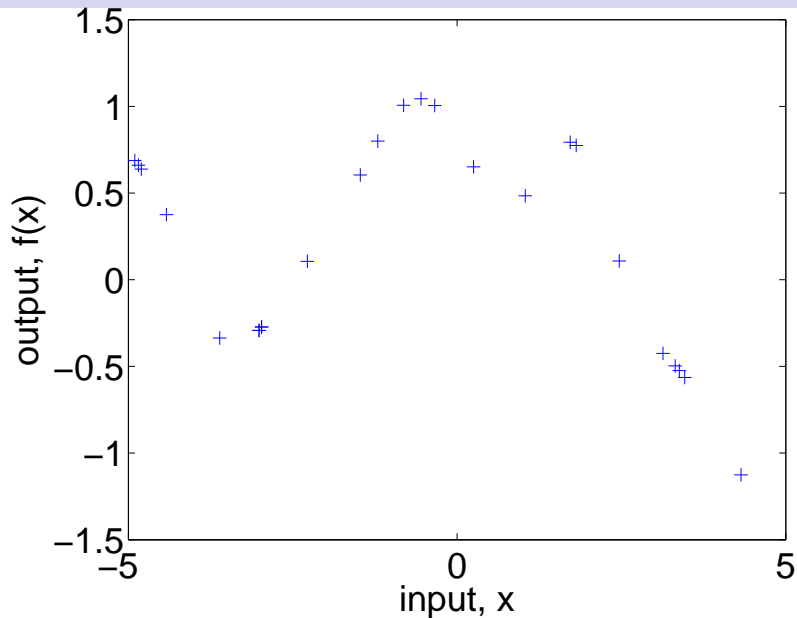
To get an indication of what this distribution over functions looks like, focus on a finite subset of function values  $\mathbf{f} = (f(x_1), f(x_2), \dots, f(x_n))^\top$ , for which

$$\mathbf{f} \sim \mathcal{N}(0, \Sigma),$$

where  $\Sigma_{ij} = k(x_i, x_j)$ .

Then plot the coordinates of  $f$  as a function of the corresponding  $x$  values.

## Some values of the random function



# Joint Generation

To generate a random sample from a  $D$  dimensional joint Gaussian with covariance matrix  $K$  and mean vector  $\mathbf{m}$ : (in octave or matlab)

```
z = randn(D,1);  
y = chol(K)'*z + m;
```

where `chol` is the Cholesky factor  $R$  such that  $R^T R = K$ .

Thus, the covariance of  $\mathbf{y}$  is:

$$\mathbb{E}[(\mathbf{y} - \bar{\mathbf{y}})(\mathbf{y} - \bar{\mathbf{y}})^T] = \mathbb{E}[R^T \mathbf{z} \mathbf{z}^T R] = R^T \mathbb{E}[\mathbf{z} \mathbf{z}^T] R = R^T I R = K.$$

# Sequential Generation

Factorize the joint distribution

$$p(f_1, \dots, f_n | \mathbf{x}_1, \dots, \mathbf{x}_n) = \prod_{i=1}^n p(f_i | f_{i-1}, \dots, f_1, \mathbf{x}_i, \dots, \mathbf{x}_1),$$

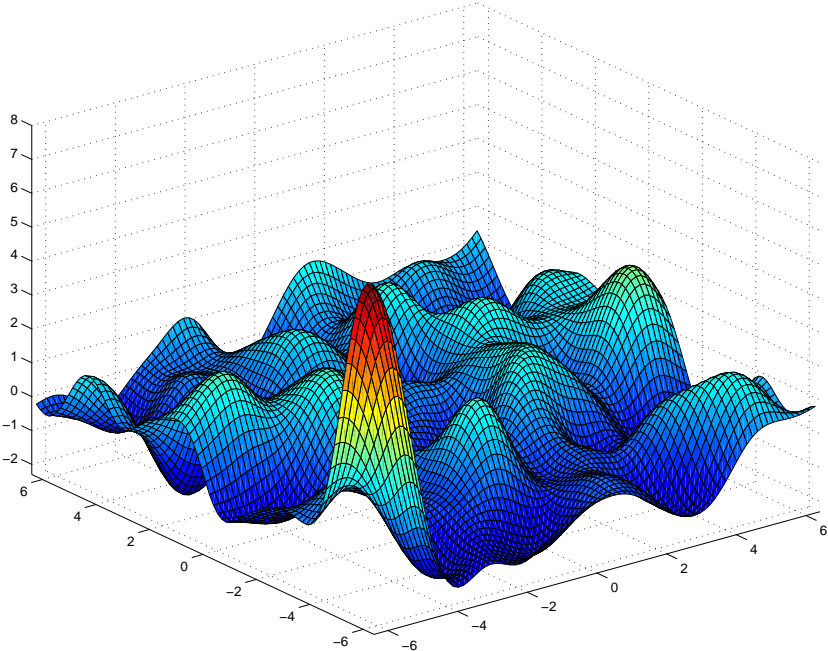
and generate function values sequentially.

What do the individual terms look like? For Gaussians:

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}\right) \implies p(\mathbf{x} | \mathbf{y}) = \mathcal{N}(\mathbf{a} + BC^{-1}(\mathbf{y} - \mathbf{b}), A - BC^{-1}B^\top)$$

Do try this at home!

# Function drawn at random from a Gaussian Process with Gaussian covariance



# Maximum likelihood, parametric model

Supervised parametric learning:

- data:  $\mathbf{x}, \mathbf{y}$
- model:  $y = f_{\mathbf{w}}(\mathbf{x}) + \varepsilon$

Gaussian likelihood:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i) \propto \prod_c \exp(-\frac{1}{2}(y_c - f_{\mathbf{w}}(\mathbf{x}_c))^2 / \sigma_{\text{noise}}^2).$$

Maximize the likelihood:

$$\mathbf{w}_{\text{ML}} = \underset{\mathbf{w}}{\operatorname{argmax}} p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i).$$

Make predictions, by plugging in the ML estimate:

$$p(y^*|\mathbf{x}^*, \mathbf{w}_{\text{ML}}, M_i)$$

# Bayesian Inference, parametric model

Supervised parametric learning:

- data:  $\mathbf{x}, \mathbf{y}$
- model:  $y = f_{\mathbf{w}}(\mathbf{x}) + \varepsilon$

Gaussian likelihood:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i) \propto \prod_c \exp(-\frac{1}{2}(y_c - f_{\mathbf{w}}(\mathbf{x}_c))^2 / \sigma_{\text{noise}}^2).$$

Parameter prior:

$$p(\mathbf{w}|M_i)$$

Posterior parameter distribution by Bayes rule  $p(a|b) = p(b|a)p(a)/p(b)$ :

$$p(\mathbf{w}|\mathbf{x}, \mathbf{y}, M_i) = \frac{p(\mathbf{w}|M_i)p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i)}{p(\mathbf{y}|\mathbf{x}, M_i)}$$

# Bayesian Inference, parametric model, cont.

Making predictions:

$$p(y^*|x^*, \mathbf{x}, \mathbf{y}, M_i) = \int p(y^*|\mathbf{w}, x^*, M_i)p(\mathbf{w}|\mathbf{x}, \mathbf{y}, M_i)d\mathbf{w}$$

Marginal likelihood:

$$p(\mathbf{y}|\mathbf{x}, M_i) = \int p(\mathbf{w}|M_i)p(\mathbf{y}|\mathbf{x}, \mathbf{w}, M_i)d\mathbf{w}.$$

Model probability:

$$p(M_i|\mathbf{x}, \mathbf{y}) = \frac{p(M_i)p(\mathbf{y}|\mathbf{x}, M_i)}{p(\mathbf{y}|\mathbf{x})}$$

Problem: integrals are intractable for most interesting models!

