NEURAL COMPUTING
NCRG
RESEARCH GROUP

Neural Computing Research Group
Dept of Computer Science & Applied Mathematics
Aston University
Birmingham B4 7ET
United Kingdom
Tel: +44 (0)121 333 4631
Fax: +44 (0)121 333 4586
http://www.ncrg.aston.ac.uk/

PREDICTION WITH GAUSSIAN PROCESSES: FROM LINEAR REGRESSION TO LINEAR PREDICTION AND BEYOND

C. K. I. WILLIAMS

Technical Report NCRG/97/012 October 27, 1997

To appear in "Learning and Inference in Graphical Models"
ed. M. I. Jordan, Kluwer Academic Press, 1998

Abstract

The main aim of this paper is to provide a tutorial on regression with Gaussian processes. We start from Bayesian linear regression, and show how by a change of viewpoint one can see this method as a Gaussian process predictor based on priors over functions, rather than on priors over parameters. This leads in to a more general discussion of Gaussian processes in section 4. Section 5 deals with further issues, including hierarchical modelling and the setting of the parameters that control the Gaussian process, the covariance functions for neural network models and the use of Gaussian processes in classification problems.

1 Introduction

In the last decade neural networks have been used to tackle regression and classification problems, with some notable successes. It has also been widely recognized that they form a part of a wide variety of non-linear statistical techniques that can be used for these tasks; other methods include, for example, decision trees and kernel methods. The books by Bishop (1995) and Ripley (1996) provide excellent overviews.

One of the attractions of neural network models is their flexibility, i.e. their ability to model a wide variety of functions. However, this flexibility comes at a cost, in that a large number of parameters may need to be determined from the data, and consequently that there is a danger of "overfitting". Overfitting can be reduced by using weight regularization, but this leads to the awkward problem of specifying how to set the regularization parameters (e.g. the parameter α in the weight regularization term $\alpha \mathbf{w}^T \mathbf{w}$ for a weight vector $\mathbf{w}$.)

The Bayesian approach is to specify an hierarchical model with a prior distribution over hyperparameters such as α, then to specify the prior distribution of the weights relative to the hyperparameters. This is connected to data via an "observations" model; for example, in a regression context, the value of the dependent variable may be corrupted by Gaussian noise. Given an observed dataset, a posterior distribution over the weights and hyperparameters (rather than just a point estimate) will be induced. However, for neural network models this posterior cannot usually be obtained analytically; computational methods used include approximations (MacKay, 1992) or the evaluation of integrals using Monte Carlo methods (Neal, 1996).

In the Bayesian approach to neural networks, a prior on the weights of a network induces a prior over functions. An alternative method of putting a prior over functions is to use a *Gaussian process* (GP) prior over functions. This idea has been used for a long time in the spatial statistics community under the name of "kriging", although it seems to have been largely ignored as a general-purpose regression method. Gaussian process priors have the advantage over neural networks that at least the lowest level of a Bayesian hierarchical model can be treated analytically. Recent work (Williams and Rasmussen, 1996, inspired by observations in Neal, 1996) has extended the use of these priors to higher dimensional problems that have been traditionally tackled with other techniques such as neural networks, decision trees *etc* and has shown that good results can be obtained.

The main aim of this paper is to provide a tutorial on regression with Gaussian processes. The approach taken is to start with Bayesian linear regression, and to show how by a change of viewpoint one can see this method as a Gaussian process predictor based on priors over functions, rather than performing the computations in parameter-space. This leads in to a more general discussion of Gaussian processes in section 4. Section 5 deals with further issues, including hierarchical modelling and the setting of the parameters that control the Gaussian process, the covariance functions for neural network models and the use of Gaussian processes in classification problems.

2 Bayesian regression

To apply the Bayesian method to a data analysis problem, we first specify a set of probabilistic models of the data. This set may be finite, countably infinite or uncountably infinite in size. An example of the latter case is when the set of models is indexed by a vector in $\mathbb{R}^m$. Let a member of this set be denoted by $\mathcal{H}_\alpha$, which will have a *prior* probability $P(\mathcal{H}_\alpha)$. On observing some data

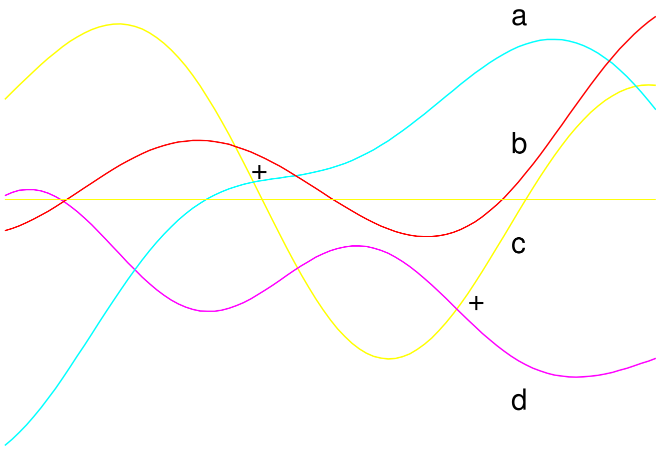


Figure 1: The four possible curves (labelled a, b, c and d) and the two data points (shown with + signs).

$\mathcal{D}$, the *likelihood* of hypothesis $\mathcal{H}_\alpha$ is $P(\mathcal{D}|\mathcal{H}_\alpha)$. The *posterior* probability of $\mathcal{H}_\alpha$ is then given by

$$\text{posterior} \propto \text{prior} \times \text{likelihood} \tag{1}$$

$$P(\mathcal{H}_\alpha|\mathcal{D}) \propto P(\mathcal{H}_\alpha)P(\mathcal{D}|\mathcal{H}_\alpha). \tag{2}$$

The proportionality can be turned into an equality be dividing through by $P(\mathcal{D}) = \sum_\alpha P(\mathcal{D}|\mathcal{H}_\alpha)P(\mathcal{H}_\alpha)$ (where the summation may be interpreted as an integration where appropriate).

Suppose we are now asked to make a prediction using this set of probabilistic models; say we are to predict some quantity y. Under each of the individual models the prediction for y is given by $P(y|\mathcal{H}_\alpha)$. The combined prediction is

$$P(y) = \sum_\alpha P(y|\mathcal{H}_\alpha)P(\mathcal{H}_\alpha|\mathcal{D}). \tag{3}$$

In this paper we will discuss the Bayesian approach to the regression problem, i.e. the discovery of the relationship between input (or independent) variables $\mathbf{x}$ and the output (or dependent) variable y. In the rest of this section we illustrate the Bayesian method with only a finite number of hypotheses; the case of an uncountably infinite set is treated in section 3.

We are given four different curves denoted $f_a(x), f_b(x), f_c(x)$ and $f_d(x)$ which correspond to the hypotheses labelled by $\mathcal{H}_a, \mathcal{H}_b, \mathcal{H}_c$ and $\mathcal{H}_d$; see the illustration in Figure 1. Each curve $\mathcal{H}_\alpha$ has a prior probability $P(\mathcal{H}_\alpha)$; if there is no reason *a priori* to prefer one curve over another, then each prior probability is $1/4$.

The data $\mathcal{D}$ is given as input-output pairs $(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \ldots, (\mathbf{x}_n, t_n)$. Assuming that the targets t_i are generated by adding independent Gaussian noise of variance σ_ν^2 to the underlying function evaluated at $\mathbf{x}_i$, the likelihood of model $\mathcal{H}_\alpha$ (for $\alpha \in \{a, b, c, d\}$) when the data point $(\mathbf{x}_i, t_i)$ is observed is

$$P(t_i|\mathbf{x}_i, \mathcal{H}_\alpha) = \frac{1}{(2\pi\sigma_\nu^2)^{1/2}} \exp -\frac{(t_i - f_\alpha(x_i))^2}{2\sigma_\nu^2}. \tag{4}$$

The likelihood of each hypothesis given all n data points is simply $\prod_i P(t_i|\mathbf{x}_i, \mathcal{H}_\alpha)$.

