Bayesian Error Bars For Regression

CAZHAOW SALIH QAZAZ

Doctor Of Philosophy

THE UNIVERSITY OF ASTON IN BIRMINGHAM

November 1996

THE UNIVERSITY OF ASTON IN BIRMINGHAM

Bayesian Error Bars For Regression

CAZHAOW SALIH QAZAZ

Doctor Of Philosophy, 1996

Thesis summary

Regression problems are concerned with predicting the values of one or more continuous quantities, given the values of a number of input variables. For virtually every application of regression, however, it is also important to have an indication of the uncertainty in the predictions. Such uncertainties are expressed in terms of the error bars, which specify the standard deviation of the distribution of predictions about the mean. Accurate estimate of error bars is of practical importance especially when safety and reliability is an issue.

The Bayesian view of regression leads naturally to two contributions to the the error bars. The first arises from the intrinsic noise on the target data, while the second comes from the uncertainty in the values of the model parameters which manifests itself in the finite width of the posterior distribution over the space of these parameters.

The Hessian matrix which involves the second derivatives of the error function with respect to the weights is needed for implementing the Bayesian formalism in general and estimating the error bars in particular. A study of different methods for evaluating this matrix is given with special emphasis on the outer product approximation method.

The contribution of the uncertainty in model parameters to the error bars is a finite data size effect, which becomes negligible as the number of data points in the training set increases. A study of this contribution is given in relation to the distribution of data in input space. It is shown that the addition of data points to the training set can only reduce the local magnitude of the error bars or leave it unchanged. Using the asymptotic limit of an infinite data set, it is shown that the error bars have an approximate relation to the density of data in input space.

Equally important is the contribution of the intrinsic noise on the targets to the error bars. A review of various methods for estimating the variance of this random variable is given and a Bayesian technique is formulated for estimating this quantity as a function of the inputs.

Keywords: Bayesian inference, Maximum likelihood, Hessian matrix, Error bars, Prediction variance, Confidence variance, Noise variance

For my mother

in

her loving memory

Acknowledgements

This thesis which represents my work in the last three years could not have been produced without the support and assistance of a number of people to whom I would like to forward thanks.

First and foremost thanks go to my supervisor, Christopher Bishop, for his guidance, criticism and support during my PhD. Special thanks also go to Christopher Williams and David Barber for all the assistance I received from them during my studies. Thanks also go to many of the fellow members of the Neural Computing Research Group, notably Alan McLachlan, Mike Tipping, Markus Svensén, Ian Nabney, Ansgar West, David Lowe, Richard Rohwer, Michael Morciniec, David Saad, Krzysztof Zapart and Francesco Vivarelli. Equally, I am grateful to Hanni Sondermann for much administrative services. In addition I am grateful to Phil Barrett, John van der Rest, Conor Doherty, Richard Lister and Jason Price for their help with software and technical matters.

My special thanks to Shara Amin, Matthew Sullivan, Elizabeth Sullivan, Haemin Mufty, Zhean Mufty, Ahmed and Arie for their constant support during my studies.

I would like to thank Gurnam, my new friend, for all here support during the final stages of my PhD.

Without the support of my family this PhD would have been impossible. Their sacrifices is noted and appreciated.

This work is dedicated to my mother who will, regrettably, never see this thesis.

Several schematics and diagrams in this thesis has been inspired by similar diagrams appearing in published works:

Figure 1.3 .... (MacKay 1994b)

Figure 1.4 .... (MacKay 1991)

The following diagrams has been used in this thesis with the permission of their author:

Figures 4.7, 4.8 and 4.9 .... (Bishop and James 1993)

Finally, this work was funded by an EPSRC postgraduate scholarship.

Contents

List of figures

List of tables

List of publications

- C. K. I. Williams, C. Qazaz, C. M. Bishop and H. Zhu (1995). On the relationship between Bayesian error bars and the input data density. In *Proceedings Fourth IEE International Conference on Artificial Neural Networks*, Cambridge, UK, pp. 160–165. IEE.

- C. Qazaz, C. K. I. Williams and C. M. Bishop (1997). An Upper Bound on the Bayesian Error Bars for Generalised Linear Regression. *Mathematics of Neural Networks: Models, Algorithms and Applications.* Eds. S W Ellacott, J C Mason and I J Anderson, Kluwer.

- C. M. Bishop and C. Qazaz (1996). Bayesian Inference of Noise Levels In Regression. In *Proceedings of the International Conference on Artificial Neural Networks*, pp. 59–64, Eds. C. von der Malsburg, W. von Seelen, J. C. Vorbruggen and B. Sendhoff, Springer, Berlin.

- C. M. Bishop and C. Qazaz (1996). Regression with Input-Dependent Noise: A Bayesian Treatment. In M. C. Mozer, M. I. Jordan and T. Petsche (Eds.) *Advances in Neural Information Processing Systems 9* , MIT Press, Cambridge, USA.

- C. M. Bishop and C. Qazaz and C. K. I. Williams and H. Zhu. Bayesian Confidence Intervals for Regression. *in preparation for submission to IEEE Transactions on Neural Networks.*

Notation

N, number of data in the training set

i, labels data points

$\boldsymbol{x}_i$, ith input vector

$\boldsymbol{t}_i$, ith target vector

$p(\boldsymbol{x})$ input data density

$p(t|\boldsymbol{x})$ conditional density of the targets

$p(t, \boldsymbol{x})$ joint density of the data

$\boldsymbol{w}$, regression weights

$\boldsymbol{u}$, noise prediction weights

k_w, number of components in the weight vector $\boldsymbol{w}$

k_u, number of components in the weight vector $\boldsymbol{u}$

γ_w, number of well determined components in $\boldsymbol{w}$

γ_u, number of well determined components in $\boldsymbol{u}$

α_w, hyperparameter for controlling the weights $\boldsymbol{w}$

α_u, hyperparameter for controlling the weights $\boldsymbol{u}$

$y(\boldsymbol{x}; \boldsymbol{w})$, model output measured at input point $\boldsymbol{x}$

$\boldsymbol{g}(\boldsymbol{x})$, vector of the derivatives of the output with respect to the weights $\boldsymbol{w}$

$E_i(\boldsymbol{w}) = \frac{1}{2}(y(\boldsymbol{x}_i; \boldsymbol{w}) - t_i)^2$, least-square error for the ith data point

$E_D(\boldsymbol{w}) = \sum_{i=1}^{N} E_i(\boldsymbol{w})$, sum-of-squares error

$E_w(\boldsymbol{w})$, penalty term for controlling $\boldsymbol{w}$

$E_u(\boldsymbol{u})$, penalty term for controlling $\boldsymbol{u}$

$S(\boldsymbol{w}) = \beta E_D(\boldsymbol{w}) + \alpha_w E_w(\boldsymbol{w})$, rigid regression error

$\boldsymbol{C}$, prior Hessian matrix

$\boldsymbol{B}_i$, Hessian matrix for the ith datum

$\boldsymbol{B}$, data Hessian matrix

$\boldsymbol{A}$, full Hessian matrix

$\boldsymbol{G}$, Outer product Hessian

s^2, true noise variance

σ_ν^2, predicted noise variance

$\beta = \sigma_\nu^{-2}$, noise level

σ_w^2, confidence variance

σ_t^2, prediction variance

$|.|$, determinant

$\boldsymbol{T}$, matrix transpose

Chapter 1

Introduction

1.1 Bayesian theory of inference

Observation is the main source from which we acquire knowledge about the world we live in and the problems we want to solve. Observation is either the result of designed experiments or accidental events yielding information about the problem in hand. Either way the result of observation is the acquisition of raw data which need to be processed and analysed in order to learn about the problem of concern. It is often the case that data obtained from observation consists of random components, generally known as noise, on which we have little or no control. Learning from noisy data lies in the domain of that branch of science known as statistics.

To learn from data, statistics employs a tool known as *model* which contains a number of same or different types of *parameters* which we collectively denote by $\boldsymbol{\theta}$ which can be adjusted in the light of the available data. For example, if we were to model an unknown density function by a Gaussian distribution, then our model is the normal distribution with the parameters as mean μ and variance σ^2 of the distribution, *i.e.* $\boldsymbol{\theta} = \{\mu, \sigma^2\}$.

Main stream statistics can be divided into two schools of thought known as the *frequentist* and *Bayesian* approaches, each having its own philosophy and methodology of problem solving. In the frequentist approach one would usually be concerned with finding a single estimate of $\boldsymbol{\theta}$ which explains the data best. This is in contrast to the Bayesian approach where parameter estimation has no meaning. Here the parameters are treated as random variables having probability distributions quantifying the degree of belief in the values of θ in the light of the data. The probability distribution over the space of parameters θ is then used to make predictions.

To see the difference between the frequentist and Bayesian mechanisms of learning and how they work let us consider the problem of predicting the value of the random variable x from a number N of observed data $X = x_1, ..., x_N$. For both frequentist and Bayesian approaches the information stored in the data can be captured by the so called *likelihood* function $p(X|\theta)$, which is the probability of the data X given the model parameters θ. Assuming that the data points are independently selected then the likelihood can be written as

$$p(X|\theta) = \prod_{i=1}^{N} p(x_i|\theta) \tag{1.1}$$

To make predictions the frequentist approach uses the *most likely* value $\widehat{\theta}$ of the parameters which is found from maximising the likelihood function (1.1) given the set of data X. This is known as the frequentist *maximum likelihood* approach. Given the estimated value of the parameters $\widehat{\theta}$ the prediction of the new data is based on the *predictive distribution* $p(x|\widehat{\theta})$. Note that the dependency of $p(x|\widehat{\theta})$ on X is implicit in the estimate $\widehat{\theta}$.

Contrary to the frequentist approach, prediction on the basis of a single parameter estimate has no place in the Bayesian inference. As mentioned earlier, the Bayesian method is concerned with defining a probability distribution over the space of parameters which is subsequently used for predictions. Bayesian learning starts by recalling prior knowledge based on experience. For example if we were to predict the outcome of throwing a die, then it is reasonable to assume that obtaining any one of the outcomes $\{1, 2, 3, 4, 5, 6\}$ is equally likely. Such knowledge is incorporated into the Bayesian formalism by defining a *prior probability distribution* $p(\theta)$ over the space of parameters which embodies our knowledge of the problem. Priors are at the heart of Bayesian statistics and also the source of criticism of the Bayesian approach from the frequentists, since they are apparently arbitrary and it is often difficult to decide what is the best prior. The next step is to combine the prior $p(\theta)$ with the likelihood $p(X|\theta)$ to obtain the *posterior probability distribution* $p(\theta|X)$ according to Bayes' rule

$$p(\theta|X) = \frac{p(X|\theta)p(\theta)}{p(X)} \tag{1.2}$$

The posterior distribution in (1.2) therefore has two contributions: one is the data part which is just the likelihood function $p(X|\theta)$ which serves as a means of extracting information from the data. The second part is the prior $p(\theta)$ which quantifies our subjective beliefs. The likelihood and the prior are competing parts of the posterior distribution. In the limit $N \to \infty$ the prior will have little influence and the posterior $p(\theta|X)$ is mostly determined by the likelihood $p(X|\theta)$, in which case Bayesian prediction approaches that of the maximum likelihood. On the other hand, if a small number of observations were available, then the posterior is significantly influenced by the prior $p(\boldsymbol{\theta})$.

Having derived the posterior for the parameters $\boldsymbol{\theta}$ we can now make predictions of a datum x. From a Bayesian point of view making predictions using the most probable value[1] $\widetilde{\boldsymbol{\theta}}$ of the parameters which explains the data best is not the right procedure to follow. In order to make sound predictions we should allow all possible values of $\boldsymbol{\theta}$ to influence the prediction of x according the value of the posterior they achieve. The result is

$$p(x|X) = \int p(x|\boldsymbol{\theta})p(\boldsymbol{\theta}|X)\,d\boldsymbol{\theta} \tag{1.3}$$

where $p(x|X)$ is the Bayesian predictive distribution. Note that, unlike the maximum likelihood predictive distribution $p(x|\widehat{\boldsymbol{\theta}})$, $p(x|X)$ is not conditioned on the parameters $\boldsymbol{\theta}$. This is a consequence of marginalisation over these parameters as in (1.3). For sufficiently large N the posterior becomes narrow and so the integral in (1.3) can be approximated by

$$\begin{aligned} p(x|X) &\approx p(x|\widehat{\boldsymbol{\theta}}) \int p(\boldsymbol{\theta}|X)\,d\boldsymbol{\theta} \\ &= p(x|\widehat{\boldsymbol{\theta}}) \end{aligned} \tag{1.4}$$

Thus we see that the Bayesian approach includes parameter estimation, which is a maximum likelihood procedure, as a limiting case. While the ability to produce predictive distributions of the form (1.3) is a merit of Bayesian methods, it is also the source of difficulty in implementing this approach. Often the posterior $p(\boldsymbol{\theta}|X)$ is of a complex nature rendering the integral (1.3) analytically intractable and so indirect methods have to be used. The inability of modern mathematical techniques to tackle complex integrations of this form analytically is a mathematical problem inherited by Bayesian methods. In Section 1.6 we will review two different approaches to implementing the Bayesian formalism, one based on approximate analytical integration of (1.3), and the other based on numerical integration techniques.

Hierarchical models

It is often the case that in the Bayesian formalism the prior distribution for the parameters $\boldsymbol{\theta}$ is conditioned on another parameter α, *i.e.* $p(\boldsymbol{\theta}|\alpha)$, which itself is treated as a random variable having a posterior probability distribution $p(\alpha|X)$. Since the parameter α controls the distribution of the other parameters $\boldsymbol{\theta}$, it is called a *hyperparameter.* Such schemes are known as *hierarchical models* and can be extended to any level. In this case, the unconditional prior $p(\boldsymbol{\theta})$ can be obtained from

$$p(\boldsymbol{\theta}) = \int p(\boldsymbol{\theta}|\alpha)p(\alpha)\,d\alpha \tag{1.5}$$

where $p(\alpha)$ is the prior over the hyperparameter α.

[1] This value can be obtained from maximising the posterior $p(\boldsymbol{\theta}|X)$.

1.2 Data and the likelihood

In real world problems data often comes in the form of a set of inputs $X = \boldsymbol{x}_1, ..., \boldsymbol{x}_N$ and a set of targets $T = \boldsymbol{t}_1, ..., \boldsymbol{t}_N$. So that for each vector of inputs $\boldsymbol{x}_i = x_i^1, ..., x_i^d$ belonging to the set X there corresponds a target vector $\boldsymbol{t}_i = t_i^1, ..., t_i^m$ belonging to the set T. The inputs can be viewed as the locations where measurements are made and the targets are the outcomes of those measurements. The set of inputs and targets together form a data set $D = \{X, T\}$ from which inference is made. In classification problems the targets are binary data taking values zero or one and represent class membership of the inputs. This is in contrast to regression where the targets are measurements of continuous variables. The most general description of data of this form is given by the predictive distribution[2] $p(\boldsymbol{t}|\boldsymbol{x})$. Assuming that the distribution of different targets are independent, then can write

$$p(\boldsymbol{t}|\boldsymbol{x}) = \prod_{j=1}^{m} p(t^j|\boldsymbol{x}) \tag{1.6}$$

The distributions $p(t^j|\boldsymbol{x})$ are not known a priori but one can attempt to model them. To see how, let us consider a regression problem where it is assumed that the targets are related to the inputs through some deterministic function $f^j(\boldsymbol{x})$ with added noise. Therefore,

$$t_i^j = f^j(\boldsymbol{x}_i) + \nu_i \tag{1.7}$$

Assuming that the errors ν_i has a Gaussian distribution with true mean

$$\langle \nu_i \rangle = 0 \tag{1.8}$$

and variance[3]

$$\langle \nu_i \nu_{i'} \rangle = s^2 \delta_{ii'} \tag{1.9}$$

where $\delta_{ii'}$ is the Kronecker delta, then t^j is a Gaussian distribution too with mean $f^j(\boldsymbol{x})$, which is a function of the inputs, and variance[4] s^2. A schematic illustration of data of this form is given in Figure 1.1. Since the targets are noisy, repeated measurements at the same input $\boldsymbol{x}_i$ will yield different values of t_i^j. In the limit of an infinite sample the mean of t_i^j will approach the true function $f^j(\boldsymbol{x}_i)$.

[2] A more complete description of the data is given by the joint distribution $p(\boldsymbol{t}, \boldsymbol{x}) = p(\boldsymbol{t}|\boldsymbol{x})p(\boldsymbol{x})$, where $p(\boldsymbol{x}) = \int p(\boldsymbol{t}, \boldsymbol{x})\, d\boldsymbol{t}$ is the density of the input data. However, since we want to model the targets $\boldsymbol{t}$ conditioned on the inputs $\boldsymbol{x}$ the density function $p(\boldsymbol{x})$ has no significance here.

[3] For simplicity of notation we have assumed that the noise component ν is the same for all targets. This assumption can be relaxed in a straightforward way by introducing a diagonal covariance matrix with its elements consisting of the variance of noise of different targets.

[4] Here we have assumed that the noise variance is independent of the inputs. Input-dependent noise variance will be considered later in this chapter as well as in Chapter 4.

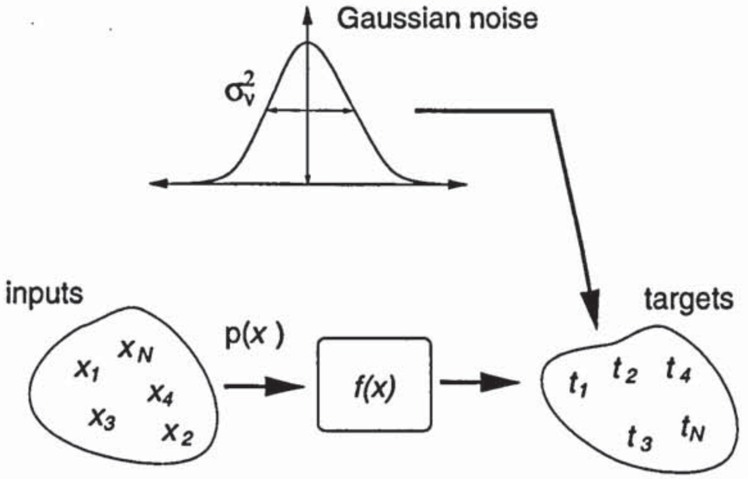

Figure 1.1: A schematic illustration of data for regression. The inputs are chosen according to some distribution $p(\boldsymbol{x})$. Due to the noise process, which is modelled as a Gaussian, the targets are shifted from their true values $f(\boldsymbol{x})$ to t as in formula (1.7).

Functions of this form are called regression functions (Nadaraya 1964; Xu et al. 1994). For each target we can write

$$p(t^j|\boldsymbol{x}) = \Big(\frac{1}{2\pi s^2}\Big)^{1/2}\exp\Big(-\frac{(f^j(\boldsymbol{x})-t^j)^2}{2s^2}\Big) \tag{1.10}$$

Since (1.10) is a Gaussian distribution, it is often convenient to summarise it by its mean $f^j(\boldsymbol{x})$ and variance s^2 which are not known a priori. Let $y^j(\boldsymbol{x};\boldsymbol{w})$ be the output of a model, *e.g.* polynomial regression or neural network, with *weights* $\boldsymbol{w}$ modelling the regression function $f^j(\boldsymbol{x})$. Likewise, let σ_ν^2 be the estimate of the true noise variance s^2. Then we can write

$$p(t^j|\boldsymbol{x},\boldsymbol{w},\beta) = \Big(\frac{\beta}{2\pi}\Big)^{1/2}\exp\Big(-\frac{\beta}{2}(y^j(\boldsymbol{x};\boldsymbol{w})-t^j)^2\Big) \tag{1.11}$$

where we have defined the noise level $\beta = \sigma_\nu^{-2}$. Note that the predictive distribution in (1.11) is now conditioned on the weights $\boldsymbol{w}$ and the estimate of the noise level β, since different values of these quantities yield different predictive distributions. Given (1.11) and assuming that the individual data points $(\boldsymbol{x}_i, \boldsymbol{t}_i)$ are independently selected we can write the likelihood function as

$$\begin{aligned} p(T|X,\boldsymbol{w},\beta) &\equiv p(D|\boldsymbol{w},\beta) \\ &= \prod_{i=1}^{N} p(\boldsymbol{t}_i|\boldsymbol{x}_i,\boldsymbol{w},\beta) \\ &= \frac{1}{Z_D(\beta)}\exp\Big(-\beta\sum_{j=1}^{m}E_D^j(\boldsymbol{w})\Big) \end{aligned} \tag{1.12}$$

where E_D^j is the so-called *sum-of-squares* error function for the jth output

$$E_D^j(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{N}\Big(y^j(\boldsymbol{x}_i;\boldsymbol{w})-t_i^j\Big)^2 \tag{1.13}$$

and $Z_D(\beta)$ is a normalising factor

$$Z_D(\beta) = \left(\frac{2\pi}{\beta}\right)^{\frac{mN}{2}} \tag{1.14}$$

Note that the appearance of the sum-of-squares error in (1.12) is a consequence of assuming a Gaussian noise on the targets. However, the use of sum-of-squares error does not require noise on the targets to have a Gaussian distribution.

As mentioned earlier, in the Bayesian approach the likelihood function is combined with a prior in order to obtain the posterior for the parameters $\boldsymbol{\theta} = (\boldsymbol{w}, \beta)$ which is subsequently used for making predictions. This is contrary to the frequentist approach which makes predictions using the single best estimate of the parameters $\widehat{\boldsymbol{\theta}} = (\widehat{\boldsymbol{w}}, \widehat{\beta})$ which are found from maximising the likelihood function $p(D|\boldsymbol{w}, \beta)$. Alternatively, one can minimise the error function $E(\boldsymbol{w}, \beta)$ defined as

$$\begin{aligned} E(\boldsymbol{w},\beta) &= -\ln p(D|\boldsymbol{w}) \\ &= \beta \sum_{j=1}^{m} E_D^j(\boldsymbol{w}) - \frac{mN}{2} \ln\beta + \text{Constant} \end{aligned} \tag{1.15}$$

These two procedures are equivalent since the negative logarithm is a monotonically decreasing function. Note that instead of minimising (1.15) we can find $\widehat{\boldsymbol{w}}$ from minimising the sum-of-squares error $E_D(\boldsymbol{w})$ as in equation (1.13). Given the sum-of-squares error our best guess of the target t^j is given by the output $y(\boldsymbol{x}; \widehat{\boldsymbol{w}})$ which is the mean of the predictive distribution $p(t^j|\boldsymbol{x}, \widehat{\boldsymbol{w}}, \widehat{\beta})$.

Contrary to regression, in classification problems the targets T are class labels taking binary values 0 or 1, *e.g.* $\boldsymbol{t}_i = (00 \cdot\cdot 1 \cdot\cdot 00)$, where the jth component being 1 indicates that the attribute $\boldsymbol{x}_i$ belongs to class j. Assuming that the input vectors are independent then the probability of $\boldsymbol{x}$ belonging to class 1, ..., m is given by

$$p(\boldsymbol{t}|\boldsymbol{x}, \boldsymbol{w}) = \prod_{j=1}^{m} \left(y^j(\boldsymbol{x}; \boldsymbol{w})\right)^{t_i^j} \tag{1.16}$$

Since the model outputs $y(\boldsymbol{x}; \boldsymbol{w})$ are viewed as the probabilities of $\boldsymbol{x}$ belonging to each of the classes, it is necessary that they i) range from 0 to 1 and ii) they sum to unity. To fulfill these requirements the outputs are chosen to be *softmax* (Bridle 1990) functions of the form

$$y^j(\boldsymbol{x}; \boldsymbol{w}) = \frac{\exp(\mathrm{a}^j(\boldsymbol{x}; \boldsymbol{w}))}{\sum_{j'=1}^{m} \exp(\mathrm{a}^{j'}(\boldsymbol{x}; \boldsymbol{w}))} \tag{1.17}$$

where a^j is some function of the inputs $\boldsymbol{x}$ and the weights $\boldsymbol{w}$. Using (1.16) we can write the likelihood as

$$\begin{aligned} p(T|X, \boldsymbol{w}) &= p(D|\boldsymbol{w}) \\ &= \prod_{i=1}^{N} \prod_{j=1}^{m} \left(y^j(\boldsymbol{x}_i; \boldsymbol{w})\right)^{t_i^j} \end{aligned} \tag{1.18}$$

As was the case for regression, the negative of the logarithm of the likelihood defines an error function, which is known as the *cross-entropy* error.

For both the sum-of-squares and cross-entropy errors, the output $y^j(\boldsymbol{x};\boldsymbol{w})$ has a simple interpretation at the minimum of the error function– in the limit $N \rightarrow \infty$, it is the conditional average $\langle t_i^j|\boldsymbol{x}\rangle$ of the targets (Bishop 1995a), *i.e.*

$$\begin{aligned} y^j(\boldsymbol{x};\widehat{\boldsymbol{w}}) &= \int t^j p(t^j|\boldsymbol{x})\, dt^j \\ &= \langle t^j|\boldsymbol{x}\rangle \end{aligned} \tag{1.19}$$

For regression problems we can write

$$\begin{aligned} y^j(\boldsymbol{x};\widehat{\boldsymbol{w}}) &= \langle t^j|\boldsymbol{x}\rangle \\ &= \langle (f^j(\boldsymbol{x})+\nu)|\boldsymbol{x}\rangle \\ &= f^j(\boldsymbol{x}) \end{aligned} \tag{1.20}$$

where we have used (1.7) and (1.8). Thus in the limit $N \rightarrow \infty$, the model will average over the noise and learns the true underlying function. For classification problems the model outputs approach the true probability of class membership.

1.3 Adaptive models

In the previous section we mentioned that the underlying function $f(\boldsymbol{x})$ is modelled using a mathematical function which is taken to be the output of an adaptive model with weights $\boldsymbol{w}$ which can be adjusted in the light of the training data. Such models can be divided into three separate groups, *parametric*, *non-parametric* and *semi-parametric* models. In the parametric case, the inputs-targets relation is assumed to have a certain functional form. For example, if we believe that the relation between a set of inputs-targets is linear then a first order polynomial would be the proper model to use in the regression. Often, however, the input-target relation is a complex one and difficult to guess. In such cases the choice of a parametric model might not be a good representative of the true functional form. Contrary to this, non-parametric models aim to discover the inputs-targets relation from the data alone. Such models, *e.g.* kernel regression (Scott 1992), typically grow in complexity in proportion to the size of the data set which makes them computationally difficult to use. Other models which aim to combine the advantages of both methods are known as semi-parametric models. Here the number of parameters can be systematically increased independent of the size of the training data giving ever more flexibility to the model. Examples of such models are *generalised linear regression*

(GLR)[5] models and *neural networks.*

1.4 Penalised maximum likelihood

The problem with the maximum likelihood approach is that it makes use of a single estimate $\widehat{\boldsymbol{w}}$ to make predictions and thus not allowing other values of the weights which may explain the data reasonably well to influence predictions. When the data set is small in relation to number of weights $\boldsymbol{w}$ the estimate $\widehat{\boldsymbol{w}}$ is poorly determined by the data. In this case the maximum likelihood solution $y(\boldsymbol{x};\widehat{\boldsymbol{w}})$ has the tendency of fine tuning to the data and hence discovering structure in the data which is due to noise rather than the true regression function $f(\boldsymbol{x})$, a problem which is known as *overfitting.* This problem can be alleviated using *early stopping* in which training is stopped when the model error on a *validation* data set reaches its minimum (Baldi and Chauvin 1991). Another technique for counter overfitting is to supply extra information. This is known as *regularization* (Tikhonov and Arsenin 1977). One particular form of regularization is the *weight decay* (Horel and Kennard 1970) leading to the so called *ridge regression* or *penalised maximum likelihood.* In this case the sum-of-squares error[6] E_D in (1.13) is replaced by the error function

$$S(\boldsymbol{w}) = \beta E_D(\boldsymbol{w}) + \frac{\alpha}{2}\,\boldsymbol{w}^T\boldsymbol{w} \tag{1.21}$$

and the error function (1.15) becomes

$$E(\boldsymbol{w},\beta) = S(\boldsymbol{w}) - \frac{mN}{2}\ln\beta + \text{Constant} \tag{1.22}$$

where $\alpha \geq 0$ is a regularising constant whose value can be determined using, for example, *cross-validation* methods (Stone 1974; Stone 1978; Wahba and Wold 1975). Note that the form of the regularising term is chosen to discourage large value of weights and thus prevent the output $y(\boldsymbol{x};\boldsymbol{w})$ from fine tuning to noisy targets. It has been empirically shown that such forms of regularization can lead to improvement in model generalisation (Hinton 1987). We will see in Section 1.6 that weight decay has a simple and natural interpretation in the Bayesian framework, it arises from the prior distribution of the weights.

[5] GLR models are reviewed in Section 3.2.

[6] For simplicity of notation we shall consider regression with a single output from now onwards. Reference to multiple outputs will be made when necessary.

1.5 Regression error bars: an overview

As we have discussed in Section 1.2, in the limit of an infinite amount of data the outputs approaches the conditional mean of the targets in which case the output $y(\boldsymbol{x};\widehat{\boldsymbol{w}})$ learns the regression function $f(\boldsymbol{x})$ by averaging over noise. However, due to the existence of intrinsic noise on the targets, we can not predict the outcome of t with certainty. The uncertainty in the predicted value of a target given an input is known as the *predictive variance* σ_t^2 and $\pm\sigma_t$ is known as the *predictive error bars* or *predictive bands.* In the limit of an infinite data $\sigma_t^2 = \sigma_\nu^2$. In reality, however, the number of data points in the training set is limited, in which case the output $y(\boldsymbol{x};\boldsymbol{w})$ can only approximate the true underlying function $f(\boldsymbol{x})$. This highlights the need for another from of uncertainty, which is the uncertainty in the predicted value of the function $f(\boldsymbol{x})$ given the input $\boldsymbol{x}$. This is known as the *confidence variance* $\sigma_w^2(\boldsymbol{x})$ and $\pm\sigma_w(\boldsymbol{x})$ is known as the *confidence error bars* or *confidence bands.* As we shall see later, the predictive variance is the sum of the noise and the confidence variances. In the rest of this chapter we will review different methods of estimating the predictive variance and its error bars using both Bayesian, maximum likelihood and *bootstrap methods.*

1.6 Bayesian framework for regression

In a regression problem the parameters are usually the weights $\boldsymbol{w}$ of the model which controls the output $y(\boldsymbol{x};\boldsymbol{w})$, the level of noise $\beta = \sigma_\nu^{-2}$ on the targets and a hyperparameter α which controls, through the conditional prior $p(\boldsymbol{w}|\alpha)$, the range of values $\boldsymbol{w}$ can take. In other words, α determines the strength of the prior on the weights $\boldsymbol{w}$. Both β and α are also treated as random variables with prior probability distributions $p(\beta)$ and $p(\alpha)$. The predictive distribution is obtained from

$$p(t|\boldsymbol{x},D) = \int p(t|\boldsymbol{x},\boldsymbol{w})p(\boldsymbol{w}|D)\,d\boldsymbol{w} \tag{1.23}$$

where $p(t|\boldsymbol{x},\boldsymbol{w})$ is the likelihood for the datum t and $p(\boldsymbol{w}|D)$ is the joint posterior probability distribution of the weights $\boldsymbol{w}$ which is obtained from

$$p(\boldsymbol{w}|D) = \iint p(\boldsymbol{w}|D,\beta,\alpha)p(\beta,\alpha|D)\,d\beta\,d\alpha \tag{1.24}$$

where $p(\boldsymbol{w}|D,\beta,\alpha)$ is the conditional posterior for $\boldsymbol{w}$ and $p(\beta,\alpha|D)$ is the posterior for β and α. Depending on needs of the user, one might be interested in predicting the target t given the input $\boldsymbol{x}$. For least-square error the best guess is to predict the mean of $p(t|\boldsymbol{x},D)$

$$y(\boldsymbol{x}) = \int y(\boldsymbol{x};\boldsymbol{w})p(\boldsymbol{w}|D)\,d\boldsymbol{w} \tag{1.25}$$

The error bars of the best guess can then be obtained from

$$\sigma_t^2(x) = \sigma_\nu^2 + \int \Big(y(x;w) - y(x)\Big)^2 p(w|D)\, dw \tag{1.26}$$

In the last few years there has been considerable progress in techniques of applying the Bayesian formalism to real world problems in the context of both regression and classification. Research into this field has focussed on two different methods, one is based on approximate analytical integration and the other is based on numerical integration techniques. In this section we will consider these two approaches in the context of regression.

1.6.1 The evidence framework

Here we consider the evidence framework which has been applied by MacKay (1991, 1992a, 1992c, 1994a), to neural networks in the context of both regression and classification problems and its usefulness as a tool for tackling real-worlds problems has been demonstrated (MacKay 1995; Thodberg 1994, 1996). Here we review the method in the context of regression. The framework involves two steps. In the first one the noise level β and the hyperparameter α are fixed to their most probable values $\widetilde{\beta}$ and $\widetilde{\alpha}$, and the conditional posterior $p(w|D,\widetilde{\beta},\widetilde{\alpha})$ is used to approximate the true posterior $p(w|D)$. This is known as the *evidence approximation.* In the second step the posterior $p(w|D,\widetilde{\beta},\widetilde{\alpha})$ is approximated by a Gaussian with mean located at $\widetilde{w}$. This is known as the *Gaussian* or *Laplace approximation* (Morris 1988).

The evidence approximation

Our task is to make predictions and estimate the error bars using (1.23) (or (1.25) and (1.26)). For this we need to know the posterior $p(w|D)$ which can be evaluated from (1.24). However, if the posterior $p(\beta,\alpha|D)$ is sharply peaked at the most probable values $\widetilde{\beta}$ and $\widetilde{\alpha}$ and if $p(w|D,\beta,\alpha)$ is a slow varying function of β and α near that peak, then the unconditional posterior can be approximated by

$$\begin{aligned} p(w|D) &\approx p(w|D,\widetilde{\beta},\widetilde{\alpha}) \iint p(\beta,\alpha|D)\, d\beta\, d\alpha \\ &= p(w|D,\widetilde{\beta},\widetilde{\alpha}) \end{aligned} \tag{1.27}$$

where

$$p(w|D,\beta,\alpha) = \frac{p(D|w,\beta)p(w|\alpha)}{p(D|\beta,\alpha)} \tag{1.28}$$

where $p(D|w,\beta)$ is the likelihood (1.12) and $p(D|\beta,\alpha)$ is the *evidence* for β and α. Equation (1.27) states that we should find the most probable values $\widetilde{\beta}$ and $\widetilde{\alpha}$ which maximise the posterior $p(\beta,\alpha|D)$ and then carry out the rest of the Bayesian formalism with the value of these parameters set to $\widetilde{\beta}$

and $\widetilde{\alpha}$. These values can be obtained from maximising the posterior distribution for β and α which is given by

$$p(\beta,\alpha|D) = \frac{p(D|\beta,\alpha)p(\beta,\alpha)}{p(D)} \tag{1.29}$$

where $p(\beta,\alpha) = p(\beta)p(\alpha)$ is the joint prior probability distribution for β and α, and $p(D)$ is a normalising constant. If we do not know what values β and α should take then we impose a vague (flat) prior expression reflecting our lack of knowledge about the values of these parameters. Such priors are *non-informative* (Berger 1985). For example, we may assume that all values of β and α are equally likely in the range $(0,\infty)$. Such priors are *improper* since they can not be normalised. In this case the posterior coincides with the peak of the *evidence* $p(D|\beta,\alpha)$, which can be evaluated from

$$p(D|\beta,\alpha) = \int p(D|w,\beta)p(w|\alpha)\, dw \tag{1.30}$$

where $p(D|w,\beta)$ is the likelihood as given by (1.12) and $p(w|\alpha)$ is the prior for the weights w. The most probable values $\widetilde{\beta}$ and $\widetilde{\alpha}$ are then found by maximising the evidence $p(D|\beta,\alpha)$, or minimising the negative of its logarithm. The approach above to handling β and α is known as the evidence approximation. It is based on techniques developed by Gull (1988, 1989) and Skilling (1991) and is computationally equivalent to *type II maximum likelihood* (Berger 1985).

Given the conditional posterior $p(w|D,\widetilde{\beta},\widetilde{\alpha})$ as an approximation to the true posterior $p(w|D)$ we can now make predictions from

$$p(t|x,D) \approx \int p(t|x,w,\widetilde{\beta})p(w|D,\widetilde{\beta},\widetilde{\alpha})\, dw \tag{1.31}$$

Alternatively, we may use

$$y(x) \approx \int y(x;w)p(w|D,\widetilde{\beta},\widetilde{\alpha})\, dw \tag{1.32}$$

and

$$\sigma_t^2(x) \approx \sigma_\nu^2 + \int \Big(y(x;w) - y(x)\Big)^2 p(w|D,\widetilde{\beta},\widetilde{\alpha})\, dw \tag{1.33}$$

Therefore, in order to make predictions we need (i) to find the most probable values $\widetilde{\beta}$ and $\widetilde{\alpha}$ and (ii) to derive a formula for the posterior $p(w|D,\beta,\alpha)$.