# Non-parametric Gaussian process models

In our non-parametric model, the “parameters” are the function itself!

Gaussian likelihood:

$$\mathbf{y}|\mathbf{x}, f(\mathbf{x}), M_i \sim \mathcal{N}(\mathbf{f}, \sigma_{\text{noise}}^2 \mathbf{I})$$

(Zero mean) Gaussian process prior:

$$f(\mathbf{x})|M_i \sim \mathcal{GP}(m(\mathbf{x}) \equiv 0, k(\mathbf{x}, \mathbf{x}'))$$

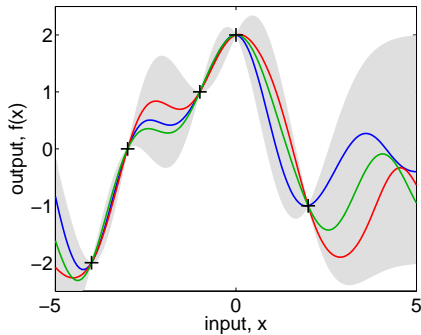
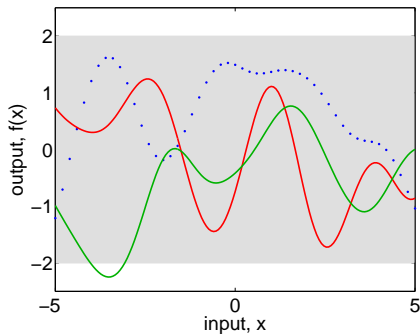
Leads to a Gaussian process posterior

$$\begin{aligned} f(\mathbf{x})|\mathbf{x}, \mathbf{y}, M_i &\sim \mathcal{GP}(m_{\text{post}}(\mathbf{x}) = k(\mathbf{x}, \mathbf{x})[K(\mathbf{x}, \mathbf{x}) + \sigma_{\text{noise}}^2 \mathbf{I}]^{-1} \mathbf{y}, \\ &k_{\text{post}}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') - k(\mathbf{x}, \mathbf{x})[K(\mathbf{x}, \mathbf{x}) + \sigma_{\text{noise}}^2 \mathbf{I}]^{-1} k(\mathbf{x}, \mathbf{x}')). \end{aligned}$$

And a Gaussian predictive distribution:

$$\begin{aligned} \mathbf{y}^*|\mathbf{x}^*, \mathbf{x}, \mathbf{y}, M_i &\sim \mathcal{N}(\mathbf{k}(\mathbf{x}^*, \mathbf{x})^\top [K + \sigma_{\text{noise}}^2 \mathbf{I}]^{-1} \mathbf{y}, \\ &k(\mathbf{x}^*, \mathbf{x}^*) + \sigma_{\text{noise}}^2 - \mathbf{k}(\mathbf{x}^*, \mathbf{x})^\top [K + \sigma_{\text{noise}}^2 \mathbf{I}]^{-1} \mathbf{k}(\mathbf{x}^*, \mathbf{x})) \end{aligned}$$

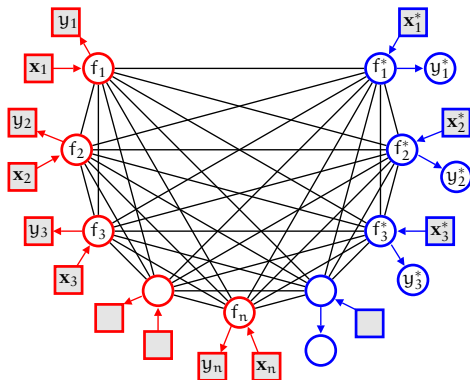
# Prior and Posterior



Predictive distribution:

$$p(y^* | x^*, \mathbf{x}, \mathbf{y}) \sim \mathcal{N}(\mathbf{k}(x^*, \mathbf{x})^\top [K + \sigma_{\text{noise}}^2 I]^{-1} \mathbf{y}, \\ \mathbf{k}(x^*, x^*) + \sigma_{\text{noise}}^2 - \mathbf{k}(x^*, \mathbf{x})^\top [K + \sigma_{\text{noise}}^2 I]^{-1} \mathbf{k}(x^*, \mathbf{x}))$$

# Graphical model for Gaussian Process



Square nodes are observed (clamped), round nodes stochastic (free).

All pairs of latent variables are connected.

Predictions  $y^*$  depend only on the corresponding single latent  $f^*$ .

Notice, that adding a triplet  $x_m^*, f_m^*, y_m^*$  does not influence the distribution. This is guaranteed by the marginalization property of the GP.

This explains why we can make inference using a finite amount of computation!

# Some interpretation

Recall our main result:

$$\mathbf{f}_* | \mathbf{X}_*, \mathbf{X}, \mathbf{y} \sim \mathcal{N}(K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1} \mathbf{y}, \\ K(\mathbf{X}_*, \mathbf{X}_*) - K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1} K(\mathbf{X}, \mathbf{X}_*)).$$

The mean is linear in two ways:

$$\mu(\mathbf{x}_*) = k(\mathbf{x}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1} \mathbf{y} = \sum_{c=1}^n \beta_c y^{(c)} = \sum_{c=1}^n \alpha_c k(\mathbf{x}_*, \mathbf{x}^{(c)}).$$

The last form is most commonly encountered in the kernel literature.

The variance is the difference between two terms:

$$V(\mathbf{x}_*) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}(\mathbf{x}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1} \mathbf{k}(\mathbf{X}, \mathbf{x}_*),$$

the first term is the *prior variance*, from which we subtract a (positive) term, telling how much the data  $\mathbf{X}$  has explained. Note, that the variance is independent of the observed outputs  $\mathbf{y}$ .

# The marginal likelihood

Log marginal likelihood:

$$\log p(\mathbf{y}|\mathbf{x}, M_i) = -\frac{1}{2}\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y} - \frac{1}{2}\log |\mathbf{K}| - \frac{n}{2}\log(2\pi)$$

is the combination of a **data fit** term and **complexity penalty**. Occam's Razor is automatic.