Let us assume that the standard deviation of the noise is much smaller (say less than 1/10) than the overall y-scale of Figure 1. On observing one data point (the left-most + in Figure 1), the likelihood (or data fit) term given by equation 4 is much higher for curves a, b and c than for curve d. Thus the posterior distribution after the first data point will now have less weight on $\mathcal{H}_d$ and correspondingly more on hypotheses $\mathcal{H}_a, \mathcal{H}_b$ and $\mathcal{H}_c$. A second data point is now observed (the right-most + in Figure 1). Only curve c fits both of of these data points well, and thus the posterior will have most of its mass concentrated on this hypothesis. If we were to make predictions at a new x point, it would be curve c that had the largest contribution to this prediction.

From this example it can be seen that Bayesian inference is really quite straightforward, at least in principle; we simply evaluate the posterior probability of each alternative, and combine these to make predictions. These calculations are easily understood when there are a finite number of hypotheses, and they can also be carried out analytically in some cases when the number of hypotheses is infinite, as we shall see in the next section.

3 From linear regression ...

Let us consider what may be called "generalized linear regression", which we take to mean linear regression using a fixed set of m basis functions $\{\phi_i(\mathbf{x})\}$. Thus the regression function will have the form $y(x) = \sum_{i=1}^{m} w_i \phi_i(x) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$, for some vector of "weights" $\mathbf{w}$. If there is a Gaussian prior distribution on the weights and we assume Gaussian noise then there are two equivalent ways of obtaining the regression function, (i) by performing the computations in weight-space, and (ii) by taking a Gaussian process view. In the rest of this section we will develop these two methods and demonstrate their equivalence.

3.1 The weight-space view

Let the weights have a prior distribution which is Gaussian and centered on the origin, $\mathbf{w} \sim N(\mathbf{0}, \Sigma_w)$, i.e.

$$P(\mathbf{w}) = \frac{1}{(2\pi)^{m/2}|\Sigma_w|^{1/2}} \exp\{-\frac{1}{2}\mathbf{w}^T \Sigma_w^{-1} \mathbf{w}\}. \tag{5}$$

Although a common choice for the prior (e.g. in ridge regression) is to set $\Sigma_w \propto I_m$, it may be more sensible to choose Σ_w so as to approximate a Gaussian process (see section 5.3).

Again, assuming that the targets t_i are generated by Gaussian noise of variance σ_ν^2 from the underlying function, the likelihood of $\mathbf{w}$ is

$$P(t_1, t_2, \ldots, t_n|\mathbf{w}) = \frac{1}{(2\pi\sigma_\nu^2)^{n/2}} \prod_{i=1}^{n} \exp\{-\frac{(t_i - y(x_i;\mathbf{w}))^2}{2\sigma_\nu^2}\}. \tag{6}$$

The posterior distribution for the weights is given by

$$P(\mathbf{w}|D) = \frac{P(D|\mathbf{w})P(\mathbf{w})}{P(D)} \tag{7}$$

where $P(D) = \int P(D|\mathbf{w})P(\mathbf{w})d\mathbf{w}$. As the prior and likelihood are Gaussian, the posterior is also Gaussian. The posterior mean value of the weights $\mathbf{w}_{MP}$ is the choice of $\mathbf{w}$ that minimizes the quadratic form

$$E = \frac{1}{2\sigma_\nu^2} \sum_{i}^{n} (t_i - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i))^2 + \frac{1}{2}\mathbf{w}^T \Sigma_w^{-1} \mathbf{w}. \tag{8}$$

Let $\beta = 1/\sigma_\nu^2$, let Φ be the $n \times m$ design matrix

$$\Phi = \begin{pmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_m(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_m(\mathbf{x}_2) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_1(\mathbf{x}_n) & \phi_2(\mathbf{x}_n) & \cdots & \phi_m(\mathbf{x}_n) \end{pmatrix} \tag{9}$$

and let $\mathbf{t}$ denote the vector of targets. Then we can rewrite equation 8 as

$$E = \frac{\beta}{2}(\mathbf{t} - \Phi\mathbf{w})^T(\mathbf{t} - \Phi\mathbf{w}) + \frac{1}{2}\mathbf{w}^T\Sigma_w^{-1}\mathbf{w} \tag{10}$$

$$= \frac{1}{2}\mathbf{w}^T(\Sigma_w^{-1} + \beta\Phi^T\Phi)\mathbf{w} - \beta\mathbf{w}^T\Phi^T\mathbf{t} + \frac{\beta}{2}\mathbf{t}^T\mathbf{t}, \tag{11}$$

and $\mathbf{w}_{MP}$ is the minimizer of this quadratic form, i.e. it is the solution of

$$(\Sigma_w^{-1} + \beta\Phi^T\Phi)\mathbf{w}_{MP} = \beta\Phi^T\mathbf{t}. \tag{12}$$

Let $A = \Sigma_w^{-1} + \beta\Phi^T\Phi$. Then $\mathbf{w}_{MP} = \beta A^{-1}\Phi^T\mathbf{t}$ and the posterior covariance matrix for the weights is A^{-1}.

The mean prediction for a new input $\mathbf{x}_*$ (under the weight-space view) is

$$\mu_{ws}(\mathbf{x}_*) = \boldsymbol{\phi}^T(\mathbf{x}_*)\mathbf{w}_{MP} = \beta\boldsymbol{\phi}^T(\mathbf{x}_*)A^{-1}\Phi^T\mathbf{t}. \tag{13}$$

We can also obtain "error bars" for this prediction; the variance about the mean is given by

$$\sigma_y^2(\mathbf{x}_*) = E_{w|\mathcal{D}}[(y(\mathbf{x}_*) - \mu_{ws}(\mathbf{x}_*))^2] \tag{14}$$

$$= \boldsymbol{\phi}^T(\mathbf{x}_*)E_{w|\mathcal{D}}[(\mathbf{w} - \mathbf{w}_{MP})(\mathbf{w} - \mathbf{w}_{MP})^T]\boldsymbol{\phi}(\mathbf{x}_*) \tag{15}$$

$$= \boldsymbol{\phi}^T(\mathbf{x}_*)A^{-1}\boldsymbol{\phi}(\mathbf{x}_*). \tag{16}$$

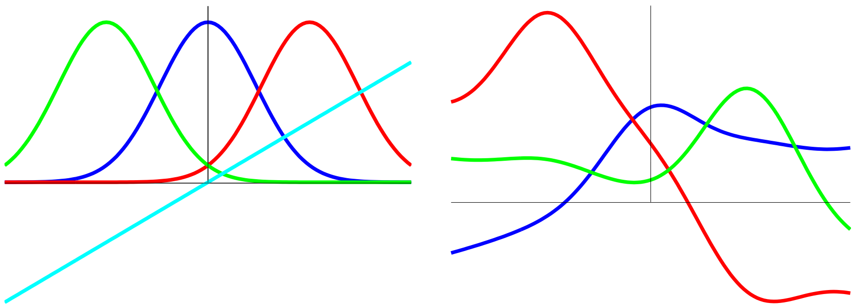
To obtain the predictive variance var $t(\mathbf{x}_*)$ it is necessary to add σ_ν^2 to $\sigma_y^2(\mathbf{x}_*)$ to account for the additional variance due to the noise, since the two sources of variation are uncorrelated.

The Bayesian approach to linear regression is discussed in most texts on Bayesian statistics, for example Box and Tiao (1973).

3.2 The function-space view

In the previous section the uncertainty in the problem was described through a probability distribution over the weights. It is also possible to deal directly with uncertainty with respect to the function values at the points we are interested in. This is the stochastic process or function-space view of the problem. A stochastic process $Y(\mathbf{x})$ is a collection of random variables indexed by $\mathbf{x}$. In the cases considered below, $\mathbf{x}$ will usually be a vector in $\mathbb{R}^p$, where p is the dimensionality of $\mathbf{x}$-space. A general stochastic process is specified by giving the probability distributions of any finite subset $(Y(\mathbf{x}_1), Y(\mathbf{x}_2), \ldots, Y(\mathbf{x}_k))$ in a consistent way. Gaussian processes are a subset of stochastic processes that can be specified by giving only the mean vector and covariance matrix for any finite subset of points. In fact we shall further specialize and consider only Gaussian processes which have zero mean.