Prior probability distribution for the weights

In order to obtain an explicit expression for the posterior $p(w|D,\beta,\alpha)$ we need to choose the form of the prior $p(w|\alpha)$. Since we assumed that the targets are generated from a smooth function we need a prior on the weights which encourages smoothness. The simplest way of incorporating this requirement is to impose a *weight decay* prior of the form

$$p(w|\alpha) = \frac{1}{Z_w(\alpha)} \exp\Big(-\alpha E_w(w)\Big) \tag{1.34}$$

where $E_w(w)$ is the weight decay regulariser

$$E_w(w) = \frac{1}{2}\, w^T w \tag{1.35}$$

and $Z_w(\alpha)$ is a normalising constant

$$Z_w(\alpha) = \left(\frac{2\pi}{\alpha}\right)^{k/2} \tag{1.36}$$

and k is the number of weights of the model. The prior in (1.34) together with the quadratic choice of $E_w(w)$ (1.35) implies that the distribution of the weights is a Gaussian with zero mean and variance α^{-1}, hence its is a *Gaussian prior.* However, the prior in (1.34) is over simplified since it does take into account that the weights in multi-layer neural networks are divided into groups according to their scaling properties (see Figure 1.2). A more elaborate from of prior which accounts for this fact divides the weights into different groups (MacKay 1991, 1994a), such as the weights belonging to different layers of a network, and assigns separate Gaussian priors to each group. Another motivation for choosing different priors for different groups of weights is related to the problem of eliminating irrelevant inputs, a procedure known as *automatic relevance determination* ARD (MacKay 1995; Neal 1996). Here a separate prior is given to the input-hidden weights. For those inputs which are irrelevant the estimated value $\widetilde{\alpha}$ is large forcing the weights to take values close to zero. Other forms of priors include *weight sharing* (Nowlan and Hinton 1992) and *Laplacian* priors (Williams 1995). The latter is particularly suitable for switching off silent weights in neural networks, a process known as *pruning.* In this thesis only single Gaussian priors will be considered.

The Gaussian approximation

Given the choice of the Gaussian prior for the weights and the Gaussian noise model, we can now rewrite the posterior probability distribution for the weights as

$$p(w|D,\beta,\alpha) = \frac{1}{Z_S(\beta,\alpha)} \exp\Big(-S(w)\Big) \tag{1.37}$$

Where

$$S(w) = \beta E_D(w) + \alpha E_w(w) \tag{1.38}$$

and

$$Z_S(\beta,\alpha) = \int \exp\Big(-S(w)\Big)\, dw \tag{1.39}$$

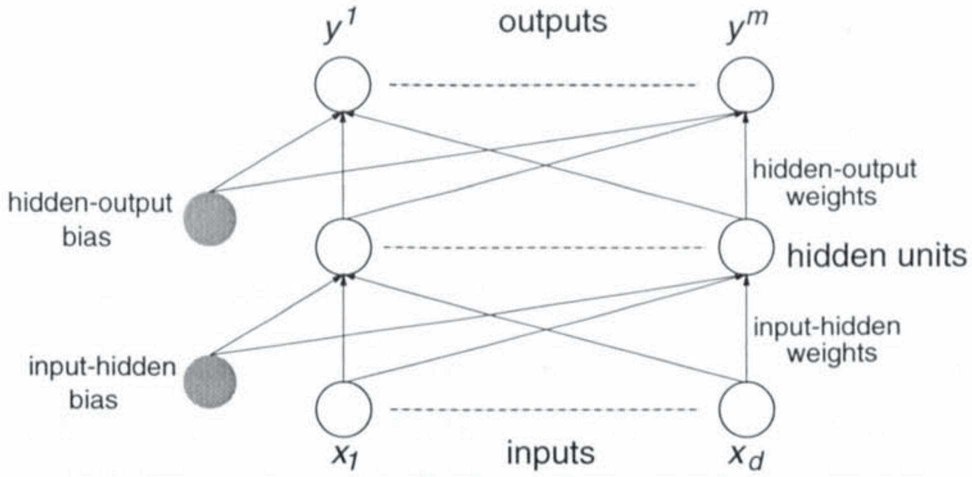

Figure 1.2: A schematic illustration of a two-layer neural network with feed-forward connections. The hidden units transforms the linear combination of the incoming inputs according to a mathematical function, *e.g.* tanh and sigmoid, which are subsequently fed into the output units. Likewise, the outputs transform the linear combination of its inputs (hidden outputs) according to a mathematical function.

For linear models, evaluation of the integral in (1.39) is straight forward since $S(\boldsymbol{w})$ is quadratic in the weights $\boldsymbol{w}$ and so $p(\boldsymbol{w}|D,\beta,\alpha)$ is a Gaussian. For non-linear models such as neural networks this is not the case, but we can attempt to approximate the posterior $p(\boldsymbol{w}|D,\beta,\alpha)$ by a multi-variate Gaussian distribution with mean centred at the most probable value $\widetilde{\boldsymbol{w}}$. This is known as the *Gaussian approximation* (MacKay 1991, 1992c; Buntine and Weigend 1991). Thus

$$p(\boldsymbol{w}|D,\beta,\alpha) = \frac{1}{Z_S(\beta,\alpha)} \exp\Big(-S(\widetilde{\boldsymbol{w}}) - \frac{1}{2}(\boldsymbol{w}-\widetilde{\boldsymbol{w}})^T \boldsymbol{A}(\boldsymbol{w}-\widetilde{\boldsymbol{w}})\Big) \tag{1.40}$$

and

$$Z_S(\beta,\alpha) = (2\pi)^{k/2}|\boldsymbol{A}|^{-1/2}\exp\Big(-S(\widetilde{\boldsymbol{w}})\Big) \tag{1.41}$$

where $|\boldsymbol{A}|$ is the determinant of the Hessian matrix $\boldsymbol{A}$

$$\boldsymbol{A} = \left.\frac{\partial^2 S(\boldsymbol{w})}{\partial \boldsymbol{w}^2}\right|_{\widetilde{\boldsymbol{w}}} \tag{1.42}$$

The motivation behind the Gaussian approximation is i) probability distributions approach Gaussian in the limit $N \to \infty$ (Walker 1969) and ii) Gaussian distributions are analytically easy to handle.

Evaluating the evidence

In order to obtain an explicit formula for the evidence $p(D|\beta,\alpha)$ we need to integrate over the weights $\boldsymbol{w}$ as in (1.30). Using (1.12), (1.34) and (1.38) in (1.30) we obtain the expression for the evidence

$$\begin{aligned} p(D|\beta,\alpha) &= \frac{1}{Z_D(\beta)Z_w(\alpha)} \int \exp\Big(-S(\boldsymbol{w})\Big)\, d\boldsymbol{w} \\ &= \frac{Z_S(\beta,\alpha)}{Z_D(\beta)Z_w(\alpha)} \end{aligned} \tag{1.43}$$

where $Z_D(\beta, \alpha)$ and $Z_w(\alpha)$ are given by (1.14) and (1.36), respectively. If we make use of the Gaussian approximation for the weights, $Z_S(\beta, \alpha)$ is then given by (1.41). Using formula (1.43), the most probable values $\widetilde{\beta}$ and $\widetilde{\alpha}$ are then found from maximising the evidence or from minimising the negative of its logarithm. These values are then used in (1.31) for making predictions.

1.6.2 Criticism of the evidence approximation

Strictly speaking, the evidence approximation is not a fully Bayesian procedure since the conditional posterior $p(w|D, \widetilde{\beta}, \widetilde{\alpha})$ is just an approximation to the true posterior $p(w|D)$ from which exact Bayesian predictions are to be made. While this fact is not in dispute, the question of how good the evidence is in approximating the Bayesian formalism has been a matter of heated debates. Wolpert (1993) criticises the evidence approach on the grounds that, since it is possible to obtain the true posterior $p(w|D)$ by integrating over the hyperparameters α and β analytically in the manner of Buntine and Weigend (1991), there is no need to go through approximation schemes. In this approach, which MacKay (1994b) calls *maximum a posteriori* MAP, the true posterior is given by

$$\begin{aligned} p(w|D) &= \iint p(w, \beta, \alpha|D)\, d\beta\, d\alpha \\ &= \frac{1}{p(D)} \int p(D|w, \beta) p(\beta)\, d\beta \int p(w|\alpha) p(\alpha)\, d\alpha \end{aligned} \tag{1.44}$$

Since β and α are scale parameters, *i.e.* control width of distributions, it is proper to impose improper priors of the form $p(\ln \beta) = 1$ and $p(\ln \alpha) = 1$ which mean

$$p(\beta) = \frac{1}{\beta} \tag{1.45}$$

$$p(\alpha) = \frac{1}{\alpha} \tag{1.46}$$

Using (1.45) and (1.46) in (1.44), we obtain the true posterior

$$\begin{aligned} p(w|D) &= \frac{p(D|w)p(w)}{p(D)} \\ &\propto \frac{\Gamma(k/2)}{\Big(2\pi E_w(w)\Big)^{k/2}} \frac{\Gamma(N/2)}{\Big(2\pi E_D(w)\Big)^{N/2}} \end{aligned} \tag{1.47}$$

where Γ is the standard gamma function.

While such objections should be a matter of concern, what matters from a practical point of view is how good the evidence framework approximates the full Bayesian predictions. This depends on how well $p(w|D, \beta, \alpha)$ approximates the true posterior $p(w|D)$. In this context, it turns out that the evidence and Gaussian approximations are related. As MacKay (1994b) and Neal (1995) argued, although integrating over β and α seems to be beneficial, it can sometimes magnify the error resulting

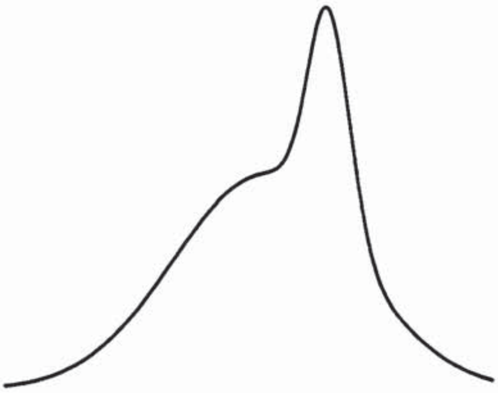

Figure 1.3: A schematic example of a posterior distribution with ill-determined parameters for which the MAP approach together with the Gaussian approximation can lead to significant error in approximating the true posterior $p(w|D)$. The MAP approach is capable of finding the most probable value $\tilde{w}$ which is subsequently used, together with the Hessian A^{-1}, to define the centre and covariance of a multi-variate Gaussian distribution for approximating the true posterior. Such an approach can lead to an entirely wrong and yet confident model.

from the Gaussian approximation to the posterior. To get an idea how this could be, let us first consider linear models. In this case integration over w can be carried exactly since the posterior $p(w|D, \widetilde{\beta}, \widetilde{\alpha})$ is Gaussian. However, if we choose to integrate over β and α as in (1.44), then the resulting posterior $p(w|D)$ is no longer Gaussian. However, in order to carry out the rest of the Bayesian analysis we have still to assume that $p(w|D)$ can be approximated by a Gaussian distribution. Since predictions are more sensitive to integration over w than integration over β and α, the MAP method can produce results more in error than the evidence procedure.

The above justification to the evidence approximation seems to have no meaning in the context of non-linear models such as neural networks since the posterior $p(w|D, \beta, \alpha)$ is not a Gaussian anyway. However, even for such models the evidence procedure can some times yield better results than the MAP. The reason is that (MacKay 1994b) if there are many *ill-determined* parameters, which is typical when the number of model parameters is large relative to the number of data, then $p(w|D)$ will tend to have a sharp peak favouring a small range of weights as shown in Figure 1.3. In this case the Gaussian approximation is not a good representative of the posterior $p(w|D)$. However, if we choose to approximate the true posterior by $p(w|D, \widetilde{\beta}, \widetilde{\alpha})$ then the subsequent Gaussian approximation will be a better representative of $p(w|D)$ and so will be less in error.

1.6.3 Bayesian model comparison

So far we have restricted our discussion of the Bayesian analysis to the case of one model. It is also possible to tackle the problem of ranking different models using the Bayesian framework. For instance, we may wish to use linear models as well as neural networks of different architecture, type and complexity to make predictions. Using Bayes' rule, we assign preferences to alternative models according to the posterior they achieve

$$p(H_j|D) = \frac{p(D|H_j)p(H_j)}{p(D)} \tag{1.48}$$

where $p(H_j)$ is the prior for the jth model, $p(D|H_j)$ is called the *model evidence.* If we have no reason to prefer one model over the others, which would be the case if the models explain the data well, then we should assign equal priors $p(H_j)$ to all models. In this case different models can be ranked according to the evidence they achieve. The evidence itself can be obtained from

$$p(D|H_j) = \iint p(D|\beta, \alpha, H_j)p(\beta, \alpha|H_j)\, d\beta\, d\alpha \tag{1.49}$$

where the evidence $p(D|\beta, \alpha, H_j)$ and the prior $p(\beta, \alpha|H_j)$ are now conditioned on the model H_j. This change will not affect our previous analysis of Bayesian inference since the probability distributions we have considered are now only conditioned on the type of the model H_j in use.

The evidence $p(D|H_j)$ and the principle of Occam's razor are intimately related (MacKay 1992a). This can be shown by writing the evidence $p(D|H_j)$ in the form

$$p(D|H_j) = \int p(D|w, H_j)p(w|H_j)\, dw \tag{1.50}$$

where $p(D|w, H_j)$ is the likelihood and $p(w|H_j)$ is the prior for the weights of the model H_j. For a sharply peaked posterior (see Figure 1.4) around $\tilde{w}$ we can approximate the integral by

$$p(D|H_j) \simeq p(D|\tilde{w}, H_j)p(\tilde{w}|H_j)\Delta w \tag{1.51}$$

where Δw is the width of the posterior weight. Taking a uniform prior over a width Δw_o we have

$$p(D|H_j) \simeq p(D|\tilde{w}, H_j)\, \frac{\Delta w}{\Delta w_o} \tag{1.52}$$

where Δw_o is the width of the prior. The likelihood $p(D|\tilde{w}, H_j)$ can be regarded as a measure of how well the model fits the data. The second term $\Delta w/\Delta w_o (< 1)$ is the Occam factor. Generally speaking, models with large number of weights achieve large values of the likelihood as they can finely tune to the data but they also have small Occam factors. The model which makes the best trade off between complexity and minimising the data misfit will achieve the best evidence. A similar result can be obtained from consideration of the minimum description length (Rissanen 1978).

In the Bayesian framework, model comparison is a way of giving preferences to different models rather than a means of selecting the model and discarding the others. In other words it is a discrete form of marginalisation. In fact the correct Bayesian procedure requires that we use the complete set of models for making predictions using the weighted average of the outputs of the models, with weighting coefficients being the posterior probabilities of the models. Thus

$$y(x) = \sum_{j=1}^{m} y_j(x)p(H_j|D) \tag{1.53}$$

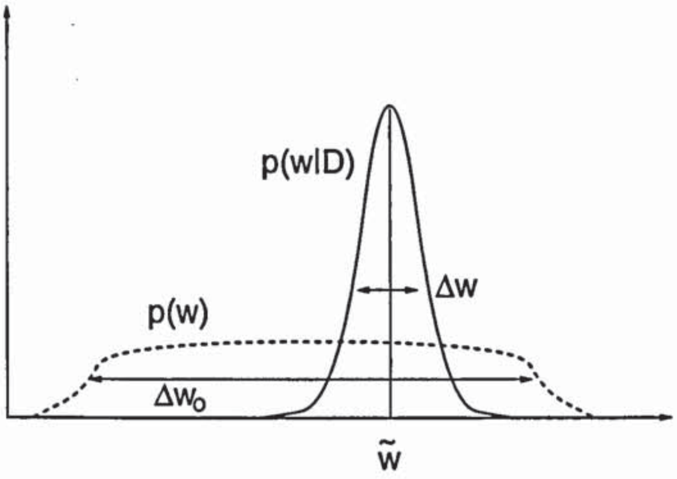

Figure 1.4: An illustration of the mechanism of Occam factor. The prior $p(w)$ is taken to be flat for a range of weights Δw_o. The arrival of data results in a posterior $p(w|D)$ with width Δw. The ratio $\Delta w/\Delta w_o$ is the Occam factor which favours simpler models.

where $y_j(x)$ is the output of the jth member of the committee of models and is given by (1.32). However, the Gaussian approximation of the posterior usually gives a poor estimation of the true evidence. Such problems have led Thodberg (1994, 1996) to use the evidence only as a criterion for selecting a committee of models whose members achieve the best evidence.

1.6.4 Non-equivalent multiple modes

So far we have considered the Gaussian approximation for the case of a single posterior mode. For neural networks the posterior have many modes some are equivalent pertaining to the symmetry of the network and some are non-equivalent modes pertaining to different solutions. In the evidence procedure the existence of multiple modes is handled using an approach similar to model comparison which involves partitioning (MacKay 1992c) the posterior space into sections each defined by the domain of its own and then regarding each partition as a sub model within a model. Using the Gaussian approximation we can approximate each model by fitting a multivariate Gaussian centred at the most probable value of the mode with covariance given by the inverse of the Hessian matrix $\boldsymbol{A}$. In this way the rest of Bayesian inference applies. However, the success of such approach depends on the assumption that the modes of the posterior are well separated so that the Gaussians do not significantly overlap.

1.6.5 Three levels of inference

Thus we can distinguish between three levels of inference for implementing the Bayesian formalism using the evidence framework:

i) In the first level we evaluate the weights $\boldsymbol{w}$ from minimising $-\ln p(\boldsymbol{w}|D,\beta,\alpha)$ given the current values of the parameters β and α.

ii) In the second level we estimate the values of β and α from minimising the evidence $-\ln p(D|\beta, \alpha)$. The above steps are repeatedly applied until the most probable values $\widetilde{w}$, $\widetilde{\beta}$ and $\widetilde{\alpha}$ are found.

iii) Finally, we can assess alternative models on the basis of the evidence $p(D|H_j)$ they achieve.

1.6.6 Predictions and error bars estimation

As mentioned earlier, evaluating the predictive distribution $p(t|x, D)$ is the ultimate goal of the Bayesian formalism. Assuming the evidence approximation $p(w|D) \approx p(w|D, \widetilde{\beta}, \widetilde{\alpha})$, the predictive distribution is obtained by integrating over w as in (1.31). Given the Gaussian approximation for the posterior $p(w|D, \widetilde{\beta}, \widetilde{\alpha})$ and the assumption that the output depends linearly on the weights w in the vicinity of $\widetilde{w}$

$$y(x; w) = y(x; \widetilde{w}) + (w - \widetilde{w})^T g(x) \tag{1.54}$$

where $g(x) = \partial y(x; w)/\partial w$ is the vector of derivatives of the output with respect to the weights measured at $\widetilde{w}$, then the distribution $p(t|x, D)$ is a Gaussian with mean $y(x; \widetilde{w})$ and variance $\sigma_t^2(x)$ given by (MacKay 1994a)

$$\sigma_t^2(x) = \sigma_\nu^2 + \sigma_w^2(x) \tag{1.55}$$

with

$$\sigma_\nu^2 = \beta^{-1} = \frac{2E_D(\widetilde{w})}{N - \gamma} \tag{1.56}$$

and

$$\sigma_w^2(x) = g^T(x) A^{-1} g(x) \tag{1.57}$$

where A is the Hessian matrix measured at the most probable value of the weights $\widetilde{w}$ (see equation (1.42)). The quantity $\gamma \leq k$ is the *number of well determined weights* (MacKay 1992a; Moody 1992) and is evaluated from

$$\gamma = k - \alpha \,\mathrm{Trace}(A^{-1}) \tag{1.58}$$

where the hyperparameter α is estimated from

$$\alpha = \frac{\gamma}{2E_w(\widetilde{w})} \tag{1.59}$$

Formulae (1.56) and (1.59) can be easily verified by minimising the negative logarithm of the evidence $p(D|\beta, \alpha)$ with respect to β and α. The linearization of the outputs as given by (1.54) is exact for linear models since the outputs depends linearly on the weights. For non-linear models such as neural networks, however, it is just an approximation which is valid only if the posterior width is narrow enough. Note that formula (1.56) is similar to the type II maximum likelihood formula for estimating the variance of N independent Gaussian random variables with mean μ and common variance σ^2, *i.e.* $\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \widehat{\mu})^2$. However, in formula (1.56) we have the factor $N-\gamma$ rather than $N-k$, since only γ out of k parameters can suppress noise. We shall come back to this point in Section 4.4.

The prediction variance $\sigma_t^2(x)$ and its error bars $\pm\sigma_t(x)$ is the centre of this thesis study. As formula (1.55) shows, $\sigma_t^2(x)$ receives contributions from two different sources. The first source is the intrinsic noise on the target data while the second source is the uncertainty in the weights due the finite size of the training data. As the size N of the data grows larger the width of the posterior becomes narrower. In the limit $N \to \infty$, $\sigma_w^2(x) \to 0$ and so the prediction variance is given by the noise variance σ_ν^2. As we shall see in Chapter 3, there is an intimate relation between the distribution of data in input space and the confidence variance $\sigma_w^2(x)$.

1.6.7 Markov chain Monte Carlo methods

Although the evidence framework provides a practical way of applying the Bayesian formalism, it suffers from the limitations of the evidence and Gaussian approximations. For non-linear models such as multi-layer neural networks of sizes large compared to the number of training data, the Gaussian approximation can be poor. An alternative approach for implementing the Bayesian formalism which avoids making these approximations is offered by numerical integration. Here instead of attempting to resort to analytical integration one can employ numerical integration techniques. However, given the high dimensionality of the posterior distributions for neural networks, direct implementation of ordinary numerical techniques is of no use given the computation overhead for implementing such schemes. Instead, one can resort to sampling techniques known as *Monte Carlo* methods. In this context, the task of implementing the Bayesian formalism is converted into the task of generating an n sample of points in the parameter space drawn from the posterior probability distribution $p(w|D)$ and evaluate the outputs and error bars from

$$\begin{aligned} y(x) &= \int y(x;w)p(w|D)\,dw \\ &\approx \frac{1}{n}\sum_{j=1}^{n} y(x;w_j) \end{aligned} \tag{1.60}$$

$$\sigma_t^2(x) \approx \frac{1}{n-1}\sum_{j=1}^{n}\Big(y(x;w_j) - y(x)\Big)^2 \tag{1.61}$$

where w_j is the jth component of the random sample. Then the task of implementing the Bayesian formalism reduces to the problem of generating a sample which is a true representative of the posterior

$p(w|D)$ in an affordable amount of time. There are several techniques for achieving this. For example, one might resort to the so called *importance sampling* where one can consider a distribution, say $q(w)$ which is easy to sample from. This method, however, can be computationally prohibitive, since for neural networks most of the posterior mass is confined to small regions of weight space and so a large sample size is required to obtain a good approximation to the integrations in (1.60) and (1.61). One particular technique which can overcome this difficulty to some extent is known as the *Markov chain Monte Carlo* technique. Here the objective is to construct a Markov chain whose equilibrium distribution is the required posterior distribution. Such a Markov chain can be obtained using Metropolis algorithm and Gibbs sampling. One of the potential limitation of such algorithms is that most of the samples might still come from low regions of posterior density space. Another potential limitation is that such algorithms can end up sampling from a single mode of the posterior distribution. Recently, a more elaborate form of sampling techniques known as *hybrid Monte Carlo* developed by Duane et al. (1987) and applied by Neal (1995, 1992) to neural networks has shown that numerical integration techniques can be a promising approach for implementing the Bayesian formalism.

1.7 Other methods of error bars estimation

So far we have discussed error bars from a Bayesian point of view. Other approaches to error bar estimation based on maximum likelihood and *bootstrap* methods are also possible. Here we review these methods.

1.7.1 Maximum likelihood methods

As pointed out earlier in this chapter, in the Bayesian approach model parameters are treated as random variables with probability distributions from which predictions are made. The posterior probability distribution over the weights together with the intrinsic noise on the targets induces a predictive probability distribution with variance $\sigma_t^2(x)$ as given by formula (1.55). It is also possible to obtain a similar result using the maximum likelihood approach, though one have to use a different kind of reasoning which is as follows (Chryssolouris et al. 1996; Tibshirani 1996; Efron and Tibshirani 1993). Let us consider a regression model which is trained on a number of n samples of size N each. Let $\widehat{w}_l$ be the estimate of the weights given the lth sample. We can write the predictive variance as

$$\sigma_t^2(x) = \left\langle \left(t - y(x; \widehat{w}_l) \right)^2 \right\rangle \tag{1.62}$$

where $\langle . \rangle$ denotes expectation defined over the n samples. If we assume that the sample size N is sufficiently large so that $\widehat{w}_l \approx w^*$, where w^* is the estimate of the weights w at the limit $N \to \infty$, then we have

$$y(x; \widehat{w}_l) \approx y(x; w^*) + (\widehat{w}_l - w^*)^T g(x) \tag{1.63}$$

where $g(x) = \partial y(x; w)/\partial w\Big|_{w^*}$. Using (1.63) in (1.62), we have

$$\sigma_t^2(x) \approx \left\langle \left(t - y(x; w^*)\right)^2 \right\rangle + g(x)^T \left\langle (\widehat{w}_l - w^*)(\widehat{w}_l - w^*)^T \right\rangle g(x) \tag{1.64}$$

The first term on the right hand side of (1.64) is precisely the value of the noise variance σ_ν^2. The quantity $\left\langle (\widehat{w}_l - w^*)(\widehat{w}_l - w^*)^T \right\rangle$ is the variance of the sample of $\widetilde{w}_l$. Therefore, the second term on the right hand side of (1.64) is the variance the outputs of the model, *i.e.* it is the confidence variance. Since in the limit $N \to \infty$, $\widehat{w}_l \to w^*$, the second term in (1.64) vanishes and $\sigma_t^2 = \sigma_\nu^2$. Assuming that $\widehat{w}_l - w^*$ is normally distributed, then we can write

$$\begin{aligned} \sigma_t^2(x) &= \sigma_\nu^2 + \sigma_w^2(x) \\ &= \sigma_\nu^2 + \sigma_\nu^2\, g(x)^T B^{-1} g(x) \end{aligned} \tag{1.65}$$

where $B = \partial^2 E_D / \partial w^2$ is the data Hessian (see Chapter 2) measured at $\widehat{w}$. The maximum likelihood formula (1.65) is similar in form to the Bayesian formula for evaluating the predictive variance. However, there is a significant difference in the way the estimate of the variance σ_ν^2 is obtained using these two different approaches. The maximum likelihood estimate of σ_ν^2 is given by

$$\sigma_\nu^2 = \frac{2E_D}{N} \tag{1.66}$$

This can be easily verified from maximising (1.15) with respect to σ_ν^2 with $m = 1$. It is a well known fact that the maximum likelihood formula (1.66) for estimating the noise variance is biased since it underestimates the noise variance while the Bayesian formula (1.56) is not. We will deal with this matter in detail in Chapter 4.

Input-dependent noise variance

So far we have assumed that the intrinsic noise on the targets is generated from a Gaussian distribution with a constant variance parameter. In practical applications this can be a too restrictive assumption leading to a poor representation of the predictive distribution. Fortunately, one can extend the analysis of this chapter to include the more general case of an input-dependent noise variance which varies as a function of the inputs. As before, the true estimate of the noise variance $s^2(x)$ is not known but one attempt to model it using the output of a model $\beta(x; u) = \sigma_\nu^{-2}(x; u)$ with weights u. In this case predictive the distribution is given by

$$p(t|\boldsymbol{x},\boldsymbol{w},\boldsymbol{u}) = \Big(\frac{\beta(\boldsymbol{x};\boldsymbol{u})}{2\pi}\Big)^{1/2} \exp\Big(-\frac{\beta(\boldsymbol{x};\boldsymbol{u})}{2}(y(\boldsymbol{x};\boldsymbol{w}) - t)^2\Big) \tag{1.67}$$

As before we interpret the negative of the logarithm of the likelihood function as an error function

$$E = \frac{1}{2}\sum_{i=1}^{N}\beta(\boldsymbol{x}_i;\boldsymbol{u})(y(\boldsymbol{x}_i;\boldsymbol{w}) - t_i)^2 - \frac{1}{2}\sum_{i=1}^{N}\ln\beta(\boldsymbol{x}_i;\boldsymbol{u}) \tag{1.68}$$

The most likely values $\widetilde{\boldsymbol{w}}$ and $\widetilde{\boldsymbol{u}}$ are then obtained from minimising the error function in (1.68). Such an approach has been used by Nix and Weigend (1994, 1995) for inferring an input-dependent noise variance. The more general case of correlated noise for multiple outputs is also integrated into the maximum likelihood formalism by Williams (1996).

Another maximum likelihood approach for estimating an input-dependent noise variance is based on the *residual error* method (Satchwell 1994). Here a regression model is trained on the data set using sum-of-squares error to obtain the outputs $y(\boldsymbol{x};\boldsymbol{w})$. To obtain the variance a new data set is created in the form $(\boldsymbol{x}_i, r_i^2)$, where $r_i^2 = \Big(y(\boldsymbol{x}_i;\boldsymbol{w}) - t_i\Big)^2$. This new data set is then used to compute the outputs $\sigma_\nu^2(\boldsymbol{x})$ again using the sum-of-squares error. The justification behind such an approach is based on the fact that since for the sum-of-squares error the outputs approximates the conditional mean of the targets (see Section 1.2) then the second model gives, in the limit $N \to \infty$, the conditional average of $\Big(t_i - \langle t_i|\boldsymbol{x}_i\rangle\Big)^2 = \Big(t_i - y(\boldsymbol{x}_i;\widehat{\boldsymbol{w}})\Big)^2$ (Bishop 1995a) which is the variance of the intrinsic noise on the targets by definition.

Another approach for estimating an input-dependent noise variance is offered by the so called *mixture density model* (Bishop 1994a; Ormoneit and Tresp 1996). Here the distribution of the targets is modelled by a Gaussian mixture with means, variances and mixing coefficients as input-dependent variables which are modelled using the outputs of a neural network. The advantage of such an approach lies in the use of a mixture of Gaussians which, if given sufficient components, can approximate, at least in principle, any distribution.

However it should be mentioned that the approaches we have considered in this Section for estimating the noise variance are biased, since they are based on the maximum likelihood method, and so they have the tendency to underestimate the noise variance and, hence, the error bars.

1.7.2 Bootstrapping

A different approach to error bar estimation is based on bootstrapping (Efron and Tibshirani 1993; Tibshirani 1996). Here a number of n of data points are drawn randomly from the available dataset with replacement. Then the process is repeated independently until a number of m samples are obtained, each having n data points from the original dataset, some appearing zero times, some appearing once or more. Each sample is then used to train a regression model and the outputs are averaged according to

$$y(x) = \frac{1}{m}\sum_{l=1}^{m} y(x; w_l) \tag{1.69}$$

where $y(x; w_l)$ is the output of the model with weights w_l estimated from the lth bootstrap sample. Equation (1.69) is called the *bagged* (for bootstrap aggregated) estimator and disregards the performance of the individual models. Another method to obtain the estimate $y(x)$ is to use a *bumping* estimator (Breiman 1994) in which one would throw away all the models except the one which achieves minimum error on the complete data set. An alternative approach to the methods described above is to strike a balance between bagging and bumping. This can be achieved by taking the weighted average (Heskes 1996) of the outputs $y(x; w_l)$.

Recalling that the targets are generated from a deterministic function of the inputs $f(x)$ which is corrupted by the addition of Gaussian noise, and that the bootstrap outputs $y(x; w_l)$ are noisy measurement of the true function $f(x)$, then the estimate of the confidence variance is given by (Tibshirani 1996, equation (3.1))

$$\sigma_w^2(x) = \frac{1}{m-1}\sum_{l=1}^{m}\Big(y(x; w_l) - y(x)\Big)^2 \tag{1.70}$$

Assuming an uncorrelated error ν we have,

$$\begin{aligned} \sigma_t^2(x) &= \langle \nu^2 \rangle + \langle (y(x; w_l) - y(x))^2 \rangle \\ &= \sigma_\nu^2 + \sigma_w^2(x) \end{aligned} \tag{1.71}$$

where $y(x)$ is given by (1.69). One of the advantages of bootstrap based methods for estimating the error bars is that it is straight forward to implement and can avoid some of the symptoms of not having sufficient data to make reliable predictions. On the other hand, a good estimate of error bars requires a number of bootstrap samples which ranges between 25 to 200 samples (Tibshirani 1996). For large neural networks this is a problem given the computation required to train the network on the bootstrap samples.

1.8 The rest of this thesis

In this chapter we have reviewed the Bayesian mechanism of learning in the context of regression. Here we saw that the error bars arise naturally from the existence of the intrinsic noise on the target data and also from the finite width of the posterior. We also reviewed other methods of estimating error bars using maximum likelihood and bootstrap based methods. In the rest of this thesis we will study and explore the behaviour of the prediction variance and its error bar. The plan of the rest of this thesis is as follows:

Chapter2: Evaluation of the Hessian matrix

In this chapter we consider analytical, exact and approximate methods of evaluating the Hessian matrix for implementing the evidence framework in general and evaluating the Bayesian error bars in particular.

Chapter3: Error bars and the distribution of input data

This chapter deals with the error bars and their relation to the distribution of the input data, focusing on the issue of whether the error bars are systematically larger in the regions of input space where the density of the input data is low.

Chapter4: Inferring an input-dependent noise variance

In this chapter we will consider the case of noisy regression where the contribution of the intrinsic noise on the targets is input-dependent. We compare Bayesian and the maximum likelihood treatments, and show that the Bayesian approach overcomes a significant problem with the maximum likelihood.

Chapter5: Summary, conclusions and directions for future work

This chapter concludes the thesis with a summary of the results as well as suggestions for future work.

Finally, I declare that the contents of the rest of this thesis is original work and has not previously appeared elsewhere with the exception of the research papers which are displayed in the list of publications.

Chapter 2

The Hessian matrix

2.1 Introduction

The elements of the Hessian matrix consists of the second partial derivatives of the error function with respect to adaptive weights and biases (weights in short) in the network. Knowledge of the Hessian is necessary for a wide range of tasks. It is needed, in the Bayesian framework, for estimation of the effective number of weights (MacKay 1992c), for estimation of evidence for model hyperparameters and for assigning error bars to network outputs. The Hessian matrix also plays an equally important role in non-Bayesian methods such as minimising system complexity by pruning of low saliency weights (Le Cun et al. 1990; Hassibi et al. 1994), second-order optimisation methods (Becker and LeCun 1989; Ricotti et al. 1988) and fast network retraining after a small change in the training data (Bishop 1991).

For the error function $S(w)$ given by (1.38) the Hessian can be written as the sum of two terms

$$\begin{aligned} A &= \frac{\partial^2 S(w)}{\partial w^2} \\ &= \beta B + \alpha C \end{aligned} \tag{2.1}$$

with

$$B = \frac{\partial^2 E_D}{\partial w^2} \tag{2.2}$$

and

$$C = \frac{\partial^2 E_w}{\partial w^2} \tag{2.3}$$

where $\beta = \sigma_\nu^{-2}$ is the noise level and α is the regularising constant controlling the weights w of the model. A common choice of E_w is the weight decay regulariser, *i.e.* $E_w = \frac{1}{2}w^T w$, which in the Bayesian formalism, corresponds to having a Gaussian prior distribution on the weights with zero mean and variance α^{-1}. In this case $C = I$, where I is the unit matrix. However, the data part B of the Hessian is not so easy to evaluate as its computation requires forward and backward passes through the network which is computationally demanding especially for large neural networks. The assumption of a diagonal Hessian has been used as a means of avoiding the computational overhead of evaluating and inverting this matrix. Such an approximation is not satisfactory as the Hessian matrix is, in general, strongly non-diagonal and in many applications it is important that all the elements of the Hessian matrix be evaluated accurately. MacKay (1991) found, for example, that the diagonal approximation scheme of Le Cun (1990) was not sufficiently accurate and therefore included the off-diagonal terms. Hassibi et al. (1993, 1994) also found it necessary to include the non-diagonal terms of the Hessian for network pruning in order to switch off the right weights.

In this chapter we will consider both exact, approximate and numerical methods of evaluating the data Hessian B, which we will refer to as the Hessian in short whenever this does not lead to any confusion. We will consider in particular the so called outer product approximation and examine its accuracy as well as its computational efficiency.

2.2 Evaluation of the Hessian Matrix

In this section we review various existing methods of evaluating the Hessian matrix. We start with consideration of the exact methods.