**Learning** in Gaussian process models involves finding

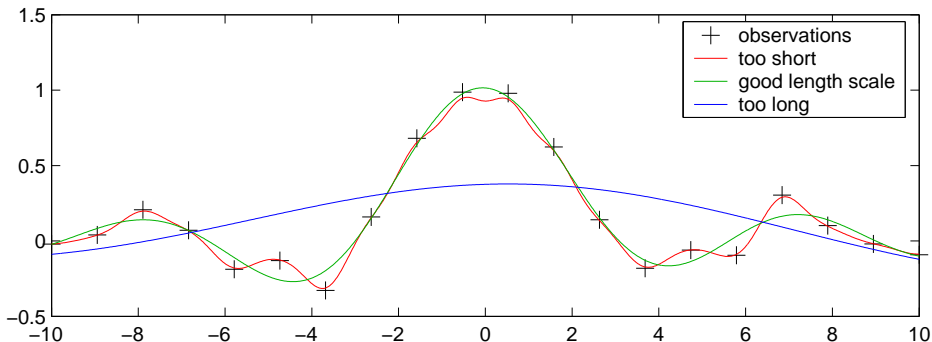
- the form of the covariance function, and
- any unknown (hyper-) parameters  $\theta$ .

This can be done by optimizing the marginal likelihood:

$$\frac{\partial \log p(\mathbf{y}|\mathbf{x}, \theta, M_i)}{\partial \theta_j} = \frac{1}{2}\mathbf{y}^\top \mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial \theta_j} \mathbf{K}^{-1} \mathbf{y} - \frac{1}{2} \text{trace}(\mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial \theta_j})$$

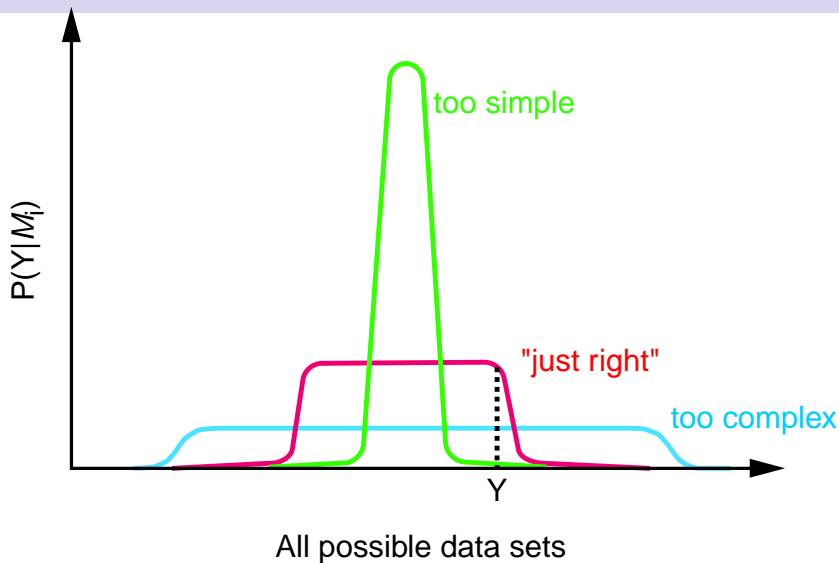
## Example: Fitting the length scale parameter

Parameterized covariance function:  $k(x, x') = \nu^2 \exp\left(-\frac{(x - x')^2}{2\ell^2}\right) + \sigma_n^2 \delta_{xx'}$ .



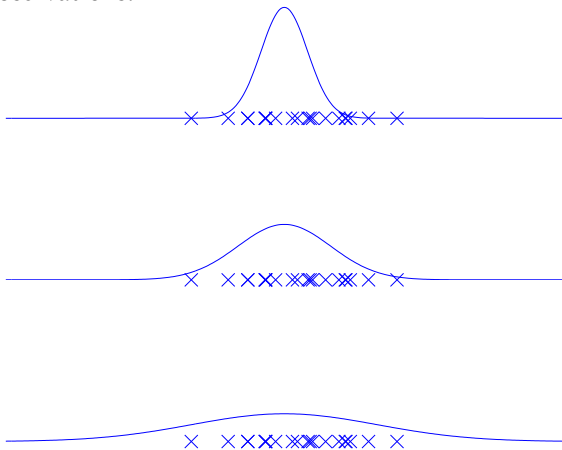
The mean posterior predictive function is plotted for 3 different length scales (the green curve corresponds to optimizing the marginal likelihood). **Notice, that an almost exact fit to the data can be achieved by reducing the length scale – but the marginal likelihood does not favour this!**

# Why, in principle, does Bayesian Inference work? Occam's Razor



# An illustrative analogous example

Imagine the simple task of fitting the variance,  $\sigma^2$ , of a zero-mean Gaussian to a set of  $n$  scalar observations.



The log likelihood is  $\log p(\mathbf{y}|\boldsymbol{\mu}, \sigma^2) = -\frac{1}{2}\mathbf{y}^\top \mathbf{I} \mathbf{y} / \sigma^2 - \frac{1}{2} \log |\mathbf{I} \sigma^2| - \frac{n}{2} \log(2\pi)$

# From random functions to covariance functions

Consider the class of linear functions:

$$f(x) = ax + b, \text{ where } a \sim \mathcal{N}(0, \alpha), \text{ and } b \sim \mathcal{N}(0, \beta).$$

We can compute the mean function:

$$\mu(x) = E[f(x)] = \iint f(x)p(a)p(b)dadb = \int axp(a)da + \int bp(b)db = 0,$$

and covariance function:

$$\begin{aligned} k(x, x') &= E[(f(x) - 0)(f(x') - 0)] = \iint (ax + b)(ax' + b)p(a)p(b)dadb \\ &= \int a^2xx'p(a)da + \int b^2p(b)db + (x + x') \int abp(a)p(b)dadb = \alpha xx' + \beta. \end{aligned}$$

# From random functions to covariance functions II

Consider the class of functions (sums of squared exponentials):

$$\begin{aligned} f(x) &= \lim_{n \rightarrow \infty} \frac{1}{n} \sum_i \gamma_i \exp(-(x - i/n)^2), \quad \text{where } \gamma_i \sim \mathcal{N}(0, 1), \forall i \\ &= \int_{-\infty}^{\infty} \gamma(u) \exp(-(x - u)^2) du, \quad \text{where } \gamma(u) \sim \mathcal{N}(0, 1), \forall u. \end{aligned}$$

The mean function is:

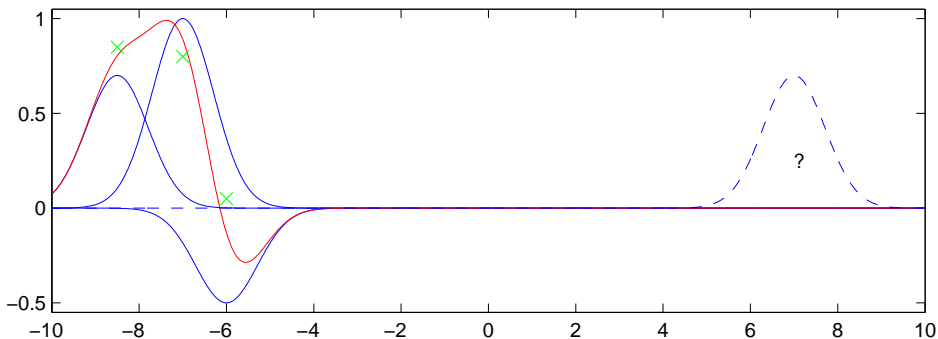
$$\mu(x) = E[f(x)] = \int_{-\infty}^{\infty} \exp(-(x - u)^2) \int_{-\infty}^{\infty} \gamma p(\gamma) d\gamma du = 0,$$

and the covariance function:

$$\begin{aligned} E[f(x)f(x')] &= \int \exp(-(x - u)^2 - (x' - u)^2) du \\ &= \int \exp\left(-2\left(u - \frac{x + x'}{2}\right)^2 + \frac{(x + x')^2}{2} - x^2 - x'^2\right) du \propto \exp\left(-\frac{(x - x')^2}{2}\right). \end{aligned}$$