In the function-space view of linear regression we consider the kinds of function that can be generated from a fixed set of basis functions with random weights. The Y value at a particular point $\mathbf{x}$ in the input space is a random variable $Y(\mathbf{x}) = \sum_j W_j\phi_j(\mathbf{x})$, which is simply a linear



(a) (b)

Figure 2: (a) shows four basis functions, and (b) shows three sample functions generated by taking different linear combinations of these basis functions.

combination of the Gaussian random variables $\mathbf{W}$, where $\mathbf{W} \sim N(\mathbf{0}, \Sigma_w)$ as in equation 5. (The notation $\mathbf{W}$ indicates that the weights are viewed as random variables.) We can calculate the mean and covariance functions for the stochastic process,

$$E_w[Y(\mathbf{x})] = 0 \tag{17}$$

$$E_w[Y(\mathbf{x})Y(\mathbf{x}')] = \boldsymbol{\phi}^T(\mathbf{x})\Sigma_w\boldsymbol{\phi}(\mathbf{x}'). \tag{18}$$

In fact since it is derived as a linear combination of Gaussian random variables it is clear that Y is a Gaussian process. Some examples of sample functions are shown in Figure 2(b), using the basis functions shown in Figure 2(a). The sample functions are obtained by drawing samples of $\mathbf{W}$ from $N(\mathbf{0}, \Sigma_w)$.

3.3 Prediction using Gaussian Processes

We will first consider the general problem of prediction using a Gaussian process, and then focus on the particular Gaussian process derived from linear regression. The key observation is that rather than deal with complicated entities such as priors on function spaces, we can consider just the function values at the $\mathbf{x}$ points that concern us, namely the training points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and the test point $\mathbf{x}_*$ whose y-value we wish to predict. Thus we need only consider finite-dimensional objects, namely covariance matrices.

Consider $n+1$ random variables $(Z_1, Z_2, \ldots, Z_n, Z_*)$ which have a joint Gaussian distribution with mean $\mathbf{0}$ and covariance matrix K_+. Let the $(n+1) \times (n+1)$ matrix K_+ be partitioned into a $n \times n$ matrix K, a $n \times 1$ vector $\mathbf{k}$ and a scalar k_*

$$K_+ = \begin{pmatrix} K & \mathbf{k} \\ \mathbf{k}^T & k_* \end{pmatrix} \tag{19}$$

If particular values are observed for the first n variables, i.e. $Z_1 = z_1, Z_2 = z_2, \ldots, Z_n = z_n$, then the conditional distribution for Z_* is Gaussian (see, e.g. von Mises, 1964, section 9.3) with

$$E[Z_*] = \mathbf{k}^T K^{-1}\mathbf{z} \tag{20}$$

$$\mathrm{var}[Z_*] = k_* - \mathbf{k}^T K^{-1}\mathbf{k} \tag{21}$$

where $\mathbf{z}^T = (z_1, z_2, \ldots, z_n)$. Notice that the predicted mean (equation 20) is a linear combination of the z's.

3.4 LINEAR REGRESSION USING THE FUNCTION-SPACE VIEW

Returning to the linear regression case, we are interested in calculating the distribution of $Y(\mathbf{x}_*) = Y_*$, given the noisy observations $(t_1, \ldots, t_n)$. To do this we calculate the joint distribution of $(T_1, T_2, \ldots, T_n, Y_*)$ and then condition on the specific values $T_1 = t_1, T_2 = t_2, \ldots, T_n = t_n$ to obtain the desired distribution $P(Y_*|\mathbf{t})$. Of course this distribution will be a Gaussian so we need only compute its mean and variance.

The distribution for $(T_1, T_2, \ldots, T_n, Y_*)$ is most easily derived by first considering the joint distribution of $\mathbf{Y}_+ = (Y_1, Y_2, \ldots, Y_n, Y_*)$. Under the linear regression model this is given by $\mathbf{Y}_+ \sim N(\mathbf{0}, \Phi_+ \Sigma_w \Phi_+^T)$, where Φ_+ is an extended Φ matrix with an additional bottom row $\boldsymbol{\phi}_* = \boldsymbol{\phi}(\mathbf{x}_*) = (\phi_1(\mathbf{x}_*), \phi_2(\mathbf{x}_*), \ldots, \phi_m(\mathbf{x}_*))$. Under the partitioning used in equation 19, the structure of $\Phi_+ \Sigma_w \Phi_+^T$ can be written

$$\Phi_+ \Sigma_w \Phi_+^T = \begin{pmatrix} \Phi \Sigma_w \Phi^T & \Phi \Sigma_w \boldsymbol{\phi}_* \\ \boldsymbol{\phi}_*^T \Sigma_w \Phi^T & \boldsymbol{\phi}_*^T \Sigma_w \boldsymbol{\phi}_* \end{pmatrix}. \tag{22}$$

The joint distribution for $T_1, \ldots T_n, Y_*$ can be found by realizing that the T's are obtained by corrupting the corresponding Y's with Gaussian noise, and hence that $(T_1, \ldots T_n, Y_*) \sim N(\mathbf{0}, \Phi_+ \Sigma_w \Phi_+^T + E_+)$, where E_+ is the $n \times n$ matrix $\sigma_\nu^2 I_n$ bordered on the right and bottom by bands of zeros. (The reason that E_+ is not $\sigma_\nu^2 I_{n+1}$ is that we are predicting Y_*, not T_*.) Conditioning on $T_1 = t_1, T_2 = t_2, \ldots, T_n = t_n$ and using equations 20 and 21, we obtain

$$E[Y_*] = \mu_{fs}(\mathbf{x}_*) = \boldsymbol{\phi}_*^T \Sigma_w \Phi^T P^{-1} \mathbf{t} \tag{23}$$

$$\mathrm{var}[Y_*] = \boldsymbol{\phi}_*^T \Sigma_w \boldsymbol{\phi}_* - \boldsymbol{\phi}_*^T \Sigma_w \Phi^T P^{-1} \Phi \Sigma_w \boldsymbol{\phi}_* \tag{24}$$

where we have defined $P = (\Phi \Sigma_w \Phi^T + \sigma_\nu^2 I_n)$. Note that if the number of basis functions (m) is less than the number of data points (n), as will often be the case, then the covariance matrix $\Phi \Sigma_w \Phi^T$ will be rank-deficient, i.e. it will have some zero eigenvalues, so that the probability of points lying outside a linear subspace will be zero. However, the addition of $\sigma_\nu^2 I_n$ ensures that P will be strictly positive definite.

The challenge is now to show that these results for the mean and variance are consistent with the expressions obtained in section 3.1. To obtain the formula for the mean as in equation 13 we note that

$$A \Sigma_w \Phi^T = \Phi^T + \beta \Phi^T \Phi \Sigma_w \Phi^T = \beta \Phi^T (\sigma_\nu^2 I + \Phi \Sigma_w \Phi^T) = \beta \Phi^T P. \tag{25}$$

By multiplying through by A^{-1} and P^{-1} we obtain $\Sigma_w \Phi^T P^{-1} = \beta A^{-1} \Phi^T$, which can be substituted into equation 23 to yield the desired result.

The equivalence of the two expressions for the variance can be proved by using the following matrix identity (the Woodbury formula, Press *et al*, 1992, section 2.7)

$$(X + YZ)^{-1} = X^{-1} - X^{-1} Y (I + Z X^{-1} Y)^{-1} Z X^{-1} \tag{26}$$

on $A^{-1} = (\Sigma_w^{-1} + \beta \Phi^T \Phi)^{-1}$ to obtain

$$A^{-1} = \Sigma_w - \Sigma_w \Phi^T (\sigma_\nu^2 I + \Phi \Sigma_w \Phi^T)^{-1} \Phi \Sigma_w, \tag{27}$$

which may be substituted into equation 16 to give equation 24.

