LETTER Communicated by Frederico Girosi

Bayesian Radial Basis Functions of Variable Dimension

C. C. Holmes
B. K. Mallick
Department of Mathematics, Imperial College, London SW7 2BZ, U.K.

A Bayesian framework for the analysis of radial basis functions (RBF) is proposed that accommodates uncertainty in the dimension of the model. A distribution is defined over the space of all RBF models of a given basis function, and posterior densities are computed using reversible jump Markov chain Monte Carlo samplers (Green, 1995). This alleviates the need to select the architecture during the modeling process. The resulting networks are shown to adjust their size to the complexity of the data.

1 Introduction

The regression of a target variable Y on an input set of covariates X given the data pairings $\mathcal{D} = \{(y_1, x_1), (y_2, x_2), \ldots, (y_N, x_N)\}$ is an important topic in data analysis. The regression curve (surface) is assumed to be the conditional mean function $m(x) = E[Y \mid X = x]$ and the recorded observations Y to be corrupted with gaussian noise ϵ, so that

$$y_i = m(x_i) + \epsilon_i, \tag{1.1}$$

where ϵ_i are independent and identically distributed (i.i.d.) $\sim N(0, \sigma^2)$. In this article, we are concerned with the approximation of $m(x)$ by an estimate $\hat{m}(x)$ of the form

$$\hat{m}(x_i) = \sum_{j=1}^{K} w_j g_j(x_i), \tag{1.2}$$

where $g()$ represents a radial basis function (RBF) and w are output coefficients (weights).

RBF networks prove ultimately flexible in the surfaces that they can fit (Powell, 1987). This can lead to severe overfitting of the data if the complexity of the model is not managed properly. This problem is commonly avoided by adjusting the number of basis functions or including a regularization term that penalizes oscillatory behavior in the models' output. Here we take a Bayesian approach and consider the number of basis functions and values

of the coefficients to be unknown. We define a joint probability distribution over both model parameters and model dimension. Using suitably adjusted Markov chain Monte Carlo methods, we can then perform predictions and make inferences by integrating over both the model dimension and the parameter values within dimensions. This suppresses the need to select a single model or set of models for comparison or averaging, or both.

When applying this approach to the analysis of RBF networks, we show that the resulting size of the most probable model is determined by the complexity of the data. Simpler problems lead to smaller networks. On this count, it can be compared with non-Bayesian approaches for choosing the network size such as the resource allocating network of Platt (1991) or the support vector methods of Vapnik, Golowich, & Smola (1997) (see also Roberts & Tarransenko, 1994, and Yingwei, Sundararajan, & Saratchandran, 1997). The ability of the network size to be driven by the complexity of the data has particular importance in the analysis of large data sets, where traditional approaches, which use a basis function located at every data point, carry computational overheads that are too large.

In section 2 we provide an overview of Bayesian approaches to neural networks. Radial basis functions are briefly discussed in the beginning of section 3, where we also deal with the computational aspects of integrating in high-dimensional spaces of varying dimension. Section 4 lists the results of our approach on a number of benchmark test series, including the robot arm data set analyzed by MacKay (1992b) and the Wolf sunspot data given in Weigend, Huberman, & Rumelhart (1992).

2 Bayesian Neural Networks

Bayesian statistics has been applied to the analysis of neural networks by, among others, MacKay (1992a, b) and Neal (1996). Bishop (1995) presents an excellent overview.

The central process of the Bayesian framework is the calculation of a probability distribution on the unknown parameter (weight) vector θ. Prior knowledge that we might have, say for small weights, is updated in the light of experimental data. These posterior distributions are used in model predictions, with point forecasts given as expectations,

$$E[Y \mid x, \mathcal{D}] = \int m(x, \theta) p(\theta \mid \mathcal{D}) d\theta, \tag{2.1}$$

where $p(\theta \mid \mathcal{D})$ represents the posterior probability of the parameters of the model $m(\cdot, \theta)$ given the training data $\mathcal{D}$. Ripley (1996) refers to this as the "predictive approach." In this article, we also wish to take account of uncertainty between models $\mathcal{M}$ of different dimension (number of basis

functions), and we make this fact explicit by writing the expectation as

$$E[Y \mid x, \mathcal{D}] = \sum_{k=0}^{K} \int m(x, \theta_k, M_k) p(\theta_k \mid \mathcal{D}, M_k) p(M_k \mid \mathcal{D}) d\theta_k, \tag{2.2}$$

where $\mathcal{M} = \{M_0, \ldots, M_K\}$ is the set of models entertained.