Thus, the squared exponential covariance function is equivalent to regression using infinitely many Gaussian shaped basis functions placed everywhere, **not just at your training points!**

# Using finitely many basis functions may be dangerous!



# Spline models

One dimensional minimization problem: find the function  $f(x)$  which minimizes:

$$\sum_{i=1}^c (f(x^{(i)}) - y^{(i)})^2 + \lambda \int_0^1 (f''(x))^2 dx,$$

where  $0 < x^{(i)} < x^{(i+1)} < 1$ ,  $\forall i = 1, \dots, n-1$ , has as solution the **Natural Smoothing Cubic Spline**: first order polynomials when  $x \in [0; x^{(1)}]$  and when  $x \in [x^{(n)}; 1]$  and a cubic polynomial in each  $x \in [x^{(i)}; x^{(i+1)}]$ ,  $\forall i = 1, \dots, n-1$ , joined to have continuous second derivatives at the knots.

The identical function is also the mean of a Gaussian process: Consider the class a functions given by:

$$f(x) = \alpha + \beta x + \lim_{n \rightarrow \infty} \frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} \gamma_i (x - \frac{i}{n})_+, \quad \text{where } (x)_+ = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

with Gaussian priors:

$$\alpha \sim \mathcal{N}(0, \xi), \quad \beta \sim \mathcal{N}(0, \xi), \quad \gamma_i \sim \mathcal{N}(0, \Gamma), \quad \forall i = 0, \dots, n-1.$$

The covariance function becomes:

$$\begin{aligned}
 k(x, x') &= \xi + xx'\xi + \Gamma \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=0}^{n-1} (x - \frac{i}{n})_+ (x' - \frac{i}{n})_+ \\
 &= \xi + xx'\xi + \Gamma \int_0^1 (x - u)_+ (x' - u)_+ du \\
 &= \xi + xx'\xi + \Gamma \left( \frac{1}{2} |x - x'| \min(x, x')^2 + \frac{1}{3} \min(x, x')^3 \right).
 \end{aligned}$$

In the limit  $\xi \rightarrow \infty$  and  $\lambda = \sigma_n^2/\Gamma$  the posterior mean becomes the natural cubic spline.

We can thus find the hyperparameters  $\sigma^2$  and  $\Gamma$  (and thereby  $\lambda$ ) by maximising the marginal likelihood in the usual way.

Defining  $h(x) = (1, x)^\top$  the posterior predictions with mean and variance:

$$\tilde{\mu}(X_*) = H(X_*)^\top \beta + K(X, X_*) [K(X, X) + \sigma_n^2 I]^{-1} (\mathbf{y} - H(X)^\top \beta)$$

$$\tilde{\Sigma}(x_*) = \Sigma(X_*) + R(X, X_*)^\top A(X)^{-1} R(X, X_*)$$

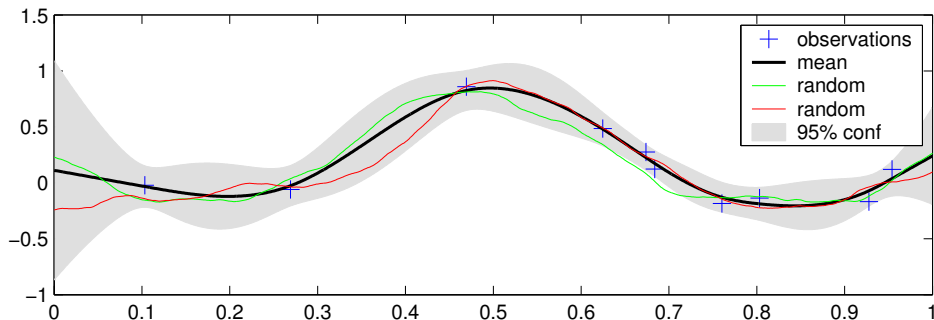
$$\beta = A(X)^{-1} H(X) [K + \sigma_n^2 I]^{-1} \mathbf{y}, \quad A(X) = H(X) [K(X, X) + \sigma_n^2 I]^{-1} H(X)^\top$$

$$R(X, X_*) = H(X_*) - H(X) [K + \sigma_n^2 I]^{-1} K(X, X_*)$$

# Cubic Splines, Example

Although this is not the fastest way to compute splines, it offers a principled way of finding hyperparameters, and uncertainties on predictions.

Note also, that although the **posterior mean** is smooth (piecewise cubic), posterior sample functions are not.



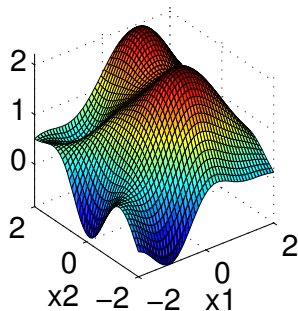
# Model Selection in Practise; Hyperparameters

There are two types of task: *form* and *parameters* of the covariance function.

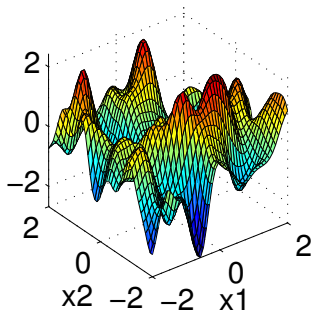
Typically, our prior is too weak to quantify aspects of the covariance function. We use a **hierarchical model** using **hyperparameters**. Eg, in ARD:

$$k(\mathbf{x}, \mathbf{x}') = v_0^2 \exp\left(-\sum_{d=1}^D \frac{(x_d - x'_d)^2}{2v_d^2}\right), \quad \text{hyperparameters } \theta = (v_0, v_1, \dots, v_d, \sigma_n^2).$$

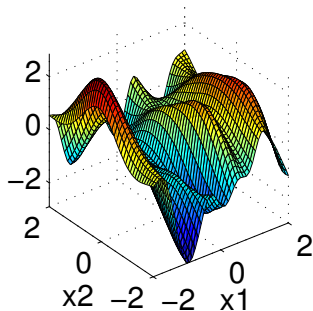
$v_1=v_2=1$



$v_1=v_2=0.32$



$v_1=0.32$  and  $v_2=1$



# Rational quadratic covariance function

The *rational quadratic* (RQ) covariance function:

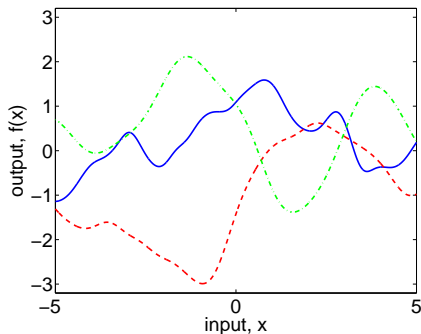
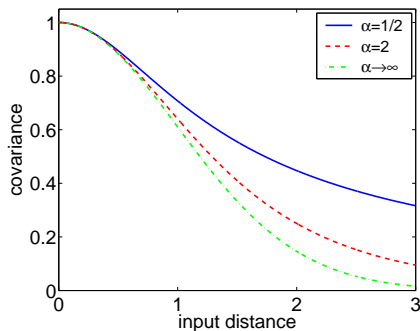
$$k_{\text{RQ}}(r) = \left(1 + \frac{r^2}{2\alpha\ell^2}\right)^{-\alpha}$$

with  $\alpha, \ell > 0$  can be seen as a *scale mixture* (an infinite sum) of squared exponential (SE) covariance functions with different characteristic length-scales.