Given that the weight-space and function-space views are equivalent, it is interesting to ask which of the two methods is computationally more efficient. In the weight-space approach it is necessary to form and invert the matrix A which is of dimension $m \times m$. Similarly for the function-space view one has to form and invert the $n \times n$ matrix P. As the inversion of a $l \times l$ matrix takes time $O(l^3)$, it is natural to choose the method which takes less time. Usually for simple linear regression problems $m \ll n$ and so the weight-space view will be preferred. But for other kinds of linear prediction (see section 4) m can be infinite and thus the function-space view is the only tenable one.

4 ... to linear prediction ...

As we have already seen in section 3.3, if the prior is a general Gaussian process and we assume a Gaussian noise model, then the predicted y-value is just some linear combination of the t-values; the method is said to be a *linear smoother* (Hastie and Tibshirani, 1990) or a *linear predictor*. In section 3 we have seen how linear regression can be seen from a function-space viewpoint. This opens up further possibilities, as linear regression with a prior on the weights is just one way of specifying the covariance between points $\mathbf{x}$ and $\mathbf{x}'$. In general the covariance function $C(\mathbf{x}, \mathbf{x}')$ of a zero-mean Gaussian process $Y(\mathbf{x})$ is defined as $E[Y(\mathbf{x})Y(\mathbf{x}')]$. Formally, the covariance function can be any function that will generate a non-negative definite covariance matrix for any set of points $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k)$. It is non-trivial to come up with functions that obey this condition, although several families are known in the literature.

One well-known class is the stationary and isotropic covariance functions where $C(\mathbf{x}, \mathbf{x}') = C(|\mathbf{x} - \mathbf{x}'|) = C(h)$, where $|\cdot|$ denotes the Euclidean norm. These can be derived as the characteristic function (or Fourier transform) of an isotropic probability density[1]. For example $C(h) = \exp(-(h/\sigma)^\nu)$ is a valid covariance function for all dimensions p and for $0 < \nu \leq 2$, corresponding to the multivariate Cauchy and Gaussian distributions when $\nu = 1$ and $\nu = 2$ respectively. Note that in this case σ sets the correlation length-scale of the random field, although other covariance functions (e.g. those corresponding to power-law spectral densities[2]) may have no preferred length scale.

Samples from Gaussian processes can have very different properties depending on the choice of the covariance function. For example in 1-d, the Ornstein-Uhlenbeck process (with covariance function $e^{-|h|}$) has very rough sample paths which are not mean-square differentiable. On the other hand, choosing $C(x, x') = \sigma_0^2 + \sigma_1^2 x x'$ (which arises from $y = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$ with $\boldsymbol{\phi}(\mathbf{x}) = (1, x)^T$ and $\mathbf{w} \sim N(\mathbf{0}, \mathrm{diag}(\sigma_0^2, \sigma_1^2))$) leads to straight-line sample paths of the form $y = w_0 + w_1 x$. It should also be noted that the covariance function $C(h) = e^{-h^2}$ gives rise to sample paths that are infinitely mean-square differentiable.

Given a covariance function which is assumed to characterize the data $\mathcal{D}$, it is easy to make predictions for new test points; we simply use equations 20 and 21; as in section 3.4 the overall covariance function will be made up of the prior covariance function and a noise term. If the covariance function is unknown (as will always be the case in practice) but is assumed to come from some parametric class, then maximum likelihood estimation or a Bayesian approach can be taken; this is discussed in section 5.2 below.

Prediction with Gaussian processes is certainly not a very recent topic; the basic theory goes back to Wiener and Kolmogorov in the 1940's and applications to multivariate regression are discussed in Whittle (1963). ARMA models for time series are Gaussian process models. Gaussian process prediction is also well known in the geostatistics field (Journel and Huijbregts, 1978; Cressie, 1993) where it is known as "kriging", although this literature naturally has focussed mostly on two- and three-dimensional input spaces. Wahba has been influential in promoting the use of spline

[1] In fact Bochner's theorem (see, e.g. Wong, 1971) states that the positive definite functions $C(h)$ which are continuous at 0 and satisfy $C(0) = 1$ are exactly the characteristic functions.

[2] For stationary covariance functions the spectral density is the Fourier transform of the covariance function.

techniques for regression problems; her work dates back to Kimeldorf and Wahba (1970), although Wahba (1990) provides a useful overview. Essentially splines correspond to Gaussian processes with a particular choice of covariance function[3].

Gaussian process prediction was also suggested by O'Hagan (1978), and is widely used in the analysis of computer experiments (e.g Sacks *et al*, 1989), although in this application it is assumed that the observations are noise-free. A connection to neural networks was made by Poggio and Girosi (1990) and Girosi, Jones and Poggio (1995) with their work on Regularization Networks. When the covariance function $C(\mathbf{x}, \mathbf{x}')$ depends only on $h = |\mathbf{x} - \mathbf{x}'|$, the predictor derived in equation 20 has the form $\sum_i c_i C(|\mathbf{x} - \mathbf{x}_i|)$ and may be called a *radial basis function* (or RBF) network.

4.1 Covariance functions and eigenfunctions

It turns out that general Gaussian processes can be viewed as Bayesian linear regression with an infinite number of basis functions. One possible basis set is the *eigenfunctions* of the covariance function. A function $\phi(\cdot)$ that obeys the integral equation over the domain $\mathcal{R}$

$$\int_{\mathcal{R}} C(\mathbf{x}, \mathbf{x}')\phi(\mathbf{x})d\mathbf{x} = \lambda\phi(\mathbf{x}') \tag{28}$$

is called an eigenfunction of C with eigenvalue λ. In general there are an infinite number of eigenfunctions, which we label $\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots$. The eigenfunctions are orthogonal and can be chosen to be normalized so that $\int_{\mathcal{R}} \phi_i(\mathbf{x})\phi_j(\mathbf{x})\, d\mathbf{x} = \delta_{ij}$ where δ_{ij} is the Kronecker delta.

Mercer's theorem (see, e.g. Wong, 1971) states that the covariance function can be expressed as

$$C(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{\infty} \lambda_i \phi_i(\mathbf{x})\phi_i(\mathbf{x}'). \tag{29}$$

This decomposition is just the infinite-dimensional analogue of the diagonalization of a real symmetric matrix. Note that if $\mathcal{R}$ is $\mathbb{R}^p$, then the summation in equation 29 can become an integral. This occurs, for example, in the spectral representation of stationary covariance functions. However, it can happen that the spectrum is discrete even if $\mathcal{R}$ is $\mathbb{R}^p$ as long as $C(\mathbf{x}, \mathbf{x}')$ decays fast enough.

The equivalence with Bayesian linear regression can now be seen by taking the prior weight matrix Σ_w to be the diagonal matrix $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots)$ and choosing the eigenfunctions as the basis functions; equation 29 and the equivalence of the weight-space and function-space views demonstrated in section 3 completes the proof.

The fact that an input vector can be expanded into an infinite-dimensional space $(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots)$ but that the necessary computations can be carried out efficiently due to Mercer's theorem has been used in some other contexts, for example in the method of potential functions (due to Aizerman, Braverman and Rozoner, 1964) and in support vector machines (Vapnik, 1995). In support vector regression the prior over functions is as described above, but instead of using a squared error loss function (which corresponds to Gaussian noise), a modified version of the l_1 error metric $|t_i - y_i|$ is used, called the ϵ-insensitive loss function. Finding the *maximum a posteriori* (or MAP) y-values for the training points and test point can now be achieved using quadratic programming (see Vapnik, 1995 for details).

[3]Technically splines require generalized covariance functions (see Cressie §5.4), and they have a power-law spectral density $S(\omega) \propto \omega^{-\beta}$ with $\beta > 0$.

5 ... and beyond ...

In this section some further details are given on the topics of modelling issues, adaptation of the covariance function, computational issues, the covariance function for neural networks and classification with Gaussian processes.