We can compare equation 2.2 with the classical "plug-in" approach,

$$\hat{y} = m(x, \hat{\theta}_k, \hat{M}_k), \tag{2.3}$$

where $\hat{\theta}_k$ represents the parameters of the "best" model $\hat{M}_k$ set to some optimum value. Equations 2.2 and 2.3 highlight the difference between classical and Bayesian methodologies for prediction. Bayesian analysis involves integrating out uncertainty in the parameter values; classical methods involve their optimization, as noted by Bishop (1995).

The predictive approach in equation 2.2 can be considered a form of model averaging, whereby each model's prediction is weighted by its posterior probability. In this respect, it is similar to such methods as stacked generalization (Wolpert, 1992). However, the weighting in a Bayesian framework is given as a combination of likelihood and prior, rather than by a value to be optimized during the modeling procedure. In addition, the method described here does not require the prior specification of the set of models $\mathcal{M}$, which we construct dynamically during the model-fitting stage. For an overview of model averaging, see Jacobs (1995), Min and Zellner (1993), and the review paper by Genest and Zidek (1986), which includes an annotated bibliography of over 90 references. The Bayesian case for averaging over different models is presented by Draper (1995).

In previous approaches to Bayesian neural networks, the model's architecture has either been fixed prior to the data analysis or a small number of models are analyzed for averaging or selection.[1] Clearly this will tend to produce suboptimal solutions if the "best" (in some sense) model happens not to be tested or the data are not well approximated by the architectures chosen. Recently Green (1995) described a Markov chain sampling method for Bayesian computation that can approximate the integrals in equation 2.2 when the set of models $\mathcal{M}$ is unknown. The technique accommodates uncertainty in both $\mathcal{M}$ and the parameter values θ_k. Green's "reversible jumps" utilize Markov chains that can switch between dimensions while at the same time exploring the parameter space within a particular dimension (Richardson & Green, 1997). This method has many practical applications and is particularly well suited to nonparametric regression techniques of the form given in equation 1.2. In the analysis that follows, we consider

[1] Numerous Bayesian model choice criteria exist, analogous to those in classical statistics (Key, 1996).

RBFs, although the methods we adopt are generic and readily applicable to other types of networks, including multilayer perceptrons and multivariate adaptive regression splines (Denison, Mallick, & Smith, in press).

3 The Bayesian RBF Model

3.1 Radial Basis Functions. RBF networks have proved a popular method for approximating data (see Girosi, Jones, & Poggio, 1995, for a review). The models are feedforward networks of radial functions where each basis is parameterized by a knot or position vector μ located in the d-dimensional covariate space x. Conventionally there are as many basis functions as data points to be approximated with the position vectors set to the data values. The model output $m(x)$ is given by a linear combination of the basis functions response and a low-order polynomial term,

$$m(x) = \sum_{i=1}^{N} w_i \phi_i(\| x - \mu_i \|) + \sum_{m=0}^{p} a_m q_m(x), \tag{3.1}$$

where $\| \cdot \|$ denotes a distance metric, usually Euclidean or Mahalanobis, and $q_m(x)$ represents a polynomial of degree m. The coefficients w and a are calculated by least squares where the constraint $\sum_{i=1}^{N} w_i q_m(x_i) = 0$, $m = 0, \ldots, p$ is imposed to ensure the uniqueness of the solution.

The theory of RBFs specifies many permissible forms that the function $\phi()$ can take. Each of the acceptable basis types corresponds to a priori assumptions on the true regression surface being approximated (Girosi, 1994). Common choices include:

- Cubic

  $$\phi(z) = z^3.$$

- Thin plate spline

  $$\phi(z) = z^2 \log z.$$

- Multiquadric

  $$\phi(z) = (z^2 + c^2)^{1/2}.$$

- Linear

  $$\phi(z) = z.$$

- Inverse multiquadric

  $$\phi(z) = (z^2 + c^2)^{-1/2}.$$

- Gaussian

  $$\phi(z) = \exp(-cz^2).$$

There is some empirical evidence to suggest that nonlocal bases, where $\phi(z) \to \infty$ as $z \to \infty$, perform better than local basis functions and are, in addition, less sensitive to user set basis parameters c (Franke, 1982; Lowe, 1995). Following this, we choose to analyze the cubic, multiquadric, and thin plate spline functions. We will always include a linear term as the low-order polynomial in our model (see equation 2.3) and from now on take the vector w to incorporate the polynomial coefficients a.

The RBF model specified by equation 2.3 interpolates the data. However, interpolation is rarely our goal when analyzing the majority of real-world data sets, which tend to be corrupted by measurement noise or fail to record salient independent variables.

In these circumstances we need to reduce the flexibility of the model. This can be achieved by adding a small, positive regularization term, η, to the diagonal terms in the design matrix of the least-squares solution of w in equation 3.1 (Wahba, 1990). The larger the value of η, the smoother the output of the model.