Using  $\tau = \ell^{-2}$  and  $p(\tau|\alpha, \beta) \propto \tau^{\alpha-1} \exp(-\alpha\tau/\beta)$ :

$$\begin{aligned} k_{\text{RQ}}(r) &= \int p(\tau|\alpha, \beta) k_{\text{SE}}(r|\tau) d\tau \\ &\propto \int \tau^{\alpha-1} \exp\left(-\frac{\alpha\tau}{\beta}\right) \exp\left(-\frac{\tau r^2}{2}\right) d\tau \propto \left(1 + \frac{r^2}{2\alpha\ell^2}\right)^{-\alpha}, \end{aligned}$$

# Rational quadratic covariance function II



The limit  $\alpha \rightarrow \infty$  of the RQ covariance function is the SE.

# Matérn covariance functions

Stationary covariance functions can be based on the Matérn form:

$$k(\mathbf{x}, \mathbf{x}') = \frac{1}{\Gamma(\nu)2^{\nu-1}} \left[ \frac{\sqrt{2\nu}}{\ell} |\mathbf{x} - \mathbf{x}'| \right]^\nu K_\nu \left( \frac{\sqrt{2\nu}}{\ell} |\mathbf{x} - \mathbf{x}'| \right),$$

where  $K_\nu$  is the modified Bessel function of second kind of order  $\nu$ , and  $\ell$  is the characteristic length scale.

Sample functions from Matérn forms are  $\lfloor \nu - 1 \rfloor$  times differentiable. Thus, the hyperparameter  $\nu$  can control the degree of smoothness

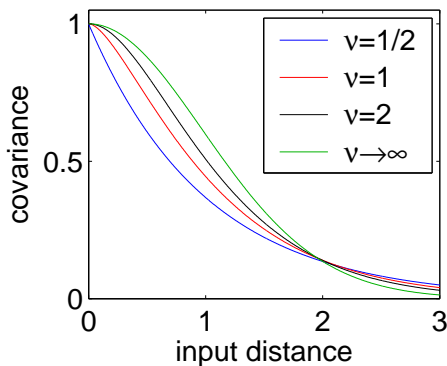
Special cases:

- $k_{\nu=1/2}(r) = \exp(-\frac{r}{\ell})$ : Laplacian covariance function, Brownian motion (Ornstein-Uhlenbeck)
- $k_{\nu=3/2}(r) = (1 + \frac{\sqrt{3}r}{\ell}) \exp(-\frac{\sqrt{3}r}{\ell})$  (once differentiable)
- $k_{\nu=5/2}(r) = (1 + \frac{\sqrt{5}r}{\ell} + \frac{5r^2}{3\ell^2}) \exp(-\frac{\sqrt{5}r}{\ell})$  (twice differentiable)
- $k_{\nu \rightarrow \infty} = \exp(-\frac{r^2}{2\ell^2})$ : smooth (infinitely differentiable)

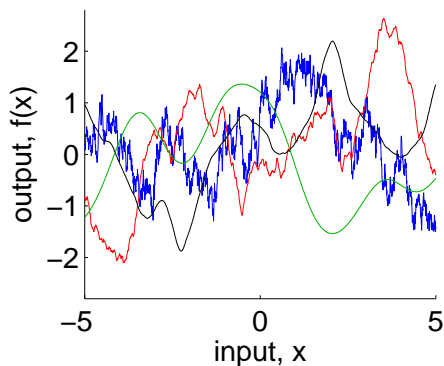
# Matérn covariance functions II

Univariate Matérn covariance function with unit characteristic length scale and unit variance:

covariance function



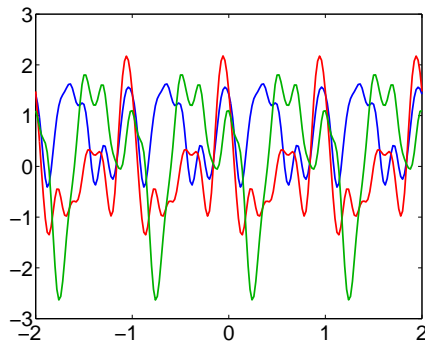
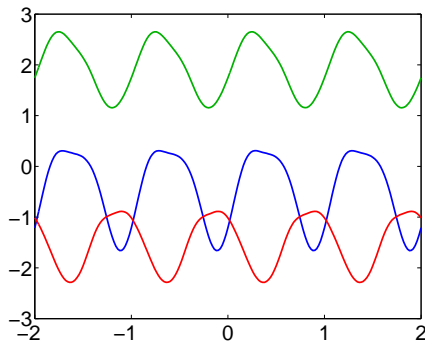
sample functions



# Periodic, smooth functions

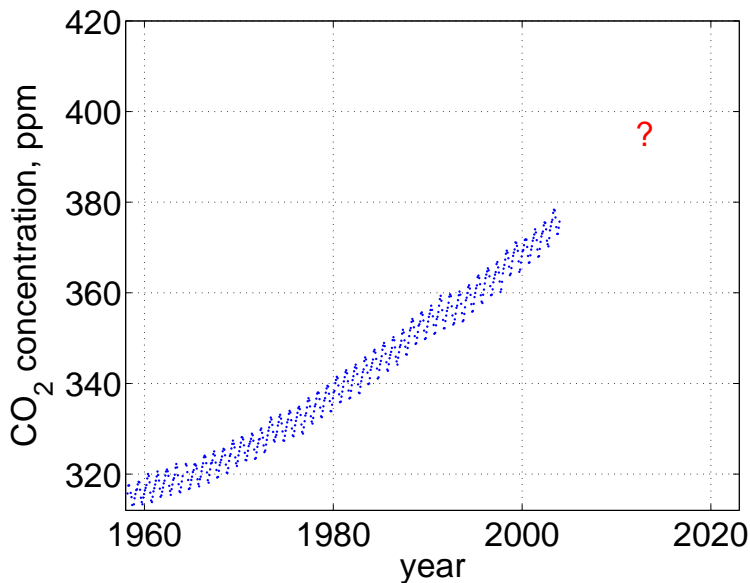
To create a distribution over periodic functions of  $x$ , we can first map the inputs to  $u = (\sin(x), \cos(x))^T$ , and then measure distances in the  $u$  space. Combined with the SE covariance function, which characteristic length scale  $\ell$ , we get:

$$k_{\text{periodic}}(x, x') = \exp(-2 \sin^2(\pi(x - x'))/\ell^2)$$



Three functions drawn at random; left  $\ell > 1$ , and right  $\ell < 1$ .