5.1 Modelling issues

As we have seen above, there is a wide variety of covariance functions that can be used. The use of stationary covariance functions is appealing as usually one would like the predictions to be invariant under shifts of the origin in input space. From a modelling point of view we wish to specify a covariance function so that nearby inputs will give rise to similar predictions. Experiments in Williams and Rasmussen (1996) and Rasmussen (1996) have demonstrated that the following covariance function seems to work well in practice:

$$\begin{aligned} C(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) &= v_0 \exp\{-\frac{1}{2}\sum_{l=1}^{p} \alpha_l (x_l^{(i)} - x_l^{(j)})^2\} \\ &\quad + a_0 + a_1 \sum_{l=1}^{p} x_l^{(i)} x_l^{(j)} + v_1 \delta(i,j), \end{aligned} \tag{30}$$

where $\boldsymbol{\theta} \stackrel{def}{=} (\log v_0, \log v_1, \log \alpha_1, \ldots, \log \alpha_p, \log a_0, \log a_1)$ is the vector of adjustable parameters. The parameters are defined to be the log of the variables in equation (30) since they are positive scale-parameters.

The covariance function is made up of three parts; the first term, a linear regression term (involving a_0 and a_1) and a noise term $v_1\delta(i,j)$. The first term expresses the idea that cases with nearby inputs will have highly correlated outputs; the α_l parameters allow a different distance measure for each input dimension. For irrelevant inputs, the corresponding α_l will become small, and the model will ignore that input. This is closely related to the Automatic Relevance Determination (ARD) idea of MacKay and Neal (MacKay, 1993; Neal 1996). The v_0 variable gives the overall scale of the local correlations, a_0 and a_1 are variables controlling the scale of the bias and linear contributions to the covariance. A simple extension of the linear regression part of the covariance function would allow a different hyperparameter a_{1i} for each of the input dimensions. The last term accounts for the noise on the data; v_1 is the variance of the noise.

The ARD idea was demonstrated for Gaussian processes in an experiment described in Williams and Rasmussen (1996), where irrelevant inputs were added to a regression task. By adapting the α parameters as described in section 5.2 below, their values for the irrelevant inputs became very small, successfully indicating the irrelevance of these inputs.

A graphical example of ARD is given in Figure 3. The left hand panel shows a sample from the prior when $\alpha_1 = \alpha_2$. Notice that the length-scale of variation is the same on both axes. In the right hand panel $\alpha_1 = 10\alpha_2$, indicating much slower variation along the x_2 axis. In the limit as $\alpha_2 \to 0$, the x_2 coordinate is irrelevant.

5.2 Adapting the covariance function

Given a covariance function it is straightforward to make predictions for new test points. However, in practical situations we are unlikely to know which covariance function to use. One option is to

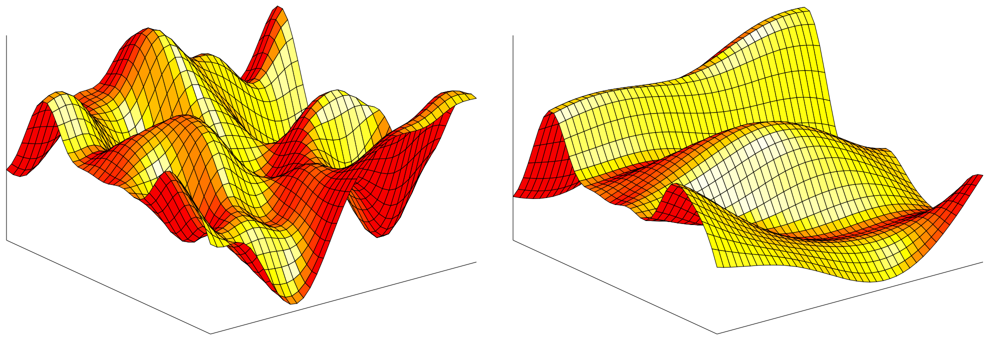


Figure 3: Functions drawn at random from the ARD prior. In the left hand plot $\alpha_1 = \alpha_2$ while on the right $\alpha_1 = 10\alpha_2$.

choose a parametric family of covariance functions (with a parameter vector $\boldsymbol{\theta}$) and then either to estimate the parameters (for example, using the method of maximum likelihood) or to use a Bayesian approach where a posterior distribution over the parameters is obtained.

These calculations are facilitated by the fact that the log likelihood $l = \log P(\mathcal{D}|\boldsymbol{\theta})$ can be calculated analytically as

$$l = -\frac{1}{2}\log\det K - \frac{1}{2}\mathbf{t}^T K^{-1}\mathbf{t} - \frac{n}{2}\log 2\pi. \tag{31}$$

It is also possible to express analytically the partial derivatives of the log likelihood with respect to the hyperparameters, using the equation

$$\frac{\partial l}{\partial \theta_i} = -\frac{1}{2}\mathrm{tr}\left(K^{-1}\frac{\partial K}{\partial \theta_i}\right) + \frac{1}{2}\mathbf{t}^T K^{-1}\frac{\partial K}{\partial \theta_i}K^{-1}\mathbf{t}, \tag{32}$$

as derived, for example, in Mardia and Marshall (1984). The evaluation of the likelihood and the partial derivatives takes time $O(n^3)$. Given l and its derivatives with respect to $\boldsymbol{\theta}$ it is straightforward to feed this information to an optimization package in order to obtain a local maximum of the likelihood. An alternative to maximum likelihood estimation of the parameters is to use a cross-validation (CV) or generalized cross-validation (GCV) method, as discussed in Wahba (1990). However, it would appear that it is difficult to use these methods when a large number of parameters are involved.

In general one may be concerned about making point estimates when the number of parameters is large relative to the number of data points, or if some of the parameters may be poorly determined, or if there may be local maxima in the likelihood surface. For these reasons the Bayesian approach of defining a prior distribution over the parameters and then obtaining a posterior distribution once the data $\mathcal{D}$ has been seen is attractive. To make a prediction for a new test point $\mathbf{x}_*$ one simply averages over the posterior distribution $P(\boldsymbol{\theta}|\mathcal{D})$, i.e.

$$P(y_*|\mathcal{D}) = \int P(y_*|\boldsymbol{\theta},\mathcal{D})P(\boldsymbol{\theta}|\mathcal{D})d\boldsymbol{\theta}. \tag{33}$$

It is not possible to do this integration analytically in general, but numerical methods may be used. If $\boldsymbol{\theta}$ is of sufficiently low dimension, then techniques involving grids in $\boldsymbol{\theta}$-space can be used. See, for example the paper by Handcock and Stein, (1993).

If $\boldsymbol{\theta}$ is high-dimensional it is very difficult to locate the regions of parameter-space which have high posterior density by gridding techniques or importance sampling. In this case Markov chain

Monte Carlo (MCMC) methods may be used. These work by constructing a Markov chain whose equilibrium distribution is the desired distribution $P(\boldsymbol{\theta}|\mathcal{D})$; the integral in equation 33 is then approximated using samples from the Markov chain.

Two standard methods for constructing MCMC methods are the Gibbs sampler and Metropolis-Hastings algorithms (see, e.g., Gelman *et al*, 1995). However, the conditional parameter distributions are not amenable to Gibbs sampling if the covariance function has the form given by equation 30, and the Metropolis-Hastings algorithm does not utilize the derivative information that is available, which means that it tends to have an inefficient random-walk behaviour in parameter-space. Following the work of Neal (1996) on Bayesian treatment of neural networks, Williams and Rasmussen (1996) and Rasmussen (1996) have used the Hybrid Monte Carlo method of Duane *et al* (1987) to obtain samples from $P(\boldsymbol{\theta}|\mathcal{D})$. Rasmussen (1996) carried out a careful comparison of the Bayesian treatment of Gaussian process regression with several other state-of-the-art methods on a number of problems and found that its performance is comparable to that of Bayesian neural networks as developed by Neal (1996), and consistently better than the other methods.