In the neural network community, it is more fashionable to control complexity by reducing the number of centers to less than the number of data points, a suggestion first advocated by Broomhead and Lowe (1988). The reduction in the dimension leads to simpler models with less variance but greater bias. In the extreme with no radial functions, we recover the standard linear model.

These two approaches have the same aim: to reduce the complexity of the full model given in equation 2.3. In the next section, we describe a Bayesian framework for the RBF network and show how the complexity management is explicitly incorporated in a prior density that we define over the model-parameter space.

3.2 RBF Likelihood and Prior. The RBF networks are uniquely determined by the number of basis functions k, the position vectors μ_k, and the output coefficients w_k. In a Bayesian framework, we are interested in the posterior probability densities of these parameters given the data set $p(k, \mu, w \mid \mathcal{D})$. This distribution is used for predictions and model inference. Following equation 2.2, we now define a "model" M_k to include the number and location of basis functions $M_k = \{k, \mu_k\}$ and the parameter θ_k to represent the coefficients w_k within the model structure. We shall use the terms $\{M_k, \theta_k\}$ and $\{k, \mu_k, w_k\}$ interchangeably. The posterior distribution $p(M_k, \theta_k \mid \mathcal{D})$ is given as a combination of likelihood and prior

$$p(M_k, \theta_k \mid \mathcal{D}) = \frac{l(\mathcal{D} \mid \theta_k, M_k)p(\theta_k, M_k)}{p(\mathcal{D})}, \tag{3.2}$$

where $l(\mathcal{D} \mid \theta_k, M_k)$ is the likelihood function and $p(\theta_k, M_k)$ is the prior density.

Assuming a normally distributed noise term, we obtain the following log-likelihood of the RBF network, up to an additive constant,

$$L(\mathcal{D} \mid \theta_k, M_k) = -n \log \sigma - \frac{1}{2\sigma^2} \sum_{i=1}^{n} [y_i - m(x_i, \theta_k, M_k)]^2, \tag{3.3}$$

The prior term in equation 3.2 is used to express subjective beliefs about the nature of the posterior density of θ_k and M_k. It is the prior density that is used to incorporate preferences for simpler models or smoother model outputs. Fixing the number of basis functions and location vectors is equivalent to setting a point prior on single values of k and μ, that is, $\mathcal{M} = \{M_k\}$. The flexibility of the model is then controlled through the use of a prior on w. Typically this will be a form of shrinkage prior, $p(w_k) \sim N(w_k \mid 0, \lambda^{-1}I)$, that penalizes large values. The prior is controlled through the precision parameter λ. This has the same effect as the regularization term η described in the previous section for classical RBF networks.

The analysis here accounts for the additional uncertainty present in the choice of M_k. We place a proper prior over the whole of RBF model space $\mathcal{M}$ of a given basis function type. Originally we looked at using a Poisson prior on k (and a uniform prior on μ). Setting the mean of this Poisson prior to a small value penalizes networks with a large number of basis functions. However, this fails to take account of the amount of smoothing that is already being achieved by the setting of λ in the prior on the output coefficients, and our beliefs really lie on the flexibility of the model rather than the dimension per se. Therefore, rather than placing a prior on the dimension of the model k, we choose to place one on the degrees of freedom (DF) which is a function of k, μ, and λ. The DF of the network is readily computable (see the appendix) and measures the amount of fitting that the model achieves.[2] We place a gamma prior on DF so that our prior term in equation 3.2 becomes

$$\begin{aligned} p(\theta_k, M_k) &= p(w_k, \mu_k, k) \\ &= p(w_k)p(\mu_k, k) \\ &= N(w_k \mid 0, \lambda^{-1}I)\text{gamma}\,(\text{DF} \mid \alpha, \beta), \end{aligned} \tag{3.4}$$

where α and β are prior parameters.[3] An advantage of using DF above model dimension is that most users have a good understanding of the concept of DF, and hence subjective beliefs are more easily elicited. For instance, setting

[2] In fact, the measure defined in the appendix is an (under)approximation of the true degrees of freedom of our networks. For computational convenience, we treat the location and number of basis functions as independent of the data. It does, however, serve as a useful approximation for the prior ratios we are interested in, as described in section 3.4.

[3] The mean of a gamma distribution is α/β, and its variance is α/β^2.

$\alpha/\beta = 1$ indicates a preference for modeling the data by the average value of the target series. In d dimensions, setting the mean equal to $d+1$ indicates a preference for a linear fit.

Ideally we should accommodate uncertainty in the values of the prior parameters $\{\alpha, \beta, \lambda\}$. This is achieved in a hierarchical set-up by placing hyperpriors on the prior parameters. Uncertainty in these hyperpriors should be expressed as further priors. In reality, there is a fast diminishing return in adding further layers to the hierarchical model. We therefore keep the prior parameters fixed throughout. (See section 5 for a discussion of allowing λ to vary.)