2.2.1 Exact methods

In the recent years exact schemes for evaluating the Hessian matrix, which make use of the efficient back-propagation technique, have been proposed (Bishop 1992; Buntine and Weigend 1993). These methods use no approximations and so they provide accurate ways of computing the Hessian which is necessary in many applications. They can be applied to networks of arbitrary topology and to any differentiable error function. While having the same advantage, the more recent algorithm by Pearlmutter (1994), called the $\mathcal{R}\{.\}$ operator method, also allows the product of the Hessian by a vector to be computed directly without evaluation the Hessian itself. Implementing these exact methods typically require a number of operations which scales like $\mathcal{O}(k^2)$, where k is the number of weights in the network. In this thesis the $\mathcal{R}\{.\}$ method will be used for exact evaluation of the Hessian matrix for multi-layer neural networks.

2.2.2 The outer product approximation method

For the least-square error[1], the data Hessian $\boldsymbol{B}$ can be written as the sum of two matrices

$$\boldsymbol{B} = \boldsymbol{G} + \boldsymbol{H} \tag{2.4}$$

with

$$\boldsymbol{G} = \sum_{i=1}^{N} \boldsymbol{g}_i \boldsymbol{g}_i^T \tag{2.5}$$

and

$$\boldsymbol{H} = \sum_{i=1}^{N} (y(\boldsymbol{x}_i : \boldsymbol{w}) - t_i) \frac{\partial \boldsymbol{g}_i}{\partial \boldsymbol{w}} \tag{2.6}$$

where the vector $\boldsymbol{g}_i \equiv \partial y(\boldsymbol{x}_i; \boldsymbol{w})/\partial \boldsymbol{w}$ is the first derivatives of the output $y(\boldsymbol{x}_i; \boldsymbol{w})$ with respect to the weights $\boldsymbol{w}$ and N is the number of data points in the training set. For linear models $\boldsymbol{H}$ is the zero matrix, as the first derivative $\boldsymbol{g}_i$ does not depend on the weights $\boldsymbol{w}$, and so $\boldsymbol{B} = \boldsymbol{G}$. For neural networks, however, $\boldsymbol{H} \to \boldsymbol{0}$ as $N \to \infty$, provided that the error is at a minimum. This can be seen from the infinite data case where the matrix $\boldsymbol{H}$ can be written as (Bishop 1995a)

$$\boldsymbol{H} = \int (y(\boldsymbol{x}; \boldsymbol{w}) - \langle t|\boldsymbol{x}\rangle) \frac{\partial \boldsymbol{g}(\boldsymbol{x}; \boldsymbol{w})}{\partial \boldsymbol{w}} \, p(\boldsymbol{x}) \, d\boldsymbol{x} \tag{2.7}$$

As mentioned Section 1.2, in the limit $N \to \infty$ the output $y(\boldsymbol{x}; \boldsymbol{w})$ represents the conditional average of the targets $\langle t|\boldsymbol{x}\rangle = \int tp(t|\boldsymbol{x}) \, d\boldsymbol{x}$, and so the quantity $(y(\boldsymbol{x}; \boldsymbol{w}) - \langle t|\boldsymbol{x}\rangle)$ vanishes and $\boldsymbol{H}$ becomes the zero matrix. Thus

$$\boldsymbol{B} = \sum_{i=1}^{N} \boldsymbol{g}_i \boldsymbol{g}_i^T \tag{2.8}$$

In reality, however, the data size is limited but equation (2.8) may still approximately hold (Levenberg 1944; Marquardt 1963). This is known as the outer product (or Levenberg-Marquardt) approximation. One way for this to be true is when the outputs of the network passes through or close to the targets. For noisy regression, however, this means over-fitting which is not desired. Instead we need the network solution to average over noise which might require that the outputs to be appreciably different from the targets in order to avoid fitting the noise. Fortunately, in cases like this $\boldsymbol{G}$ might still approximate the Hessian $\boldsymbol{B}$ well. This can be seen from the fact that, since the outputs $y(\boldsymbol{x}_i; \boldsymbol{w})$ of a trained network approximates the true function $f(\boldsymbol{x})$, the quantity $(y(\boldsymbol{x}_i; \boldsymbol{w}) - t_i)$ is a Gaussian random variable

[1] With no loss of generality we restrict our analysis of the outer product approximation to the case of one output. The results can easily be extended to networks with an arbitrary number of outputs.

with zero mean, which is statistically independent of the derivative $\partial g(x_i; w)/\partial w$. Therefore, for a sufficiently large number data points the right hand side of (2.6) becomes negligible.

One of the advantages of the outer product approximation is the simplicity of its implementation as it involves evaluating the first derivatives only, which requires a number of operations scaling like $\mathcal{O}(k)$ using standard back-propagation. The elements of the Hessian matrix can then be found in $\mathcal{O}(k^2)$ steps using simple multiplications, which is the same as the number of operations required for evaluating the Hessian using exact methods. Beside this, the outer product approximation has the interesting property of ensuring that the Hessian is positive definite. But the question that remains is 'How accurate is the outer product approximation for evaluating various quantities which depend on the Hessian matrix?'. In the next section we will try to answer this question.

Accuracy of the outer product approximation

In many applications we need to compute the trace or determinant of the Hessian. The determinant of the Hessian is needed, for example, to evaluate the evidence for the hyperparameters α and β, and its trace is needed for estimating the effective number of weights of the network. These quantities can all be computed from the knowledge of the eigenvalues of the Hessian matrix. From a practical view point, therefore, a comparison of the eigenvalues of the Hessian obtained by exact and outer product methods can provide a good test of the accuracy of the outer product approximation. Since in practice we deal with the full Hessian matrix $\boldsymbol{A}$ (see equation (2.1)), rather than the data Hessian $\beta \boldsymbol{B}$, we shall compare the eigenvalues of $\boldsymbol{A}$ with those of $\bar{\boldsymbol{A}} = \beta \boldsymbol{G} + \alpha \boldsymbol{I}$, where we have taken $\boldsymbol{C} = \boldsymbol{I}$. From equations (2.1)) and (2.4), we have

$$\begin{aligned} \boldsymbol{A} &= \beta \boldsymbol{G} + \beta \boldsymbol{H} + \alpha \boldsymbol{I} \\ &= \bar{\boldsymbol{A}} + \beta \boldsymbol{H} \end{aligned} \tag{2.9}$$

Here we will show that there is a limit to the accuracy in computing the eigenvalues of the Hessian matrix using the outer product method. Let $\{\lambda(\boldsymbol{A})\}$, $\{\lambda(\bar{\boldsymbol{A}})\}$ and $\{\beta\lambda(\boldsymbol{H})\}$ be the eigenvalues of $\boldsymbol{A}$, $\bar{\boldsymbol{A}}$ and $\beta \boldsymbol{H}$. Since these matrices are real symmetric, they belong to the group of so called *Hermitian* matrices which are characterised by having real eigenvalues which can, therefore, be arranged in increasing order such that $\lambda_{\min} = \lambda_1 < ... \lambda_l \cdots < \lambda_k = \lambda_{\max}$. According to Weyl's theorem (Horn and Johnson 1985) on the variational description of the eigenvalues of Hermitian matrices, for any three matrices $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$, we have

$$\lambda_{\min}(\boldsymbol{c}) \leq \lambda_l(\boldsymbol{a}) - \lambda_l(\boldsymbol{b}) \leq \lambda_{\max}(\boldsymbol{c}) \tag{2.10}$$

provided that $a = b + c$. Since the matrices A, $\bar{A}$ and βH satisfy this relation and that the Hessian A is positive definite[2], we can write

$$\frac{\beta\lambda_{\min}(H)}{\lambda_l(A)} \leq \frac{\lambda_l(A) - \lambda_l(\bar{A})}{\lambda_l(A)} \leq \frac{\beta\lambda_{\max}(H)}{\lambda_l(A)} \tag{2.11}$$

The quantity $(\lambda_l(A) - \lambda_l(\bar{A}))/\lambda_l(A)$ is the relative error in computing the lth eigenvalue of the Hessian matrix using the outer product approximation. The inequality (2.11) shows that this error is at least as large as $(\beta\lambda_{\min}(H))/\lambda_l(A)$ and that the smaller the eigenvalue $\lambda_l(A)$ the larger this error may become.

A demonstration of this discrepancy between the eigenvalues of the exact and outer product Hessians is shown in Figure 2.1. Here a two-layer neural network with 4 hidden units is trained on a toy data consisting of 30 data points with targets generated from $\sin(x)$ with the addition of zero mean Gaussian noise of variance $\sigma_\nu^2 = 0.01$. A single Gaussian prior was imposed on the weights w of the network and its hyperparameter α together with the noise level $\beta = \sigma_\nu^{-2}$ were both estimated from the training data using the evidence framework (formulae (1.59) and (1.56)). At the end of training the estimated value of α and β converged to 0.293 and 69.05, respectively. The use of this weight decay regularization was to ensure that the outer product approximation does not become trivially valid as a result of overfitting the data. Part (a) of the figure shows network error at each training cycle which is performed using the BFGS algorithm (Polak 1971; Luenberger 1984). As the training progresses the difference between the exact and outer product Hessians, defined as $\sum_{l,m=1}^{k}\left((A_{lm} - \bar{A}_{lm})/A_{lm}\right)^2$, decreases until it becomes negligible at the end of training. This is shown in part (b) of the figure. While part (c) shows the difference in the eigenvalues of the Hessians, defined as $\sum_{l=1}^{k}\left(\lambda_l(A) - \lambda_l(\bar{A}))/\lambda_l(A)\right)^2$, during training, which remains significant even at the end of training session. This error is due, largely, to the discrepancy in the small eigenvalues of the exact and outer product Hessians which we expect according to (2.11). This fact is confirmed by the results of part (d) of the figure which shows the logarithmic plot of the exact eigenvalues $\lambda(A)$ against the outer product eigenvalues $\lambda(\bar{A})$ of the Hessian matrix. Note the deviation of the plot of the small eigenvalues from the line with unit slope. So how does this affect the trace and the determinant of the full Hessian A? The error in computing the small eigenvalues of the Hessian using the outer product approximation will have negligible effect on evaluation of the trace $\sum_{l=1}^{k}(\beta\lambda_l + \alpha)$, while it could jeopardise the evaluation of the determinant $\prod_{l=1}^{k}(\beta\lambda_l + \alpha)$ depending on how small the hyperparameter α is. The reason is that the error in evaluating the small eigenvalues can affect the product of eigenvalues more than the sum. This also implies that the trace and the determinant of the inverse Hessian A^{-1} may be significantly different from those of $\bar{A}^{-1}$. A comparison of the

[2] Since βG and αI are positive definite by design so is $\bar{A}$.

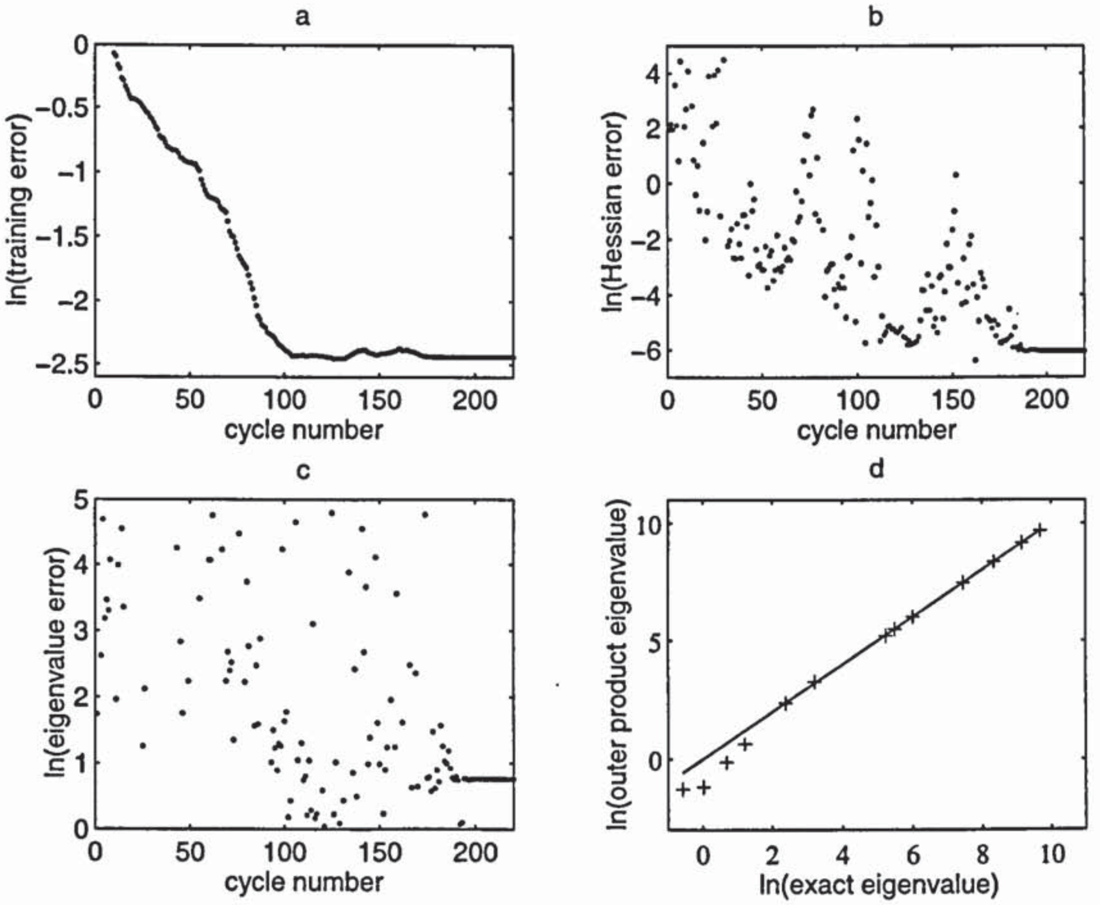

Figure 2.1: (a) Plot of the logarithm of training error against number of training cycles for a two layer network with four hidden units. (b) Plot of the logarithm of the difference between the exact and outer product Hessian, defined as $\sum_{l,m=1}^{k}((A_{lm} - \bar{A}_{lm})/A_{lm})^2$, against training cycle. Note that the outer product approximates the Hessian well once the network is sufficiently trained. (c) Logarithm of the difference, defined as $\sum_{l=1}^{k}((\lambda_l(A) - \lambda_l(\bar{A}))/\lambda_l(A))^2$, between the eigenvalues of the exact and outer product Hessians during training. (d) Plot of the logarithm of the exact and outer product eigenvalues of the Hessian matrix.

determinant and trace of the exact Hessian matrix and its outer product approximation and their inverses is given in Table 2.1.

2.2.3 Finite differences methods

The Hessian matrix can also be evaluated using the numerical method of finite differences. Here we perturb the weight w_{lm} by a small amount $\pm\epsilon$ and then approximate the first derivative $\partial E_i/\partial w_{lm}$ by

$$\frac{\partial E_i}{\partial w_{lm}} \approx \frac{1}{2\epsilon}\Big(E_i(w_{lm}+\epsilon) - E_i(w_{lm}-\epsilon)\Big) + \mathcal{O}(\epsilon^2) \tag{2.12}$$

where $E_i(w) = \frac{1}{2}(y(x_i; w) - t_i)^2$ is the least square error for the ith data point. The central differences formula (2.12) evaluates the first derivatives in a number of $\mathcal{O}(k^2)$ operations with error $\mathcal{O}(\epsilon^2)$. This can be compared with the computational efficiency of the exact methods which scales like $\mathcal{O}(k)$. To

	A	$\bar{A}$	A^{-1}	$\bar{A}^{-1}$
trace	2.240x10^4	2.230x10^4	4.432	10.460
determinant	1.667x10^{23}	2.91x10^{21}	5.9980x10^{-24}	3.437x10^{-23}

Table 2.1: A comparison of trace and determinant of the exact Hessian matrix $A = \beta B + \alpha I$, where $\alpha = 0.293$ and $\beta = 69.05$, and its outer product approximation $\bar{A} = \beta G + \alpha I$ together with their inverses.

evaluate the second derivative $\partial^2 E_i/\partial w_{lm}\partial w_{np}$, we re-apply formula (2.12) to $E(w_{lm} \pm \epsilon)$, with w_{np} begin the weight perturbed. This yields the result (Bishop 1995a)

$$\begin{aligned}\frac{\partial^2 E_i}{\partial w_{lm}\partial w_{np}} \approx\ & \frac{1}{4\epsilon^2}\Big(E(w_{lm}+\epsilon, w_{np}+\epsilon) - E(w_{lm}+\epsilon, w_{np}-\epsilon) \\ & -E(w_{lm}-\epsilon, w_{np}+\epsilon) + E(w_{lm}-\epsilon, w_{np}-\epsilon)\Big) + \mathcal{O}(\epsilon^2) \qquad (2.13)\end{aligned}$$

Formula (2.13) ensures that the elements of the Hessian are evaluated with an accuracy of the order $\mathcal{O}(\epsilon^2)$. This residual error can not be made arbitrarily small as choosing too small a value for ϵ might causes machine round off error which could outweigh the residual error. To evaluate an element of the Hessian four forward passes through the network are required each taking a number of operations scaling like $\mathcal{O}(k)$. So the total number of operations necessary for evaluating the Hessian in this way is $\mathcal{O}(k^3)$. This can be compared with the exact and outer product methods where the number of operations scales like $\mathcal{O}(k^2)$. For large neural networks this difference is significant.

A more computationally efficient way of using central differences is to evaluate the first derivatives by analytical means, *i.e.* backpropagation, and then apply central differences to the first derivatives to obtain the elements of the Hessian matrix. This leads to

$$\frac{\partial^2 E_i}{\partial w_{lm}\partial w_{np}} \approx \frac{1}{2\epsilon}\left\{\frac{\partial E(w_{lm}+\epsilon)}{\partial w_{np}} - \frac{\partial E(w_{lm}-\epsilon)}{\partial w_{np}}\right\} + \mathcal{O}(\epsilon^2) \qquad (2.14)$$

In this way the Hessian matrix is evaluated with a number of operations scaling like $\mathcal{O}(k^2)$. A comparison of the amount of cpu time required for evaluating the Hessian matrix for multi-layer neural networks of different sizes, using exact, outer product and central difference methods is shown in Figure 2.2.

2.3 Summary and conclusions

The exact methods provide satisfactory means for computing the Hessian matrix in terms of accuracy and speed. These methods require a number of operations which scales like $\mathcal{O}(k^2)$. While the outer product approximation is just as efficient as these methods, it produces significant error in computing small eigenvalues of the Hessian. The effect of this error varies depending on what we need to evaluate.

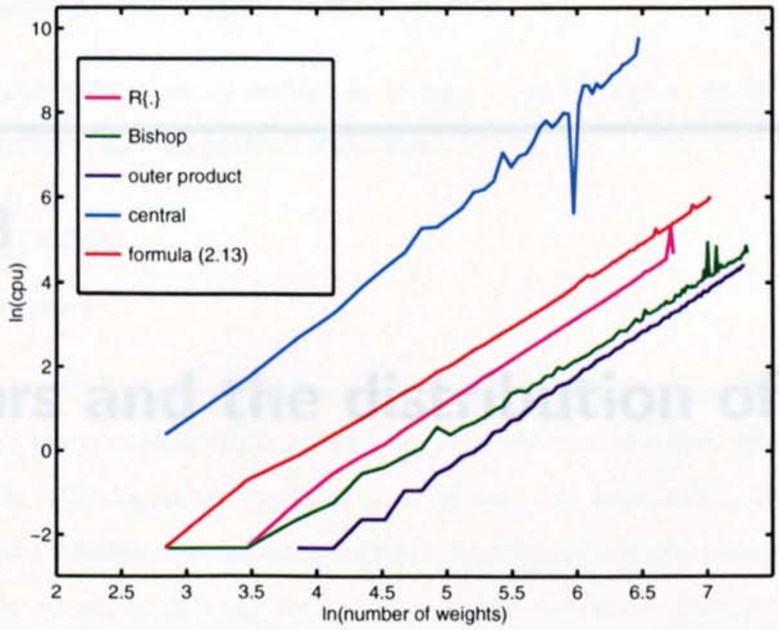

Figure 2.2: Logarithm of the cpu time taken for evaluating the Hessian matrix $\boldsymbol{B}$ for a two layer network, plotted against number of weights of the network. For each network size the Hessian is evaluated using the exact methods of Bishop and $\mathcal{R}\{.\}$, the outer product approximation and the numerical methods of equations (2.13) and (2.14). Every effort is made to make the comparison a fair one: each implementation was written in C++ in a similar style and the code was executed on the same type of machine. The plots corresponding to the exact methods of Bishop and $\mathcal{R}\{.\}$ as well as the numerical method of (2.14) have slope ≈ 2. This can be compared to the slope of the line corresponding to the central differences method which scales like ≈ 3. This difference in scaling of the cpu time is significant for large networks.

For quantities depending on the sum of the eigenvalues the effect of this inaccuracy is negligible, while it can be significant for quantities depending on the product of these eigenvalues. Furthermore, the outer product is not a robust method as its validity depends on the number of data points and how well the network is trained, *i.e.* how well the output $y(\boldsymbol{x};\boldsymbol{w})$ averages over noise. Nevertheless, it has the important property of ensuring that the Hessian is positive definite. It might be the case sometimes that the training procedure yields a bad local minimum of the error. In situations like this the exact Hessian may not be positive definite but has a number of negative eigenvalues which have to be discarded using an arbitrary cut-off procedure. In cases like this the eigenvalues of the outer product Hessian can be used as a means of avoiding this arbitrary cut-off.

As for the numerical methods they are highly inefficient in computing the Hessian as they require a number of operations scaling like $\mathcal{O}(k^3)$. In addition there is a round-off error $\mathcal{O}(\epsilon^2)$ which can not be made arbitrarily small due to machine precision. Nevertheless, they are useful as tools for checking software for evaluating the Hessian using other methods.

In conclusion, the exact methods offer the best available way of evaluating the Hessian in terms of both speed and accuracy of computation. In the rest of this thesis the $\mathcal{R}\{.\}$ method will be used for evaluating the Hessian matrix of neural networks, while for linear models the outer product method will be used since it is exact.

Chapter 3

Error bars and the distribution of input data

3.1 Introduction

We saw in Chapter 1 that, in the Bayesian framework for regression, the uncertainty in predictions $\sigma_t^2(x)$ arises from two different sources. The first source is the intrinsic noise on the target data σ_ν^2, while the second arises from the uncertainty in the model weights $\sigma_w^2(x)$ as a consequence of having a limited number of data in the training set. These distinct sources of uncertainty make additive contributions to the prediction variance so that $\sigma_t^2(x) = \sigma_\nu^2 + \sigma_w^2(x)$. In this chapter we will investigate the behaviour of the contribution $\sigma_w^2(x)$ in relation to both individual data points and the data set as a whole. One key result obtained is that, under certain circumstances, the magnitude of $\sigma_w^2(x)$ exhibits an approximate inverse proportionality to the density of the input data $p(x)$.

Since the analyses of this chapter are primarily made for the so-called generalised linear regression GLR models a brief review of these models will be given first.

3.2 Generalised linear regression models

A GLR model is specified by a set of weights $w = \{w_1, ..., w_k\}^T$, and a set of basis functions $\phi(x) = \{\phi_1(x), ..., \phi_k(x)\}^T$, and has outputs of the form

$$\begin{aligned} y(x; w) &= \sum_{l=1}^{k} w_l \phi_l(x) \\ &= w^T \phi(x) \end{aligned} \tag{3.1}$$

Here the basis $\phi_l(x)$ are fixed non-linear continuous functions of the inputs x, with generally one of them $\phi_1 = 1$, so that the corresponding parameter w_1 plays the role of a bias. Some examples of basis functions used in GLR models are Gaussian, sigmoid, tanh and polynomials. Given a sufficient number of weights and a suitable choice of basis functions, such models can approximate any function, and they also have the advantage of being linear in the adaptive weights w. The principal limitation of these models, however, is the exponential increase in the number of weights as the dimensionality of the input is increased, a form of the *curse of dimensionality* (Bellman 1961; Bishop 1995a).

Depending on the type of the basis functions, a GLR model might also depend on parameters other than w. In the case of Gaussian basis functions, for example, these extra parameters are the locations and width of the Gaussian functions. Similarly, if sigmoid or tanh basis functions are used then the parameters will be the locations and steepness of the basis functions. Here we use a simple approach to fixing these parameters which involves placing the locations of the basis functions on a regular grid defined by the data in input space. Then the width (or steepness) of the basis is set to $\Delta x/n$, where Δx is defined in equation (A.2), n^d is the number of the basis functions and d is the dimensionality of input space. For the case of Gaussian basis functions this will ensure that the basis functions will sufficiently overlap (neither too peaked or too flat) which is necessary for obtaining a smooth regression function. For the cases of sigmoid and tanh basis functions this will ensure that the basis functions do not significantly overlap in the regions of inputs where they have reached saturation. Further details of this procedure for fixing the basis functions are given in Appendix A.1. Other methods of fixing the locations and width of basis functions are also possible and will be considered in Chapter 4.

For models of the form (3.1) the posterior $p(w|D, \beta, \alpha)$, given least-square error, is Gaussian with mean centred at the most probable value $\widetilde{w}$ which can be found from minimising the error function

$$S(w) = \frac{1}{2}\beta \sum_{i=1}^{N} (w^T \phi(x_i) - t_i)^2 + \frac{1}{2} w^T C w \tag{3.2}$$

The choice of $C = \alpha I$, where I is the unit matrix, corresponds to the having the standard weight decay prior. The most probable weights $\widetilde{w}$ is the vector which minimises the error in (3.2). It is given by

$$\tilde{w} = \beta A^{-1} \Phi^T t \tag{3.3}$$

where

$$\Phi = \begin{pmatrix} \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_k(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & \cdots & \phi_k(x_2) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_1(x_N) & \phi_2(x_N) & \cdots & \phi_k(x_N) \end{pmatrix} \tag{3.4}$$

is the Nxk *design matrix*, t is the vector of the targets t_i, and A is the Hessian matrix

$$\begin{aligned} A &= \beta B + C \\ &= \beta \Phi^T \Phi + C \end{aligned} \tag{3.5}$$

At the most probable value $\tilde{w}$, the outputs of a GLR model is given by

$$\begin{aligned} y(x;\tilde{w}) &= \tilde{w}^T \phi(x) \\ &= \beta \phi^T(x) A^{-1} \Phi^T t \\ &= k^T(x) t \end{aligned} \tag{3.6}$$

where we have introduced the effective kernel k (Hastie and Tibshirani 1990)

$$k^T(x) = \beta \phi^T(x) A^{-1} \Phi^T \tag{3.7}$$

Equation (3.6) shows that the outputs of a GLR model can be written as a linear combination of the targets with weighting coefficients determined by the components of the kernel $k(x)$. Hence it is a *linear smoother*. Note that if we omit the prior, *i.e.* $C = 0$, the kernel has the property

$$\sum_{i=1}^{N} k_i(x) = 1 \tag{3.8}$$

where the sum is taken over the training data. To prove this result we note from (3.5) and (3.7) that

$$\begin{aligned} \sum_{i=1}^{N} k_i(x) \phi^T(x_i) &= \phi^T(x) (\Phi^T \Phi)^{-1} \Phi^T \Phi \\ &= \phi^T(x) \end{aligned} \tag{3.9}$$

If one of the basis functions is a bias, say $\phi_1(x) = 1$, then taking the $l = 1$ component of (3.9) we obtain (3.8). Intuitively we would expect the effective kernels to be localised functions of the inputs, giving most weights to the data points close to the vector of input x. Experimental study of the kernels will be considered later in this chapter.

Given formula (3.1) for the outputs, we have $g(x) = \partial y(x;w)/\partial w = \phi(x)$ and the so the predictive variance is given by

$$\begin{aligned}\sigma_t^2(x) &= \sigma_\nu^2 + \sigma_w^2(x) \\ &= \sigma_\nu^2 + \phi^T(x)A^{-1}\phi(x)\end{aligned} \tag{3.10}$$

which is exact for GLR models. Note that the expression for $\sigma_t^2(x)$ is independent of the targets t_i and the weights w. For models with outputs which are non-linear in the weights this will no longer be true since the Hessian A and the derivatives $g(x)$ now depend on t_i and the most probable value of the weights $\widetilde{w}$.

3.3 Analysis in terms of discrete data

In order to understand the relationship between the magnitude of the error bars and the distribution of data in input space, we consider two complementary approaches. In Section 3.4, we discuss a representation in terms of continuous probability density functions. First, however, we consider discrete data points. We shall see that this leads to an upper bound on the magnitude of the error bars.

3.3.1 Contributions from an isolated data point

Here we consider the change in the magnitude of the error bars after a single data point is observed. We assume that the value of the noise variance $\sigma_\nu^2 = \beta^{-1}$ is known a priori. In the absence of data the Hessian is $A = C$ and the prediction variance is given by

$$\begin{aligned}\sigma_t^2(x) &= \sigma_\nu^2 + \sigma_w^2(x) \\ &= \sigma_\nu^2 + \phi^T(x)C^{-1}\phi(x)\end{aligned} \tag{3.11}$$

The second term in (3.11) is the confidence variance $\sigma_w^2(x)$ due to prior uncertainty in the model weights, and is typically much larger than the noise variance σ_ν^2. If we now add a data point at the input point $\bar{x}$, then the Hessian becomes $A = \beta\phi(\bar{x})\phi^T(\bar{x}) + C$. Using the identity

$$(M + vv^T)^{-1} = M^{-1} - \frac{(M^{-1}v)(v^TM^{-1})}{1 + v^TM^{-1}v} \tag{3.12}$$

where v is a column vector, we compute A^{-1} to obtain

$$A^{-1} = C^{-1} + \frac{C^{-1}\phi(\bar{x})\phi^T(\bar{x})C^{-1}}{\sigma_\nu^2 + \phi(\bar{x})^TC^{-1}\phi(\bar{x})} \tag{3.13}$$

Using (3.13) we can now consider the error bars at the point $\bar{x}$. From (3.10) we have

$$\sigma_t^2(\bar{x}) = \left(1 + \frac{\rho}{1+\rho}\right)\sigma_\nu^2 \tag{3.14}$$

where ρ is the ratio of the prior confidence variance to the noise variance, *.i.e.*

$$\rho = \frac{\phi^T(\bar{x})C^{-1}\phi(\bar{x})}{\sigma_\nu^2} \tag{3.15}$$

Since ρ is always positive, the ratio $\rho/(1+\rho)$ lies in the interval [0 1], and so the right hand side of (3.14) is always smaller than 2 indicating that

$$\sigma_t^2(\bar{x}) \leq 2\sigma_\nu^2 \tag{3.16}$$

Formula (3.16) is a key result, which indicates that the prediction variance at the location of the input point $\bar{x}$ is no larger than twice the noise variance and the contribution of $\sigma_w^2(\bar{x})$ to the error bars is always smaller than σ_ν^2 at a data point. This effect is illustrated in Figure 3.1.

It is also possible to consider a total of N data points all located at the input $\bar{x}$. In this case we have

$$\sigma_t^2(\bar{x}) = \left(1 + \frac{\rho}{1+N\rho}\right) \sigma_\nu^2 \tag{3.17}$$

Thus even for a few number of data points located at the input $\bar{x}$ the prediction error bars $\sigma_t^2(\bar{x}) \approx \sigma_\nu^2$.

In the analysis above we have restricted ourselves to error bars measured at the input point where the data was added. This can be extended to error bars measured at arbitrary points of the input space. Using the matrix identity (3.12), in (3.10), we have

$$\sigma_t^2(x) = \sigma_\nu^2 + V_o(x,x) - \frac{V_o^2(x,\bar{x})}{\sigma_\nu^2 + V_o(\bar{x},\bar{x})} \tag{3.18}$$

where we have defined the prior covariance function as $V_o(x_i, x_j) = \phi^T(x_i)C^{-1}\phi(x_j)$, with the property that $V_o(x_i, x_j) = V_o(x_j, x_i)$, provided that C is symmetric. Note that for $x_i = x_j$, the covariance function $V_o(x_i, x_i)$ is just the prior confidence variance $\sigma_w^2(x_i)$ measured at the input point x_i. The first two terms on the right hand side of (3.18) are simply the prior prediction variance $\sigma_t^2(x)$ measured at x. Since the third term in (3.18) is always positive, the effect of the addition of a data point is to reduce the prediction variance from its prior value anywhere in the input space. The scale of this reduction is related to the prior covariance function $V_o(x, \bar{x})$.

The precise form of the covariance function depends on the choice of the prior Hessian C and the choice of the basis functions $\phi(x)$. We note, however, that a simple diagonal prior of the form $C = \alpha I$ is inconsistent since if the type of the basis function is changed then the covariance nature of the prior also changes. One implication of this is the manner in which the reduction in the error bars occurs when a data point is added. To illustrate this, let us consider a GLR model with one basis function only. It is easy to show that if we chose $\phi(x) = x$, as is the case in a linear regression problem, then the reduction in the error bars is given by

$$\Delta\sigma_t^2(x) = -\frac{\alpha^{-2}x^2\bar{x}^2}{\sigma_\nu^2 + \alpha^{-1}\bar{x}^2} \tag{3.19}$$

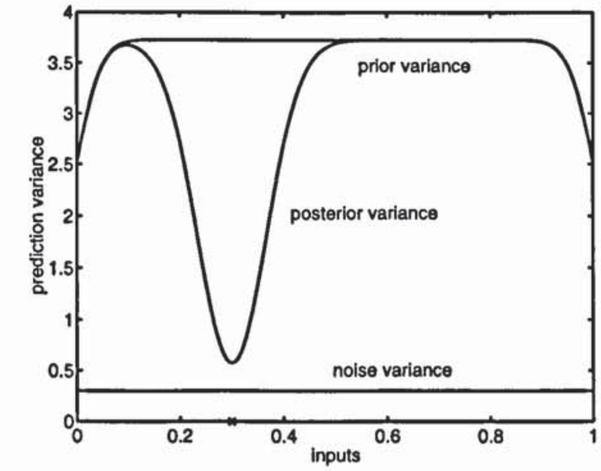

Figure 3.1: A simple example of prediction variance for a one-dimensional input space and a set of 30 equally spaced Gaussian basis functions. The prior variance is typically larger than the noise variance. With the addition of a single data point at $\bar{x} = 0.3$ (indicated by the cross) the variance at the location of the data point itself is reduced to less than twice the noise variance. We see that the reduction in prediction variance is large at and around $\bar{x} = 0.3$. For distant points, however, this reduction is negligible.

Formula (3.19) shows the decrease in the prediction variance depends, given $\bar{x}$, on x. Intuitively, however, we may expect the addition of a data point to be more informative about the regression at the vicinity of $\bar{x}$ resulting in a larger reduction in the magnitude of the error bars at $\bar{x}$ and nearby points (see Figure 3.1). Logically, the weight prior should be chosen so as to specify a particular prior covariance structure over the network outputs. One approach is to specify the prior covariance structure directly (Williams 1997).

3.3.2 An upper bound on the error bars

The analysis of Section 3.3.1 can be extended to the more general case where a GLR model is trained on a data set consisting of N data points. Then a single data point is added and the model is retrained on the $N+1$ data points. As we shall see, this will yield an upper bound on the error bars measured at an arbitrary point x of the input space. Here we will relax the assumption that the noise variance σ_ν^2 is known a priori and instead we allow this quantity to be measured from the original data set according to formula (1.56) (MacKay 1992a). However, we do need to assume that the addition of a data point to the training set does not lead to any significant change in the estimated value of σ_ν^2 after the model is retained on the $N+1$ data points.

The addition of the new data point at $\bar{x}$ will change the Hessian of the model to $\bar{A}$, which can be related to the old Hessian A through

$$\bar{A} = A + \beta\phi(\bar{x})\phi^T(\bar{x}) \tag{3.20}$$

Using the identity (3.12) we write $\bar{A}^{-1}$ in terms of A^{-1}

$$\bar{A}^{-1} = A^{-1} + \frac{A^{-1}\phi(\bar{x})\phi^T(\bar{x})A^{-1}}{\sigma_\nu^2 + \phi^T(\bar{x})A^{-1}\phi(\bar{x})} \tag{3.21}$$

We now need to estimate the new confidence variance $\bar{\sigma}_w^2(x)$ at an arbitrary input x. Using (3.21) in (3.10), we obtain

$$\bar{\sigma}_w^2(x) = \sigma_w^2(x) - \frac{V^2(x,\bar{x})}{\sigma_\nu^2 + V(\bar{x},\bar{x})} \tag{3.22}$$

where we have defined the posterior covariance function as $V(x,\bar{x}) = \phi^T(x)A^{-1}\phi(\bar{x})$, with the property $V(x,\bar{x}) = V(\bar{x},x)$, since A is symmetric. The posterior covariance $V(\bar{x},\bar{x}) = \phi^T(\bar{x})A^{-1}\phi(\bar{x})$ is the magnitude of the confidence variance $\sigma_w^2(\bar{x})$ measured at the input location $\bar{x}$ before the addition of the new data point. Since the Hessian is positive definite so is its inverse, implying that the second term on the right hand side of (3.22), is always positive. This yields the result

$$\bar{\sigma}_w^2(x) \leq \sigma_w^2(x) \tag{3.23}$$

Thus the addition of a data point at an arbitrary point $\bar{x}$ can only lead to a decrease in the magnitude of the error bars anywhere in the input space or leave it unchanged. The reduction in the error bars as a result of the addition of a new data point can be understood in a simple intuitive way – Since the arrival of new data conveys some information about the regression function, the effect of this addition can only reduce the uncertainty in the regression or leave it unchanged depending how relevant (how informative) the new data is.