5.3 Computational issues

Equations 20 and 21 require the inversion of a $n \times n$ matrix. When n is of the order of a few hundred then this is quite feasible with modern computers. However, once $n \sim O(1000)$ these computations can be quite time consuming, especially if this calculation must be carried out many times in an iterative scheme as discussed above in section 5.2. It is therefore of interest to consider approximate methods.

One possible approach is to approximate the matrix inversion step needed for prediction, i.e. the computation of $K^{-1}\mathbf{z}$ in equation 20. Gibbs and MacKay (1997a) have used the conjugate gradients (CG) algorithm for this task, based on the work of Skilling (1993). The algorithm iteratively computes an approximation to $K^{-1}\mathbf{z}$; if it is allowed to run for n iterations it takes time $O(n^3)$ and computes the exact solution to the linear system, but by stopping the algorithm after $k < n$ iterations an approximate solution is obtained. Note also that when adjusting the parameters of the covariance matrix, the solution of the linear system that was obtained with the old parameter values will often be a good starting point for the new CG iteration.

When adjusting the parameters one also needs to be able to calculate quantities such as $\mathrm{tr}(K^{-1}\partial K/\partial\theta_i)$. For large matrices this computation can be approximated by using the "randomized trace method". Observe that if $\mathbf{d} \sim N(\mathbf{0}, I_n)$, then $E[\mathbf{d}^T M \mathbf{d}] = \mathrm{tr}M$, and thus the trace of a matrix M may be estimated by averaging $\mathbf{d}^T M \mathbf{d}$ over several $\mathbf{d}$'s. This method has been used in the splines literature by Hutchinson (1989) and Girard (1989) and also by Gibbs and MacKay (1997a) following the independent work of Skilling (1993). Similar methods can be brought to bear on the calculation of $\log\det K$.

An alternative approximation scheme for Gaussian processes is to project the covariance kernel onto a finite number of basis functions, i.e. to find the Σ_w in equation 5 which leads to the best approximation of the covariance function. This method has been discussed by Silverman (1985), Wahba (1990) Zhu and Rohwer (1996) and Hastie (1996). However, if we also can choose which basis functions we wish to use then the m eigenfunctions which have the largest eigenvalues should be used. This truncated expansion makes sense when the eigenvalues in equation 29 decay to zero fast enough so that the sum $\sum_{i=1}^{\infty} \lambda_i$ converges. The technique is an infinite-dimensional analogue of principal components analysis, and is discussed, for example, in Zhu *et al* (1997).

There may well be other methods of speeding up the computations; one possible example is to ignore data points which are far away from the test point when making predictions, thereby producing a

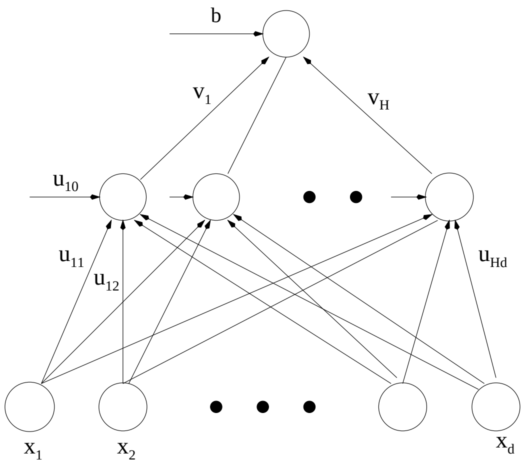


Figure 4: The architecture of a network with a single hidden layer.

smaller matrix to be inverted.

5.4 The covariance function of neural networks

My own interest in using Gaussian processes for regression was sparked by Radford Neal's observation (Neal, 1996), that under a Bayesian treatment, the functions produced by a neural network with certain kinds of prior distribution over its weights will tend to a Gaussian process prior over functions as the number of hidden units in the network tends to infinity. In the remainder of this section the covariance function for a neural network with a single hidden layer is derived.

Consider a network which takes an input $\mathbf{x}$, has one hidden layer with H units and then linearly combines the outputs of the hidden units with a bias b to obtain $f(\mathbf{x})$. The mapping can be written

$$f(\mathbf{x}) = b + \sum_{j=1}^{H} v_j h(\mathbf{x}; \mathbf{u}_j) \tag{34}$$

where $h(\mathbf{x}; \mathbf{u})$ is the hidden unit transfer function (which we shall assume is bounded) which depends on the input-to-hidden weights $\mathbf{u}$. This architecture is important because it has been shown by Hornik (1993) that networks with one hidden layer are universal approximators as the number of hidden units tends to infinity, for a wide class of transfer functions (but excluding polynomials). Let b and the v's have independent zero-mean distributions of variance σ_b^2 and σ_v^2 respectively, and let the weights $\mathbf{u}_j$ for each hidden unit be independently and identically distributed. Denoting all weights by $\mathbf{w}$, we obtain (following Neal, 1996)

$$\begin{aligned} E_{\mathbf{w}}[f(\mathbf{x})] &= 0 && (35)\\ E_{\mathbf{w}}[f(\mathbf{x})f(\mathbf{x}')] &= \sigma_b^2 + \sum_j \sigma_v^2 E_{\mathbf{u}}[h_j(\mathbf{x}; \mathbf{u}) h_j(\mathbf{x}'; \mathbf{u})] && (36)\\ &= \sigma_b^2 + H\sigma_v^2 E_{\mathbf{u}}[h(\mathbf{x}; \mathbf{u}) h(\mathbf{x}'; \mathbf{u})] && (37) \end{aligned}$$

where equation 37 follows because all of the hidden units are identically distributed. The final term in equation 37 becomes $\omega^2 E_{\mathbf{u}}[h(\mathbf{x}; \mathbf{u}) h(\mathbf{x}'; \mathbf{u})]$ by letting σ_v^2 scale as ω^2/H.

The sum in equation 36 is over H identically and independently distributed random variables. As the transfer function is bounded, all moments of the distribution will be bounded and hence the Central Limit Theorem can be applied, showing that the stochastic process will converge to a Gaussian process in the limit as $H \to \infty$.

By evaluating $E_{\mathbf{u}}[h(\mathbf{x})h(\mathbf{x}')]$ for all $\mathbf{x}$ and $\mathbf{x}'$ in the training and testing sets we can obtain the covariance function needed to describe the neural network as a Gaussian process. These expectations are, of course, integrals over the relevant probability distributions of the biases and input weights. For some choices of transfer function and weight priors these integrals can be calculated analytically. For Gaussian weight priors and a transfer function that is either (i) the error function $\Phi(z) = 2/\sqrt{\pi}\int_0^z e^{-t^2} dt$ or (ii) a Gaussian, explicit expressions for the covariance functions are given in Williams (1997a) and Williams (1997b).

One attraction of using infinite neural networks (represented as GPs) is that the full Bayesian computation should be much easier than with finite networks in some circumstances. Finite networks require integration over the joint posterior in weight-space and (hyper)parameter space, and currently this high-dimensional integration can only be tackled with MCMC methods. With GPs, in effect the integration over the weights can be done exactly (using equations 20 and 21) so that only the integration over the parameters remains. This should lead to improved computational efficiency for GP predictions over neural networks, particularly for problems where n is not too large.

5.5 Classification problems

Given an input $\mathbf{x}$, the aim of a classifier is to produce an estimate of the posterior probabilities for each class $P(k|\mathbf{x})$, where $k = 1, \ldots C$ indexes the C classes. Naturally we require that $0 \leq P(k|\mathbf{x}) \leq 1$ for all k and that $\sum_k P(k|\mathbf{x}) = 1$. A naïve application of the regression method for Gaussian processes using, say, targets of 1 when an example of class k is observed and 0 otherwise will not obey these constraints.