# The Prediction Problem



# Covariance Function

The covariance function consists of several terms, parameterized by a total of 11 *hyperparameters*:

- long-term smooth trend (**squared exponential**)

$$k_1(x, x') = \theta_1^2 \exp(-(x - x')^2 / \theta_2^2),$$

- seasonal trend (**quasi-periodic smooth**)

$$k_2(x, x') = \theta_3^2 \exp\left(-2 \sin^2(\pi(x - x')) / \theta_5^2\right) \times \exp\left(-\frac{1}{2}(x - x')^2 / \theta_4^2\right),$$

- short- and medium-term anomaly (**rational quadratic**)

$$k_3(x, x') = \theta_6^2 \left(1 + \frac{(x - x')^2}{2\theta_8\theta_7^2}\right)^{-\theta_8}$$

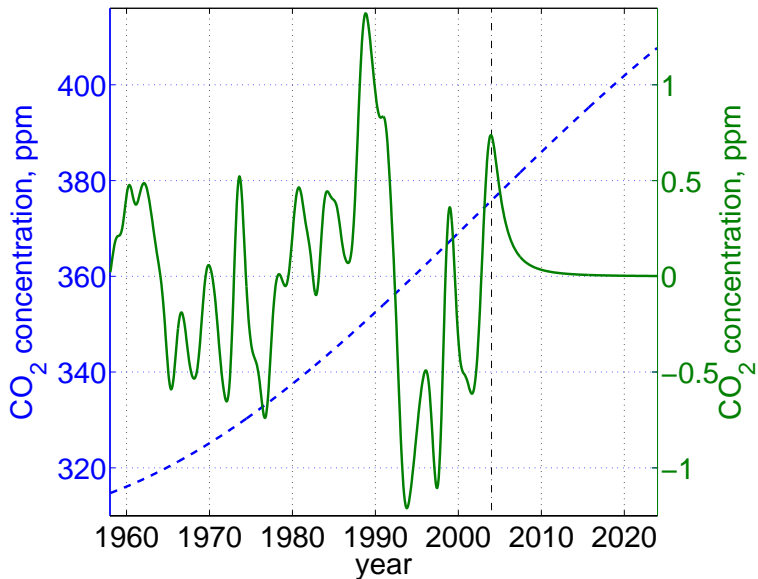
- noise (**independent Gaussian, and dependent**)

$$k_4(x, x') = \theta_9^2 \exp\left(-\frac{(x - x')^2}{2\theta_{10}^2}\right) + \theta_{11}^2 \delta_{xx'}.$$

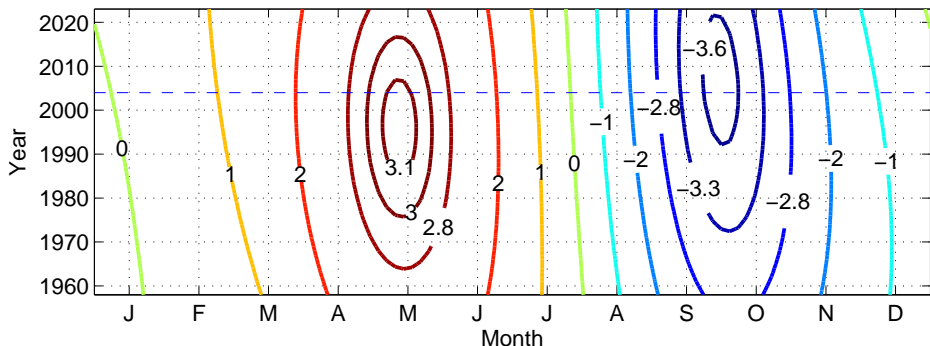
$$k(x, x') = k_1(x, x') + k_2(x, x') + k_3(x, x') + k_4(x, x')$$

Let's try this with the gpml software (<http://www.gaussianprocess.org/gpml>).

# Long- and medium-term mean predictions



# Mean Seasonal Component



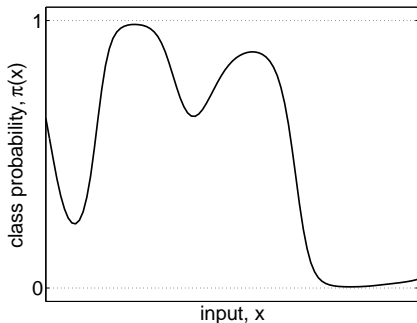
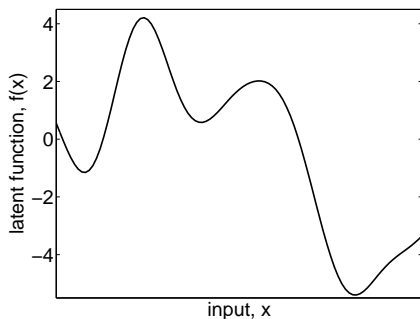
Seasonal component: magnitude  $\theta_3 = 2.4$  ppm, decay-time  $\theta_4 = 90$  years.

Dependent noise, magnitude  $\theta_9 = 0.18$  ppm, decay  $\theta_{10} = 1.6$  months.

Independent noise, magnitude  $\theta_{11} = 0.19$  ppm.

Optimize or integrate out? See MacKay [3].

# Binary Gaussian Process Classification



The class probability is related to the *latent* function,  $f$ , through:

$$p(y = 1 | f(\mathbf{x})) = \pi(\mathbf{x}) = \Phi(f(\mathbf{x})),$$

where  $\Phi$  is a sigmoid function, such as the **logistic** or **cumulative Gaussian**. Observations are independent given  $f$ , so the likelihood is

$$p(\mathbf{y} | \mathbf{f}) = \prod_{i=1}^n p(y_i | f_i) = \prod_{i=1}^n \Phi(y_i f_i).$$

# Prior and Posterior for Classification

We use a Gaussian process prior for the latent function:

$$\mathbf{f}|X, \theta \sim \mathcal{N}(\mathbf{0}, K)$$

The posterior becomes:

$$p(\mathbf{f}|\mathcal{D}, \theta) = \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|X, \theta)}{p(\mathcal{D}|\theta)} = \frac{\mathcal{N}(\mathbf{f}|\mathbf{0}, K)}{p(\mathcal{D}|\theta)} \prod_{i=1}^m \Phi(y_i|f_i),$$

which is non-Gaussian.

The latent value at the test point,  $f(\mathbf{x}^*)$  is

$$p(f_*|\mathcal{D}, \theta, \mathbf{x}_*) = \int p(f_*|\mathbf{f}, X, \theta, \mathbf{x}_*)p(\mathbf{f}|\mathcal{D}, \theta) d\mathbf{f},$$

and the predictive class probability becomes

$$p(y_*|\mathcal{D}, \theta, \mathbf{x}_*) = \int p(y_*|f_*)p(f_*|\mathcal{D}, \theta, \mathbf{x}_*) df_*,$$

both of which are intractable to compute.

