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

![Graph showing CO₂ concentration over years](image)

**CO₂ concentration, ppm**

Year:
- 1960
- 1980
- 2000
- 2020

**CO₂ concentration, ppm**

- 320
- 340
- 360
- 380
- 400
- 420

Rasmussen (Engineering, Cambridge)

Gaussian Process Regression

August 30th - September 10th, 2009
The Prediction Problem

\[ \text{CO}_2 \text{ concentration, ppm} \]

\[ \text{year} \]

Rasmussen (Engineering, Cambridge)
The Prediction Problem

CO₂ concentration, ppm

year
The Prediction Problem

\[ \text{CO}_2 \text{ concentration, ppm} \]

\[ 1960 \quad 1980 \quad 2000 \quad 2020 \]

\[ 320 \quad 340 \quad 360 \quad 380 \quad 400 \quad 420 \]

year

Rasmussen (Engineering, Cambridge)
Maximum likelihood, parametric model

Supervised parametric learning:

- data: $x, y$
- model: $y = f_w(x) + \varepsilon$

Gaussian likelihood:

$$p(y|x, w) \propto \prod_c \exp\left(-\frac{1}{2}(y_c - f_w(x_c))^2 / \sigma^2_{\text{noise}} \right).$$

Maximize the likelihood:

$$w_{\text{ML}} = \arg\max_w p(y|x, w).$$

Make predictions, by plugging in the ML estimate:

$$p(y^*|x^*, w_{\text{ML}})$$
Bayesian Inference, parametric model

Supervised parametric learning:

- data: \( x, y \)
- model: \( y = f_w(x) + \epsilon \)

Gaussian likelihood:

\[
p(y|x, w) \propto \prod_c \exp\left(-\frac{1}{2} (y_c - f_w(x_c))^2 / \sigma^2_{\text{noise}}\right).
\]

Parameter prior:

\[
p(w)
\]

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

\[
p(w|x, y) = \frac{p(w)p(y|x, w)}{p(y|x)}
\]
Bayesian Inference, parametric model, cont.

Making predictions:

\[ p(y^*|x^*, x, y) = \int p(y^*|w, x^*) p(w|x, y) \, dw \]

Marginal likelihood:

\[ p(y|x) = \int p(w)p(y|x, w) \, dw. \]
The Gaussian distribution is given by

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

where \( \mu \) is the mean vector and \( \Sigma \) the covariance matrix.
Both the **conditionals** and the **marginals** of a joint Gaussian are again Gaussian.
In algebra, if \( x \) and \( y \) are jointly Gaussian

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

the marginal distribution of \( x \) is

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

and the conditional distribution of \( x \) given \( y \) is

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

where \( x \) and \( y \) 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.

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

\[
 f = (f_1, \ldots, f_n)\top \sim \mathcal{N}(\mu, \Sigma), \quad \text{indexes } i = 1, \ldots, n
\]

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:

\[ p(x) = \int p(x, y) dy. \]

For Gaussians:

\[ p(x, y) = \mathcal{N}\left( \begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix} \right) \implies p(x) = \mathcal{N}(a, A) \]
Random functions from a Gaussian Process

Example one dimensional Gaussian process:

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

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

\[ f \sim \mathcal{N}(0, \Sigma), \]

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

$$
z = \text{randn}(D,1);
$$
$$
y = \text{chol}(K)\,^t\!z + m;
$$

where $\text{chol}$ is the Cholesky factor $R$ such that $R^\top R = K$.

Thus, the covariance of $y$ is:

$$
\mathbb{E}[(y - \bar{y})(y - \bar{y})^\top] = \mathbb{E}[R^\top zz^\top R] = R^\top \mathbb{E}[zz^\top]R = R^\top IR = K.
$$
Sequential Generation

Factorize the joint distribution

$$p(f_1, \ldots, f_n | x_1, \ldots x_n) = \prod_{i=1}^{n} p(f_i | f_{i-1}, \ldots, f_1, x_i, \ldots, x_1),$$

and generate function values sequentially.