For the two-class classification problem it is only necessary to represent $P(1|\mathbf{x})$, since $P(2|\mathbf{x}) = 1 - P(1|\mathbf{x})$. An easy way to ensure that the estimate $\pi(\mathbf{x})$ of $P(1|\mathbf{x})$ lies in $[0,1]$ is to obtain it by passing an unbounded value $y(\mathbf{x})$ through a the logistic transfer function $\sigma(z) = 1/(1+e^{-z})$ so that $\pi(\mathbf{x}) = \sigma(y(\mathbf{x}))$. The input $y(\mathbf{x})$ to the logistic function will be called the *activation*. In the simplest method of this kind, logistic regression, the activation is simply computed as a linear combination of the inputs, plus a bias, i.e. $y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$. Using a Gaussian process or other flexible methods allow $y(\mathbf{x})$ to be a non-linear function of the inputs. An early reference to this approach is the work of Silverman (1978).

For the classification problem with more than two classes, a simple extension of this idea using the "softmax" function (Bridle, 1990) gives the predicted probability for class k as

$$\pi(k|\mathbf{x}) = \frac{\exp y_k(\mathbf{x})}{\sum_m \exp y_m(\mathbf{x})}. \tag{38}$$

For the rest of this section we shall concentrate on the two-class problem; extension of the methods to the multi-class case is relatively straightforward.

By defining a Gaussian process prior over the activation $y(\mathbf{x})$ automatically induces a prior over $\pi(\mathbf{x})$, as illustrated in Figure 5.5. To make predictions for a test input $\mathbf{x}_*$ when using fixed parameters in the GP we would like to compute $\hat{\pi}_* = \int \pi_* P(\pi_*|\mathbf{t}, \boldsymbol{\theta})\, d\pi_*$, which requires us to find $P(\pi_*|\mathbf{t}, \boldsymbol{\theta}) = P(\pi(\mathbf{x}_*)|\mathbf{t}, \boldsymbol{\theta})$ for a new input $\mathbf{x}_*$. This can be done by finding the distribution

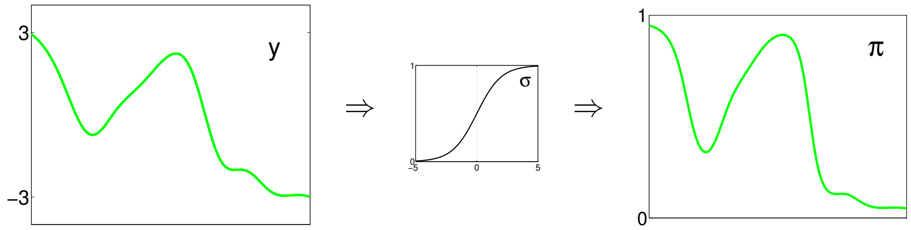


Figure 5: $\pi(\mathbf{x})$ is obtained from $y(\mathbf{x})$ by "squashing" it through the sigmoid function σ.

$P(y_*|\mathbf{t},\boldsymbol{\theta})$ (y_* is the activation of π_*) as given by

$$P(y_*|\mathbf{t},\boldsymbol{\theta}) = \int P(y_*,\mathbf{y}|\mathbf{t},\boldsymbol{\theta})d\mathbf{y} = \frac{1}{P(\mathbf{t}|\boldsymbol{\theta})}\int P(y_*,\mathbf{y}|\boldsymbol{\theta})P(\mathbf{t}|\mathbf{y})d\mathbf{y} \tag{39}$$

and then using the appropriate Jacobian to transform the distribution. When $P(\mathbf{t}|\mathbf{y})$ is Gaussian then the integral in equation 39 can be computed exactly to give equations 20 and 21. However, the usual expression for $P(\mathbf{t}|\mathbf{y}) = \prod_i \pi_i^{t_i}(1-\pi_i)^{1-t_i}$ for classification data (where the t's take on values of 0 or 1), means that the marginalization to obtain $P(y_*|\mathbf{t},\boldsymbol{\theta})$ is no longer analytically tractable.

Faced with this problem there are two routes that we can follow: (i) to use an analytic approximation to the integral in equation 39 or (ii) to use Monte Carlo methods, specifically MCMC methods, to approximate it. These two methods will be considered in turn.

The first analytic method we shall consider is Laplace's approximation, where the integrand $P(y_*,\mathbf{y}|\mathbf{t},\boldsymbol{\theta})$ is approximated by a Gaussian distribution centered at a maximum of this function with respect to $y_*,\mathbf{y}$ with an inverse covariance matrix given by $-\nabla\nabla\log P(y_*,\mathbf{y}|\mathbf{t},\boldsymbol{\theta})$. Finding a maximum can be carried out using the Newton-Raphson (or Fisher scoring) iterative method on $\mathbf{y}$, which then allows the approximate distribution of y_* to be calculated. This is also the method used to calculate the *maximum a posteriori* estimate of y_*. For more details see Green and Silverman (1994) §5.3 and Barber and Williams (1997).

An alternative analytic approximation is due to Gibbs and MacKay (1997b). Instead of using the Laplace approximation they use variational methods to find approximating Gaussian distributions that bound the
marginal likelihood $P(\mathbf{t}|\boldsymbol{\theta})$ above and below, and then use these approximate distributions to predict $P(y_*|\mathbf{t},\boldsymbol{\theta})$ and thus $\hat{\pi}(\mathbf{x}_*)$.

For the analytic approximation methods, there is also the question of what to do about the parameters $\boldsymbol{\theta}$. Maximum likelihood and GCV approaches can again be used as in the regression case (e.g. O'Sullivan *et al*, 1986). Barber and Williams (1997) used an approximate Bayesian scheme based on the Hybrid Monte Carlo method whereby the marginal likelihood $P(\mathbf{t}|\boldsymbol{\theta})$ (which is not available analytically) is replaced by the Laplace approximation of this quantity. Gibbs and MacKay (1997b) estimated $\boldsymbol{\theta}$ by maximizing their lower bound on $P(\mathbf{t}|\boldsymbol{\theta})$.

Recently Neal (1997) has developed a MCMC method for the Gaussian process classification model. This works by generating samples from $P(\mathbf{y},\boldsymbol{\theta}|\mathcal{D})$ in a two stage process. Firstly, for fixed $\boldsymbol{\theta}$, each of the n individual y_i's are updated sequentially using Gibbs sampling. This "sweep" takes time $O(n^2)$ once the matrix K^{-1} has been computed (in time $O(n^3)$), so it actually makes sense to perform quite a few Gibbs sampling scans between each update of the parameters, as this probably makes the Markov chain mix faster. Secondly, the parameters are updated using the Hybrid Monte Carlo method.

It should also be noted that MCMC method for Gaussian processes can be applied to other situations. For example Neal (1997) describes how to use it for a regression problem where the noise model is assumed to be t-distributed rather than the standard Gaussian distribution. This is important as the t-distribution is widely used when it is desired that the noise model be "robust", i.e. less sensitive to outliers than the Gaussian model. The MCMC approach can also be used in a hierarchical regression model where it is assumed that the noise process has a variance that depends on $\mathbf{x}$, and that this noise-field $N(\mathbf{x})$ is drawn from a prior generated from an independent Gaussian process $Z(\mathbf{x})$ by $N(\mathbf{x}) = \exp Z(\mathbf{x})$ (see Goldberg *et al*, 1997).

6 Discussion

In this paper I have shown how to move from simple Bayesian linear regression to regression with Gaussian processes, and have discussed some of the issues in using Gaussian process prediction for the kinds of problem that neural networks have also been used on.

There are still further elaborations that can be made. One weakness of the method described above is that the covariance functions used are stationary, which means that the length-scales over which interactions occur are the same everywhere in the input space. This is a very strong assumption, and can be relaxed in a number of ways. One appealing method is to warp the original input space $\mathbf{x}$ into another space, say $\boldsymbol{\xi}$, which may be taken to be of the same dimension as $\mathbf{x}$, and then to use a stationary covariance function in $\boldsymbol{\xi}$-space. This is, roughly speaking, the approach proposed by Sampson and Guttorp (1992). Of course this warping is defined by co-ordinate functions $\xi_i(\mathbf{x}), i = 1, \ldots p$, which may again be modelled with Gaussian processes[4], so one can derive a hierarchical Gaussian process model.