A further corollary of the result (3.23) is that, if we consider the error bars due to each of a set of N data points individually, then the envelope of those error bars define an upper bound on the error bars of the data set as a whole. This is illustrated in Figure 3.2.

3.3.3 Global Averages

g So far we have dealt with the local magnitude of the error bars measured at an arbitrary point x. Useful insights can also be obtained by considering the global averages defined on the training data set. It can be shown (see Appendix B.1 for the proof) that the global average $\langle\sigma_t^2(x)\rangle$ defined as

$$\langle\sigma_w^2(x)\rangle = \frac{1}{N}\sum_{i=1}^{N}\sigma_w^2(x_i) \tag{3.24}$$

satisfies the relation

$$\langle\sigma_w^2(x)\rangle = \frac{\gamma}{N}\,\sigma_\nu^2 \tag{3.25}$$

where γ is the number of well determined weights (MacKay 1992a; Moody 1992), N is the number of the data points in the training set and σ_ν^2 is the variance of the intrinsic noise on the targets. Note

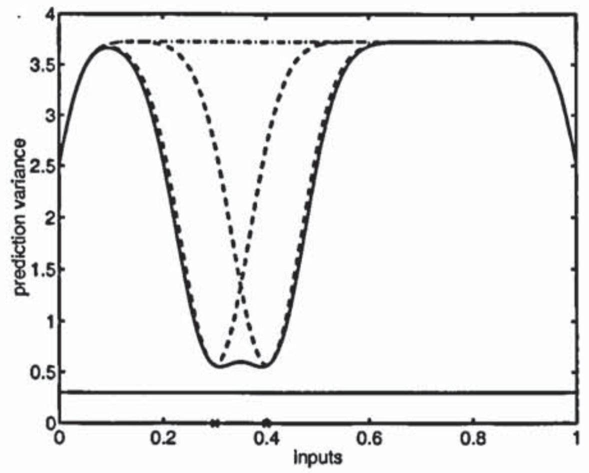

Figure 3.2: As in Figure 3.1 except that two data points are added this time at inputs $x = 0.3$ and $x = 0.4$ as shown by the crosses. The top curve shows the prior prediction variance, the dashed curves show the prediction variance resulting from taking one data point at a time, and the solid curve shows the prediction variance due to the complete data set. The envelope of the dashed curves constitutes an upper bound on the error bars while the noise level (shown by the lower solid line) constitutes a lower bound.

that the size of the training data reduces the error bars by a factor of N^{-1} implying that for $N \gg \gamma$, $\sigma_w^2(x) \ll \sigma_\nu^2$ and so $\sigma_t^2(x) \approx \sigma_\nu^2$ at the location of the input points . We have already seen this inverse dependency (see formula (3.17)) on the number of the training data indicating that for large N $\sigma_w^2(x)$ is negligible. Using (3.25) we can write the average of the prediction error bars as

$$\langle \sigma_t^2(x) \rangle = \left(1 + \frac{\gamma}{N}\right) \sigma_\nu^2 \tag{3.26}$$

The appearance of γ rather than the number of parameters k in the above formula is not a surprise. Since only γ out of k weights determine the regression, the contribution of the weights to the prediction error bars depends on the factor γ rather than k. However, for $\gamma > N$, equation (3.26), seems to suggest that $\langle \sigma_t^2(x) \rangle > 2\sigma_\nu^2$. If so this will bring us to conflict with the findings of section (3.3.1) and the inequality (3.16) in particular. However, this is not the case since we always have $\gamma \leq N$. The reason is if $k > N$, then $\beta \boldsymbol{B}$ will have at most N non-zero eigenvalues. Taking $\boldsymbol{C} = \alpha \boldsymbol{I}$, we have

$$\begin{aligned} \gamma &= k - \alpha \text{Trace}\left(\boldsymbol{A}^{-1}\right) \\ &= \sum_{j=1}^{k} \frac{\lambda_j}{\lambda_j + \alpha} \\ &= \sum_{j=1}^{N} \frac{\lambda_j}{\lambda_j + \alpha} \end{aligned} \tag{3.27}$$

where λ_j is the jth eigenvalue of $\beta \boldsymbol{B}$. Since $\lambda_j/(\lambda_j + \alpha)$ lies in the closed interval [0,1], $\gamma \leq N$, yielding $\langle \sigma_t^2(x) \rangle \leq 2\sigma_\nu^2$.

We have already commented on the appearance of γ in formula (3.26), which shows that more flexible models have larger error bars. While this is true for fixed σ_ν^2 it may or may not hold otherwise, depending on the estimated values of γ and σ_ν^2 which depend one on the another in a non-linear way. In reality the noise variance is not known a priori and has to be estimated from the data according to formula (1.56). For a model with an insufficient flexibility γ is typically small but, due to data misfit, σ_ν^2 is large. On the other hand if we choose models which are too flexible, γ is large while σ_ν^2 is small due to over fitting. The effect of the change in γ on the estimated value of the prediction variance is shown in Figure 3.3.

Average change in the error bars

In Section 3.3.1 we considered the change in the local magnitude of the error bars as a result of the addition of a data point and we showed that the introduction of a new data point to the training set cannot lead to any increase in the value of the error bars anywhere in the input space. We will now consider the reduction in the global average of the error bars. Let $\langle \Delta\sigma_t^2(\boldsymbol{x})\rangle$ be this average defined as

$$\langle \Delta\sigma_t^2(\boldsymbol{x})\rangle = \frac{1}{N}\sum_{i=1}^{N}\left(\bar{\sigma}_w^2(\boldsymbol{x}_i) - \sigma_w^2(\boldsymbol{x}_i)\right) \tag{3.28}$$

where $\bar{\sigma}_w^2(\boldsymbol{x}_i)$ is the prediction variance after the addition of the data point at $\bar{\boldsymbol{x}}$. It can be shown (see Appendix B.2 for the proof) that $\langle \Delta\sigma_t^2(\boldsymbol{x})\rangle$ is subject to the closed bound

$$-\frac{\sigma_\nu^2}{N} \leq \langle \Delta\sigma_t^2(\boldsymbol{x})\rangle \leq 0 \tag{3.29}$$

The upper bound can also be seen as a direct result of (3.23) that the addition of a new data can not increase the error bars. The N^{-1} factor in (3.29) indicates that the larger the size of the data set on which the model is trained the smaller the difference the addition of a new data point will make. Since in the limit $N \to \infty$ the model learns perfectly about the regression, the addition of data will become irrelevant conveying no further information about the regression. For fixed N, however, the maximum reduction will be obtained if the data is maximally informative. Such data points usually belong to regions of input space where the error bars are the largest (MacKay 1992b).

3.4 Analysis in terms of continuous distributions

So far we have studied the error bars based on a consideration of discrete data points in a data set of finite size. In this section we turn to a complementary approach in which we consider continuous probability distribution functions $p(\boldsymbol{x})$ of data in the input space. It has been widely observed that the error bars are small in the regions of input space where there is a lot of data, and relatively large in regions of little data. This effect is illustrated in Figure 3.4. Such empirical observations led Bishop

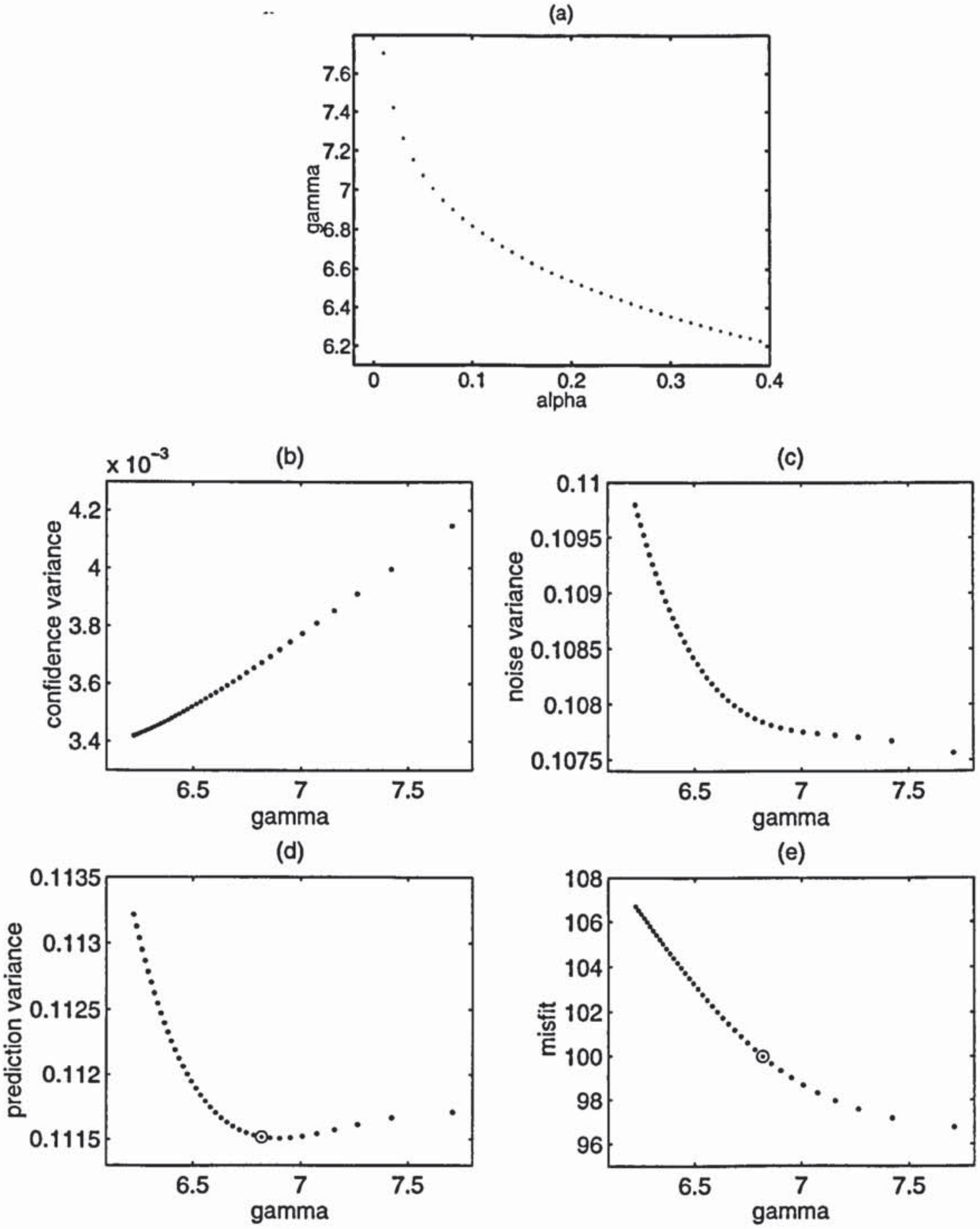

Figure 3.3: An illustration of the dependency of the average value of the prediction variance $\langle \sigma_t^2(x) \rangle$ on the number of well determined weights γ. Here a model with 49 Gaussian basis functions (with specifications given in Table A.1) is used to fit a data set of size $N = 200$ with targets generated from $\sin(x)$ plus the addition of Gaussian noise with zero mean and variance 0.1. The flexibility of the model, *i.e.* value of γ, is controlled through the hyperparameter α. This is shown in part (a) of the figure. Part (b) shows that as the flexibility of the model is decreased $\langle \sigma_w^2(x) \rangle$ becomes smaller. However, the decrease in the value of γ is accompanied by an increase in the estimated value of σ_ν^2, due to data misfit as shown in part (c). The overall affect on the average value of the prediction variance $\langle \sigma_t^2(x) \rangle$ is shown in part (d). Part (e) of the figure shows the error function $S = \beta E_D + \alpha E_w$ (see equation (1.21)), also known as total misfit, against the number of effective parameters of the model. At their most probable values, α and β satisfy $\widetilde{\alpha} = \gamma/2E_w$ (equation (1.59)) and $\widetilde{\beta} = (N-\gamma)/2E_D$ (equation (1.56)). This implies that the optimum value of misfit is given by $S = N/2$ (MacKay 1991). It is interesting to note that when the misfit reaches this optimum $N/2 = 100$ (indicated by the circle in (e)), the average of the prediction variance is close to its minimum value (indicated by the circle in (d)).

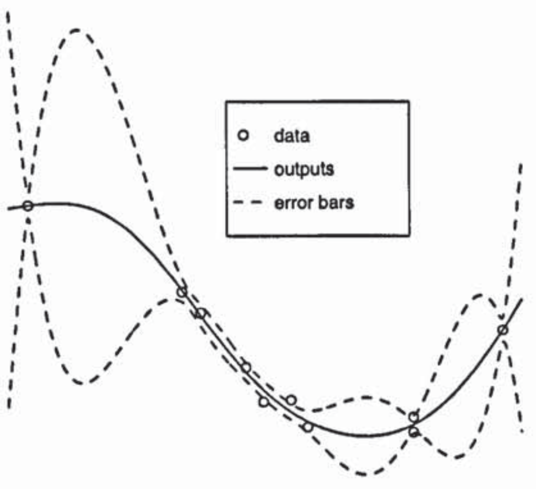

Figure 3.4: An example of confidence error bars for a simple one dimensional toy problem with 10 data points. The result of fitting a GLR model with 20 Gaussian basis functions is shown by the solid curve. $\pm\sigma_w(x)$ (dashed curves) were evaluated from $(g^T(x)A^{-1}g(x))^{1/2}$. It is notable that the error bars grow larger in the regions of input space away from the training data.

(1994b) to conjecture a relation of the form $\sigma_w^2(x) \propto p^{-1}(x)$. In this section we will attempt to show analytically that, under certain circumstances, there is indeed an approximate inverse proportionality relation between the magnitude of $\sigma_w^2(x)$ and the density of the input data $p(x)$. To start with we shall first consider a toy model with disjoint basis functions as this will provide useful insight into the behaviour of error bars and its relation to the input data density.

3.4.1 A toy model

Here we consider the variance $\sigma_t^2(x)$ for a simple GLR model in which the basis functions are disjoint. For simplicity of expression we assume that the basis functions are arbitrary constants[1] within the domain of support, so that for an input x_i, the response of the basis function is $\phi_j(x_i) = h_j I_j(x_i)$, where h_j is the hight of $\phi_j(x_i)$ and $I_j(x_i)$ is an indicator such that $I_j(x_i) = 1$ if x_i lies within the domain of the jth basis function, and $I_j(x_i) = 0$ otherwise. A schematic of this model is given in Figure 3.5. If we choose the prior Hessian to be diagonal, $\boldsymbol{C} = \alpha \boldsymbol{I}$, then the Hessian $\boldsymbol{A}$ also becomes diagonal with elements given by

$$A_{jl} = (\beta n_j h_j^2 + \alpha)\delta_{jl} \tag{3.30}$$

where n_j is the number of the data points falling within the domain of the jth basis function, and δ_{jl} is the Kronecker delta. The prediction variance for this model is then given by

$$\sigma_t^2(x) = \sigma_\nu^2 + \frac{\sigma_\nu^2 h_j^2}{\sigma_\nu^2 \alpha + n_j h_j^2} \tag{3.31}$$

Again we see that the effect of adding data points is to reduce the magnitude of the error bars. If the number of the data points n_j in the domain of the jth basis function is large, the contribution from

[1] This analysis is also applicable to arbitrary non-constant basis functions so long as they remain disjoint.

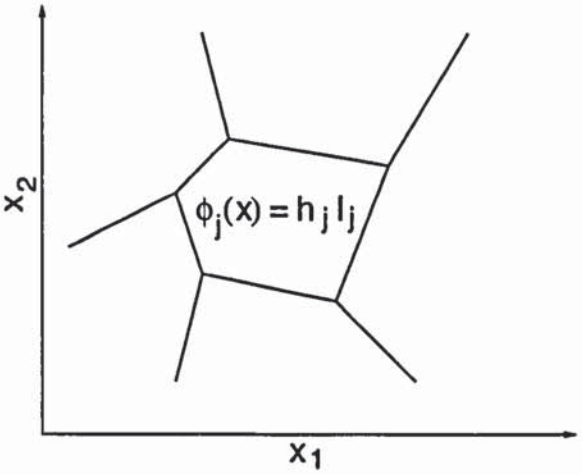

Figure 3.5: A schematic illustration of a toy model consisting of disjoint basis functions in a 2-dimensional input space. The input space is divided into disjoint regions and one basis functions is defined for each region such that each basis function $\phi_j(\boldsymbol{x})$ is zero except within its own region.

the prior α can be neglected. Then

$$\sigma_t^2(\boldsymbol{x}) = \sigma_\nu^2 + \frac{\sigma_\nu^2}{n_j} \tag{3.32}$$

$$= \sigma_\nu^2 + \frac{\sigma_\nu^2}{NV_j\widehat{p}(\boldsymbol{x})} \tag{3.33}$$

where N is the total number of training data, V_j is the volume of input space within the domain of the jth basis function and $\widehat{p}(\boldsymbol{x})$ is the normalised histogram estimate of the density inside V_j. Note the $1/n_j$ dependence of the variance $\sigma_w^2(\boldsymbol{x})$ as implied by (3.32), which shows that the reduction in $\sigma_w^2(\boldsymbol{x})$ is a $1/n_j$ effect. This can be understood in terms of the standard $1/n_j$ dependence of the variance of the mean of n_j independent identically distributed variables with a common variance σ_ν^2. Equation (3.33) shows that the $\sigma_w^2(\boldsymbol{x})$ depends locally on the inverse of the density $\widehat{p}(\boldsymbol{x})$ and globally on N^{-1}. As we shall see later, such dependencies will also occur in the case of more realistic GLR models. It should also be noted from (3.32) that even when only a few data points fall within the domain of the jth basis function the variance σ_ν^2 of the intrinsic noise on the targets dominates (3.33).

3.4.2 The effective kernel

In Section 3.2 we expressed the output of a GLR model as a linear combination of the targets with weighting coefficients defined by the effective kernel in its discrete from. It is also possible to obtain a similar result for the continuous case of $N \to \infty$. Here we consider representations in terms of integrals over continuous input probability distribution functions. To this end we make use of the approximation

$$\frac{1}{N}\sum_{i=1}^{N} Q(\boldsymbol{x}_i) \approx \int Q(\boldsymbol{x})p(\boldsymbol{x})\, d\boldsymbol{x} \tag{3.34}$$

where $Q(\boldsymbol{x})$ is an arbitrary function of the inputs $\boldsymbol{x}$, and the points $\boldsymbol{x}_i$ are independently selected from the density $p(\boldsymbol{x})$. Our goal is to express the outputs $y(\boldsymbol{x};\boldsymbol{w})$ and the variance $\sigma_t^2(\boldsymbol{x})$ in terms of the effective kernels. As we shall see, the latter will lead us to an approximate inverse relationship between $\sigma_w^2(\boldsymbol{x})$ and the density of the input data which we have already mentioned. In the limit $N \to \infty$, the data part of the Hessian $\boldsymbol{B}$ overwhelms the prior part $\boldsymbol{C}$, and the Hessian can be written as

$$\boldsymbol{A} \approx \beta N \int \boldsymbol{\phi}(\boldsymbol{x})\boldsymbol{\phi}^T(\boldsymbol{x})p(\boldsymbol{x})\,d\boldsymbol{x} \tag{3.35}$$

We can now write the output in (3.6) in its continuous form

$$\begin{aligned} y(\boldsymbol{x};\widetilde{\boldsymbol{w}}) &\approx N\iint \beta\boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\boldsymbol{\phi}(\boldsymbol{z})tp(t|\boldsymbol{z})p(\boldsymbol{z})\,dt\,d\boldsymbol{z} \\ &= \int K(\boldsymbol{z};\boldsymbol{x})h(\boldsymbol{z})\,d\boldsymbol{z} \end{aligned} \tag{3.36}$$

where the function $h(\boldsymbol{z})$ and the effective kernel $K(\boldsymbol{z};\boldsymbol{x})$ are defined as

$$h(\boldsymbol{z}) = \int tp(t|\boldsymbol{z})\,dt \tag{3.37}$$

$$K(\boldsymbol{z};\boldsymbol{x}) = \beta N\boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\boldsymbol{\phi}(\boldsymbol{z})p(\boldsymbol{z}) \tag{3.38}$$

where $K(\boldsymbol{z};\boldsymbol{x})$ has the property

$$\int K(\boldsymbol{z};\boldsymbol{x})\,d\boldsymbol{z} = 1 \tag{3.39}$$

To prove result we note from (3.35) and (3.38) that

$$\begin{aligned} \int K(\boldsymbol{z};\boldsymbol{x})\boldsymbol{\phi}^T(\boldsymbol{z})\,d\boldsymbol{z} &= \boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\Big[\beta N\int \boldsymbol{\phi}(\boldsymbol{z})\boldsymbol{\phi}^T(\boldsymbol{z})p(\boldsymbol{z})\,d\boldsymbol{z}\Big] \\ &= \boldsymbol{\phi}^T(\boldsymbol{x}) \end{aligned} \tag{3.40}$$

Taking the bias component of $\boldsymbol{\phi}(\boldsymbol{x})$, *i.e.* $\phi_1(\boldsymbol{x}) = 1$, in (3.40) we obtain (3.39). We now express the prediction variance in terms of the effective kernel. To do this we make use of the result (3.39)

$$\int \frac{K^2(\boldsymbol{z};\boldsymbol{x})}{p(\boldsymbol{z})}\,d\boldsymbol{z} = N\beta\boldsymbol{\phi}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\boldsymbol{\phi}(\boldsymbol{x}) \tag{3.41}$$

which is easily verified by substituting (3.38) into (3.41) and making use of (3.35). Using the general form (3.10), we obtain the following result

$$\sigma_t^2(\boldsymbol{x}) \approx \sigma_\nu^2 + \frac{\sigma_\nu^2}{N}\int \frac{K^2(\boldsymbol{z};\boldsymbol{x})}{p(\boldsymbol{z})}\,d\boldsymbol{z} \tag{3.42}$$

As functions of the inputs $\boldsymbol{x}$ we can expect the kernels to be localised around the input point $\boldsymbol{z} = \boldsymbol{x}$. If we assume that the kernels are indeed localised in this way, and if the density $p(\boldsymbol{x})$ does not change appreciably in the vicinity of $\boldsymbol{x}$, then we can collapse the integral in (3.42) to obtain

$$\begin{aligned} \sigma_t^2(\boldsymbol{x}) &= \sigma_\nu^2 + \sigma_w^2(\boldsymbol{x}) \\ &= \sigma_\nu^2 + \frac{\sigma_\nu^2}{NV(\boldsymbol{x})p(\boldsymbol{x})} \end{aligned} \tag{3.43}$$

where we have defined

$$\frac{1}{V(x)} \equiv \int K^2(z;x)\,dz \tag{3.44}$$

We can interpret $V(x)$ as the volume in input space of the effective kernel. For example, if $K(z;x)$ is approximated by a d-dimensional spherical Gaussian centred at x with standard deviation σ, then $V(x) = \pi^{d/2}(2\sigma)^d$. If $V(x)$ is a reasonably slowly varying function of x then we see from (3.43) that $\sigma_w^2(x)$ exhibits an approximate inverse proportionality relation with the density $p(x)$ of the input data

$$\sigma_w^2(x) \propto p^{-1}(x) \tag{3.45}$$

A relation of this kind was first conjectured by Bishop (1994b). In the next section we will make a numerical study of this relationship.

3.4.3 Experimental results

In this section we explore experimentally the relationship between the confidence variance $\sigma_w^2(x)$ and the density of the input data distribution $p(x)$. Again we shall omit the prior part C of the Hessian matrix since we are primarily interested in the high data density regions. We start will a numerical study of the effective kernels. The specifications of the basis functions of some of the GLR models used in the experiments of this section are given in Appendix A.

Numerical results show that in the regions of high input data density the effective kernels are localised functions of the inputs and that further away from these regions they spread out. Accordingly, we can expect that the confidence variance $\sigma_w^2(x)$ to have an inverse proportionality relation with the density of the input data $p(x)$ (provided that $1/V(x)$ is constant) in the regions of high input data density, while such a relation will not hold in the regions of low input data density. Some examples of the effective kernels for GLR models with different types of basis function are shown in Figures 3.6 and 3.7. Experimental study of the kernels also show that as the number of weights k increases the effective kernels become more tightly peaked. This is illustrated in Figure 3.8.

Numerical results also show that in the regions of high input data density $1/V(x)$ is roughly constant but varies on a much larger scale outside those regions. Some example of $1/V(x)$ for GLR models with different types of basis functions is shown in Figure 3.9. The experiments we have performed so far have provided some evidence that in the regions of high input data density the function $1/V(x)$ is roughly constant and that the effective kernels are localised function of the inputs. These findings indicate that the inverse proportionality formula (3.45) holds in the the regions of high input data density. As a demonstration, Figure 3.10 displays the plot of the inverse of confidence variance $\sigma_w^2(x)$ for GLR models with a range of basis functions along with the density of the input

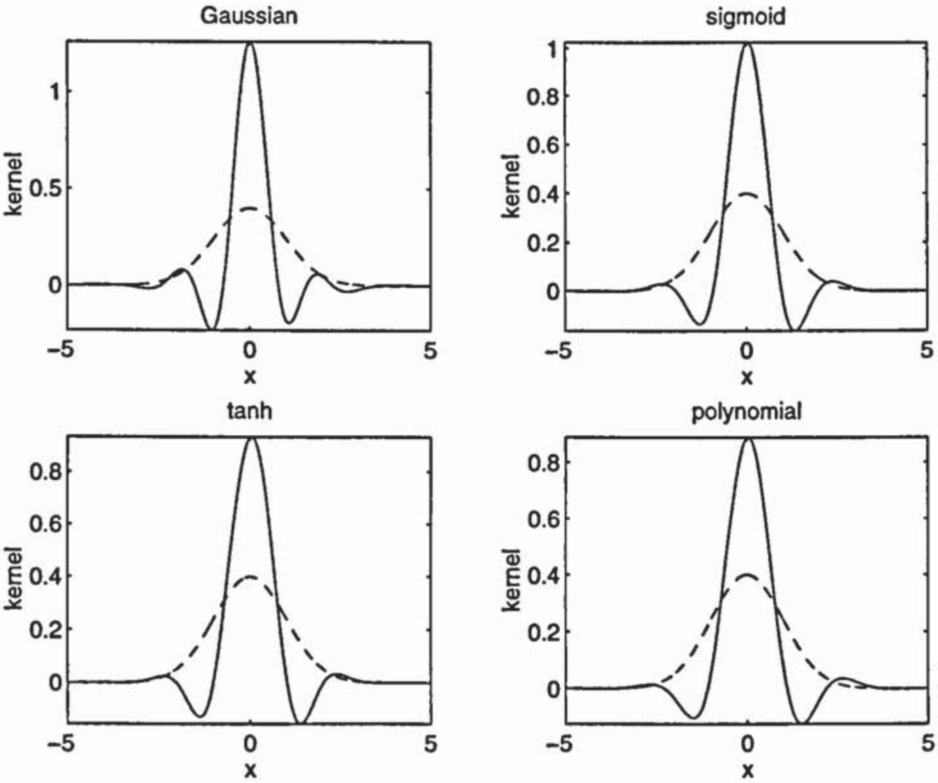

Figure 3.6: Some examples of the effective kernels (solid curves) for GLR models with Gaussian, sigmoid, tanh and polynomial basis functions. Each model consisted of 7 basis functions and a bias (8 weights in total). The effective kernels are centred at the input $x = 0$ which corresponds to the regions of high input data density $p(x)$ (dashed curves). We note that the effective kernels are peaked functions around their centres at $x = 0$.

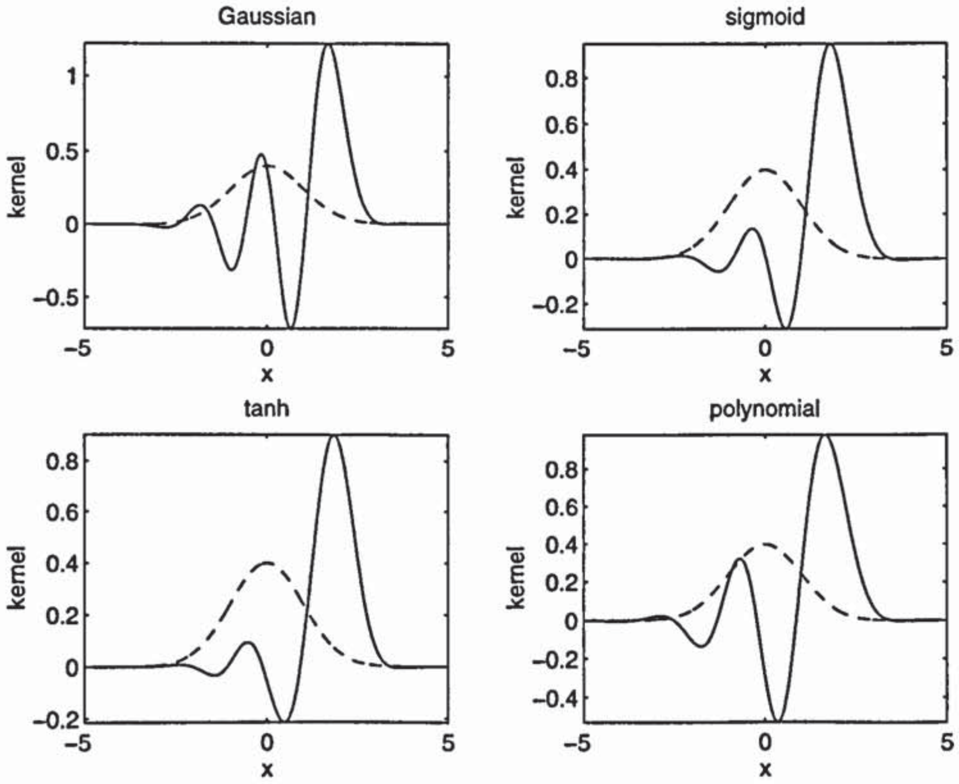

Figure 3.7: As in Figure 3.6 except that the effective kernels (solid curves) are centred at $x = 2$ which lies in the regions of low input data density $p(x)$ (dashed curves). We note that the kernels are no longer well localised functions of the inputs.

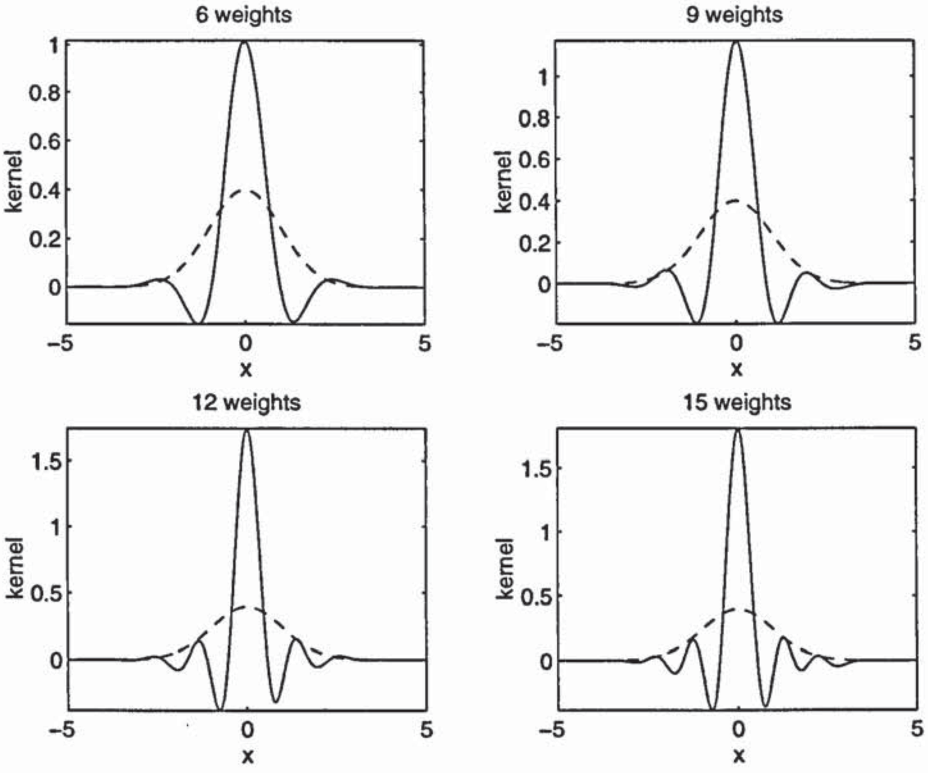

Figure 3.8: Some examples of the effective kernels (solid curves) for GLR models with different number of Gaussian basis functions. The effective kernels are centred at $x = 0$ which corresponds to high input data density (dashed curves). We see that the width of the kernels become narrower as the number of weights increases.

data $p(x)$. The figures show that the inverse variance has the same general shape as the density function despite the differences in the type of the basis functions of the GLR models. In order to make the comparison of the density and the inverse variance clearer, we show in Figure 3.11 a log-log plot of the density versus confidence variance for different types of GLR models. We see that at high regions of input data density there is an approximate inverse relation between the confidence variance and the density and that this relation starts to break down in regions of low density.

The results discussed so far have been based on a finite data set drawn from a density function, in which the Hessian is evaluated numerically as a finite sum. For the particular choice of Gaussian basis functions, however, it is possible to evaluate the Hessian matrix analytically using the continuous density representation of Section 3.4.2. In this case, expression (3.35) for the Hessian becomes the convolution of Gaussian functions, provided that the density $p(x)$ is Gaussian too, which is easily evaluated. The average slope of $\ln \sigma_w^2(x)$ versus $\ln p(x)$ curve is then calculated by taking 10 samples of size 5000 each from the true distribution $p(x)$. Linear regression is used to determine the slope, taking into account only points for which $p(x) > 0.1p_{max}(x)$. The reason for this arbitrary cut-off is to reduce the influence of those data points which belong to low regions of input data density on the measurement of the slope. This is done for two kinds of reasons. First, we are interested in the

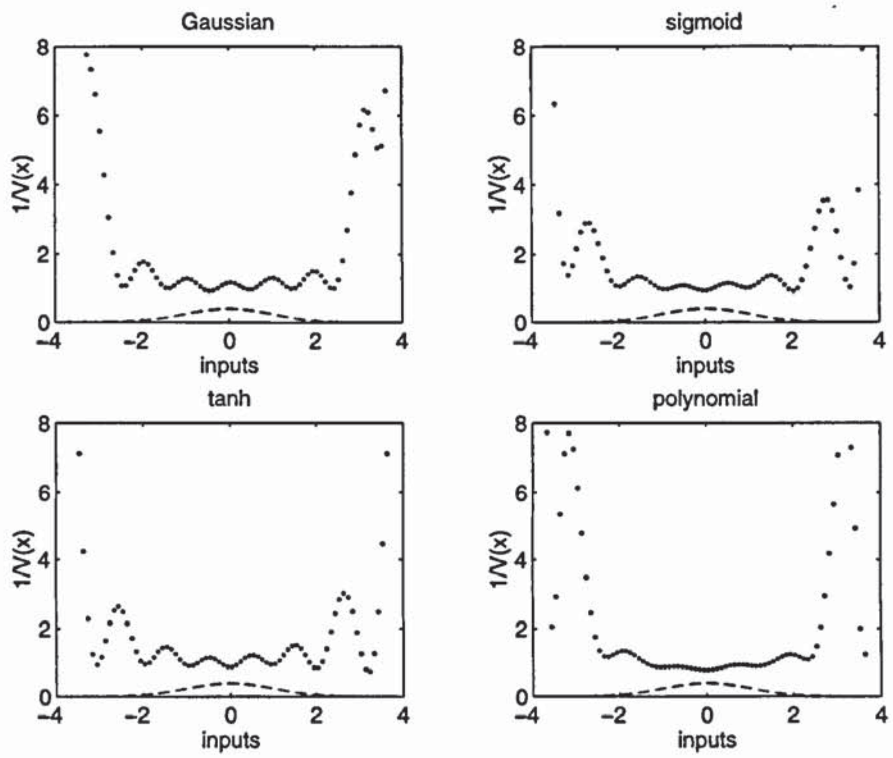

Figure 3.9: Plots of $1/V(x) = \int K^2(z;x)\,dz$ (doted curves) against the inputs for a range of GLR models with Gaussian, sigmoid, tanh and polynomial basis functions. Each model consisted of 7 basis functions and a bias, giving 8 weights in total. The function $1/V(x)$ was evaluated from the numerical integration of formula (3.44) using *Simpson's rule* (Press, Teukolsky, Vetterling, and Flannery 1992). Note that $1/V(x)$ is roughly constant in the regions of high input data density $p(x)$ (dashed curves), while varying on a much larger scale in the regions of lower density.

regions of high input data density, and second the inclusion of these isolated data points make the measurements of the slope noisy. This step is then repeated for various number of basis functions for data sets of input dimensions 1 and 2, and the results are shown in Figure 3.12. We note that as the number of basis functions increases the slope gets closer to -1.