What do the individual terms look like? For Gaussians:

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

Do try this at home!
Supervised parametric learning:

- data: \( x, y \)
- model: \( y = f_w(x) + \varepsilon \)

Gaussian likelihood:

\[
p(y|x, w, M_i) \propto \prod_c \exp\left(-\frac{1}{2}(y_c - f_w(x_c))^2 / \sigma^2_{\text{noise}}\right).
\]

Maximize the likelihood:

\[
w_{\text{ML}} = \arg\max_w p(y|x, w, M_i).
\]

Make predictions, by plugging in the ML estimate:

\[
p(y^*|x^*, w_{\text{ML}}, M_i)
\]
Bayesian Inference, parametric model

Supervised parametric learning:

- data: x, y
- model: \( y = f_w(x) + \varepsilon \)

Gaussian likelihood:

\[
p(y|x, w, M_i) \propto \prod_c \exp\left(-\frac{1}{2}(y_c - f_w(x_c))^2 / \sigma_{\text{noise}}^2\right).
\]

Parameter prior:

\[
p(w|M_i)
\]

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

\[
p(w|x, y, M_i) = \frac{p(w|M_i)p(y|x, w, M_i)}{p(y|x, M_i)}
\]
Bayesian Inference, parametric model, cont.

Making predictions:

\[ p(y^* | x^*, x, y, M_i) = \int p(y^* | w, x^*, M_i) p(w | x, y, M_i) dw \]

Marginal likelihood:

\[ p(y | x, M_i) = \int p(w | M_i) p(y | x, w, M_i) dw. \]

Model probability:

\[ p(M_i | x, y) = \frac{p(M_i) p(y | x, M_i)}{p(y | x)} \]

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:

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

(Zero mean) Gaussian process prior:

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

 Leads to a Gaussian process posterior

\[ f(x) | x, y, M_i \sim \mathcal{GP}(m_{\text{post}}(x) = k(x, x)[K(x, x) + \sigma_{\text{noise}}^2 I]^{-1}y, \]

\[ k_{\text{post}}(x, x') = k(x, x') - k(x, x)[K(x, x) + \sigma_{\text{noise}}^2 I]^{-1}k(x, x') \].

And a Gaussian predictive distribution:

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

\[ k(x^*, x^*) + \sigma_{\text{noise}}^2 - k(x^*, x)\top [K + \sigma_{\text{noise}}^2 I]^{-1}k(x^*, x) \)
Prior and Posterior

Predictive distribution:

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

\[
\begin{align*}
  f_\ast | X_\ast, X, y & \sim \mathcal{N}(K(X_\ast, X)[K(X, X) + \sigma_n^2 I]^{-1} y,
  
  & \quad (K(X_\ast, X_\ast) - K(X_\ast, X)[K(X, X) + \sigma_n^2 I]^{-1} K(X, X_\ast)).
\end{align*}
\]

The mean is linear in two ways:

\[
\mu(x_\ast) = k(x_\ast, X)[K(X, X) + \sigma_n^2]^{-1} y = \sum_{c=1}^{n} \beta_c y^{(c)} = \sum_{c=1}^{n} \alpha_c k(x_\ast, x^{(c)}).
\]

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

The variance is the difference between two terms:

\[
V(x_\ast) = k(x_\ast, x_\ast) - k(x_\ast, X)[K(X, X) + \sigma_n^2 I]^{-1} k(X, x_\ast),
\]

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

Log marginal likelihood:

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

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:

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

P(Y|M_i)

All possible data sets

"just right"

too simple

too complex
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

\[
\log p(y|\mu, \sigma^2) = -\frac{1}{2} y^\top I y / \sigma^2 - \frac{1}{2} \log |I\sigma^2| - \frac{n}{2} \log(2\pi)
\]
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:

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

and covariance function:

\[ k(x, x') = E[(f(x) - 0)(f(x') - 0)] = \int \int (ax + b)(ax' + b)p(a)p(b)dadb \]

\[ = \int a^2xx'p(a)da + \int b^2p(b)db + (x + x')\int abp(a)p(b)dadb = \alpha xx' + \beta. \]
Consider the class of functions (sums of squared exponentials):

\[
f(x) = \lim_{n \to \infty} \frac{1}{n} \sum_{i} \gamma_i \exp(-(x - i/n)^2), \quad \text{where } \gamma_i \sim \mathcal{N}(0, 1), \ \forall i
\]

\[
= \int_{-\infty}^{\infty} \gamma(u) \exp(-(x - u)^2) du, \quad \text{where } \gamma(u) \sim \mathcal{N}(0, 1), \ \forall u.
\]

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:

\[
E[f(x)f(x')] = \int \exp\left(- (x - u)^2 - (x' - u)^2\right) du
\]

\[
= \int \exp\left(- 2(u - \frac{x + x'}{2})^2 + \frac{(x + x')^2}{2} - x^2 - x'^2\right) du \propto \exp\left(- \frac{(x - x')^2}{2}\right).
\]

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 $0 < x^{(i)} < x^{(i+1)} < 1$, $\forall i = 1, \ldots, n - 1$, 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 $x \in [x^{(i)}; x^{(i+1)}]$, $\forall i = 1, \ldots, n - 1$, joined to have continuous second derivatives at the knots.