# Gaussian Approximation to the Posterior

We approximate the non-Gaussian posterior by a Gaussian:

$$p(\mathbf{f}|\mathcal{D}, \theta) \simeq q(\mathbf{f}|\mathcal{D}, \theta) = \mathcal{N}(\mathbf{m}, A)$$

then  $q(f_*|\mathcal{D}, \theta, \mathbf{x}_*) = \mathcal{N}(f_*|\mu_*, \sigma_*^2)$ , where

$$\mu_* = \mathbf{k}_*^\top K^{-1} \mathbf{m}$$

$$\sigma_*^2 = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\top (K^{-1} - K^{-1} A K^{-1}) \mathbf{k}_*.$$

Using this approximation with the cumulative Gaussian likelihood

$$q(y_* = 1|\mathcal{D}, \theta, \mathbf{x}_*) = \int \Phi(f_*) \mathcal{N}(f_*|\mu_*, \sigma_*^2) df_* = \Phi\left(\frac{\mu_*}{\sqrt{1 + \sigma_*^2}}\right)$$

# Laplace's method and Expectation Propagation

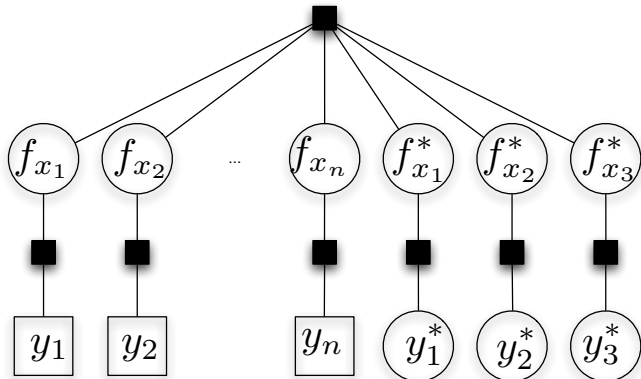
How do we find a good Gaussian approximation  $\mathcal{N}(\mathbf{m}, A)$  to the posterior?

**Laplace's method:** Find the Maximum A Posteriori (MAP) latent values  $\mathbf{f}_{\text{MAP}}$ , and use a local expansion (Gaussian) around this point as suggested by Williams and Barber [8].

**Variational bounds:** bound the likelihood by some tractable expression  
A **local variational bound for each likelihood term** was given by Gibbs and MacKay [1]. A **lower bound based on Jensen's inequality** by Opper and by Seeger [5].

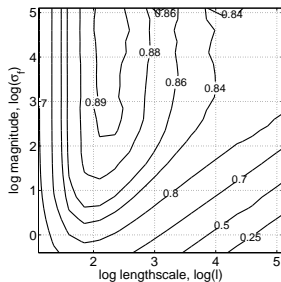
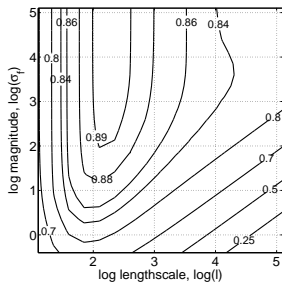
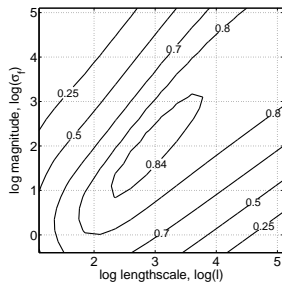
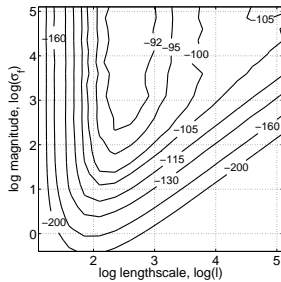
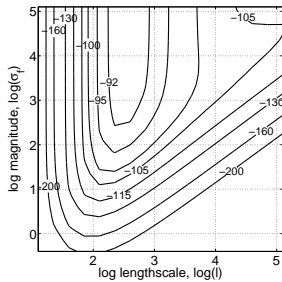
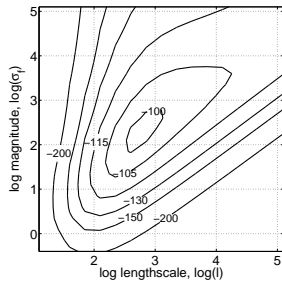
**Expectation Propagation:** use an approximation of the likelihood, such that the moments of the marginals of the approximate posterior match the (approximate) moment of the posterior, Minka [4].

# Expectation Propagation

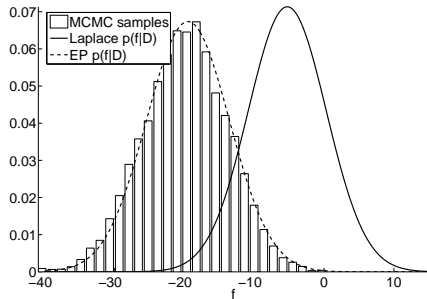
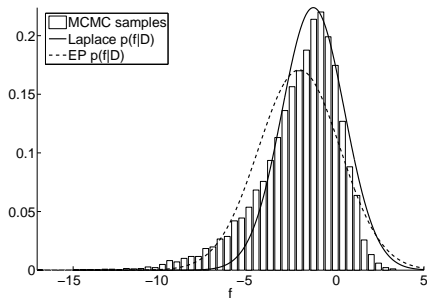


$$\tilde{f}_i = \frac{\text{proj}[p(y_i|f_i)q^i]}{q^i}, \quad q^i = p(\mathbf{f}) \prod_{j \neq i} \tilde{f}_j.$$

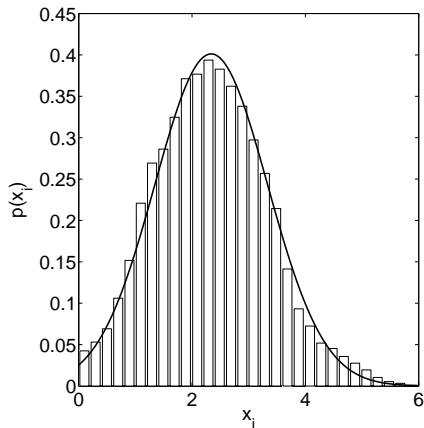
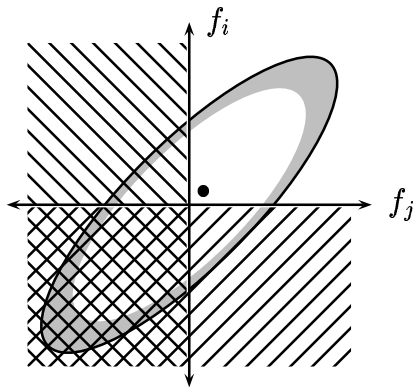
# USPS Digits, 3s vs 5s



# USPS Digits, 3s vs 5s

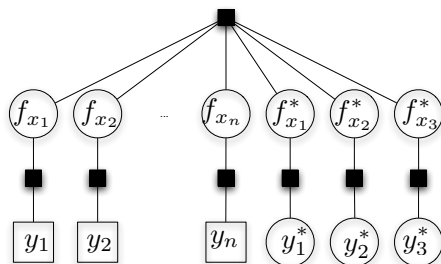


# The Structure of the posterior



# Sparse Approximations

Recall the graphical model for a Gaussian process. Inference is expensive because the latent variables are fully connected.