3.5 Multi-layer neural networks

Although the theoretical analysis of this chapter has been performed for GLR models, many of the results apply also to non-linear multi-layer neural networks if we make the Gaussian approximation to the posterior weight distribution and linearise the outputs in the vicinity of the most probable weights $\tilde{w}$. The inequality (3.23) we derived in Section 3.3.2 is also applicable to neural networks provided we make the additional assumption that the addition of a new data point to the training set will not lead to significant changes in the value of the weights. This assumption is used by Cohn (1994, 1995) for deriving a formula similar to (3.22) for use in *optimum experimental design* OED, also known as *active learning*. Given the same approximations, the inequality (3.29) of Section 3.3.3 is

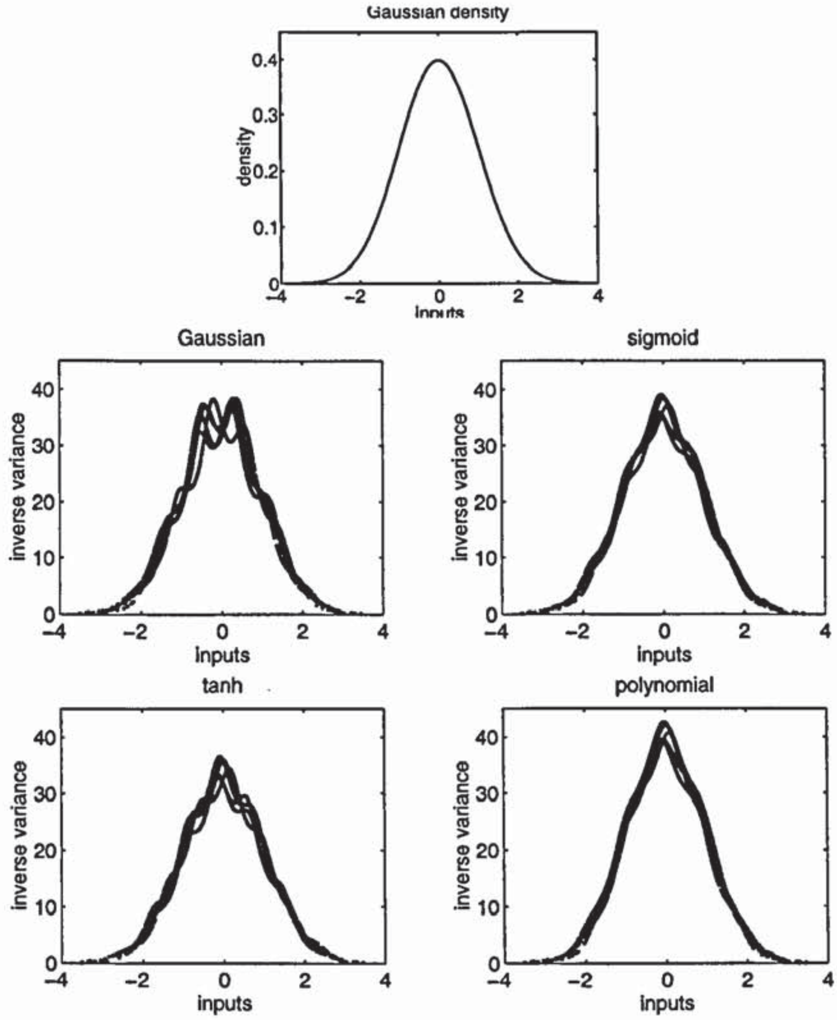

Figure 3.10: Plots of $1/\sigma_w^2(x)$ for GLR models with different types of basis functions against the inputs for 5 data sets of size 1000 each with inputs selected from a Gaussian density function $p(x)$ (top figure). Despite the difference between the types of the basis functions we see that the plots of $1/\sigma_w^2(x)$ have the same general shape as the density $p(x)$.

also applicable to neural networks. Furthermore, if the outputs of the network have linear activation functions then, for the least-square error, it is effectively a GLR model with adaptive basis functions. It is therefore a linear smoother. Therefore, we can expect the inverse relation (3.45) between the confidence variance and the density of the input data to approximately hold within the limitation of the approximations made to obtain this formula which were discussed in the previous section. As a numerical demonstration, Figure 3.13 shows the log-log plot of density versus the confidence variance for a two-layer network with tanh hidden activation functions and linear output units. Again we see that in the regions of high input data density there is an inverse relation between the confidence variance and the density of the input data, while this relation breaks down in the regions of low density.

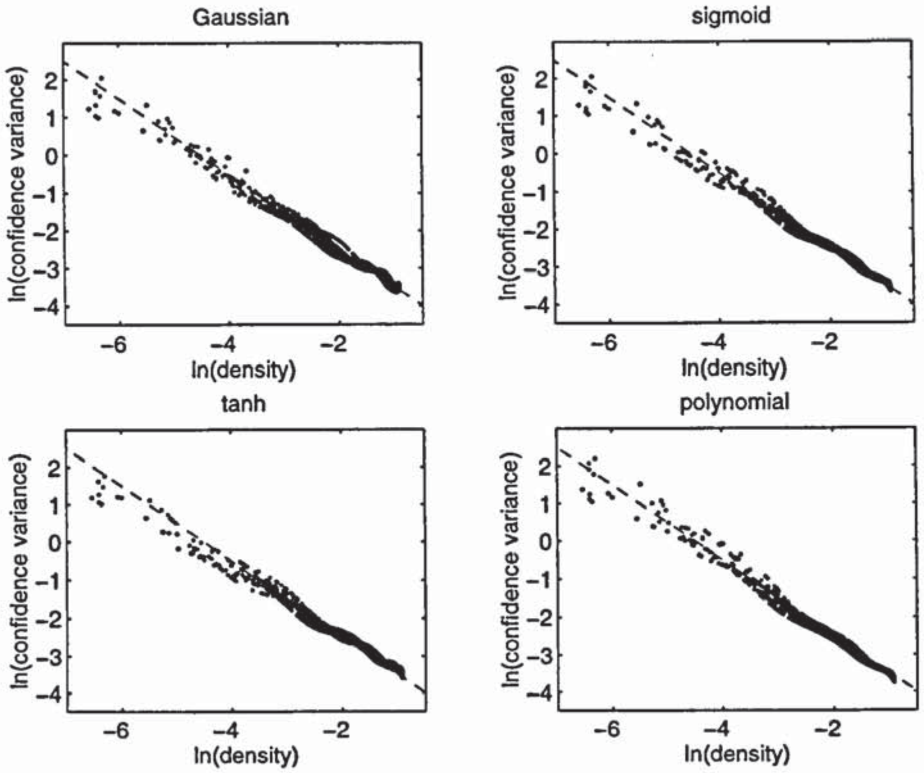

Figure 3.11: Plots of ln $\sigma_w^2(x)$ versus ln $p(x)$ for GLR models with different types of basis functions. The inputs are the same as in Figure 3.10. Each point corresponds to a data point in the training set. Note that in the regions of high input data density the points lie close to the line with slope -1, indicating an approximate inverse proportionality relationship between $\sigma_w^2(x)$ and the density of input data $p(x)$.

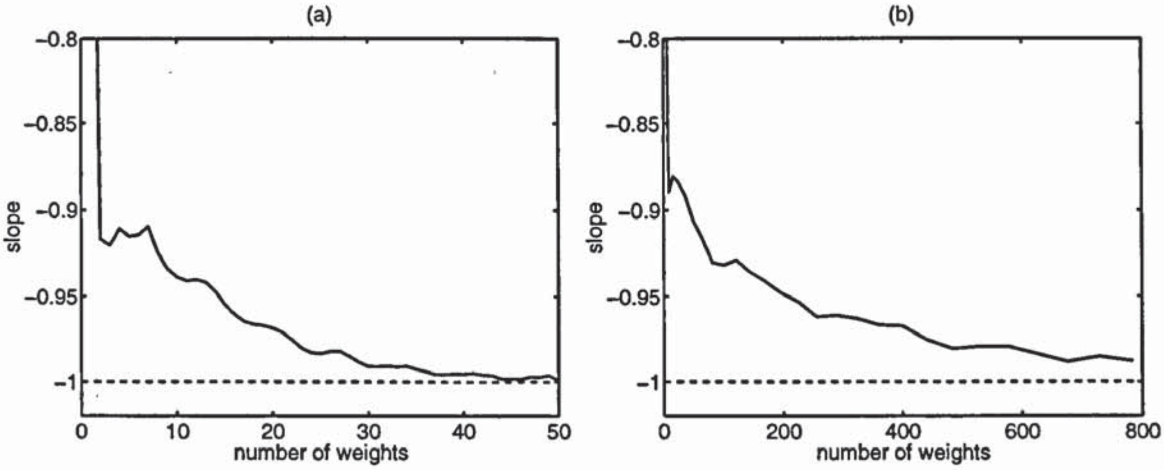

Figure 3.12: (a) Plot of the slope of $\ln \sigma_w^2(x)$ versus $\ln p(x)$ curve in the regions of input space where $p(x) \geq 0.1 p_{max}(x)$, versus number of weights for a one-dimensional input data set. (b) Same as part (a) but showing the results for a two-dimensional input data set. In this case the basis functions are arranged on a regular two-dimensional grid in input space.

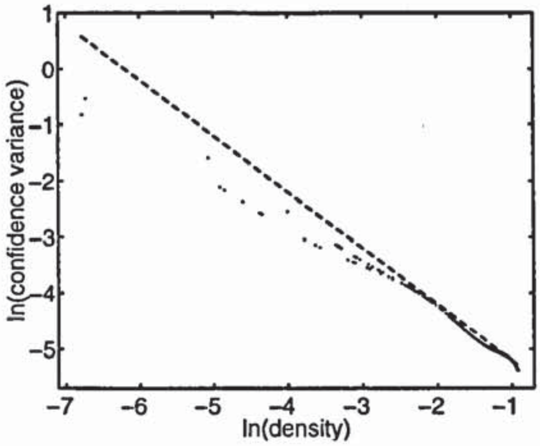

Figure 3.13: Plot of $\ln \sigma_w^2(x)$ versus $\ln p(x)$ for a two-layer neural network with 2 hidden units (7 weights) having tanh activation functions and linear outputs. The network was trained on a data set of size $N = 100$ with inputs chosen from the density function of Figure 3.10 and targets generated from the sin function plus the addition of zero mean Gaussian noise. Each point corresponds to a data point in the training set. Note that in the high density regions the points lie close to the line of slope -1.

3.6 Relation to optimal experimental design

In previous sections we have considered the effect of the addition of a single data point to the training set and its effect on both local and average values of the error bars. The conclusion obtained has been that the addition of a new datum can only reduce the magnitude of the error bars or leave it unchanged anywhere in the input space. Furthermore, we obtained a bound on the magnitude of the reduction in the value of the prediction variance $\sigma_t^2(x)$ as the result of the addition of a data point to the training set. However, the question which raises itself is how does the reduction in the error bars depend on the location of the input where the new data is introduced. Ideally we would like to choose the data at an input point which conveys maximum information about the regression. Such data points are said to be *maximally informative* leading to improvement, at least in principle, in the generalisation error. Selecting data in this manner is the subject of optimum experimental design.

Using an entropy based technique MacKay (1992b) found that data points located in regions of input space where the variance $\sigma_w^2(x)$ is high are maximally informative. However, using such criteria alone for active data selection requires searching outside the input space where data did not occur, which is computationally infeasible. Fortunately, we can still use this approach coupled with the constraint of limiting the search for maximally informative data in a certain region of input space. This search can be conducted using random sampling in the input space. For high dimensional input spaces one can instead use gradient information with a kind of constraint that limits the search for maximally informative data in a limited region of input space (Cohn 1994; Cohn et al. 1995). This is a reasonable approach since in regression problems we are primarily interested in a limited region of input space anyway, namely the region (or regions) where the training data occurred. But can we not apply formula (3.45) to conduct this search for maximally informative data? The answer is no. The reason is that the inverse relationship between the variance $\sigma_w^2(x)$ and the density $p(x)$ is valid only in the regions of high density where the error bars are small while active data gathering based on the magnitude of the error bars requires us to search outside such regions, in the domain of which the error bars are determined mainly the prior, *i.e.* $\sigma_w^2(x) \approx \phi^T(x)C^{-1}\phi(x)$ which depends on the properties of the model rather than the input data density.

3.7 Summary and conclusions

In this chapter we have studied the behaviour of the error bars both locally and globally. For the case of a single isolated data point we have shown that the error bar is pulled down close to σ_ν, and that the length scale over which this effect occurs is characterised by the prior covariance function. We have also provided theoretical and empirical evidence that, in regions of high input data density, the variance $\sigma_w^2(x)$ exhibits an approximate inverse proportionality relationship with the density of input data $p(x)$. Also we have noted that this contribution, in the high input density regions, is insignificant compared to the contribution arising from the variance of intrinsic noise on the targets. This highlights the significance of accurate evaluation of the variance of noise on the targets for obtaining reliable estimate of the error bars.

Chapter 4

Inferring an input dependent noise variance

4.1 Introduction

In Chapter 3 we studied the properties of the Bayesian error bars in relation with the distribution of data in input space. One key result obtained was that the prediction variance $\sigma_t^2(x)$ is no larger than than twice the noise variance at the location of the data points. This brought us about to the conclusion that accurate measurement of the noise variance is essential in making reliable estimate of prediction error bars. This conclusion is also supported by the fact that, since $\sigma_w^2(x)$ depends on the noise variance $\sigma_\nu^2(x)$, mis-estimating $\sigma_\nu^2(x)$ will lead to mis-estimating $\sigma_w^2(x)$ too.

Typically it is assumed that the noise variance is independent of the inputs. This assumption is particularly restrictive as in many applications it will be more realistic to allow the noise variance itself to vary as a function of the inputs. To see how, let us consider the situation in which there is a lot of data in one region of input space and few isolated data points outside that region. If the noise variance is modelled as a constant, then its estimate will be dominated by the data points in the regions of high density. However, as we have seen, the error bars will be pulled down to less than twice the noise variance in the location of the isolated data points and their neighbourhoods. The model is therefore highly confident about the regression in these regions even though there are few number of data points. If, however, we relax the assumption of a constant noise variance, then in the

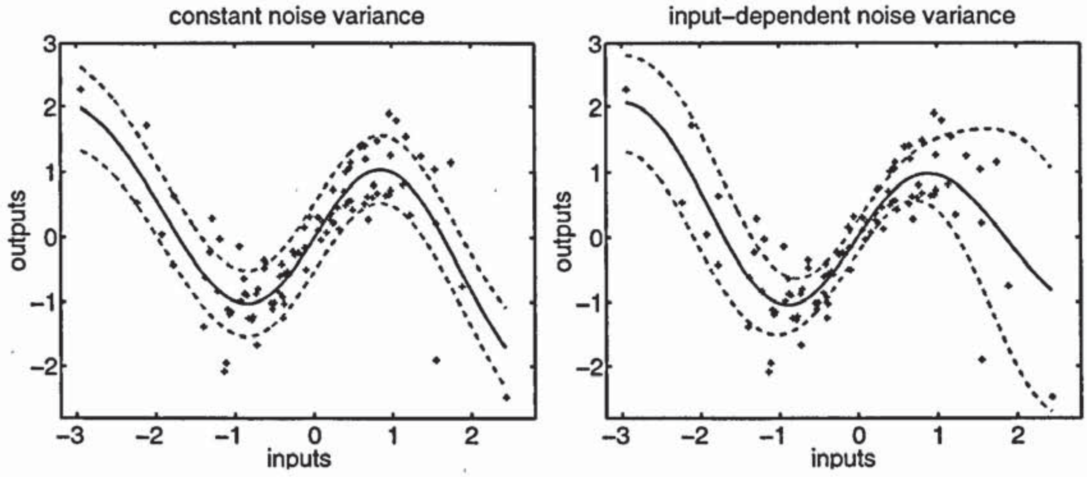

Figure 4.1: Bayesian error bars for a GLR model consisting of 5 Gaussian basis functions and a bias (see Table A.4 for the specifications of the basis functions). The data set consisted of 100 data points with targets generated from a *sin* function plus the addition of Gaussian noise with true variance $s^2(x) = 0.05 + 0.2x^2$. In both parts of the figure the solid curve represents the model outputs and the dashed curves represent the prediction error bars $\sigma_t(x) = \pm(\sigma_\nu^2 + \sigma_w^2(x))^{1/2}$. In the left part of the figure the noise variance is treated as a constant and whose estimate is obtained using the evidence framework of Chapter 1. We see that the error bars are dominated by the estimate of the average noise variance and as a result the model is particularly confident about the regression in the regions of low density where the true noise variance is actually large. This can be contrasted to the right part of the figure where the noise variance is allowed to vary as a function of the inputs and its estimate was obtained using the evidence framework formulated in this chapter. Here we see the error bars are small in the region of high density where the noise variance is low and the further away from this region the error bars become larger due to the increase in the magnitude of the noise variance.

regions of isolated data points there is little evidence to suggest a small value of the noise variance and so we expect much larger error bars. An illustration of the difference in the estimate of the error bars using constant and input-dependent noise variance is given in Figure 4.1.

In the rest of this chapter we will study regression with an input-dependent noise. To start with, we will first consider the maximum likelihood view of the problem and show that it is a biased estimator of noise variance. Then using the evidence procedure we will develop an approximate Bayesian formalism for tackling regression problems with input-dependent noise variance, which includes constant noise variance as a special case, and which overcomes the bias of the maximum likelihood.

4.2 Regression with input-dependent noise

The data set consists of N input-target pairs $D \equiv \{x_i, t_i\}$, with the inputs independently selected from some distribution function $p(x)$. We assume that the targets $\{t_i\}$ are related to the inputs through a smooth function $f(x)$ plus the addition of noise $\nu(x)$. Thus, for the input x_i the observed target t_i is given by

$$t_i = f(x_i) + \nu(x_i) \tag{4.1}$$

We further assume that the noise component $\nu(\boldsymbol{x})$ is generated from a random process having a Gaussian distribution of zero mean and true variance $s^2(\boldsymbol{x})$ which varies as a function of the inputs.

Our aim is to predict the function $f(\boldsymbol{x})$ and the noise variance $s^2(\boldsymbol{x})$. To this end we need two outputs. The first output $y(\boldsymbol{w};\boldsymbol{x})$, which represents the regression function, is governed by the set of weights $\boldsymbol{w}$ and predicts $f(\boldsymbol{x})$. The second output $\beta(\boldsymbol{u};\boldsymbol{x}) = \sigma_\nu^{-2}(\boldsymbol{u};\boldsymbol{x})$, which represents the noise variance, is governed by the set of weights $\boldsymbol{u}$ and predicts $s^2(\boldsymbol{x})$. This is illustrated in Figure 4.2. The conditional distribution of the target t_i given the input $\boldsymbol{x}_i$ is then modelled by a normal distribution $p(t_i|\boldsymbol{x}_i,\boldsymbol{w},\boldsymbol{u}) = \mathcal{N}(t_i|y_i,\beta_i^{-1})$, where $y_i = y(\boldsymbol{x}_i;\boldsymbol{w})$ and $\beta_i \equiv \beta(\boldsymbol{x}_i;\boldsymbol{u})$. Assuming that the data points are independently selected the likelihood function[1] can be written as

$$\begin{aligned} p(D|\boldsymbol{w},\boldsymbol{u}) &= \prod_{i=1}^{N} p(t_i|\boldsymbol{x}_i,\boldsymbol{w},\boldsymbol{u}) \\ &= \frac{1}{Z_D}\exp\left(-\sum_{i=1}^{N}\beta_i E_i\right) \end{aligned} \tag{4.2}$$

where E_i is the least-square error

$$E_i = \frac{1}{2}(y_i - t_i)^2 \tag{4.3}$$

and the normalising factor Z_D is given by[2]

$$Z_D = \frac{(2\pi)^{N/2}}{\prod_{i=1}^{N}\beta_i^{1/2}} \tag{4.4}$$

Having chosen the likelihood function we shall now consider the problem of regression with input-dependent noise. We start with the maximum likelihood approach.

4.3 The maximum likelihood approach

Regression with input-dependent noise variance has been studied by Satchwell (1994) and Nix and Weigend (1994, 1995) using the maximum likelihood method. More recently Williams (1996) has extended this approach to the case of outputs with input-dependent correlations. We have already discussed the maximum likelihood technique in Chapter 1 and highlighted its approach to making predictions which is based on single best parameter estimates. In the context of regression with input-dependent noise this means finding the most likely values $\widehat{\boldsymbol{w}}$ and $\widehat{\boldsymbol{u}}$ by maximising of the likelihood

[1] In favour of a simpler notation, I choose to write the likelihood as $p(D|\boldsymbol{w},\boldsymbol{u})$ which should be read as the probability of the targets $\{t_i\}$ given the inputs $\{\boldsymbol{x}_i\}$ and the weights $\boldsymbol{w}$ and $\boldsymbol{u}$, *i.e.* $p(\{t_i\}|\{\boldsymbol{x}_i\},\boldsymbol{w},\boldsymbol{u})$. Similarly any distribution appearing in this chapter having the form $p(D|...)$ should be understood as $p(\{t_i\}|\{\boldsymbol{x}_i\},...)$.

[2] For m outputs with identical noise components $Z_D = \frac{(2\pi)^{mN/2}}{\prod_{i=1}^{N}\beta_i^{m/2}}$.

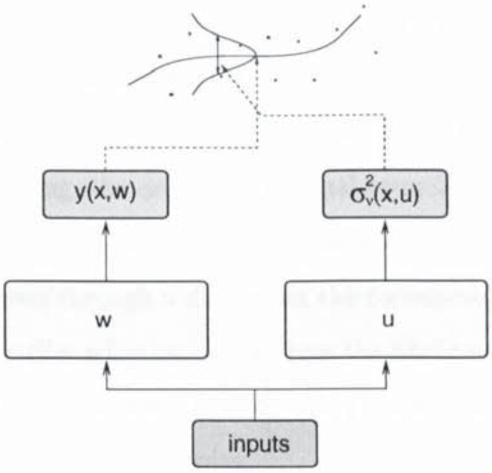

Figure 4.2: Schematic of model architecture for solving regression problems with input-dependent noise variance. The conditional mean of the targets are governed by the output $y(\boldsymbol{x};\boldsymbol{w})$ which is controlled by the weights $\boldsymbol{w}$ and the variance of noise on the targets is controlled by the output $\sigma_\nu^2(\boldsymbol{x};\boldsymbol{u})$ which is controlled by the weights $\boldsymbol{u}$.

function (4.2) with respect to $\boldsymbol{w}$ and $\boldsymbol{u}$. Equivalently, we may minimise the error function $E(\boldsymbol{w},\boldsymbol{u})$, which apart from some additive constants, can be written as

$$\begin{aligned} E(\boldsymbol{w},\boldsymbol{u}) &= -\ln p(D|\boldsymbol{w},\boldsymbol{u}) \\ &= \sum_{i=1}^{N}\beta_i E_i - \frac{1}{2}\sum_{i}^{N}\ln\beta_i \end{aligned} \tag{4.5}$$

It is common practice to add penalty terms to the error function (4.5). This will lead to the penalised error

$$E(\boldsymbol{w},\boldsymbol{u}) = \sum_{i=1}^{N}\beta_i E_i - \frac{1}{2}\sum_{i=1}^{N}\ln\beta_i + \alpha_w E_w + \alpha_u E_u \tag{4.6}$$

where

$$E_w(\boldsymbol{w}) = \frac{1}{2}\,\boldsymbol{w}^T\boldsymbol{w} \tag{4.7}$$

and

$$E_u(\boldsymbol{u}) = \frac{1}{2}\,\boldsymbol{u}^T\boldsymbol{u} \tag{4.8}$$

This regularization procedure, which leads to the so called 'penalised maximum likelihood', is known to reduce over-fitting by encouraging small weight values and hence preventing large output curvature. Note that, since the weights $\boldsymbol{w}$ and $\boldsymbol{u}$ have, in general, different scales, they have been assigned different regularising constants α_w and α_u. The optimum values $\widehat{\alpha}_w$ and $\widehat{\alpha}_u$ can be estimated from the data using cross-validation methods (Bishop 1995a). Given, however, the two dimensional nature of the problem, finding these estimates are computationally expensive and also wasteful of data since the available data set has to be partitioned. As we shall see later, in the Bayesian framework there is no need for such a procedure.

It is not satisfactory to minimise the error function (4.5) or (4.6) jointly with respect to $\boldsymbol{w}$ and $\boldsymbol{u}$. In the process of fitting the data, the weights $\boldsymbol{w}$ will unavoidably fit some of the noise because some of the noise components are indistinguishable from the data. When the regression passes close to a target point due to overfitting, the estimated residual error $2E_i$, on which the estimate of the noise variance is based, becomes small giving rise to an under-estimate of noise variance. In extreme cases, where the regression passes through a data point the corresponding estimate of noise variance can go to zero corresponding to $\beta(\boldsymbol{x};\boldsymbol{u}) \rightarrow \infty$. In this case the likelihood diverges. However, it should be mentioned that such cases can be avoided by controlling model complexity by using ,for example, proper regularization procedures or early stopping.

The solution to this problem has already been mentioned in Section 1.7.1 and was first suggested in this context by MacKay 1991. In order to obtain an unbiased estimate of the noise variance $\sigma_\nu^2(\boldsymbol{x};\boldsymbol{u})$ we must find the marginal distribution of $\sigma_\nu^2(\boldsymbol{x};\boldsymbol{u})$ in which we have integrated out the dependence on $\boldsymbol{w}$. Equivalently, we may estimate $\boldsymbol{u}$ from its marginal distribution, in which we have integrated out the dependence on $\boldsymbol{w}$ and then use $\boldsymbol{u}$ to find the estimate of $\sigma_\nu^2(\boldsymbol{x};\boldsymbol{u})$. This leads, as we shall in Section 4.7, to a hierarchical Bayesian analysis.

4.4 A toy problem

Before proceeding with developing the general Bayesian treatment of regression for the case of input-dependent noise it is useful to consider a toy problem involving estimating the mean and variance of a sample $X = \{x_1, ..., x_N\}$ of Gaussian random variables of unknown mean μ and variance s^2. As we shall see, this will highlight the inadequacy of the maximum likelihood approach as a biased estimator of the noise variance and also shows how marginalisation, which is a Bayesian concept, can correct for this bias.

A maximum likelihood approach to finding the unknown parameters μ and σ^2 is to maximise the the likelihood jointly with respect to μ and σ^2, which corresponds to finding the most likely values $\widehat{\mu}$ and $\widehat{\sigma}^2$ which explain the data best. This yields the standard results

$$\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{4.9}$$

and

$$\widehat{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \widehat{\mu})^2 \tag{4.10}$$

It is well known that the maximum likelihood estimate $\widehat{\sigma}^2$ in (4.10) is biased, since its average over samples

$$\langle \widehat{\sigma}_\nu^2 \rangle = \frac{N-1}{N}\, s^2 \tag{4.11}$$

where $\langle . \rangle$ denotes average over samples of size N, is not equal to the true value s^2 unless $N \to \infty$. By adopting the Bayesian approach this bias can be removed. Here we compute the estimate σ^2 of the variance by integrating over the mean μ. Assuming a flat prior $p(\mu)$ we obtain

$$\begin{aligned} p(D|\sigma^2) &= \int p(D|\mu,\sigma^2)p(\mu)\, d\mu \\ &\propto \frac{1}{\sigma^{N-1}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(x_i-\widehat{\mu})^2\right) \end{aligned} \tag{4.12}$$

Maximising the above result with respect to σ^2 yields

$$\widetilde{\sigma}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i-\widehat{\mu})^2 \tag{4.13}$$

which is unbiased. The effect of marginalisation on the estimated value of the noise variance is illustrated in Figure 4.3 which shows the contours of $p(D|\mu,\sigma^2)$ together with the marginal likelihood $p(D|\sigma^2)$ and the conditional likelihood $p(D|\widehat{\mu},\sigma^2)$. Note that for large N the effect of marginalisation is small. However, in the case of regression problems there is generally a much larger number of degrees of freedom, *i.e.* the number of weights, in relation to the size N of the training data, in which case the effect of the bias of the maximum likelihood is significant. To see this let use recall the maximum and Bayesian formulae for estimating the noise variance which we reviewed in Chapter 1, which are

$$\begin{aligned} \sigma_\nu^2 &= \frac{2E_D}{N} \\ &= \frac{1}{N}\sum_{i=1}^{N}(y_i-t_i)^2 \end{aligned} \tag{4.14}$$

and

$$\begin{aligned} \sigma_\nu^2 &= \frac{2E_D}{N-\gamma} \\ &= \frac{1}{N-\gamma}\sum_{i=1}^{N}(y_i-t_i)^2 \end{aligned} \tag{4.15}$$

respectively, where γ is the number of well determined parameters and $E_D = \frac{1}{2}\sum_{i=1}^{N}(y_i-t_i)^2$ is the sum-of-squares error function. We note that the principal difference between the two formulae above for estimating the noise variance is the presence of the term γ. This difference becomes significant when γ is comparable to the size of the training data N. We can understand the bias of the maximum likelihood in estimating the noise variance in the context of regression to be the result of estimating the noise variance directly from the residual errors $(y_i-t_i)^2$, and not taking into account the uncertainty in the weights and, hence, in the outputs y_i on which this error depends. The implication of this is that when the regression passes through or close to a target due to overfitting, the contribution of that data point to the estimate of the noise variance is zero or small, leading to an underestimation of the noise variance. This bias is corrected by integrating over the weights and then estimating σ_ν^2 by maximising the posterior $p(\beta, \alpha|D)$ (or the evidence $p(D|\beta, \alpha)$ if we impose a flat prior) of formula (1.29). This leads to the result (4.15). The presence of γ in this formula takes account of the fact that only γ out of k parameters are well-determined by the data and, therefore, only these weights can suppress the noise as a result of over fitting. If we set the hyperparameter $\alpha = 0$ then k will replace γ in formula (4.15). In fact as it turns out, there is an intimate connection between the Bayesian formula (4.15) for the estimate of the noise variance and the confidence variance $\sigma_w^2(x)$. It can be shown that the Bayesian estimate of the noise variance can be written as (see Appendix C.2 for the proof)

$$\tilde{\sigma}_\nu^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - t_i)^2 + \frac{1}{N}\sum_{i=1}^{N}\sigma_w^2(x_i) \tag{4.16}$$

The second term on the right hand side of (4.16) is the average value of the confidence variance defined over the training set. Thus the effect of marginalisation over the weights is to shift the estimated value of the error bars by an amount equal to the magnitude of the average value of confidence variance. Since the variance $\sigma_w^2(x)$ is relatively large in the regions of low density, the contribution of isolated data points to the estimate of the noise variance does not vanish but may be significant even if overfitting occurs. It should be noted that when the data set is large the contribution $\langle\sigma_w^2(x)\rangle$ to the noise variance becomes negligible and the maximum likelihood estimate of σ_ν^2 approaches that of the Bayesian approach. However, in real-world problems such sufficiently large data sets are seldom available in which case the results of Bayesian formalism are significantly different from those of the maximum likelihood.

4.5 The evidence framework for regression with input-dependent noise

In the rest of this chapter we will develop an approximate Bayesian formalism for regression with input-dependent noise which is based on the evidence framework discussed in Chapter 1. The aim is to make predictions and estimate error bars of those predictions from

$$y(x) = \int y(x; w)p(w|D)\, dw \tag{4.17}$$

$$\sigma_t^2(x) = \sigma_\nu^2(x) + \int \Big(y(x; w) - y(x)\Big)^2 p(w|D)\, dw \tag{4.18}$$

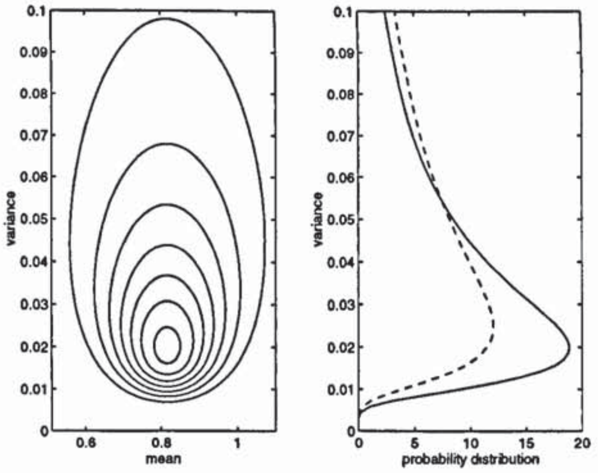

Figure 4.3: The left hand plot shows the contours of the likelihood function $p(D|\mu, \sigma^2)$ for a sample of size $N = 5$ drawn from a Gaussian distribution having zero mean and true variance $s^2 = 0.05$. The right hand plot shows the marginal likelihood function $p(D|\sigma^2)$ (dashed curve) and the conditional likelihood function $p(D|\widehat{\mu}, \sigma^2)$ (solid curve), where $\widehat{\mu}$ is estimated from the data. It can be seen that the skewed contours result in a value of $\widehat{\sigma}^2$ smaller than the estimated value $\widetilde{\sigma}^2$ which is estimated from the marginal likelihood function.

where $y(\boldsymbol{x})$ is the marginal output of the model, $\sigma_t^2(\boldsymbol{x})$ is the prediction variance and $p(\boldsymbol{w}|D)$ is the posterior probability distribution for the weights $\boldsymbol{w}$ which is given by

$$p(\boldsymbol{w}|D) = \iiint p(\boldsymbol{w}|D, \boldsymbol{u}, \alpha_w) p(\boldsymbol{u}|D, \alpha_w, \alpha_u) p(\alpha_w, \alpha_u|D) \, d\boldsymbol{u} \, d\alpha_w \, d\alpha_u \tag{4.19}$$

where $p(\boldsymbol{w}|D, \boldsymbol{u}, \alpha_w)$ is the conditional posterior for the weights $\boldsymbol{w}$ which control the regression, $p(\boldsymbol{u}|D, \alpha_w, \alpha_u)$ is the posterior for the weights $\boldsymbol{u}$ which controls the noise variance and finally $p(\alpha_w, \alpha_u|D)$ is the posterior distribution for the hyperparameters α_w and α_u which control $\boldsymbol{w}$ and $\boldsymbol{u}$, respectively. Our task is to perform the integral in (4.17) and (4.18) analytically. To this end we need to approximate the integral in (4.19), for evaluating the true posterior $p(\boldsymbol{w}|D)$, which involves integration over two types of parameters which are the weights $\boldsymbol{u}$ and the hyperparameters α_w and α_u. We have already discussed the evidence approximation to handling the hyperparameters in Chapter 1. If the posterior $p(\alpha_w, \alpha_u|D)$ is sharply peaked around the most probable values $\widetilde{\alpha}_w$ and $\widetilde{\alpha}_u$, which would be the case if α_w and α_u are well determined by the data, then integration over these hyperparameters is very much like estimating them from the data. As a result, we have

$$\begin{aligned} p(\boldsymbol{w}|D) &\approx \int p(\boldsymbol{w}|D, \boldsymbol{u}, \widetilde{\alpha}_w) p(\boldsymbol{u}|D, \widetilde{\alpha}_w, \widetilde{\alpha}_u) \, d\boldsymbol{u} \left[\iint p(\alpha_w, \alpha_u|D) \, d\alpha_w \, d\alpha_u \right] \\ &= \int p(\boldsymbol{w}|D, \boldsymbol{u}, \widetilde{\alpha}_w) p(\boldsymbol{u}|D, \widetilde{\alpha}_w, \widetilde{\alpha}_u) \, d\boldsymbol{u} \end{aligned} \tag{4.20}$$

Thus our task is to find the most probable values $\widetilde{\alpha}_w$ and $\widetilde{\alpha}_u$ which requires the knowledge of the posterior $p(\alpha_w, \alpha_u|D)$ which we will deal with later. For now let us assume that these values are known.