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

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

with Gaussian priors:

$$
\alpha \sim \mathcal{N}(0, \xi), \quad \beta \sim \mathcal{N}(0, \xi), \quad \gamma_i \sim \mathcal{N}(0, \Gamma), \quad \forall i = 0, \ldots, n - 1.
$$
The covariance function becomes:

\[
k(x, x') = \xi + xx'\xi + \Gamma \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} (x - \frac{i}{n})_+ (x' - \frac{i}{n})_+ \\
= \xi + xx'\xi + \Gamma \int_0^1 (x - u)_+ (x' - u)_+ du \\
= \xi + xx'\xi + \Gamma \left( \frac{1}{2} |x - x'| \min(x, x')^2 + \frac{1}{3} \min(x, x')^3 \right).
\]

In the limit \( \xi \to \infty \) and \( \lambda = \sigma^2_n / \Gamma \) 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:

\[
\tilde{\mu}(X_*) = H(X_*)^\top \beta + K(X, X_*)[K(X, X) + \sigma^2_n I]^{-1} (y - H(X)^\top \beta) \\
\tilde{\Sigma}(x_*) = \Sigma(x_*) + R(X, X_*)^\top A(X)^{-1} R(X, X_*) \\
\beta = A(X)^{-1} H(X) [K + \sigma^2_n I]^{-1} y, \quad A(X) = H(X) [K(X, X) + \sigma^2_n I]^{-1} H(X)^\top \\
R(X, X_*) = H(X_*) - H(X)[K + \sigma^2_n I]^{-1} K(X, X_*)
\]
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.
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:

\[ k(x, x') = v_0^2 \exp \left(- \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{2v_d^2} \right), \]

hyperparameters \( \theta = (v_0, v_1, \ldots, v_d, \sigma_n^2) \).

\[ \begin{align*}
  v1 &= v2 = 1 \\
  v1 &= v2 = 0.32 \\
  v1 &= 0.32 \text{ and } v2 = 1
\end{align*} \]
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 \( p(\tau|\alpha, \beta) \propto \tau^{\alpha-1} \exp(-\alpha \tau/\beta) \):

\[
k_{\text{RQ}}(r) = \int p(\tau|\alpha, \beta) k_{\text{SE}}(r|\tau) d\tau \\
\propto \int \tau^{\alpha-1} \exp\left(-\frac{\alpha \tau}{\beta}\right) \exp\left(-\frac{\tau r^2}{2}\right) d\tau \propto \left( 1 + \frac{r^2}{2\alpha\ell^2} \right)^{-\alpha},
\]
The limit $\alpha \to \infty$ of the RQ covariance function is the SE.
Matérn covariance functions

Stationary covariance functions can be based on the Matérn form:

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

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 \([\nu - 1]\) 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, Browninan 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)
- \(k_{\nu \to \infty} = \exp(-\frac{r^2}{2\ell^2})\): smooth (infinitely differentiable)
Univariate Matérn covariance function with unit characteristic length scale and unit variance:

![Covariance function](image1)

![Sample functions](image2)

\( \nu = \frac{1}{2} \)

\( \nu = 1 \)

\( \nu = 2 \)

\( \nu \to \infty \)
Periodic, smooth functions

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

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

Three functions drawn at random; left \( \ell > 1 \), and right \( \ell < 1 \).
The Prediction Problem

CO₂ concentration, ppm

year

1960 1980 2000 2020
320 340 360 380 400 420

Gaussian Process Regression

Rasmussen (Engineering, Cambridge)

August 30th - September 10th, 2009
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

CO₂ concentration, ppm

year


320 340 360 380 400

CO₂ concentration, ppm

−1 0 0.5 1

Gaussian Process Regression
August 30th - September 10th, 2009
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].
The class probability is related to the *latent* function, $f$, through:

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

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

$$p(y|f) = \prod_{i=1}^{n} p(y_i|f_i) = \prod_{i=1}^{n} \Phi(y_if_i).$$
Prior and Posterior for Classification

We use a Gaussian process prior for the latent function:

\[ f|X, \theta \sim \mathcal{N}(0, K) \]

The posterior becomes:

\[
p(f|D, \theta) = \frac{p(y|f)p(f|X, \theta)}{p(D|\theta)} = \frac{\mathcal{N}(f|0, K)}{p(D|\theta)} \prod_{i=1}^{m} \Phi(y_if_i),
\]

which is non-Gaussian.