It is also interesting to consider the differences between finite neural network and Gaussian process priors. One difference is that in functions generated from finite neural networks, the effects of individual basis functions can be seen. For example, with sigmoidal units a steep step may be observed where one basis function (with large weights) comes into play. A judgement about whether this type of behaviour is appropriate or not should depend on prior beliefs about the problem at hand. Of course it is also possible to compare finite neural networks and Gaussian process predictions empirically. This has been done by Rasmussen (1996), where GP predictions (using MCMC for the parameters) were compared to those from Neal's MCMC Bayesian neural networks. These results show that for a number of problems, the predictions of GPs and Bayesian neural networks are similar, and that both methods outperform several other widely-used regression techniques.

Acknowledgements

I thank David Barber, Chris Bishop, David MacKay, Radford Neal, Manfred Opper, Carl Rasmussen, Richard Rohwer, Francesco Vivarelli and Huaiyu Zhu for helpful discussions about Gaussian processes over the last few years, and David Barber, Chris Bishop and David MacKay for comments on the manuscript.

[4] It may be desirable to impose the condition that the $\mathbf{x}$ to $\boldsymbol{\xi}$ mapping should be bijective.

References

Aizerman, M. A., E. M. Braverman, and L. I. Rozoner (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control* **25**, 821–837.

Barber, D. and C. K. I. Williams (1997). Gaussian Processes for Bayesian Classification via Hybrid Monte Carlo. In M. C. Mozer, M. I. Jordan, and T. Petsche (Eds.), *Advances in Neural Information Processing Systems 9*. MIT Press.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford: Clarendon Press.

Box, G. E. P. and G. C. Tiao (1973). *Bayesian Inference in Statistical Analysis*. Reading, Mass.: Addison-Wesley.

Bridle, J. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fougelman-Soulie and J. Herault (Eds.), *NATO ASI series on systems and computer science*. Springer-Verlag.

Cressie, N. A. C. (1993). *Statistics for Spatial Data*. New York: Wiley.

Duane, S., A. D. Kennedy, B. J. Pendleton, and D. Roweth (1987). Hybrid Monte Carlo. *Physics Letters B* **195**, 216–222.

Gelman, A., J. B. Carlin, H. S. Stern, and D. B. Rubin (1995). *Bayesian Data Analysis*. London: Chapman and Hall.

Gibbs, M. and D. J. C. MacKay (1997a). Efficient Implementation of Gaussian Processes. Draft manuscript, available from `http://wol.ra.phy.cam.ac.uk/mackay/homepage.html`.

Gibbs, M. and D. J. C. MacKay (1997b). Variational Gaussian Process Classifiers. Draft manuscript, available via `http://wol.ra.phy.cam.ac.uk/mackay/homepage.html`.

Girard, D. (1989). A fast "Monte Carlo cross-validation" procedure for large least squares problems with noisy data. *Numer. Math.* **56**, 1–23.

Girosi, F., M. Jones, and T. Poggio (1995). Regularization Theory and Neural Networks Architectures. *Neural Computation* **7(2)**, 219–269.

Goldberg, P. W., C. K. I. Williams, and C. M. Bishop (1997). Regression with Input-dependent Noise: A Gaussian Process Treatment. Accepted to NIPS*97.

Green, P. J. and B. W. Silverman (1994). *Nonparametric regression and generalized linear models*. London: Chapman and Hall.

Handcock, M. S. and M. L. Stein (1993). A Bayesian Analysis of Kriging. *Technometrics* **35(4)**, 403–410.

Hastie, T. (1996). Pseudosplines. *Journal of the Royal Statistical Society B* **58**, 379–396.

Hastie, T. J. and R. J. Tibshirani (1990). *Generalized Additive Models*. London: Chapman and Hall.

Hornik, K. (1993). Some new results on neural network approximation. *Neural Networks* **6**(8), 1069–1072.

Hutchinson, M. (1989). A stochastic estimator for the trace of the influence matrix for Laplacian smoothing splines. *Communications in statistics:Simulation and computation* **18**, 1059–1076.

Journel, A. G. and C. J. Huijbregts (1978). *Mining Geostatistics*. Academic Press.

Kimeldorf, G. and G. Wahba (1970). A correspondence between Bayesian estimation of stochastic processes and smoothing by splines. *Annals of Mathematical Statistics* **41**, 495–502.

MacKay, D. J. C. (1992). A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation* **4(3)**, 448–472.

MacKay, D. J. C. (1993). Bayesian Methods for Backpropagation Networks. In J. L. van Hemmen, E. Domany, and K. Schulten (Eds.), *Models of Neural Networks II*. Springer.

Mardia, K. V. and R. J. Marshall (1984). Maximum likelihood estimation for models of residual covariance in spatial regression. *Biometrika* **71(1)**, 135–146.

Neal, R. M. (1997). Monte Carlo Implementation of Gaussian Process Models for Bayesian Regression and Classification. Draft manuscript, available from `http://www.cs.toronto.edu/~radford/`.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. New York: Springer. Lecture Notes in Statistics 118.

O'Hagan, A. (1978). Curve Fitting and Optimal Design for Prediction (with discussion). *Journal of the Royal Statistical Society B* **40(1)**, 1–42.

O'Sullivan, F., B. S. Yandell, and W. J. Raynor (1986). Automatic Smoothing of Regression Functions in Generalized Linear Models. *Journal of the American Statistical Association* **81**, 96–103.

Poggio, T. and F. Girosi (1990). Networks for approximation and learning. *Proceedings of IEEE* **78**, 1481–1497.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical Recipes in C* (second ed.). Cambridge University Press.

Rasmussen, C. E. (1996). *Evaluation of Gaussian Processes and Other Methods for Non-linear Regression*. Ph. D. thesis, Dept. of Computer Science, University of Toronto. Available from `http://www.cs.utoronto.ca/~carl/`.

Ripley, B. (1996). *Pattern Recognition and Neural Networks*. Cambridge, UK: Cambridge University Press.

Sacks, J., W. J. Welch, T. J. Mitchell, and H. P. Wynn (1989). Design and Analysis of Computer Experiments. *Statistical Science* **4(4)**, 409–435.

Sampson, P. D. and P. Guttorp (1992). Nonparametric estimation of nonstationary covariance structure. *Journal of the American Statistical Association* **87**, 108–119.

Silverman, B. W. (1978). Density Ratios, Empirical Likelihood and Cot Death. *Applied Statistics* **27(1)**, 26–33.

Silverman, B. W. (1985). Some aspects of the spline smoothing approach to non-parametric regression curve fitting (with discussion). *J. Roy. Stat. Soc. B* **47(1)**, 1–52.

Skilling, J. (1993). Bayesian numerical analysis. In W. T. Grandy, Jr. and P. Milonni (Eds.), *Physics and Probability*. Cambridge University Press.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. New York: Springer Verlag.

von Mises, R. (1964). *Mathematical Theory of Probability and Statistics*. Academic Press.

Wahba, G. (1990). *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics. CBMS-NSF Regional Conference series in applied mathematics.

Whittle, P. (1963). *Prediction and regulation by linear least-square methods*. English Universities Press.

Williams, C. K. I. (1997a). Computation with infinite neural networks. Submitted to *Neural Computation*.

Williams, C. K. I. (1997b). Computing with infinite networks. In M. C. Mozer, M. I. Jordan, and T. Petsche (Eds.), *Advances in Neural Information Processing Systems 9*. MIT Press.

Williams, C. K. I. and C. E. Rasmussen (1996). Gaussian processes for regression. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8*, pp. 514–520. MIT Press.

Wong, E. (1971). *Stochastic Processes in Information and Dynamical Systems*. New York: McGraw-Hill.

Zhu, H. and R. Rohwer (1996). Bayesian Regression Filters and the Issue of Priors. *Neural Computing and Applications* **4**, 130–142.

Zhu, H., C. K. I. Williams, R. J. Rohwer, and M. Morciniec (1997). Gaussian Regression and Optimal Finite Dimensional Linear Models. Technical Report NCRG/97/011, Aston University, UK. Available from `http://www.ncrg.aston.ac.uk/Papers/` .