Next step we need to handle the rest of the integral in (4.20) which involves integration over the weights $\boldsymbol{u}$. There are two options. First is to integrate over the weights $\boldsymbol{u}$ analytically. In this case we may assume that the posterior $p(\boldsymbol{u}|D, \widetilde{\alpha}_w, \widetilde{\alpha}_u)$ can be approximated by a multivariate Gaussian with mean located at the most probable value $\widetilde{\boldsymbol{u}}$. This is just the Gaussian approximation discussed in Chapter 1. The second option is to assume that the posterior $p(\boldsymbol{u}|D, \widetilde{\alpha}_w, \widetilde{\alpha}_u)$ is sharply peaked around $\widetilde{\boldsymbol{u}}$ and so the integral in (4.20) can be written as

$$\begin{aligned} p(\boldsymbol{w}|D) &\approx p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w) \int p(\boldsymbol{u}|D, \widetilde{\alpha}_w, \widetilde{\alpha}_u)\, d\boldsymbol{u} \\ &= p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w) \end{aligned} \tag{4.21}$$

This step is similar to handling the hyperparameters α_w and α_u but not quite the same. The reason is the weights $\boldsymbol{u}$ have the same status as $\boldsymbol{w}$ and can have as many components as $\boldsymbol{w}$. Therefore, it is doubtful that except for sufficiently large data sets, the approximation above is reasonably valid. Thus we have to choose between these two options. Here we go for the latter for a number of reasons. First, the posterior $p(\boldsymbol{u}|D, \widetilde{\alpha}_w, \alpha_u)$ is not quadratic in the weights and this is true even for GLR models, as we shall see later, and so the best we can do is to represent it by a Gaussian which is only an approximation. Second, for GLR models the posterior $p(\boldsymbol{w}|\widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)$ is a Gaussian enabling us to integrate over $\boldsymbol{w}$ in an exact manner, while the posterior $p(\boldsymbol{w}|D, \widetilde{\alpha}_w)$ is not. Since obtaining the marginal output is more sensitive to integration over $\boldsymbol{w}$ than integration over $\boldsymbol{u}$, it is more likely that integration over $\boldsymbol{u}$ as in (4.20) be counterproductive. In this context, non-linear models such as neural networks provide another reason for using the posterior $p(\boldsymbol{w}|\widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)$ rather than $p(\boldsymbol{w}|D)$ for making predictions. As explained in Section 1.6.2, representing the true posterior $p(\boldsymbol{w}|D)$ by the Gaussian approximation can sometimes lead to more prediction error than representing the conditional posterior $p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)$ using the same approximation.

Choosing $p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)$ to represent the true posterior $p(\boldsymbol{w}|D)$ we can now make predictions and estimate the error bars from

$$y(\boldsymbol{x}) \approx \int y(\boldsymbol{x}; \boldsymbol{w}) p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)\, d\boldsymbol{w} \tag{4.22}$$

and

$$\sigma_t^2(\boldsymbol{x}) \approx \sigma_\nu^2(\boldsymbol{x}) + \int \Big(y(\boldsymbol{x}; \boldsymbol{w}) - y(\boldsymbol{x})\Big)^2 p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)\, d\boldsymbol{w} \tag{4.23}$$

The above integrals are exact for GLR models, while for non-linear models such as neural networks we need to make the Gaussian approximation to the posterior $p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)$. In Section 4.7 we will formulate this approximate framework for the application of the Bayesian formalism to regression with input-dependent noise variance.

4.6 Choice of the model

So far we have not specified the outputs $y(\boldsymbol{x};\boldsymbol{w})$ and $\beta(\boldsymbol{x};\boldsymbol{u})$, which can be either the outputs of a GLR model or a neural network. Generally we can write

$$y(\boldsymbol{x};\boldsymbol{w}) = \mathrm{a}(\boldsymbol{x};\boldsymbol{w}) \tag{4.24}$$

where $\mathrm{a}(\boldsymbol{x};\boldsymbol{w})$ is a function of the inputs $\boldsymbol{x}$ and the weights $\boldsymbol{w}$. For GLR models the expression simplifies to $\mathrm{a}(\boldsymbol{x};\boldsymbol{w}) = \boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x})$, where the vector of the basis functions $\boldsymbol{\phi}(\boldsymbol{x})$ has k_w components. Since the noise variance $\sigma_\nu^2(\boldsymbol{x})$ is always positive we choose it to be an exponential function of the form

$$\sigma_\nu^2(\boldsymbol{x};\boldsymbol{u}) = \exp(\mathrm{b}(\boldsymbol{x};\boldsymbol{u})) \tag{4.25}$$

This will prevent the noise variance from taking negative values. For GLR models $\mathrm{b}(\boldsymbol{x};\boldsymbol{w}) = \boldsymbol{u}^T\boldsymbol{\psi}(\boldsymbol{x})$, where the vector of basis functions $\boldsymbol{\psi}(\boldsymbol{x})$ has k_u components. The use of the exponential function is not uncommon and has been used by several authors (Jacobs et al. 1991; Nowlan and Hinton 1992; Williams 1996) for modelling scale parameters, which are positive by definition, using a representation of the form $\zeta = \exp(\eta)$. One particular way of choosing a prior for η is to impose a uniform prior $p(\eta) = c$, where c is a constant. It is straightforward to show (Berger 1985) that such a prior on η will induce a corresponding prior on ζ which has the form $p(\zeta) = \zeta^{-1}$. More generally, we can work out the form of the prior for ζ given the prior for η. Further discussion of this matter is given in Appendix C.1.

4.7 Three levels of inference

4.7.1 First level

In this level of Bayesian inference we need to derive an expression for the posterior distribution $p(\boldsymbol{w}|D,\boldsymbol{u},\alpha_w)$ from which we infer the most probable value $\widetilde{\boldsymbol{w}}$. Our starting point is the likelihood function, as given by (4.2), which we combine with the prior $p(\boldsymbol{w}|\alpha_w)$, using Bayes' rule to obtain

$$p(\boldsymbol{w}|D,\boldsymbol{u},\alpha_w) = \frac{p(D|\boldsymbol{w},\boldsymbol{u})p(\boldsymbol{w}|\alpha_w)}{p(D|\boldsymbol{u},\alpha_w)} \tag{4.26}$$

The denominator $p(D|\boldsymbol{u},\alpha_w)$, which is the evidence for $\boldsymbol{u}$ and α_w, acts as a normalising factor and has no significance in this level of inference. If we choose a weight decay prior for $\boldsymbol{w}$, we have

$$p(\boldsymbol{w}|\alpha_w) = \frac{1}{Z_w(\alpha_w)}\exp(-\alpha_w E_w(\boldsymbol{w})) \tag{4.27}$$

where $Z_w(\alpha_w)$ is the normalising constant

$$Z_w(\alpha_w) = \left(\frac{2\pi}{\alpha_w}\right)^{k_w/2} \tag{4.28}$$

and $E_w(w)$ is the weight decay regulariser of (4.7). Rewriting the posterior using (4.2) and (4.27), we have

$$p(w|D, u, \alpha_w) = \frac{1}{Z_s} \exp(-S(w)) \tag{4.29}$$

where

$$S(w) = \sum_{i=1}^{N} \beta_i E_i + \alpha_w E_w \tag{4.30}$$

where $E_i = \frac{1}{2}(y_i - t_i)^2$ is the least-square error, $y_i \equiv y(x_i; w)$, and $\beta_i \equiv \beta(x_i; u)$ and the normalising factor $Z_S(u, \alpha_w)$ is given by

$$Z_S(u, \alpha_w) = \int \exp\Big(- S(w)\Big)\, dw \tag{4.31}$$

For GLR models the posterior $p(w|D, u, \alpha_w)$ is a Gaussian with mean centred at the most probable weights $\widetilde{w}$ which can be found from minimising the error $S(w)$ in (4.30) to give

$$\widetilde{w} = A^{-1}\Phi^T \beta t \tag{4.32}$$

where β is an NxN diagonal matrix with elements $\beta(x_i, u)$, Φ is the design matrix of equation (3.4), t is the vector of the targets t_i and A is the Hessian matrix

$$\begin{aligned} A &= \frac{\partial^2 S(w)}{\partial w^2}\Big|_{\widetilde{w}} \\ &= \sum_{i=1}^{N} \beta_i B_i + \alpha_w I \end{aligned} \tag{4.33}$$

where the matrix $B_i = \partial^2 E_i / \partial w^2$ is the Hessian matrix for a the ith data point. Using Taylor expansion of the error $S(w)$,

$$S(w) = S(\widetilde{w}) + \frac{1}{2}(w - \widetilde{w})^T A (w - \widetilde{w}) \tag{4.34}$$

in (4.31) we obtain

$$Z_S(u, \alpha_w) = (2\pi)^{k_w/2} |A|^{-1/2} \exp(-S(\widetilde{w})) \tag{4.35}$$

The expansion in (4.34) is exact for GLR models as the error function $S(w)$ is quadratic in the weights, *i.e.* $p(w|D, u, \alpha_w)$ is Gaussian, while for neural networks it is an approximation. From now onwards, all quantities depending on the weights w are measured at the most probable value $\widetilde{w}$.

4.7.2 Second level

In the first level of inference we obtained a formula for the conditional posterior distribution $p(\boldsymbol{w}|D, \boldsymbol{u}, \alpha_w)$. Now we are interested in $p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)$ which requires the knowledge of $\widetilde{\boldsymbol{u}}$ and $\widetilde{\alpha}_w$. In this level of inference we will infer $\widetilde{\boldsymbol{u}}$. We begin from the posterior $p(\boldsymbol{u}|D, \alpha_w, \alpha_u)$ which can be written, using Bayes' rule, as.

$$p(\boldsymbol{u}|D, \alpha_w, \alpha_u) = \frac{p(D|\boldsymbol{u}, \alpha_w)p(\boldsymbol{u}|\alpha_u)}{p(D|\alpha_w, \alpha_u)} \tag{4.36}$$

The evidence $p(D|\boldsymbol{u}, \alpha_w)$ has already appeared in the previous level of inference as a normalising factor in (4.26). It is given by

$$\begin{aligned} p(D|\boldsymbol{u}, \alpha_w) &= \int p(\boldsymbol{w}, D|\boldsymbol{u}, \alpha_w)\, d\boldsymbol{w} \\ &= \int p(D|\boldsymbol{w}, \boldsymbol{u})p(\boldsymbol{w}|\alpha_w)\, d\boldsymbol{w} \\ &= \frac{Z_S(\boldsymbol{u}, \alpha_w)}{Z_D(\boldsymbol{u})Z_w(\alpha_w)} \end{aligned} \tag{4.37}$$

where $Z_D(\boldsymbol{u})$, $Z_w(\boldsymbol{w})$ and $Z_S(\boldsymbol{u}, \alpha_w)$ are given by (4.4), (4.28) and (4.35), respectively. Note that the posterior in (4.36) is not conditioned on the weights $\boldsymbol{w}$. We have already explained that in order to obtain an unbiased estimate of the variance we should integrate over the mean. In the context of regression with input-dependent noise variance, this implies that we should estimate the noise variance $\sigma_\nu^2(\boldsymbol{x}; \boldsymbol{u})$ irrespective of the regression function $y(\boldsymbol{x}; \boldsymbol{w})$. In other words, we should estimate the weights $\boldsymbol{u}$ after we have integrated over $\boldsymbol{w}$. This is why we could not optimise $\boldsymbol{u}$ simultaneously with $\boldsymbol{w}$ as would be the case in the maximum likelihood approach.

To complete the current level of inference, we also need to choose a prior for the weights $\boldsymbol{u}$. Since inferring noise variance is essentially a regression problem we require the outputs $\beta(\boldsymbol{x}_i; \boldsymbol{u}) = \sigma_\nu^2(\boldsymbol{x}; \boldsymbol{u})$ to be smooth, and so we choose a zero-mean Gaussian prior[3]

$$p(\boldsymbol{u}|\alpha_u) = \frac{1}{Z_u(\alpha_u)} \exp(-\alpha_u E_u(\boldsymbol{u})) \tag{4.38}$$

where $E_u(\boldsymbol{u})$ is given by (4.8) and the normalising constant can be written as

$$Z_u(\alpha_u) = \left(\frac{2\pi}{\alpha_u}\right)^{k_u/2} \tag{4.39}$$

Using (4.37) together with (4.38) in (4.36), we obtain

$$\begin{aligned} p(\boldsymbol{u}|D, \alpha_u, \alpha_w) &\propto \frac{Z_S(\boldsymbol{u}, \alpha_w)}{Z_D(\boldsymbol{u})Z_w(\alpha_w)Z_u(\alpha_u)} \exp(-\alpha_u E_u(\boldsymbol{u})) \\ &\propto (\alpha_w)^{k_w/2}(\alpha_u)^{k_u/2} \exp(-M(\boldsymbol{u})) \end{aligned} \tag{4.40}$$

[3] Further consideration to the prior over the space of weights $\boldsymbol{u}$ is given in Appendix C.1.

where we have defined the error function $M(\boldsymbol{u})$ as

$$M(\boldsymbol{u}) = \sum_{i=1}^{N} \beta_i E_i + \alpha_u E_u - \frac{1}{2}\sum_{i=1}^{N} \ln \beta_i + \frac{1}{2}\ln|\boldsymbol{A}| \tag{4.41}$$

The most probable value $\widetilde{\boldsymbol{u}}$ is then found from minimising $M(\boldsymbol{u})$. One of the interesting terms which appear in (4.41) is the logarithm of the determinant of the Hessian matrix $\ln|\boldsymbol{A}|$. The presence of this term is due to marginalisation over the regression weights $\boldsymbol{w}$. In fact this term is the only difference between the error $E(\boldsymbol{w}, \boldsymbol{u})$ of (4.6) of the maximum likelihood and $M(\boldsymbol{u})$ as far as estimation of $\widetilde{\boldsymbol{u}}$ is concerned. To gain a *crude* idea about the effect of this term on the inferred value of noise variance, let us minimise M with respect to β_i. From differentiating this error with respect to β_i, and using

$$\begin{aligned} \frac{\partial}{\partial \beta_i} \ln|\boldsymbol{A}| &= \text{Trace}(\boldsymbol{A}^{-1}\boldsymbol{B}_i) \\ &= \boldsymbol{g}^T(\boldsymbol{x}_i)\boldsymbol{A}^{-1}\boldsymbol{g}(\boldsymbol{x}_i) \\ &= \sigma_w^2(\boldsymbol{x}_i) \end{aligned} \tag{4.42}$$

we obtain

$$\sigma_\nu^2(\boldsymbol{x}_i) = 2E_i + \sigma_w^2(\boldsymbol{x}_i) \tag{4.43}$$

The quantity $2E_i$ is the square of the residual regression error measured at the most probable value of the weights $\widetilde{\boldsymbol{w}}$ and $\sigma_w^2(\boldsymbol{x}_i)$ is the confidence variance due to uncertainty in these weights measured at the input point $\boldsymbol{x}_i$. Thus the noise variance at an input point $\boldsymbol{x}_i$ is given by the sum of the residual error $2E_i$ and the error bars $\sigma_w^2(\boldsymbol{x}_i)$. We have already obtained a similar result (see formula (4.16)) for the case of a constant noise variance in Section 4.4. However, it must be mentioned here that in the case of an input-dependent noise we estimate the noise variance by minimising M with respect to the weights $\boldsymbol{u}$ rather than the β_i's directly.

4.7.3 Third level

The Bayesian formalism we have developed so far involves two levels of inference. In the first level we derived a formula for the posterior $p(\boldsymbol{w}|D, \boldsymbol{u}, \alpha_w)$ which is conditioned on $\boldsymbol{u}$, α_w. Naturally we choose this posterior to be measured at $\widetilde{\boldsymbol{u}}$ and $\widetilde{\alpha}_w$. Thus we are interested in $p(\boldsymbol{w}|\widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w)$. To find $\widetilde{\boldsymbol{u}}$ we needed a second level of inference in which we inferred $\widetilde{\boldsymbol{u}}$ irrespective of $\boldsymbol{w}$. This was accomplished using marginalisation over the weights which led to the posterior $p(\boldsymbol{u}|D, \alpha_w, \alpha_u)$ from which $\widetilde{\boldsymbol{u}}$ was found. We can now proceed in a similar manner to implement the next level of inference in which we infer the most probable value of the hyperparameters. Here we need to derive the posterior $p(\alpha_w, \alpha_u|D)$ which, using Bayes' rule, can be written as

$$p(\alpha_u, \alpha_w|D) = \frac{p(D|\alpha_u, \alpha_w)p(\alpha_u, \alpha_w)}{p(D)} \tag{4.44}$$

where $p(D|\alpha_u, \alpha_w)$ is the evidence for the hyperparameters and $p(\alpha_u, \alpha_w)$ is the prior. Since α_w and α_u are assumed to be independent, we can split the prior into the product of two independent terms, *i.e.* $p(\alpha_w, \alpha_u) = p(\alpha_w)p(\alpha_u)$. Note that the denominator $p(D)$ has no role here as it does not depend on the hyperparameters.

If we chose the priors $p(\alpha_w)$ and $p(\alpha_u)$ to be flat, which corresponds to our lack of knowledge about which range of values α_w and α_u can take, then we can estimate $\widetilde{\alpha}_w$ and $\widetilde{\alpha}_u$ from the evidence $p(D|\alpha_w, \alpha_u)$ alone. The evidence itself can be evaluated from the previous level of inference

$$\begin{aligned} p(D|\alpha_u, \alpha_w) &= \int p(\boldsymbol{u}, D|\alpha_u, \alpha_w)\, d\boldsymbol{u} \\ &= \int p(D|\boldsymbol{u}, \alpha_w) p(\boldsymbol{u}|\alpha_u)\, d\boldsymbol{u} \\ &= \frac{1}{Z_w(\alpha_w) Z_u(\alpha_u)} \int \frac{Z_S(\boldsymbol{u}, \alpha_w)}{Z_D(\boldsymbol{u})} \exp(-\alpha_u E_u(\boldsymbol{u}))\, d\boldsymbol{u} \end{aligned} \tag{4.45}$$

where we have made use of (4.37) and (4.38). We now need to evaluate the integral in (4.45). By using (4.41) together with some algebraic manipulations we can write

$$\begin{aligned} p(D|\alpha_u, \alpha_w) &\propto \frac{\exp(-\alpha_w E_w)}{Z_w(\alpha_w) Z_u(\alpha_u)} \left[\int \left(\prod_i^N \beta_i^{1/2} \right) |\boldsymbol{A}|^{-1/2} \right. \\ &\qquad \left. \exp\left(-\sum_{i=1}^N \beta_i E_i - \alpha E_u(\boldsymbol{u}) \right) d\boldsymbol{u} \right] \\ &\propto \frac{\exp(-\alpha_w E_w)}{Z_w(\alpha_w) Z_u(\alpha_u)} \int \exp(-M(\boldsymbol{u}))\, d\boldsymbol{u} \end{aligned} \tag{4.46}$$

We have already seen the error function $M(\boldsymbol{u})$, given by (4.41), appearing in the second level of inference. Being nonlinear in $\boldsymbol{u}$ implies that we can not perform the integral in (4.46) in an exact manner. However, if we assume that the integrand $\exp(-M(\boldsymbol{u}))$ can be approximated by a Gaussian centred at $\widetilde{\boldsymbol{u}}$, then using Taylor expansion of $M(\boldsymbol{u})$

$$M(\boldsymbol{u}) = M(\widetilde{\boldsymbol{u}}) + \frac{1}{2} (\boldsymbol{u} - \widetilde{\boldsymbol{u}})^T \boldsymbol{H} (\boldsymbol{u} - \widetilde{\boldsymbol{u}}) \tag{4.47}$$

where

$$\boldsymbol{H} = \frac{\partial^2 M(\boldsymbol{u})}{\partial \boldsymbol{u}^2}\Big|_{\widetilde{\boldsymbol{u}}} \tag{4.48}$$

in (4.46) will give the approximate formula

$$p(D|\alpha_u, \alpha_w) \propto (\alpha_w)^{k_w/2} (\alpha_u)^{k_u/2}\, |\boldsymbol{H}|^{-1/2} \exp(-\alpha_w E_w) \exp(-M(\widetilde{\boldsymbol{u}})) \tag{4.49}$$

The matrix $\boldsymbol{H}$ (see Appendix C.3 for its evaluation) is the Hessian of the posterior $p(\boldsymbol{u}|D, \alpha_w, \alpha_u)$ which involves the second derivatives of the error $M(\boldsymbol{u})$ with respect to the weights $\boldsymbol{u}$ measured at the most probable value $\widetilde{\boldsymbol{u}}$. To find the optimum value of the hyperparameters $\widetilde{\alpha}_u$ and $\widetilde{\alpha}_w$ we need to maximise the evidence. Alternatively we may choose to minimise the error $W(\alpha_w, \alpha_u) = -\ln p(\alpha_w, \alpha_u|D)$ which can be written , apart from additive constants, as

$$\begin{aligned} W(\alpha_w, \alpha_u) &= -\ln p(D|\alpha_u, \alpha_w) \\ &= \alpha_w E_w + \alpha_u E_u - \frac{k_w}{2}\ln\alpha_w - \frac{k_u}{2}\ln\alpha_u \\ &\quad + \frac{1}{2}\ln|\boldsymbol{A}| + \frac{1}{2}\ln|\boldsymbol{H}| \end{aligned} \tag{4.50}$$

The presence of $\ln|\boldsymbol{A}|$ and $\ln|\boldsymbol{H}|$ in (4.50) is due to marginalisation over $\boldsymbol{w}$ and $\boldsymbol{u}$, respectively. At the minimum of the error $W(\alpha_w, \alpha_u)$ the most probable values of the hyperparameters satisfy

$$\widetilde{\alpha}_w = \frac{\gamma_w}{2E_w} \tag{4.51}$$

$$\widetilde{\alpha}_u = \frac{\gamma_u}{2E_u} \tag{4.52}$$

where we have defined the effective number of $\boldsymbol{w}$ and $\boldsymbol{u}$ weights γ_w and γ_u as

$$\gamma_w = k_w - \alpha_w \mathrm{Trace}\Big(\boldsymbol{A}^{-1} - \boldsymbol{H}^{-1}\frac{\partial \boldsymbol{H}}{\partial \alpha_w}\Big) \tag{4.53}$$

$$\gamma_u = k_u - \alpha_u \mathrm{Trace}(\boldsymbol{H}^{-1}) \tag{4.54}$$

The quantities γ_w and γ_u always lie in the intervals $(0, k_w)$ and $(0, k_u)$, depending on how well the weights are determined by the data. If the data set size is large relative to the size of the model, *i.e.* $N \gg k_w$ $(N \gg k_u)$, then α_w (α_u) is negligible in which case the weights $\boldsymbol{w}$ $(\boldsymbol{u})$ are well determined by the data. Therefore $\gamma_w \approx k_w$ $(\gamma_u \approx k_u)$, suggesting that

$$\widetilde{\alpha}_w \approx \frac{k_w}{2E_w} \tag{4.55}$$

$$\widetilde{\alpha}_u \approx \frac{k_u}{2E_u} \tag{4.56}$$

Formulae (4.55) and (4.56) do not distinguish between the the actual number of parameters and number of well determined parameters and they are particularly easy to implement as they do not require evaluation of the Hessian $\boldsymbol{H}$ which is computationally demanding. However, these formulae can be used only if the data size N is sufficiently large. For small N these formulae will favour large values of α_w and α_u dragging $\boldsymbol{w}$ and $\boldsymbol{u}$ towards smaller values.

4.8 Adapting the formalism for the case of multiple modes

In the Bayesian formalism for noisy regression which we have developed so far we have assumed that the posterior distribution $p(\boldsymbol{w}|D, \widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w, \widetilde{\alpha}_u)$ is uni-modal. While this is correct for GLR models, the posterior for neural networks may well have many modes, some pertaining to the symmetry of the network and some pertaining to unrelated maxima. So we have to handle the presence of these multiple modes. One simple approach is to limit ourselves to only one mode and carry out the Bayesian formalism for that mode, but one can also do better. In this case we chop the posterior space into sections each dominated by its own mode (MacKay 1992c). To find these modes we train the same network several times starting from different locations of weights $\boldsymbol{w}$. After finding a number of modes and their corresponding value $\widetilde{\boldsymbol{w}}$ we can then apply the evidence framework. In this case the model predictions is given by

$$\begin{aligned} y(\boldsymbol{x}) &= \sum_{j=1}^{m} \int y(\boldsymbol{x};\boldsymbol{w}) p_j(\boldsymbol{w}|\widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w, \widetilde{\alpha}_u)\, d\boldsymbol{w} \\ &= \frac{1}{m} \sum_{j=1}^{m} y_j(\boldsymbol{x};\widetilde{\boldsymbol{w}}) \qquad (4.57) \end{aligned}$$

where m is the number of modes found and $p_j(\boldsymbol{w}|\widetilde{\boldsymbol{u}}, \widetilde{\alpha}_w, \widetilde{\alpha}_u)$ is a Gaussian centred around $\boldsymbol{w}_j$ of the jth mode. Note that the evidence for the modes are approximated by $1/m$ in the above formula since the Gaussian approximation leads to a poor estimate of this quantity as mentioned in Section 1.6.3. It must be mentioned, however, that the success of this approach to multiple modes depends on the validity of the assumption that the modes of the posterior are well separated so that there is no significant overlap between the Gaussian distributions used to approximate the modes.

4.9 Implementing the three levels of inference

Thus we can distinguish between three levels of inference for implementing the Bayesian formalism of this chapter:

Step1 : Estimate the weights $\boldsymbol{w}$ for the current values of the weights $\boldsymbol{u}$ and the hyperparameter α_w by minimising the error function $S(\boldsymbol{w})$ in equation (4.30).

Step2 : Estimate the weights $\boldsymbol{u}$ for the current values of the hyperparameters α_w and α_u from minimising the error function $M(\boldsymbol{u})$ in equation (4.41).

Step3 : Finally, estimate the value of the hyperparameters α_w and α_u from minimising the error function $W(\alpha_w, \alpha_u)$ in equation (4.50). Since, however, the most probable value of the hyperparameters satisfy formulae (4.51) and (4.52) one can, instead of minimising $W(\alpha_w, \alpha_u)$, apply these expressions as re-estimation rules for updating the value of the hyperparameters during training. This can speed up the convergence of the algorithm.

The above procedures are then repeated until convergence is obtained, at which point the estimates of the weights and the hyperparameters should yield the most probable values $\widetilde{\boldsymbol{w}}$, $\widetilde{\boldsymbol{u}}$, $\widetilde{\alpha}_w$ and $\widetilde{\alpha}_u$.

Since optimisation of the weights w for GLR models moves on a faster time scale than optimisation of u, it would save appreciable computation overhead if u is updated from an outer loop within which w is continuously re-estimated. Within this outer loop one might also choose to optimise the hyperparameters which is easy to do if the approximate re-estimation formulae of (4.55) and (4.56) are used. However, for neural networks this optimisation arrangement might not be the best since the weights w are also to be optimised from an iterative procedure. One can instead update the weights w and u for a few cycles each followed by updating the hyperparameters. Finally, it may speed up the convergence of the algorithm if minimising the errors $S(w)$ and $M(u)$ is started from initial weights values which are good representative of the scale of the targets and the noise variance. This can result in appreciable time gain especially in optimising u since it involves evaluation of the Hessian A which is computationally expensive.

4.10 Prediction error bars

As we saw before, in the Bayesian framework the model parameters are treated as random variables with probability distributions. The distribution over the model parameters gives rise to distribution over model outputs which can be written as

$$\begin{aligned} p(t|x,D) &= \iiiint p(t|x,w,u)p(w,u,\alpha_w,\alpha_u|D)\, dw\, du\, d\alpha_w\, d\alpha_u \\ &\approx \int p(t|x,w,\tilde{u})p(w|D,\tilde{u},\tilde{\alpha}_w)\, dw \end{aligned} \tag{4.58}$$

where $p(w,u,\alpha_w,\alpha_u|D) = p(w|D,u,\alpha_w)p(u|D,\alpha_w,\alpha_u)p(\alpha_w,\alpha_u|D)$. If we assume that the posterior $p(w|D,\tilde{u},\tilde{\alpha}_w)$ is a Gaussian and that the outputs $y(x;w)$ depends linearly on the weights in the vicinity of $\tilde{w}$ (these are just approximations for neural networks) so that

$$y(x;w) \approx y(x;\tilde{w}) + (w-\tilde{w})^T g(x) \tag{4.59}$$

then the integral (4.58) is easily performed to give a predictive distribution $p(t|x,D)$ which is a Gaussian with mean $y(x) = y(x;\tilde{w})$ and variance $\sigma_t^2(x)$, where

$$\begin{aligned} \sigma_t^2(x) &= \sigma_\nu^2(x) + \sigma_w^2(x) \\ &= \sigma_\nu^2(x) + g^T(x)A^{-1}g(x) \end{aligned} \tag{4.60}$$

Again we see that the prediction variance $\sigma_t^2(x)$ is given by the sum of the noise variance $\sigma_\nu^2(x)$, which is now input-dependent, and the confidence variance $\sigma_w^2(x)$.

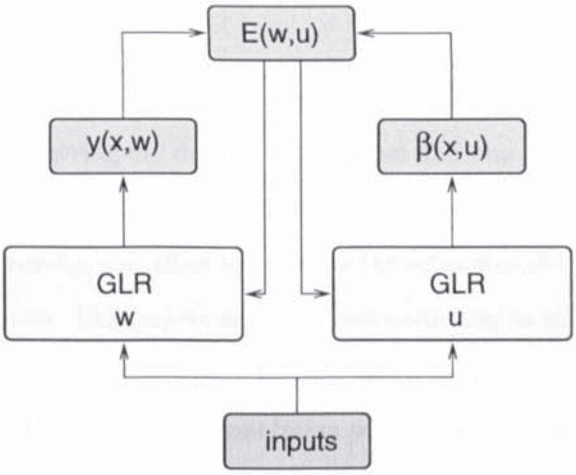

Figure 4.4: Model architecture for implementing regression with input-dependent noise variance using maximum likelihood approach. The system consists of two GLR models one with output $y(\boldsymbol{x};\boldsymbol{w})$ predicting the regression and the other with output $\sigma_\nu^2(\boldsymbol{x};\boldsymbol{u})$ predicting the noise variance. Both the regression and the noise variance are predicted from the joint minimisation of the error function $E(\boldsymbol{w},\boldsymbol{u})$ (formula (4.6)) with respect to $\boldsymbol{w}$ and $\boldsymbol{u}$.

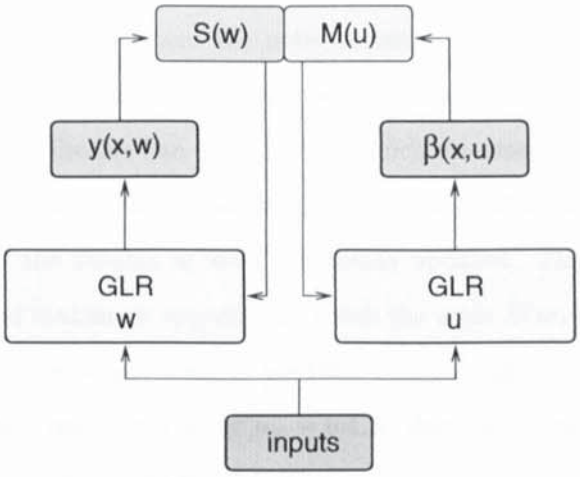

Figure 4.5: Model architecture for implementing regression with input-dependent noise variance using the Bayesian approach. The system consists of two GLR models one with output $y(\boldsymbol{x};\boldsymbol{w})$ predicting the regression and the other with output $\beta(\boldsymbol{x};\boldsymbol{u})$ predicting the noise variance. Contrary to the maximum likelihood, in the Bayesian approach the regression and noise variance are inferred in two steps involving minimisation of $S(\boldsymbol{w})$ (formula (4.30)) with respect to $\boldsymbol{w}$ and minimisation of $M(\boldsymbol{u})$ (formula (4.41)) with respect to $\boldsymbol{u}$.

4.11 Experimental results

In this section we will test the theoretical results we have obtained using two experiments. The first one is designed to compare the predicted noise variance of the Bayesian approach with that of maximum likelihood, while the second experiment is designed to test whether modelling the noise as input-dependent will improve the generalisation ability or not. These experiments are conducted using the arrangements shown in Figures 4.4 and 4.5.

4.11.1 Experiment 1

As a demonstration of the Bayesian treatment of regression with an input-dependent noise variance, we consider an experiment involving toy data with one input and one output. Here, 1000 data points are independently selected from a Gaussian distribution with zero mean and unit variance. The choice of the input density as a Gaussian will allow to compare the estimates of the noise variance in regions of low and high input densities. The targets are generated according to $\sin(0.35\pi x) + \nu(x)$, where the true noise variance is given by $s^2(x) = 0.05 + 0.05x^2$. Since an estimator is a random variable, its merit should be judged by the quality of a population of its estimates. For this reason the data set is divided into 100 sub-sets each containing 10 data points, and the model is trained on each sub-set in turn and tested on the remaining 99 sub-sets. Both outputs $y(\boldsymbol{x}; \boldsymbol{w})$ and $\beta(\boldsymbol{x}; \boldsymbol{u})$ have 4 uniformly distributed Gaussian basis functions (and a bias) each with width chosen equal to the spacing of the centres (see Appendix A.1 for further details).

As described above, the regression and the noise variance were estimated using both Bayesian and the penalised maximum likelihood techniques and the results are shown in Fig 4.6. For the Bayesian case training process involved an outer loop in which the most probable value $\widetilde{\boldsymbol{u}}$ is found by minimisation of $M(\boldsymbol{u})$ (4.41), using the scaled conjugate gradient algorithm (Williams 1991; Møller 1993). While looking for $\widetilde{\boldsymbol{u}}$ the weights $\boldsymbol{w}$ were continually updated. Then the same experiment is repeated using the penalised maximum approach in which the error $E(\boldsymbol{w}, \boldsymbol{u})$ in (4.6) is minimised to obtain the most likely estimates $\widehat{\boldsymbol{w}}$ and $\widehat{\boldsymbol{u}}$ which are then used to make predictions. In both cases the hyperparameters were given fixed values $\alpha_w = \alpha_u = 0.1$ as this allows the maximum likelihood and Bayesian approaches to be treated on an equal footing.

Figure 4.6 shows the results of this experiment averaged over the 100 trials. The reason for this averaging is that this is the definition of bias and that is what we are interested in. It is clear that the maximum likelihood approach systematically under-estimates the noise variance especially in the regions of low input density, while the Bayesian results shows improved estimates of the noise variance. This is born out by evaluating the logarithm of the likelihood for the test data. The Bayesian approach obtains -9.5 for the log likelihood per data point averaged over 100 runs. Due to overfitting the maximum likelihood occasionally gives extremely large negative values of the log likelihood corresponding to small estimates of the noise variance. Even omitting these extreme values, the maximum likelihood still gives an average log likelihood per data point of -17.5 which is substantially smaller than the Bayesian result.

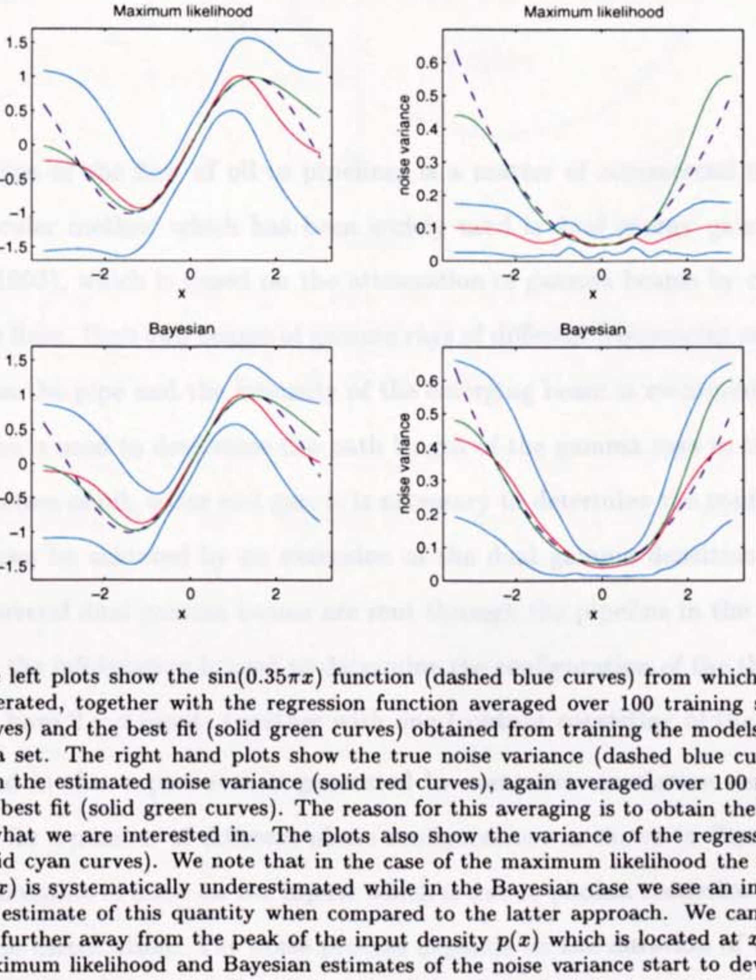

Figure 4.6: The left plots show the $\sin(0.35\pi x)$ function (dashed blue curves) from which the data was generated, together with the regression function averaged over 100 training sets (solid red curves) and the best fit (solid green curves) obtained from training the models on the entire data set. The right hand plots show the true noise variance (dashed blue curves) together with the estimated noise variance (solid red curves), again averaged over 100 data sets, and the best fit (solid green curves). The reason for this averaging is to obtain the bias and that is what we are interested in. The plots also show the variance of the regression and noise (solid cyan curves). We note that in the case of the maximum likelihood the noise variance $\sigma_{\nu}^{2}(x)$ is systematically underestimated while in the Bayesian case we see an improvement in the estimate of this quantity when compared to the latter approach. We can also see that the further away from the peak of the input density $p(x)$ which is located at $x = 0$ both the maximum likelihood and Bayesian estimates of the noise variance start to deviate from the true variance as a result of overfitting the data points which is due to the small amount of data present in these regions. However, in the Bayesian case this deviation is less compared to the maximum likelihood. Contrary to the difference in the quality of the estimates of the noise variance of the Bayesian and maximum likelihood approaches, we see from the left plots that the performance of these approaches are similar in predicting the regression function. This is because, unlike the estimate of the noise variance, the maximum likelihood estimate of the mean is not biased.