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

\[
p(f_*|D, \theta, x_*) = \int p(f_*|f, X, \theta, x_*) p(f|D, \theta) df,
\]

and the predictive class probability becomes

\[
p(y_*|D, \theta, x_*) = \int p(y_*|f_*) p(f_*|D, \theta, x_*) df_*,
\]

both of which are intractable to compute.
Gaussian Approximation to the Posterior

We approximate the non-Gaussian posterior by a Gaussian:

\[ p(f|D, \theta) \approx q(f|D, \theta) = N(m, A) \]

then \( q(f_*|D, \theta, x_*) = N(f_*|\mu_*, \sigma_*^2) \), where

\[
\mu_* = k_*^\top K^{-1} m \\
\sigma_*^2 = k(x_*, x_*) - k_*^\top (K^{-1} - K^{-1}AK^{-1}) k_*.
\]

Using this approximation with the cumulative Gaussian likelihood

\[
q(y_* = 1|D, \theta, x_*) = \int \Phi(f_*) N(f_*|\mu_*, \sigma_*^2) df_* = \Phi\left(\frac{\mu_*}{\sqrt{1 + \sigma_*^2}}\right)
\]
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**

\[
\tilde{f}_i = \frac{\text{proj}[p(y_i|f_i)q^i]}{q^i}, \quad q^i = p(f) \prod_{j \neq i} \tilde{f}_j.
\]
USPS Digits, 3s vs 5s

Rasmussen (Engineering, Cambridge)
Gaussian Process Regression
August 30th - September 10th, 2009 51 / 62
USPS Digits, 3s vs 5s

Rasmussen (Engineering, Cambridge)  Gaussian Process Regression  August 30th - September 10th, 2009  52 / 62
The Structure of the posterior

\[ p(x_i) \]

\[ f_i \]

\[ f_j \]
Sparse Approximations

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

Exact inference: $O(n^3)$.

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

Algorithm: Subset of data.

Are there better ways to sparsify?
Because of the marginalization property, we can introduce more latent variables without changing the distribution of the original variables.

\[ f(x_1), f(x_2), \ldots, f(x_n), f^*(x_1), f^*(x_2), f^*(x_3) \]

The \( u = (u_1, u_2, \ldots)^\top \) are called *inducing variables*.

The inducing variables have associated *inducing inputs*, \( s \), but no associated output values.

The marginalization property ensures that

\[ p(f, f^*) = \int p(f, f^*, u) du \]
The Central Approximations

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

Assume: $p(f, f_*) \simeq q(f, f_*)$, where

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

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
The exact training and test conditionals are:

\[
p(f|u) = \mathcal{N}(K_{f,u}K_{f,f}^{-1}u, K_{f,f} - Q_{f,f})
\]

\[
p(f_*|u) = \mathcal{N}(K_{f_*,u}K_{f,f}^{-1}u, K_{f_*,f_*} - Q_{f_*,f_*}),
\]

where \( Q_{a,b} = K_{a,u}K_{u,u}^{-1}K_{u,b} \).

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

The effective prior is:

\[
q(f, f_*) = \mathcal{N}(0, \begin{bmatrix} K_{f,f} & Q_{f_*,f} \\ Q_{f_*} & K_{f_*,f_*} \end{bmatrix})
\]
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:

\[ q(f|u) = \mathcal{N}(K_{f,u}K_{f,f}^{-1}u, \text{diag}[K_{f,f} - Q_{f,f}]) \]
\[ q(f_*|u) = p(f_*|u). \]

The effective prior becomes

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

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\text{Fully Independent Training Conditional}
Complex non-linear inference problems can be solved by manipulating plain old Gaussian distributions
More on Gaussian Processes

Rasmussen and Williams

Gaussian process web (code, papers, etc): http://www.GaussianProcess.org
A few references