Exact inference:  $\mathcal{O}(n^3)$ .

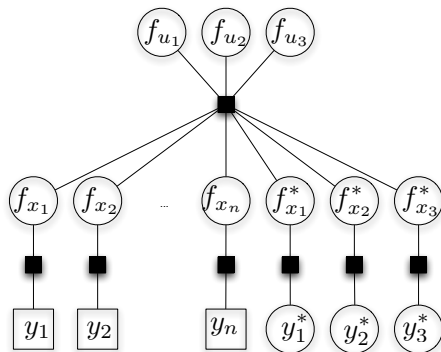
Sparse approximations: solve a smaller, sparse, approximation of the original problem.

Algorithm: Subset of data.

Are there better ways to sparsify?

# Inducing Variables

Because of the marginalization property, we can introduce more latent variables without changing the distribution of the original variables.



The  $\mathbf{u} = (u_1, u_2, \dots)^\top$  are called *inducing variables*.

The inducing variables have associated *inducing inputs*,  $\mathbf{s}$ , but no associated output values.

The marginalization property ensures that

$$p(\mathbf{f}, \mathbf{f}^*) = \int p(\mathbf{f}, \mathbf{f}^*, \mathbf{u}) d\mathbf{u}$$

# The Central Approximations

In a unifying treatment, Candela and Rasmussen [2] assume that training and test sets are *conditionally independent* given  $\mathbf{u}$ .

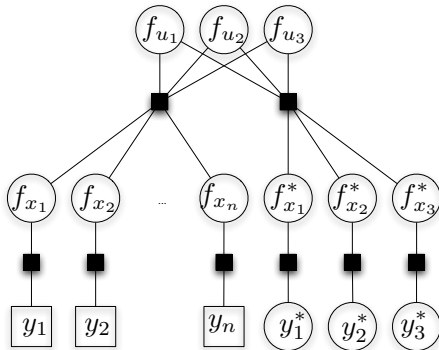
Assume:  $p(\mathbf{f}, \mathbf{f}_*) \simeq q(\mathbf{f}, \mathbf{f}_*)$ , where

$$q(\mathbf{f}, \mathbf{f}_*) = \int q(\mathbf{f}_*|\mathbf{u})q(\mathbf{f}|\mathbf{u})p(\mathbf{u})d\mathbf{u}.$$

The inducing variables *induce* the dependencies between training and test cases.

Different sparse algorithms in the literature correspond to different

- choices of the inducing inputs
- further approximations



# Training and test conditionals

The exact training and test conditionals are:

$$p(\mathbf{f}|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}}K_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{u}, K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}})$$
$$p(\mathbf{f}_*|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f}_*,\mathbf{u}}K_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{u}, K_{\mathbf{f}_*,\mathbf{f}_*} - Q_{\mathbf{f}_*,\mathbf{f}_*}),$$

where  $Q_{\mathbf{a},\mathbf{b}} = K_{\mathbf{a},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},\mathbf{b}}$ .

These equations are easily recognized as the usual predictive equations for GPs.

The *effective prior* is:

$$q(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K_{\mathbf{f},\mathbf{f}} & Q_{*,\mathbf{f}} \\ Q_{\mathbf{f},*} & K_{*,*} \end{bmatrix}\right)$$

## Example: Sparse parametric Gaussian processes

Snelson and Ghahramani [6] introduced the idea of sparse GP inference based on a pseudo data set, integrating out the targets, and optimizing the inputs.

Equivalently, in the unifying scheme:

$$\begin{aligned}q(\mathbf{f}|\mathbf{u}) &= \mathcal{N}(K_{\mathbf{f},\mathbf{u}}K_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{u}, \text{diag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}]) \\q(\mathbf{f}_*|\mathbf{u}) &= p(\mathbf{f}_*|\mathbf{u}).\end{aligned}$$

The effective prior becomes

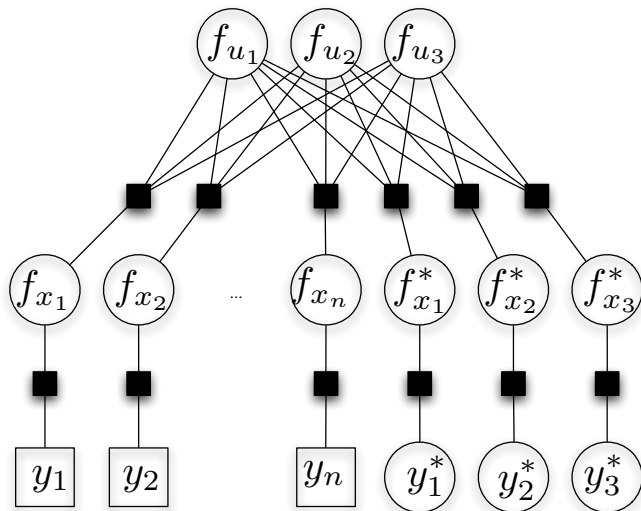
$$q_{\text{FITC}}(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} - \text{diag}[Q_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{f}}] & Q_{*,\mathbf{f}} \\ Q_{\mathbf{f},*} & K_{*,*} \end{bmatrix}\right),$$

which can be computed efficiently.

The Bayesian Committee Machine [7] uses block diag instead of diag, and the inducing variables to be the test cases.

# The FITC Factor Graph

Sparse Parametric Gaussian Process  $\equiv$  Fully Independent Training Conditional



Complex non-linear inference problems can be solved by manipulating plain old Gaussian distributions

# More on Gaussian Processes

Gaussian Processes for  
Machine Learning



Carl Edward Rasmussen and Christopher K. I. Williams

Rasmussen and Williams

*Gaussian Processes for Machine Learning*,  
MIT Press, 2006.

<http://www.GaussianProcess.org/gpml>

Gaussian process web (code, papers, etc): <http://www.GaussianProcess.org>

# A few references

- [1] Gibbs, M. N. and MacKay, D. J. C. (2000). Variational Gaussian Process Classifiers. *IEEE Transactions on Neural Networks*, 11(6):1458–1464.
- [2] Joaquin Quiñonero-Candela and Carl Edward Rasmussen (2005). A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959.
- [3] MacKay, D. J. C. (1999). Comparison of Approximate Methods for Handling Hyperparameters. *Neural Computation*, 11(5):1035–1068.
- [4] Minka, T. P. (2001). *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology.
- [5] Seeger, M. (2003). *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. PhD thesis, School of Informatics, University of Edinburgh. <http://www.cs.berkeley.edu/~mseeger>.
- [6] Snelson, E. and Ghahramani, Z. (2006). Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18*. MIT Press.
- [7] Tresp, V. (2000). A Bayesian Committee Machine. *Neural Computation*, 12(11):2719–2741.
- [8] Williams, C. K. I. and Barber, D. (1998). Bayesian Classification with Gaussian Processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351.