4.11.2 Experiment 2

In this section we will compare the performance of the Bayesian formalism which we have developed in this chapter for input-dependent noise with that of the Bayesian formalism for constant noise. Our aim is to see whether modelling the noise variance as a function of the inputs can actually lead to any improvement in the generalisation ability. To this end we shall use a data set arising from the monitoring of multi-phase flows in oil pipelines (Bishop and James 1993). A brief description of this data set is given next.

Oil flow data

Accurate determination of the flow of oil in pipelines is a matter of commercial interest to the oil industry. One particular method which has been widely used is dual energy gamma densitometry (Bishop and James 1993), which is based on the attenuation of gamma beams by oil, water and gas passing through pipe lines. Here two beams of gamma rays of different frequencies are passed through the same chord across the pipe and the intensity of the emerging beam is measured (see Figure 4.7). Then this information is used to determine the path length of the gamma rays in oil, water and gas. To determine the fraction of oil, water and gas, it is necessary to determine the configuration of these three phases. This can be achieved by an extension of the dual gamma densitometry (Bishop and James 1993). Here several dual gamma beams are sent through the pipeline in the manner shown in Figure 4.8, and then the information is used to determine the configuration of the three phases. Thus for each measure we have $2 \times d$ inputs together with one 1 output consisting of the fraction of oil.

The data set used in this experiment is generated by computer simulation using the technique described above and for a number of different phase configurations as shown in Figure 4.9. Attention has been made to simulation of noise on the inputs which is due to photon statistics which is therefore governed by a Poisson distribution. The noise process depends on the duration of time in which the gamma ray was sent as well as on the path length of the beam inside the phase fractions. The latter implies dependency of the noise level on the inputs. The noise on the inputs induces a noise on the targets, *i.e.* measurement of oil fractions. This will shift the observed targets from their true values according to

$$t = f(x + \epsilon) + \zeta(x) \tag{4.61}$$

where $\zeta(x)$ represents other sources of noise which we assume to have a Gaussian distribution with zero mean. From linear expansion of f around ϵ to first order we have

$$t \approx f(x) + \epsilon^T \frac{\partial f}{\partial x} + \zeta(x) \tag{4.62}$$

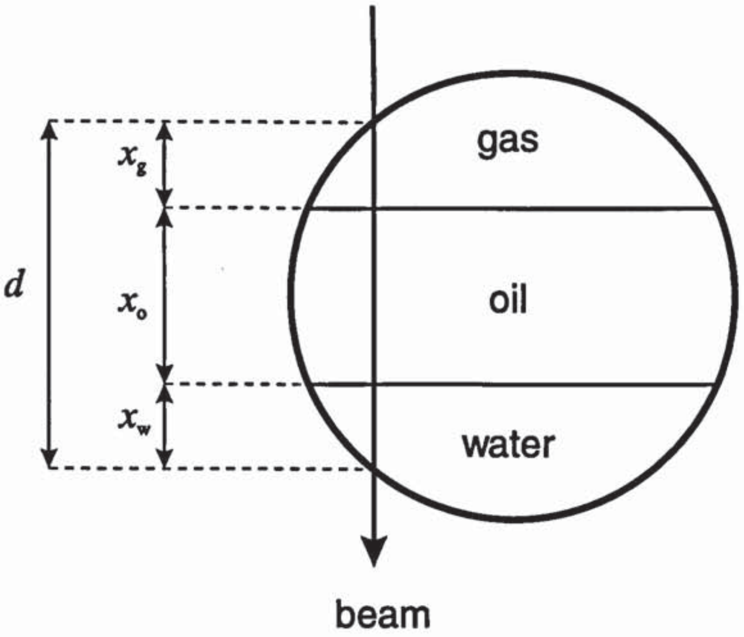

Figure 4.7: Cross section of a pipeline with oil, water and gas in a stratified configuration. Two gamma rays of different wave lengths are passed along the same line and the intensity of the emerging beams are measured from the photon counts. From the information obtained the path length of the gamma rays x_o, x_w and x_g in oil, water and gas can then measured.

Assuming that the ϵ is a Gaussian with zero mean then the total noise on the targets is also a Gaussian with zero mean and variance $\sigma_\nu^2(x)$ which depends on the inputs and so the Bayesian formalism of this chapter applies.

The experiment

Here we compare the generalisation ability of two models, one trained using Bayesian methods with constant noise variance and the other trained using the Bayesian formalism for input-dependent noise variance. In both cases the regression is obtained using a GLR model with 100 Gaussian basis functions with linear outputs[4] $y(x; w) = w^T\phi(x)$. The locations of the basis functions were randomly chosen from the data set (Lowe 1989) and the width of each basis is set equal to the average distance of the data points from its centre. This ensures that each basis function has most of the data within its domain of response. For the case of input-dependent noise, a GLR model with 100 basis Gaussian functions is used and the noise level is taken to be an exponential function of its output, *i.e.* $\beta(x; u) = \exp(u^T\psi(x))$. The location and width of the basis functions are chosen in the same way described above. To save programming effort and computational time the hyperparameters are estimated from (4.55) and (4.56) which makes no distinction between the number of well determined and actual

[4] Since the targets are fractions of oil within the 3 phase configuration, it is proper to use softmax output activation functions since it will ensure that the predicted oil fractions will remain within the closed interval $[0, 1]$ and also that the 3 phase fractions sum to 1. However, such procedures are not necessary here since our objective is only to compare the performance of two different methods under similar conditions.

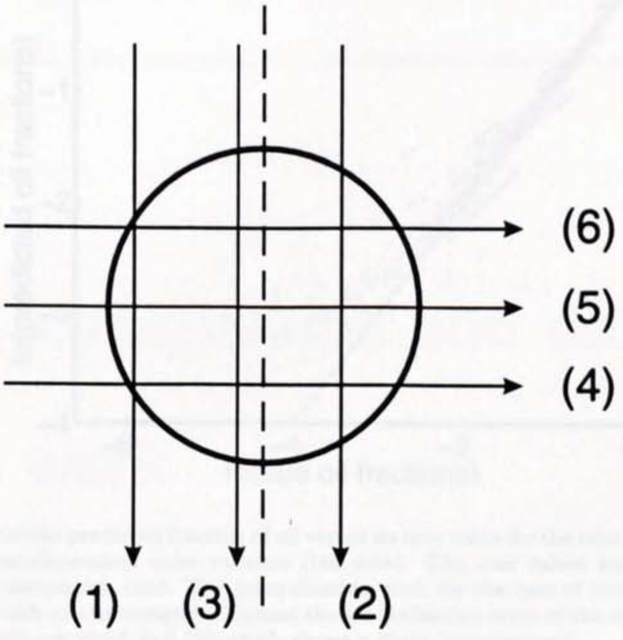

Figure 4.8: Arrangement of 12 gamma beams (2 on each cord) from which the oil flow data set is created.

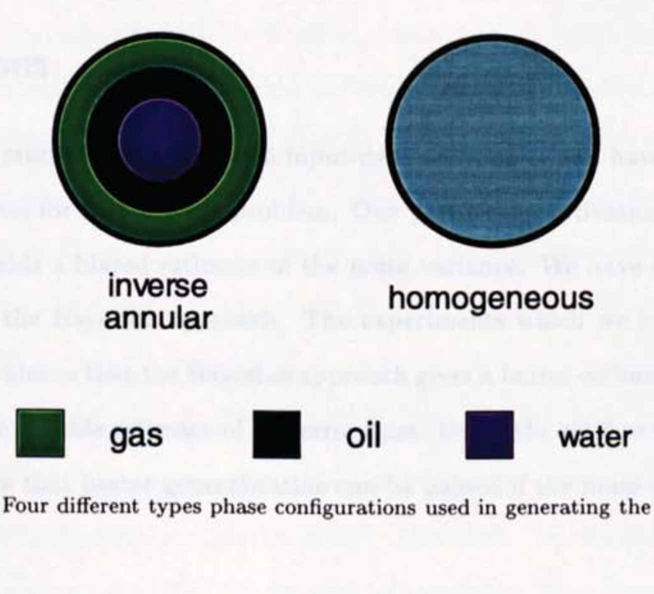

Figure 4.9: Four different types phase configurations used in generating the oil flow data.

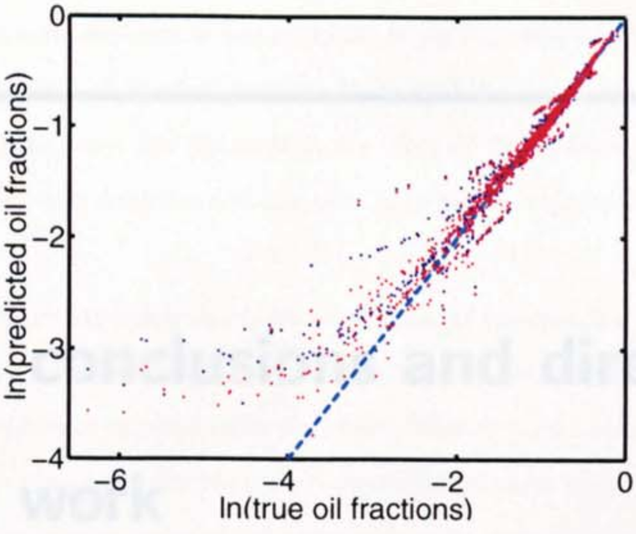

Figure 4.10: Log-log of the predicted fraction of oil versus its true value for the case of constant (blue dots) and input-dependent noise variance (red dots). The true values were obtained from long photon integration time. The generalisation error for the case of constant noise variance is 4.477 which can be compared against the generalisation error of the case of input-dependent noise variance which is 3.796 which shows a slight improvement.

number of parameters. The models were trained on a data set consisting of 1000 data points and then tested on a data set of the same size. The regression results are shown in Figure 4.10. The Bayesian approach of constant noise variance gives 2.755 for the log likelihood per test data point and 4.477 for the test error. This can be compared with the Bayesian approach of input-dependent noise variance which achieves 2.87 for the log likelihood per test data point and 3.796 for the test error. The predicted fractions of oil for both methods are shown in Figure 4.10. Thus we note that there is a small improvement in the generalisation ability of the model if noise variance is modelled as input-dependent.

4.12 Conclusions

In this chapter we have studied regression with input-dependent noise and have developed an approximate Bayesian formalism for tackling the problem. One particular motivation was the fact that the maximum likelihood yields a biased estimate of the noise variance. We have shown this bias can be compensated for using the Bayesian approach. The experiments which we have carried out in this chapter provide some evidence that the Bayesian approach gives a better estimate of the noise variance and hence yields a more reliable estimate of the error bars. Using the oil flow data we have provided some numerical evidence that better generalisation can be gained if the noise variance is modelled as input-dependent.

Chapter 5

Summary, conclusions and directions for future work

5.1 Summary of thesis and conclusions

Bayesian error bars for regression have been the focal point of this thesis. As we saw in Chapter 1, the error bars arise naturally as a result of i) the existence of intrinsic noise on the targets and ii) the uncertainty in the model parameters due to the finite amount of data in the training set.

As an important tool for implementing the Bayesian machinery in general and estimating the error bars in particular, in Chapter 2 we reviewed and studied both exact and approximate ways of evaluation of the Hessian matrix. Our key finding was that there can be appreciable differences between the small eigenvalues of the exact Hessian matrix and its outer product approximation for neural networks. From this we concluded that this approximation can give significant error in estimating quantities which depend on the product of the eigenvalues. In conclusion we found that the best ways of evaluating the Hessian matrix, both in terms of accuracy and computational efficiency, are offered by exact methods.

In chapter 3 we studied the properties of the prediction variance $\sigma_t^2(x)$ in relation to the distribution of input data using two complementary approaches, one based on consideration of discrete data sets and the other based on continuous probability density functions. In the first approach we showed that the prediction variance measured at the location of the data points cannot be larger than twice the value of the variance of the intrinsic noise on the targets. This has led us to the conclusion that accurate evaluation of the noise variance is important for obtaining reliable estimate of the error bars (This result prompted the work of Chapter 4 on the Bayesian inference of an input-dependent noise variance). More generally, we have also shown that the effect of the addition of a data point to the training set can only reduce the magnitude of the error bars anywhere in the input space or leave it unchanged.

Given the above results on the behaviour of the local value of the error bars, we have also studied the global averages of the error bars. One finding was that the average value of the error bars depends on the number of well determined parameters in the model. However, we have demonstrated that this does not necessarily imply that more flexible models posses larger error bars. We have also derived a bound on the change in the average value of the error bars as a result of the addition of a data point to the training set. These findings, which were based on the discrete data set approach, also apply to neural networks provided that we make the Gaussian approximation to the posterior and linearise the outputs in the vicinity of the most probable value of the weights.

Based on a consideration of continuous probability distribution functions, we have provided both theoretical and experimental evidence that for GLR models the confidence error bars exhibit an approximate inverse relation of the form $\sigma_w(x) \propto p^{-1/2}(x)$, where $p(x)$ is the density of the input data, in regions of input space where the data density is high. We also provided some numerical evidence that a similar relation holds for trained neural networks. However, it must be mentioned that the inverse proportionality relation between the density of input data can hold only in the regions of high density. In such regions, however, the prediction variance is dominated by the noise variance.

In Chapter 4 we developed a Bayesian formalism, using the evidence framework, for tackling regression problems with input-dependent noise variance. Using a toy data set we have demonstrated that this method can yield an improved estimate of noise variance compared to the maximum likelihood approach. We have also shown, using the oil flow data, that improved generalisation error can be gained if the noise variance is modelled as input-dependent. However, implementing this approach has proven to be computationally more expensive than maximum likelihood as it requires evaluation of the Hessian matrices $\boldsymbol{A}$ and $\boldsymbol{H}$. However, this should not deter us from using this Bayesian formalism if we have reason to believe that the noise variance depends significantly on the inputs values.

In the Bayesian formalism developed in Chapter 4 we fixed the weights $\boldsymbol{u}$, as well as the hyperparameters α_w and α_u and carried out the rest of Bayesian inference with these parameters fixed to their most probable values. This was done on the grounds that integration over these parameters can render the Gaussian approximation over $\boldsymbol{w}$ from being a good representative of the unconditional posterior distribution $p(\boldsymbol{w}|D)$. However, it should be noted that fixing the weights $\boldsymbol{u}$ to the most probable value is not exactly like fixing the hyperparameters. The reason is that $\boldsymbol{u}$ is multi-dimensional vector with possibly as many components as the regression weights $\boldsymbol{w}$. Although such an approach can still yield an unbiased estimate of the noise variance, it also excludes the posterior for $\boldsymbol{u}$ from influencing predictions. The width of the posterior over $\boldsymbol{u}$ will make an additional contribution to the predictive error bars. Similarly, there will also be contributions arising from the finite width of the posterior over the hyperparameters α_w and α_u. If desired these additional contributions could be evaluated using the formalism of Chapter 4.

5.2 What if the evidence framework fails?

As mentioned in the introductory Chapter 1 what matters ultimately is how accurate are the predictions. Usually the evidence approximation is thought to give reasonable performance when the size of the data set is large compared to the size of the model. However, it has been recently argued by Neal (1995) that, except for computational reasons, we need not restrict the size of the model. This is a point for concern when using the evidence procedure, as increase in the size of the model will eventually lead to the breakdown of the Gaussian approximation. In cases like this one can use Markov chain Monte Carlo techniques.

5.3 Directions for future work

There are many directions in which future work can be carried out.

1) On the issue of noise

The evidence framework of MacKay and its extension of Chapter 4 to the case of input-dependent noise assumes Gaussian noise on the targets. It is also interesting to extend the evidence procedure to cases where the noise is non-Gaussian. A parametric approach would be to model the noise variance using a non-Gaussian exponential family distribution such as gamma or chi-square distributions. Alternatively, one can use non-parametric methods such as Gaussian mixture models (Bishop 1994a) which was mentioned in Section 1.7.1.

Another issue relates to the presence of noise on the *input* data. We have already seen an example of a data set with noisy inputs in Section 4.11.2 where we dealt with the oil flow data. To tackle this problem we used the approximation that the noise on the inputs induces an additive noise on the targets which can be approximated by a Gaussian and so the standard Bayesian formalism can be applied. However, such approximation will be in significant error if the magnitude of noise is large. The problem of noisy inputs has already been investigated by some authors (Bishop 1995b; Matsuoka 1992) but a Bayesian treatment of this problem is yet to be carried out.

2) Multiple outputs with correlated noise

The evidence formalism of Chapter 4 is applicable to the case of m-dimensional outputs. In this case the intrinsic noise on the targets will be, in general, different functions of the inputs and so m auxiliary outputs are required to infer the noise variances as functions of the inputs, assuming that the noise on the targets is not correlated. The more general case of correlated noise has been studied by Williams (1996) using the maximum likelihood approach. It would be interesting to extend the Bayesian formalism to this case.

3) Markov chain Monte Carlo methods

In this thesis we have used the formula $\boldsymbol{g}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\boldsymbol{g}(\boldsymbol{x})$ for evaluating the error bars, which is exact for GLR models. For neural networks, however, it is an approximation depending on the assumptions that i) the posterior weight distribution can be approximated by a Gaussian and ii) the outputs depend linearly on the weights in the vicinity of $\widetilde{\boldsymbol{w}}$. The validity of these approximations is doubtful for neural networks whose number of parameters is comparable to or larger than the size of the training data. It would be interesting to study the property of the error bars using the full Bayesian formalism which can be obtained using sampling methods.

Appendix A

A.1 Experimental details

In this appendix we shall give further details of the experiments carried out in this thesis. The experiments involving GLR models made use of Gaussian, sigmoid and *tanh* basis functions as well as polynomials of various degrees. The procedure for fixing the centres of the basis functions made use of a rectangular gridding system over the range of the input data, using the outermost data points to define the coordinates of the rectangle. Once the rectangle was defined the basis functions were then uniformly distributed (see Figure A.1) inside the rectangle and the width of the basis functions were set equal to

$$\sigma = \frac{\Delta x}{n} \tag{A.1}$$

were n^d is the number of basis functions, d is the dimensionality of input space and $\Delta x = (\Delta x_1, ..., \Delta x_d)$ is a vector with elements

$$\Delta x_j = |x_j^{\mathrm{max}} - x_j^{\mathrm{min}}| \tag{A.2}$$

Note that $|x_j^{\mathrm{max}} - x_j^{\mathrm{min}}|$ is the longest distance between the data points in the direction of the jth input. In the case of sigmoid and *tanh* basis functions, which were used only in the experiments involving 1-dimensional input data, σ should be understood as the steepness of the basis functions. The experiment of Section 4.11.2 used a different procedure for fixing the centres and width of the basis functions. This will be discussed shortly. It should also be noted that in those experiments which involved more than one data set, the basis functions were fixed separately for each data set. With regard to the experiments of Section 3.4.3, the input data set was generated independently from

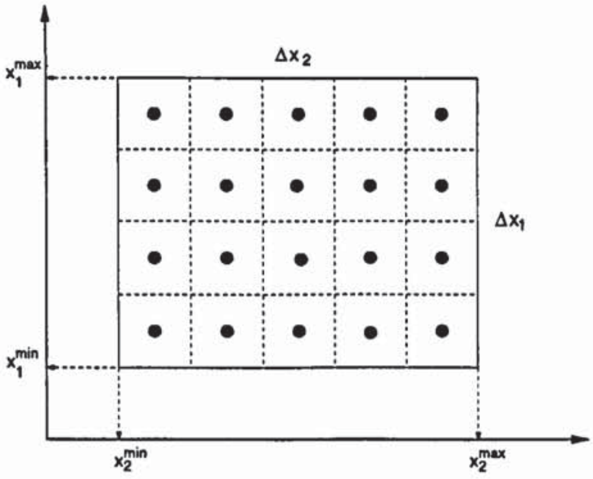

Figure A.1: Schematic diagram for fixing the locations of the basis functions for a 2-dimensional input data. The outermost data points where used to define the coordinates of the rectangle and the vector $\Delta \boldsymbol{x} = (\Delta x_1, \Delta x_2)$ (see equations (A.1) and (A.2)). The basis function were then uniformly located (solid circles) on the regular grid.

a Gaussian probability density function with zero mean and unit standard deviation and the basis functions were fixed according to the gridding procedure described above. The exact specifications of the GLR basis functions used in the experiments of Figures 3.6 , 3.7 and 3.9 are displayed in Table A.2. Specifications of the basis functions for the experiments of Figures 3.3, 3.8 and 4.1 are given in Tables A.1, A.3 and A.4.

The experiments discussed so far have been based on a finite data set drawn from a density function, in which the Hessian is evaluated numerically as a finite sum. For the particular choice of Gaussian basis functions, however, it is possible to evaluate the Hessian matrix analytically using the continuous density representation of Section 3.4.2. In this case, expression (3.35) for the Hessian becomes the convolution of Gaussian functions, provided that the density $p(\boldsymbol{x})$ is Gaussian too, which is easily evaluated. Let $\boldsymbol{\mu}_i$, $\boldsymbol{\mu}_j$ and $\boldsymbol{\mu}_p$ be the means of the ith, jth basis functions and the density $p(\boldsymbol{x})$, respectively. Also let $\boldsymbol{C}_i$, $\boldsymbol{C}j$ and $\boldsymbol{C}_p$ be their variance matrices[1]. Using formula (3.35)[2], we can write a typical element of the Hessian as

$$\begin{aligned} A_{ij} &= \int p(\boldsymbol{x})\phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})\, d\boldsymbol{x} \\ &= \frac{1}{(2\pi)^{d/2}|\boldsymbol{C}_p|^{1/2}} \int \Big[\exp\Big[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_p)^T \boldsymbol{C}_p^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_p)\Big] \exp\Big[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T \boldsymbol{C}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\Big] \\ &\qquad \exp\Big[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^T \boldsymbol{C}_j^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_j)\Big]\, d\boldsymbol{x}\Big] \end{aligned} \tag{A.3}$$

[1] These matrices are diagonal with elements $diag(\sigma_1^2, ..., \sigma_d^2)$, where d is the dimensionality of the input space (see equation (A.1)).

[2] Note that we have dropped the factor βN in the right hand side of this equation since it plays only the role of a scaling factor.

where d is the dimensionality of the input vector $\boldsymbol{x}$. After some simple algebra we obtain

$$A_{ij} = \frac{1}{(2\pi)^{d/2}|C_p|^{1/2}} \exp\Big[-\frac{1}{2}\, a\Big] \int \exp\Big[-\frac{1}{2}\, \boldsymbol{x}^T C^{-1} \boldsymbol{x}\Big] \exp\Big[\boldsymbol{\mu}^T \boldsymbol{x}\Big]\, d\boldsymbol{x} \qquad \text{(A.4)}$$

where

$$C^{-1} = C_p^{-1} + C_i^{-1} + C_j^{-1} \qquad \text{(A.5)}$$

$$\boldsymbol{\mu} = C_p^{-1}\boldsymbol{\mu}_p + C_i^{-1}\boldsymbol{\mu}_i + C_j^{-1}\boldsymbol{\mu}_j \qquad \text{(A.6)}$$

$$a = \boldsymbol{\mu}_p^T C_p^{-1}\boldsymbol{\mu}_p + \boldsymbol{\mu}_i^T C_i^{-1}\boldsymbol{\mu}_i + \boldsymbol{\mu}_j^T C_j^{-1}\boldsymbol{\mu}_j \qquad \text{(A.7)}$$

By integrating (A.4) we obtain our final result

$$A_{ij} = \frac{|C|^{1/2}}{|C_p|^{1/2}} \exp\Big[-\frac{1}{2}\, a\Big] \exp\Big[\frac{1}{2}\, \boldsymbol{\mu}^T C \boldsymbol{\mu}\Big] \qquad \text{(A.8)}$$

Using (A.8) the average slope of $\ln \sigma_w^2(\boldsymbol{x})$ versus $\ln p(\boldsymbol{x})$ curve is then calculated by taking 10 samples of size 5000 each from the true distribution $p(\boldsymbol{x})$. Linear regression is used to determine the slope, taking into account only points for which $p(\boldsymbol{x}) > 0.1 p_{max}(\boldsymbol{x})$. The reason for this arbitrary cut-off is to reduce the influence of those data points which belong to low regions of input data density on the measurement of the slope. This is done for two kinds of reasons. First, we are interested in the regions of high input data density, and second the inclusion of these isolated data points make the measurements of the slope noisy. This step is then repeated for various number of basis functions and for data sets of input dimensions 1 and 2, and the results are shown in Figure 3.12. It should also be mentioned that as the number of basis functions was increased the Hessian matrix started to show ill-conditioning. This problem was solved by adding a regularising term to the Hessian with its parameter α set equal to 10^{-10}.

In the experiments we have discussed so far the locations of the basis functions were fixed using the gridding system described earlier in this appendix. This method is particularly suitable for data sets with uncorrelated inputs and low input dimensionality. However, the experiment of Section 4.11.2 involved the oil flow data with 12 correlated inputs. In this case 100 data points where selected in random from the training data and used to define the location of the 100 basis functions (Lowe 1989). The width of each basis functions is then set equal to the average distance between its location and the data points in the training set. An illustration of the location of the basis functions for this experiment is shown in Figure A.2.

$\mu_2 = -2.8865$	$\mu_3 = -2.7768$	$\mu_4 = -2.6671$	$\mu_5 = -2.5573$	$\mu_6 = -2.4476$	$\mu_7 = -2.3379$
$\mu_8 = -2.2281$	$\mu_9 = -2.1184$	$\mu_{10} = -2.0087$	$\mu_{11} = -1.8989$	$\mu_{12} = -1.7892$	$\mu_{13} = -1.6795$
$\mu_{14} = -1.5697$	$\mu_{15} = -1.4600$	$\mu_{16} = -1.3503$	$\mu_{17} = -1.2405$	$\mu_{18} = -1.1308$	$\mu_{19} = -1.0211$
$\mu_{20} = -0.9113$	$\mu_{21} = -0.8016$	$\mu_{22} = -0.6919$	$\mu_{23} = -0.5821$	$\mu_{24} = -0.4724$	$\mu_{25} = -0.3627$
$\mu_{26} = -0.2530$	$\mu_{27} = -0.1432$	$\mu_{28} = -0.0335$	$\mu_{29} = 0.0762$	$\mu_{30} = 0.1860$	$\mu_{31} = 0.2957$
$\mu_{32} = 0.4054$	$\mu_{33} = 0.5152$	$\mu_{34} = 0.6249$	$\mu_{35} = 0.7346$	$\mu_{36} = 0.8444$	$\mu_{37} = 0.9541$
$\mu_{38} = 1.0638$	$\mu_{39} = 1.1736$	$\mu_{40} = 1.2833$	$\mu_{41} = 1.3930$	$\mu_{42} = 1.5028$	$\mu_{43} = 1.6125$
$\mu_{44} = 1.7222$	$\mu_{45} = 1.8320$	$\mu_{46} = 1.9417$	$\mu_{47} = 2.0514$	$\mu_{48} = 2.1612$	$\mu_{49} = 2.2709$
$\mu_{50} = 2.3806$					

Table A.1: Table of the locations of the basis functions of the experiments displayed in Figure 3.3. The width of the basis functions were set equal to 0.1075.

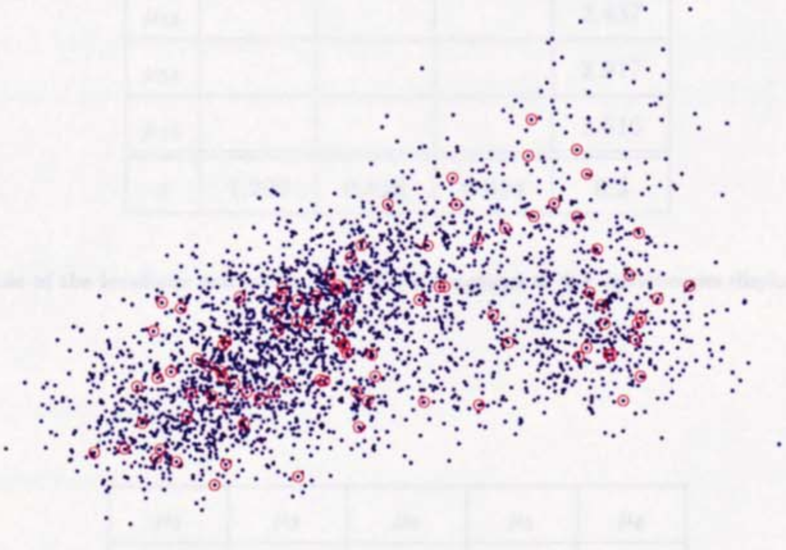

Figure A.2: Scatter plot of two inputs of the training data together with the locations (indicated by the circles) of the Gaussian basis functions used in the experiment of Section 4.11.2.

μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	μ_8
-3.2263	-2.1474	-1.0686	0.0103	1.0892	2.1680	3.2469

Table A.2: Table óf the locations of the basis functions of the experiments displayed in Figures 3.6, 3.7 and 3.9. The width of the basis functions were set equal to 0.9247.

k	5	8	11	14
μ_2	−3.010	−3.293	−3.422	−3.495
μ_3	−1.5	−2.349	−2.735	−2.956
μ_4	0.010	−1.405	−2.049	−2.417
μ_5	1.520	−0.461	−1.362	−1.877
μ_6	3.031	0.482	−0.676	−1.338
μ_7		1.426	0.010	−0.798
μ_8		2.370	0.693	−0.259
μ_9		3.314	1.383	0.280
μ_{10}			2.069	0.819
μ_{11}			2.756	1.358
μ_{12}			3.443	1.898
μ_{13}				2.437
μ_{14}				2.977
μ_{15}				3.516
σ	1.208	0.826	0.624	0.5

Table A.3: Table of the locations and width of the basis functions of the experiments displayed in Figure 3.8.

μ_2	μ_3	μ_4	μ_5	μ_6
-2.4037	-1.3283	-0.2530	0.8224	1.8978

Table A.4: Table of the locations of the basis functions of the experiments displayed in Figure 4.1. The width of the basis functions were set equal to 0.8603.

Appendix B

B.1 Average value of the prediction variance

Here we derive an expression for the average value of the prediction variance $\sigma_t^2(x)$ defined over the training data. Using formula (1.55) and (1.57) we have

$$\begin{aligned} \sigma_t^2(x) &= \sigma_\nu^2 + \sigma_w^2(x) \\ &= \sigma_\nu^2 + g^T(x) A^{-1} g(x) \end{aligned} \tag{B.1}$$

where $g(x) = \partial y(x;w)/\partial w$ is the derivatives of the outputs and A is the Hessian matrix, both measured at the most probable value of the weights $\widetilde{w}$. For GLR models (B.1) is exact while for neural networks it is just an approximation based on the Gaussian approximation for the posterior and the linearization of the outputs in the vicinity of $\widetilde{w}$. Taking the average of (B.1) we have

$$\begin{aligned} \langle \sigma_t^2(x) \rangle &= \sigma_\nu^2 + \frac{1}{N} \sum_{i=1}^{N} \sigma_w^2(x_i) \\ &= \sigma_\nu^2 + \frac{1}{N} \sum_{i=1}^{N} g^T(x_i) A^{-1} g(x_i) \end{aligned} \tag{B.2}$$

Using the rule

$$v^T M v = \text{Trace}(M v v^T) \tag{B.3}$$

where v is column vector, in (B.2) we obtain

$$\begin{aligned} \langle \sigma_t^2(x) \rangle &= \sigma_\nu^2 + \frac{1}{N} \sum_{i=1}^{N} \text{Trace}\Big(A^{-1} g(x_i) g^T(x_i) \Big) \\ &= \sigma_\nu^2 + \frac{1}{N} \text{Trace}\Big(A^{-1} \sum_{i=1}^{N} g(x_i) g^T(x_i) \Big) \\ &= \sigma_\nu^2 + \frac{1}{N} \text{Trace}\Big(A^{-1} B \Big) \end{aligned} \tag{B.4}$$

where $\boldsymbol{B}$ is the data Hessian matrix as given by the outer product approximation (2.8). We now make use of the relation

$$\boldsymbol{B} = \sigma_\nu^2 (\boldsymbol{A} - \boldsymbol{C}) \tag{B.5}$$

in (B.4) and take $\boldsymbol{C} = \alpha \boldsymbol{I}$, to obtain

$$\begin{aligned}
\langle \sigma_t^2(\boldsymbol{x}) \rangle &= \sigma_\nu^2 + \frac{\sigma_\nu^2}{N} \mathrm{Trace}\Big(\boldsymbol{I} - \alpha \boldsymbol{A}^{-1})\Big) \\
&= \sigma_\nu^2 + \frac{\sigma_\nu^2}{N}\Big(k - \alpha \mathrm{Trace}(\boldsymbol{A}^{-1})\Big) \\
&= \sigma_\nu^2 + \frac{\sigma_\nu^2 \gamma}{N}
\end{aligned} \tag{B.6}$$

where we have used the expression $k - \alpha\mathrm{Trace}(\boldsymbol{A}^{-1})$ for the number of well determined parameters γ.

B.2 Average change in the value of prediction variance

We will derive a closed bound on the average change in the value of the prediction variance as a result of the addition of a data point at $\bar{\boldsymbol{x}}$ to the training set. Let $\sigma_t^2(\boldsymbol{x})$ be the prediction variance for a model trained on a data set of size N. Also let $\bar{\sigma}_t^2(\boldsymbol{x})$ be the prediction variance after the introduction of the new data point. Then we can write the change in the prediction variance as

$$\Delta\sigma_t^2(\boldsymbol{x}) = \bar{\sigma}_t^2(\boldsymbol{x}) - \sigma_t^2(\boldsymbol{x}) \tag{B.7}$$

From (3.23) we have the upper bound $\Delta\sigma^2(\boldsymbol{x}) \leq 0$ for all $\boldsymbol{x}$. If we take the average of (B.7) over the training data and use (3.20), we have

$$\begin{aligned}
\langle \Delta\sigma_t^2(\boldsymbol{x}) \rangle &= \langle \boldsymbol{g}^T(\boldsymbol{x})\bar{\boldsymbol{A}}^{-1}\boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{g}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\boldsymbol{g}(\boldsymbol{x}) \rangle \\
&= \langle \boldsymbol{g}^T(\boldsymbol{x})(\boldsymbol{A} + \boldsymbol{g}(\bar{\boldsymbol{x}})\boldsymbol{g}^T(\bar{\boldsymbol{x}}))^{-1}\boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{g}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\boldsymbol{g}(\boldsymbol{x}) \rangle
\end{aligned} \tag{B.8}$$

where $\langle . \rangle = \frac{1}{N}\sum_{i=1}^{N}$. Using the matrix identity (3.12), we have

$$\begin{aligned}
\langle \Delta\sigma_t^2(\boldsymbol{x}) \rangle &= -\frac{\langle \boldsymbol{g}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\boldsymbol{g}(\bar{\boldsymbol{x}})\boldsymbol{g}^T(\bar{\boldsymbol{x}})\boldsymbol{A}^{-1}\boldsymbol{g}(\boldsymbol{x}) \rangle}{1 + \boldsymbol{g}^T(\bar{\boldsymbol{x}})\boldsymbol{A}^{-1}\boldsymbol{g}^T(\bar{\boldsymbol{x}})} \\
&= -\frac{\langle \boldsymbol{g}^T(\bar{\boldsymbol{x}})\boldsymbol{A}^{-1}\boldsymbol{g}(\boldsymbol{x})\boldsymbol{g}^T(\boldsymbol{x})\boldsymbol{A}^{-1}\boldsymbol{g}(\bar{\boldsymbol{x}}) \rangle}{1 + \boldsymbol{g}^T(\bar{\boldsymbol{x}})\boldsymbol{A}^{-1}\boldsymbol{g}^T(\bar{\boldsymbol{x}})} \\
&= -\frac{\boldsymbol{g}^T(\bar{\boldsymbol{x}})\boldsymbol{A}^{-1}\langle \boldsymbol{g}(\boldsymbol{x})\boldsymbol{g}^T(\boldsymbol{x}) \rangle \boldsymbol{A}^{-1}\boldsymbol{g}^T(\bar{\boldsymbol{x}})}{1 + \boldsymbol{g}^T(\bar{\boldsymbol{x}})\boldsymbol{A}^{-1}\boldsymbol{g}(\bar{\boldsymbol{x}})} \\
&= -\frac{1}{N}\,\frac{\boldsymbol{g}^T(\bar{\boldsymbol{x}})\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{A}^{-1}\boldsymbol{g}(\bar{\boldsymbol{x}})}{1 + \boldsymbol{g}^T(\bar{\boldsymbol{x}})\boldsymbol{A}^{-1}\boldsymbol{g}^T(\bar{\boldsymbol{x}})}
\end{aligned} \tag{B.9}$$

where we have used $\langle g(x)g^T(x)\rangle = B/N$. Using (B.5) in (B.9) we obtain

$$\begin{aligned}
\langle \Delta\sigma_t^2(x)\rangle &= -\frac{\sigma_\nu^2}{N}\frac{g^T(\bar{x})A^{-1}(A-C)A^{-1}g^T(\bar{x})}{1+g^T(\bar{x})A^{-1}g^T(\bar{x})} \\
&= -\frac{\sigma_\nu^2}{N}\frac{g^T(\bar{x})A^{-1}g(\bar{x}) - g^T(\bar{x})A^{-1}CA^{-1}g^T(\bar{x})}{1+g^T(\bar{x})A^{-1}g^T(\bar{x})} \\
&= -\frac{\sigma_\nu^2}{N}\frac{\sigma_w^2(\bar{x}) - g^T(\bar{x})A^{-1}CA^{-1}g^T(\bar{x})}{1+g^T(\bar{x})A^{-1}g^T(\bar{x})} \qquad \text{(B.10)}
\end{aligned}$$

Since $g^T(\bar{x})A^{-1}CA^{-1}g^T(\bar{x})$ is positive semi-definite, we can omit it to obtain

$$\langle \Delta\sigma_t^2(x)\rangle \geq -\frac{\sigma_\nu^2}{N}\frac{\sigma_w^2(\bar{x})}{1+\sigma_w^2(\bar{x})} \qquad \text{(B.11)}$$

Finally, we note that $\sigma_w^2(x)$ is a positive quantity and so the ratio $\sigma_w^2(\bar{x})/(1+\sigma_w^2(\bar{x}))$ lies in the interval $(0,1)$. Hence, we can also obtain a simplified lower bound given by

$$\langle \Delta\sigma_t^2(x)\rangle \geq -\frac{\sigma_\nu^2}{N} \ . \qquad \text{(B.12)}$$

Appendix C

C.1 Prior probability distribution of u

In the Bayesian formalism of Chapter 4 for inferring an input-dependent noise we imposed a zero mean Gaussian prior on the weights $\boldsymbol{u}$. The motivation behind such a prior was due to the fact that inferring noise is essentially a regression problem and so we require the outputs $\beta(\boldsymbol{x}_i;\boldsymbol{u}) = \sigma_\nu^{-2}(\boldsymbol{x};\boldsymbol{u})$ to be smooth. However, since the weights $\boldsymbol{u}$ do not have a direct physical meaning the quality of the prior for these weights should be judged upon from considering the quality of the prior $p(\sigma_\nu^2)$ that it induces on the noise variance. It turns out that a zero mean Gaussian prior will result in a prior on the noise variance that is not flexible enough in the sense that its mean and variance can only vary in the range $[1,\infty]$. Ideally, however, we would like the prior $p(\sigma_\nu^2)$ to allow its mean and variance assume any values in the interval $[0,\infty]$. This problem can be overcome by introducing a Gaussian prior with its mean centred at $\boldsymbol{u}_o$. Thus

$$p(\boldsymbol{u}|\boldsymbol{u}_o,\alpha_u) = \Big(\frac{\alpha_u}{2\pi}\Big)^{k_u/2} \exp\Big(-\frac{\alpha_u}{2}\,(\boldsymbol{u}-\boldsymbol{u}_o)^T(\boldsymbol{u}-\boldsymbol{u}_o)\Big) \tag{C.1}$$

where k_u is the length of the vector $\boldsymbol{u}$. To see why (C.1) is a better prior let us evaluate the corresponding prior for the noise variance. We write[1]

$$p(\sigma_\nu^2) = \int \delta\Big(\sigma_\nu^2 - \exp(-\boldsymbol{u}^T\psi(\boldsymbol{x}))\Big)p(\boldsymbol{u}|\boldsymbol{u}_o,\alpha_u)\,d\boldsymbol{u} \tag{C.2}$$

where δ is the Dirac delta function. Using the transformation of variables $\widehat{\boldsymbol{u}} = \boldsymbol{u}-\boldsymbol{u}_o$ and $\gamma = \exp(-\boldsymbol{u}_o\psi(\boldsymbol{x}))$ together with formula (C.1), we have

$$p(\sigma_\nu^2) = \Big(\frac{\alpha}{2\pi}\Big)^{k_u/2} \int \delta\Big(\sigma_\nu^2 - \gamma\exp(-\widehat{\boldsymbol{u}}^T\psi(\boldsymbol{x}))\Big)\exp\Big(-\frac{\alpha_u}{2}\,\widehat{\boldsymbol{u}}^T\widehat{\boldsymbol{u}}\Big)\,d\widehat{\boldsymbol{u}} \tag{C.3}$$

[1] This is applicable only to GLR models.

Note that the delta function argument depends on the scalar product $\widehat{\boldsymbol{u}}^T\psi(\boldsymbol{x})$ which in turn depends on the magnitude and relative orientation of the two vectors $\widehat{\boldsymbol{u}}^T$ and $\psi(\boldsymbol{x})$. Thus without loss of generality we can write $\psi(\boldsymbol{x}) = (0,0,|\psi(\boldsymbol{x})|,0,0,.....)$, where $|\psi(\boldsymbol{x})| = \sqrt{\psi^T(\boldsymbol{x})\psi(\boldsymbol{x})}$, which implies that $\widehat{\boldsymbol{u}}^T\psi(\boldsymbol{x}) = \widehat{u}_i|\psi(\boldsymbol{x})|$. Note that $\widehat{u}_i$ is an arbitrary component of $\widehat{\boldsymbol{u}}$. This means that we can integrate out the non-interacting components, after this coordinate transformation, to give

$$p(\sigma_\nu^2) = \Big(\frac{\alpha_u}{2\pi}\Big)^{1/2}\int \delta\Big(\sigma_\nu^2 - \gamma\exp(-\widehat{u}_i|\psi(\boldsymbol{x})|)\Big)\exp\Big(-\frac{\alpha_u}{2}\,\widehat{u}_i^2\Big)\,d\widehat{u}_i \tag{C.4}$$

Using the transformation $t = \gamma\exp\Big(-\widehat{u}_i|\psi(\boldsymbol{x})|\Big)$ in the above result we obtain

$$p(\sigma_\nu^2) = \Big(\frac{\alpha_u}{2\pi}\Big)^{1/2}\frac{1}{|\psi(\boldsymbol{x})|}\int \delta(\sigma_\nu^2 - t)\,\frac{1}{t}\exp\Big(-\frac{\alpha_u}{2|\psi(\boldsymbol{x})|^2}(\ln t - \ln\gamma)^2\Big)\,dt \tag{C.5}$$

which integrates to

$$p(\sigma_\nu^2) = \Big(\frac{\alpha_u}{2\pi}\Big)^{1/2}\frac{1}{\sigma_\nu^2|\psi(\boldsymbol{x})|}\exp\Big(-\frac{\alpha_u}{2|\psi(\boldsymbol{x})|^2}\,(\ln\sigma_\nu^2 + \boldsymbol{u}_o^T\psi(\boldsymbol{x}))^2\Big) \tag{C.6}$$

Formula (C.6) is the prior probability distribution for the noise variance σ_ν^2 which has the form of a log normal distribution. To compute the prior mean and variance of the noise we use

$$\langle\sigma_\nu^2\rangle = \int \exp(-\boldsymbol{u}^T\psi(\boldsymbol{x}))p(\boldsymbol{u}|\boldsymbol{u}_o,\alpha_u)\,d\boldsymbol{u} \tag{C.7}$$

and

$$\begin{aligned}\langle(\sigma_\nu^2 - \langle\sigma_\nu^2\rangle)^2\rangle &= \langle(\sigma_\nu^2)^2\rangle - \langle\sigma_\nu^2\rangle^2\\ &= \int \exp(-2\boldsymbol{u}^T\psi(\boldsymbol{x}))p(\boldsymbol{u}|\boldsymbol{u}_o,\alpha_u)\,d\boldsymbol{u}\\ &\quad -\Big(\int \exp(-\boldsymbol{u}^T\psi(\boldsymbol{x}))p(\boldsymbol{u}|\boldsymbol{u}_o,\alpha_u)\,d\boldsymbol{u}\Big)^2\end{aligned} \tag{C.8}$$

Again using the transformation $\widehat{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_o$ together with (C.1), it is straight forward to show that the mean and variance satisfy

$$\langle\sigma_\nu^2\rangle = \exp\Big(-\boldsymbol{u}_o^T\psi(\boldsymbol{x})\Big)\exp\Big(\frac{|\psi(\boldsymbol{x})|^2}{2\alpha_u}\Big) \tag{C.9}$$

$$\langle(\sigma_\nu^2 - \langle\sigma_\nu^2\rangle)^2\rangle = \exp\Big(-2\boldsymbol{u}_o^T\psi(\boldsymbol{x})\Big)\Big[\exp\Big(\frac{2|\psi(\boldsymbol{x})|^2}{\alpha_u}\Big) - \exp\Big(\frac{|\psi(\boldsymbol{x})|^2}{\alpha_u}\Big)\Big] \tag{C.10}$$

Note that in the special case of a zero mean Gaussian prior $p(\boldsymbol{u}|\alpha)$ the mean and variance of $p(\sigma_\nu^2)$ ranges between $[1,\infty]$ as α_u varies from $[0,\infty]$. This in contrast to the more general case of $\boldsymbol{u}_o \neq 0$ where the mean can assume any value in the range $[0,\infty]$.

C.1.1 Fixing $\boldsymbol{u}_o$

While the introduction of a non-zero mean Gaussian prior leads to a more meaningful prior for the noise variance it also introduces further complications into the evidence framework which we have developed as we have now an extra vector of parameters which we need to fix. Here is a suggestion for dealing with this complication.

Our task is to choose a value for $\boldsymbol{u}_o$ which reflects our knowledge of the mean and variance of the noise. We first note from (C.6) that the prior variance depends on the form of the basis functions. This implies that different types of basis functions yield different priors. This is unfortunate since if the basis functions of the model are changed so will the prior. Ideally we would prefer the prior to be dictated by our prior knowledge rather than the type of the basis functions of the GLR model in use. Unfortunately we can not make the prior independent of the basis functions but what we can do is to make its mean and variance independent of the basis functions at selected points of input space. One difficulty in setting the prior mean and variance of noise is due to the fact that $\boldsymbol{u}_o$ does not have a physical interpretation which makes it hard to be fixed by hand. These two complications can be overcome in the following manner. If we use the transformation

$$\widehat{\boldsymbol{\psi}}(\boldsymbol{x}) = \frac{1}{|\boldsymbol{\psi}(\boldsymbol{x})|}\,\boldsymbol{\psi}(\boldsymbol{x}) \tag{C.11}$$

then the new set of basis functions $\widehat{\boldsymbol{\psi}}(\boldsymbol{x})$ become normalised in the sense that $|\widehat{\boldsymbol{\psi}}(\boldsymbol{x})|^2 = 1$. In this case the prior mean and variance become

$$\langle\sigma_\nu^2\rangle = \exp\Big(-\boldsymbol{u}_o^T\widehat{\boldsymbol{\psi}}(\boldsymbol{x})\Big)\exp\Big(\frac{1}{2\alpha_u}\Big) \tag{C.12}$$

and

$$\langle(\sigma_\nu^2 - \langle\sigma_\nu^2\rangle)^2\rangle = \exp\Big(-2\boldsymbol{u}_o^T\widehat{\boldsymbol{\psi}}(\boldsymbol{x})\Big)\Big[\exp\Big(\frac{2}{\alpha_u}\Big) - \exp\Big(\frac{1}{\alpha_u}\Big)\Big] \tag{C.13}$$

Next we expand $\boldsymbol{u}_o^T\boldsymbol{\psi}(\boldsymbol{x})$ in terms of the normalised basis functions

$$\boldsymbol{u}_o^T\widehat{\boldsymbol{\psi}}(\boldsymbol{x}_j) = \sum_{i=1}^{n}\xi_i\widehat{\boldsymbol{\psi}}^T(\boldsymbol{x}_i)\widehat{\boldsymbol{\psi}}(\boldsymbol{x}_j) = c_j \tag{C.14}$$

where the ξ_i's are the coefficients of the expansion and c_j is a constant which we assume that its value is known as a priori. Note that the points $\boldsymbol{x}_i$ need not be inputs from the training set but rather are points from the input space where we want to impose a prior value for the mean and variance of $p(\sigma_\nu^2)$. From (C.14) we have

$$\begin{aligned}\boldsymbol{u}_o^T\begin{pmatrix}\widehat{\boldsymbol{\psi}}(\boldsymbol{x}_1)\\ \vdots\\ \widehat{\boldsymbol{\psi}}(\boldsymbol{x}_n)\end{pmatrix} &= \begin{pmatrix}\xi_1\\ \vdots\\ \xi_n\end{pmatrix}^T\begin{pmatrix}\widehat{\boldsymbol{\psi}}^T(\boldsymbol{x}_1)\widehat{\boldsymbol{\psi}}(\boldsymbol{x}_1)\cdots\widehat{\boldsymbol{\psi}}^T(\boldsymbol{x}_1)\widehat{\boldsymbol{\psi}}(\boldsymbol{x}_n)\\ \vdots\\ \widehat{\boldsymbol{\psi}}^T(\boldsymbol{x}_1)\widehat{\boldsymbol{\psi}}(\boldsymbol{x}_n)\cdots\widehat{\boldsymbol{\psi}}^T(\boldsymbol{x}_n)\widehat{\boldsymbol{\psi}}(\boldsymbol{x}_n)\end{pmatrix} = \begin{pmatrix}c_1\\ \vdots\\ c_n\end{pmatrix}\\ &= \quad\boldsymbol{\xi}\qquad\qquad\qquad\qquad M\qquad\qquad\qquad\qquad = \quad\boldsymbol{c}\end{aligned} \tag{C.15}$$

Solving (C.15) for $\boldsymbol{\xi}$ we obtain

$$\boldsymbol{\xi} = M^{-1}\boldsymbol{c} \tag{C.16}$$

By using (C.14) and (C.16), we have

$$u_o = \sum_{i=1}^{n}[M^{-1}c]_i \psi(x_i) \tag{C.17}$$

Using (C.14) in (C.12) and (C.13) we obtain the following expressions for the mean and variance of prior noise variance

$$\langle \sigma_\nu^2 \rangle_i = \exp\Big(-c_i\Big)\exp\Big(\frac{1}{2\alpha_u}\Big) \tag{C.18}$$

$$\langle (\sigma_\nu^2 - \langle \sigma_\nu^2 \rangle)^2 \rangle_i = \exp\Big(-2c_i\Big)\Big[\exp\Big(\frac{2}{\alpha_u}\Big) - \exp\Big(\frac{1}{\alpha_u}\Big)\Big] \tag{C.19}$$

for all $i = 1, ..., n$. Note that we can associate a physical meaning to the elements of the vector c. They are the negative of the logarithm, up to a constant, of the mean prior variance at the x_i's input points. Thus we see that the problem of fixing u_o is now reduced to fixing the elements of the vector c which have a direct physical interpretation. Now our task is choose suitable values for c_i's and α_u which reflects our prior knowledge of the noise variance. For example, if we choose to be particularly vague about the noise variance we might then choose values for c_i's and α_u[2] which yield large values of $\langle (\sigma_\nu^2 - \langle \sigma_\nu^2 \rangle)^2 \rangle$. Similarly, if we were to expect large values of noise on the targets at the input point x_i we may then fix the c_i and α_u which yield large values for the mean $\langle \sigma_\nu^2 \rangle$.

C.2 Proof of formula (4.16)

In the evidence framework discussed in Section 1.6.1, the noise variance σ_ν^2 is fixed to its most probable value which is found from minimising the negative logarithm of the evidence $p(D|\beta, \alpha)$ which is given by (MacKay 1992a)

$$-\ln p(D|\beta,\alpha) = \beta E_D + \alpha E_w - \frac{N}{2}\ln\beta - \frac{k}{2}\ln\alpha + \frac{1}{2}\ln|A| \tag{C.20}$$

apart from some additive constants. Minimising the above with respect to σ_ν^2 gives

$$\sigma_\nu^2 = \frac{2E_D}{N} + \frac{1}{N}\,\mathrm{Trace}(A^{-1}B) \tag{C.21}$$

where B is the data Hessian matrix. By using the identity (B.3) and the outer product formula to the data Hessian (2.8) in the above we obtain

$$\begin{aligned} \sigma_\nu^2 &= \frac{2E_D}{N} + \frac{1}{N}\sum_{i=1}^{N} g_i^T A^{-1} g_i \\ &= \frac{1}{N}\sum_{i=1}^{N}(y_i - t_i)^2 + \frac{1}{N}\sum_{i=1}^{N}\sigma_w^2(x_i) \end{aligned} \tag{C.22}$$

[2] We may still choose to evaluate α_u from the data (see Section 4.7.3). In this case we may update the values of c_i as we update the values of α_u in a way that the prior mean and variance remain the same.

C.3 Evaluation of the derivatives of $M(u)$ with respect to u

Here we derive an expression for the first and second derivatives of the error function $M(\boldsymbol{u})$ with respect to $\boldsymbol{u}$. These derivatives are needed for finding the most probable value $\widetilde{\boldsymbol{u}}$ while the second derivatives are needed for estimating the most probable values of the hyperparameters $\widetilde{\alpha}_w$ and $\widetilde{\alpha}_u$. We have

$$M(\boldsymbol{u}) = \sum_{i=1}^{N} \beta_i E_i + \alpha_u E_u - \frac{1}{2}\sum_{i=1}^{N} \ln \beta_i + \frac{1}{2}\ln|A| \tag{C.23}$$

where $\beta_i \equiv \beta(\boldsymbol{x}_i;\boldsymbol{u})$, $E_i = \frac{1}{2}(y(\boldsymbol{x}_i;\widetilde{\boldsymbol{w}}) - t_i)^2$, $E_u(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{u}^T\boldsymbol{u}$ and the Hessian $\boldsymbol{A}$ is evaluated at the most probable value of the weights $\widetilde{\boldsymbol{w}}$. By making use of

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{u}} \ln|A| &= \sum_{i=1}^{N} \frac{\partial \ln|A|}{\partial \beta_i}\frac{\partial \beta_i}{\partial \boldsymbol{u}} \\
&= \sum_{i=1}^{N} \mathrm{Trace}(\boldsymbol{A}^{-1}\boldsymbol{g}_i\boldsymbol{g}_i^T)\frac{\partial \beta_i}{\partial \boldsymbol{u}} \\
&= \sum_{i=1}^{N} \boldsymbol{g}_i^T\boldsymbol{A}^{-1}\boldsymbol{g}_i\frac{\partial \beta_i}{\partial \boldsymbol{u}}
\end{aligned} \tag{C.24}$$

where $\boldsymbol{g}_i = \boldsymbol{g}(\boldsymbol{x}_i)$, in (C.23) we write the first derivatives as

$$\frac{\partial M(\boldsymbol{u})}{\partial \boldsymbol{u}} = \sum_{i=1}^{N}\Big(E_i + \frac{1}{2}\,\boldsymbol{g}_i^T\boldsymbol{A}^{-1}\boldsymbol{g}_i - \frac{1}{2\beta_i}\Big)\frac{\partial \beta_i}{\partial \boldsymbol{u}} + \alpha_u\boldsymbol{u} \tag{C.25}$$

with the property $\frac{\partial M(\boldsymbol{u})}{\partial \boldsymbol{u}} = 0$ at $\widetilde{\boldsymbol{u}}$. Using the identity

$$\frac{\partial M^{-1}}{\partial a}M + M^{-1}\frac{\partial M}{\partial a} = 0 \tag{C.26}$$

where a is a scalar, and

$$\frac{\partial \boldsymbol{A}}{\partial \boldsymbol{u}} = \sum_{i=1}^{N}\frac{\partial \boldsymbol{A}}{\partial \beta_i}\frac{\partial \beta_i}{\partial \boldsymbol{u}} \tag{C.27}$$

we can write the second derivatives of $M(\boldsymbol{u})$ as

$$\begin{aligned}
\boldsymbol{H} &= \frac{\partial^2 M(\boldsymbol{u})}{\partial \boldsymbol{u}^2} \\
&= \sum_{i=1}^{N}\Big(E_i + \frac{1}{2}\,\boldsymbol{g}_i^T\boldsymbol{A}^{-1}\boldsymbol{g}_i - \frac{1}{2\beta_i}\Big)\frac{\partial^2 \beta_i}{\partial \boldsymbol{u}^2} - \frac{1}{2}\sum_{i,j=1}^{N}\Big(\boldsymbol{g}_i^T\boldsymbol{A}^{-1}\boldsymbol{g}_j\Big)^2\frac{\partial \beta_i}{\partial \boldsymbol{u}}\frac{\partial \beta_j^T}{\partial \boldsymbol{u}} \\
&\quad + \frac{1}{2}\sum_{i=1}^{N}\frac{1}{\beta_i^2}\frac{\partial \beta_i}{\partial \boldsymbol{u}}\frac{\partial \beta_i^T}{\partial \boldsymbol{u}} + \alpha_u\boldsymbol{I}
\end{aligned} \tag{C.28}$$

For GLR models $\boldsymbol{g}_i = \boldsymbol{\phi}_i$, where $\boldsymbol{\phi}_i$ is the vector of basis functions measured at $\boldsymbol{x}_i$, and $\beta_i = \exp(\boldsymbol{u}^T\boldsymbol{\psi}_i)$. In this case (C.28) simplifies to

$$\begin{aligned}
\boldsymbol{H} &= \frac{\partial^2 M(\boldsymbol{u})}{\partial \boldsymbol{u}^2} \\
&= \sum_{i=1}^{N}\beta_i\Big(E_i + \frac{1}{2}\,\boldsymbol{\phi}_i^T\boldsymbol{A}^{-1}\boldsymbol{\phi}_i\Big)\boldsymbol{\psi}_i\boldsymbol{\psi}_i^T - \frac{1}{2}\sum_{i,j=1}^{N}\beta_i\beta_j\Big(\boldsymbol{\phi}_i^T\boldsymbol{A}^{-1}\boldsymbol{\phi}_j\Big)^2\boldsymbol{\psi}_i\boldsymbol{\psi}_j^T + \alpha_u\boldsymbol{I}
\end{aligned} \tag{C.29}$$

C.4 Evaluation of the derivatives of H with respect to α_w

We will now evaluate the first derivatives of the Hessian $\boldsymbol{H}$ with respect to the hyperparameters α_w. By using the identity (B.7) and $\partial \boldsymbol{A}/\partial \alpha_w = \boldsymbol{I}$ in (C.28) we obtain

$$\frac{\partial \boldsymbol{H}}{\partial \alpha_w} = -\frac{1}{2}\sum_{i=1}^{N} \boldsymbol{g}_i^T \boldsymbol{A}^{-2} \boldsymbol{g}_i \frac{\partial^2 \beta_i}{\partial \boldsymbol{u}^2} + \sum_{i,j=1}^{N} (\boldsymbol{g}_i^T \boldsymbol{A}^{-1} \boldsymbol{g}_j)(\boldsymbol{g}_i^T \boldsymbol{A}^{-2} \boldsymbol{g}_j) \frac{\partial \beta_i}{\partial \boldsymbol{u}} \frac{\partial \beta_j^T}{\partial \boldsymbol{u}} \tag{C.30}$$

where $\boldsymbol{A}^{-2} \equiv (\boldsymbol{A}^{-1})^2$. For GLR models this formula simplifies to

$$\frac{\partial \boldsymbol{H}}{\partial \alpha_w} = -\frac{1}{2}\sum_{i=1}^{N} \beta_i \boldsymbol{\phi}_i^T \boldsymbol{A}^{-2} \boldsymbol{\phi}_i \boldsymbol{\psi}_i \boldsymbol{\psi}_i^T + \frac{1}{2}\sum_{i,j=1}^{N} \beta_i \beta_j (\boldsymbol{\phi}_i^T \boldsymbol{A}^{-1} \boldsymbol{\phi}_j)(\boldsymbol{\phi}_i^T \boldsymbol{A}^{-2} \boldsymbol{\phi}_j) \boldsymbol{\psi}_i \boldsymbol{\psi}_j^T \tag{C.31}$$

Bibliography

Baldi, P. and Y. Chauvin (1991). Temporal evolution of generalisation during learning in linear networks. *Neural Computation 3*(4), 589–603.

Becker, S. and Y. LeCun (1989). Improving the convergence of back-propagation learning with second order methods. In D. Touretzky, G. E. Hinton, and T. J. Sejnowski (Eds.), *Proceedings of the 1988 Connectionist Models Summer School*, San Mateo, CA, pp. 29–37. Morgan Kaufmann.

Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour.* New Jersey: Princeton University Press.

Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis* (Second ed.). New York: Springer-Verlag.

Bishop, C. M. (1991). A fast procedure for retraining the multilayer perceptron. *International Journal of Neural Systems 2*(3), 229–236.

Bishop, C. M. (1992). Exact calculation of the Hessian matrix for the multilayer perceptron. *Neural Computation 4*(4), 494–501.

Bishop, C. M. (1994a). Mixture density networks. Technical Report NCRG 4288, Neural Computing Research Group, Aston University, Birmingham, UK.

Bishop, C. M. (1994b). Novelty detection and neural network validation. *IEE Proceedings: Vision, Image and Signal Processing 141*(4), 217–222. Special issue on applications of neural networks.

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

Bishop, C. M. (1995b). Training with noise is equivalent to Tikhonov regularization. *Neural Computation* 7(1), 108–116.

Bishop, C. M. and G. D. James (1993). Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research A327*, 580–593.

Breiman, L. (1994). Bagging predictions, technical report, University of California, Berkley.

Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fogelman Soulié and J. Hérault (Eds.), *Neurocomputing: Algorithms, Architectures and Applications*, pp. 227–236. New York: Springer-Verlag.

Buntine, W. L. and A. S. Weigend (1991). Bayesian back-propagation. *Complex Systems 5*, 603–643.

Buntine, W. L. and A. S. Weigend (1993). Computing second derivatives in feed-forward networks: a review. *IEEE Transactions on Neural Networks 5*(3), 480–488.

Chryssolouris, G., M. Lee, and A. Ramsey (1996). Confidence interval predictions for neural network models. *IEEE Transactions on Neural Networks 7*(1), 229–232.

Cohn, D. A. (1994). Neural network exploration using optimal experimental design. In S. J. Hanson, J. D. Cowan, and C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 6, San Mateo, CA, pp. 679–686. Morgan Kaufmann.

Cohn, D. A., Z. Ghahramani, and M. I. Jordan (1995). Active learning with statistical models. In G. Tesauro, D. S. Touretzky, and T. K. Leen (Eds.), *Advances in Neural Information Processing Systems*, Volume 7, Cambridge MA, pp. 705–712. MIT Press.

Duane, S., A. D. Kennedy, B. J. Pendleton, and D. Roweth (1987). Hybrid Monte Carlo. *Physics Letters B 195*(2), 216–222.

Efron, B. and R. J. Tibshirani (1993). *An Introduction to the Bootstrap.* Chapman & Hall.

Gull, S. F. (1988). Bayesian inductive inference and maximum entropy. In G. J. Erickson and C. R. Smith (Eds.), *Maximum-Entropy and Bayesian Methods in Science and Engineering, Vol. 1: Foundations*, pp. 53–74. Dordrecht: Kluwer.

Gull, S. F. (1989). Developments in maximum entropy data analysis. In J. Skilling (Ed.), *Maximum Entropy and Bayesian Methods, Cambridge, 1988*, pp. 53–71. Dordrecht: Kluwer.

Hassibi, B. and D. G. Stork (1993). Second order derivatives for network pruning: optimal brain surgeon. In S. J. Hanson, J. D. Cowan, and C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 5, San Mateo, CA, pp. 164–171. Morgan Kaufmann.

Hassibi, B., D. G. Stork, and G. Wolff (1994). Optimal brain surgeon: Extensions and performance comparisons. In S. J. Hanson, J. D. Cowan, and C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 6, San Mateo, CA, pp. 263–270. Morgan Kaufmann.

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

Heskes, T. (1996). Balancing between bagging and bumping. *Neural Computation 8*(1), 152–163.

Hinton, G. E. (1987). Learning translation invariant recognition in massively parallel networks. In J. W. de Bakker, A. J. Nijman, and P. C. Treleaven (Eds.), *Proceedings PARLE Conference on Parallel Architectures and Languages Europe*, Berlin, pp. 1–13. Springer-Verlag.

Horel, A. E. and R. W. Kennard (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics 12*(3), 55–67.

Horn, R. and C. Johnson (1985). *Matrix Analysis.* Cambridge University Press.

Jacobs, R. A., M. I. Jordan, S. J. Nowlan, and G. E. Hinton (1991). Adaptive mixtures of local experts. *Neural Computation 3*(1), 79–87.

Le Cun, Y., J. S. Denker, and S. A. Solla (1990). Optimal brain damage. In D. S. Touretzky (Ed.), *Advances in Neural Information Processing Systems*, Volume 2, San Mateo, CA, pp. 598–605. Morgan Kaufmann.

Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathematics II*(2), 164–168.

Lowe, D. (1989, October). Adaptive radial basis function non-linearities, and the problem of generalisation. In *Proceedings of the First Interntational Conference on Artificial Neural Networks*, London, pp. 171–175. IEE.

Luenberger, D. G. (1984). *Linear and Nonlinear Programming* (Second ed.). Reading, MA: Addison-Wesley.

MacKay, D. J. C. (1991). Bayesian Methods for Adaptive Models. Ph. D. thesis, California Institute of Technology.

MacKay, D. J. C. (1992a). Bayesian interpolation. *Neural Computation 4*(3), 415–447.

MacKay, D. J. C. (1992b). Information-based objective functions for active data selection. *Neural Computation 4*(4), 590–604.

MacKay, D. J. C. (1992c). A practical Bayesian framework for back-propagation networks. *Neural Computation 4*(3), 448–472.

MacKay, D. J. C. (1994a). Bayesian methods for backpropagation networks. In E. Domany, J. L. van Hemmen, and K. Schulten (Eds.), *Models of Neural Networks III*, Chapter 6. New York: Springer-Verlag.

MacKay, D. J. C. (1994b). Hyperparameters: optimise or integrate out? In G. Heidbreder (Ed.), *Maximum Entropy and Bayesian Methods, Santa Barbara 1993*, Dordrecht. Kluwer.

MacKay, D. J. C. (1995). Bayesian non-linear modelling for the 1993 energy prediction competition. In G. Heidbreder (Ed.), *Maximum Entropy and Bayesian Methods, Santa Barbara 1993*, Dordrecht. Kluwer.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society of Industrial and Applied Mathematics 11*(2), 431–441.

Matsuoka, K. (1992). Noise injection into inputs in back-propagation training. *IEEE Transactions on Systems, Man and Cybernetics 22*, 436–440.

Møller, M. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks 6*(4), 525–533.

Moody, J. E. (1992). The effective number of parameters: an analysis of generalization and regularization in nonlinear learning systems. In J. E. Moody, S. J. Hanson, and R. P. Lippmann (Eds.), *Advances in Neural Information Processing Systems*, Volume 4, San Mateo, CA, pp. 847–854. Morgan Kaufmann.

Morris, C. N. (1988). Approximating posterior distributions and posterior moments. In D. V. L. J. M. Bernardo, M. H. Degroot and A. F. M. .Smith (Eds.), *Proceedings of the Third Valencia International Meeting*, Volume 3, Oxford, pp. 327–344. Clarendon Press.

Nadaraya, É. A. (1964). On estimating regression. *Theory of Probability and its Applications 9*(1), 141–142.

Neal, R. M. (1992). Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Technical Report CRG-TR-92-1, Department of Computer Science, University of Toronto, Canada.

Neal, R. M. (1995). *Bayesian Learning for Neural Networks.* Ph. D. thesis, University of Toronto, Canada.

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

Nix, A. D. and A. S. Weigend (1995). Estimating the mean and variance of the target probability distribution. In G. Tesauro, D. S. Touretzky, and T. K. Leen (Eds.), *Advances in Neural Information Processing Systems*, Volume 7, Cambridge MA, pp. 489–496. MIT Press.

Nix, D. A. and A. S. Weigend (1994). Learning local error bars for nonlinear regression. In *Proceedings of the IEEE International Conference on Neural Networks*, Volume 1, New York, pp. 55–60. IEEE.

Nowlan, S. J. and G. E. Hinton (1992). Simplifying neural networks by soft weight sharing. *Neural Computation 4*(4), 473–493.

Ormoneit, D. and V. Tresp (1996). Improved gaussian mixture density estimates using bayesian penalty terms and network averaging. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems*, Volume 8, Cambridge MA, pp. 542–548. MIT Press.

Pearlmutter, B. A. (1994). Fast exact multiplication by the Hessian. *Neural Computation 6*(1), 147–160.

Polak, E. (1971). *Computational Methods in Optimization: A Unified Approach.* New York: Academic Press.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical Recipes in C: The Art of Scientific Computing* (Second ed.). Cambridge University Press.

Ricotti, L. P., S. Ragazzini, and G. Martinelli (1988). Learning of word stress in a sub-optimal second order backpropagation neural network. In *Proceedings of the IEEE International Conference on Neural Networks*, Volume 1, San Diego, CA, pp. 355–361. IEEE.

Rissanen, J. (1978). Modelling by shortest data description. *Automatica 14*, 465–471.

Satchwell, C. (1994). Finding error bars (the easy way). Networks, Official Newsletter of the Neural Computing Applications Forum, Edition 5.

Scott, D. W. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization.* New York: John Wiley.

Skilling, J. (1991). On parameter estimation and quantified MaxEnt. In W. T. Grandy and L. H. Schick (Eds.), *Maximum Entropy and Bayesian Methods, Laramie, 1990*, Dordrecht, pp. 267–273. Kluwer.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, B 36*(1), 111–147.

Stone, M. (1978). Cross-validation: A review. *Math. Operationsforsch. Statist. Ser. Statistics 9*(1), 127–139.

Thodberg, H. H. (1994). Bayesian backpropagation in action: Pruning, committees, error bars and an application to spectroscopy. In S. J. Hanson, J. D. Cowan, and C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 6, San Mateo, CA, pp. 208–215. Morgan Kaufmann.

Thodberg, H. H. (1996). A review of Bayesian neural networks with an application to near infrared spectroscopy. *IEEE Transactions on Automatic Control 7*(1), 56–72.

Tibshirani, R. (1996). A comparison of some error estimates for neural networks. *Neural Computation 8*(1), 152–163.

Tikhonov, A. N. and V. Y. Arsenin (1977). *Solutions of Ill-Posed Problems.* Washington, DC: V. H. Winston.

Wahba, G. and S. Wold (1975). A completely automatic French curve: fitting spline functions by cross-validation. *Communications in Statistics, Series A 4*(1), 1–17.

Walker, A. M. (1969). On the asymptotic behaviour of posterior distributions. *Journal of the Royal Statistical Society, B 31*(1), 80–88.

Williams, C. K. I. (1997). Regression with Gaussian Processes. In S. W. Ellacott, J. C. Mason, and I. J. Anderson (Eds.), *Mathematics of Neural Networks: Models, Algorithms and Applications.* Kluwer. Paper presented at the Mathematics of Neural Networks and Applications Conference, Oxford, UK, June 1995.

Williams, P. M. (1991). A Marquardt algorithm for choosing the step-size in backpropagation learning with conjugate gradients. Technical Report CSRP 299, University of Sussex, Brighton, UK.

Williams, P. M. (1995). Bayesian regularization and pruning using a Laplace prior. *Neural Computation 7*(1), 117–143.

Williams, P. M. (1996). Using neural networks to model conditional multivariate densities. *Neural Computation 8*(4), 843–854.

Wolpert, D. H. (1993). On the use of evidence in neural networks. In S. J. Hanson, J. D. Cowan, and C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 5, San Mateo, CA, pp. 539–546. Morgan Kaufmann.

Xu, L., A. Krzyżak, and A. Yuille (1994). On radial basis function nets and kernel regression: statistical consistency, convergence rates, and receptive fields. *Neural Networks 7*(4), 609–628.