Finally, we place a vague gamma prior on the precision (inverse variance) of the noise term in equation 3.3, $p(\sigma^{-2}) \sim \text{Ga}(10^{-3}, 10^{-3})$.

To recall, our ultimate aim is to use the densities defined in equation 3.2 for the predictions given in equation 2.2 by integrating over the distributions. Unfortunately, these posterior densities are typically complex and of varying dimension. This makes the integration intractable to analytical methods, and we must turn to approximation methods. The methods we use are described in the next section.

3.3 Markov Chain Monte Carlo and Reversible Jump Samplers. Bayesian inference involves integration, and Markov chain Monte Carlo (MCMC) methods form an important tool for approximating the integrals we are interested in. In this section, we provide a brief overview of this method. (For further details, see Bernardo & Smith, 1994; Tierney, 1994; Smith & Roberts, 1993; and Besag, Green, Higdon, & Mengerson, 1996.)

Suppose we wish to make model predictions using an integral of the form given in equation 2.1:

$$I = \int f(\pi) p(\pi \mid \mathcal{D}) d\pi. \tag{3.5}$$

MCMC methods proceed by drawing samples of π in direct proportion to their probability $p(\pi \mid \mathcal{D})$ and then approximating equation 3.5 by

$$I \approx \frac{1}{N - n_0} \sum_{t=n_0}^{N} f(\pi_t), \tag{3.6}$$

where N is the total number of samples generated (chain length), and n_0 is a "burn-in" period. The burn-in ensures that the Markov chain generating this sample has converged to the stationary distribution of interest $p(\pi \mid \mathcal{D})$.

The Metropolis-Hastings algorithm provides one example of this technique (Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953; Hastings, 1970). To begin, an initial sample point π_1 is drawn; this could be from the prior $p(\pi)$. A candidate for the next point in the chain, $\hat{\pi}_2$, is then proposed

based on the current point π_1 and a randomly generated vector u drawn from a (possibly symmetric) proposal distribution S, giving

$$\hat{\pi}_2 = \pi_1 + u, \tag{3.7}$$

where u is drawn from S and the hat, $\hat{\pi}$, indicates that π is currently only a proposal for the next state. The proposal distribution S is commonly chosen to be $N(0, \Sigma)$. Generally we can write $\hat{\pi}_{t+1} = \pi_t + u$.

The proposed sample is now "accepted" with probability

$$\alpha(\pi_t, \hat{\pi}_{t+1}) = \min\left\{1, \frac{p(\hat{\pi}_{t+1})S(\hat{\pi}_{t+1}, \pi_t)}{p(\pi_t)S(\pi_t, \hat{\pi}_{t+1})}\right\}, \tag{3.8}$$

where $p(x)$ is the probability of x, and $S(x, x')$ is the probability of proposing a move from x to x'. Obviously the proposal ratio, $S(x, x')/S(x', x)$, equals 1 if $S()$ is symmetric.

If the proposed state is accepted, then the next point in the chain, π_{t+1}, is set to $\hat{\pi}_{t+1}$; otherwise it is set to the previous point ($\pi_{t+1} = \pi_t$). The new sample π_{t+1} now forms the starting point for the next proposed state, and the algorithm is iterated until a large number of samples have been drawn. Each updated state is just a function of the previous state, and they are hence referred to as Markov chains. Independent samples from the distribution $p(\pi \mid \mathcal{D})$ (the distribution of interest) are obtained by first discarding an initial portion of the chain, to ensure the chain has converged, and then taking every nth sample to remove correlations (Tierney, 1994).

Until recently MCMC methods were mainly restricted to densities of fixed dimension. However, Green (1995) developed an MCMC method that samples from $p(\pi)$, where π is of unknown dimension.

3.4 Reversible Jump Radial Basis Functions. Green's (1995) reversible jump MCMC method allows us to approximate the integral in equation 2.2 when the number of basis functions is unknown. The algorithm proceeds by augmenting the usual proposal step of a conventional Metropolis-Hastings sampler with a number of other possible move types surrounding a change in the dimension of the density. At each iteration, in addition to the possibility of attempting a move within a particular parameter subspace, the sampler can propose to "jump" dimension, either up or down, by adding or removing a basis function from the network. We refer to these jumps as birth and death steps. The probability of attempting a birth or death step when the current state has k basis functions is given by b_k and d_k, respectively. In this article, we set $d_1 = 0$, $b_N = 0$, and $b_k = d_k = 0.1$ for all other values of k, where N is the number of data points.

It is common when jumping between dimensions to generate some random vector u that augments the current state π to form π'. For a birth step, this vector u is just a datum (position) vector drawn at random from those

points that do not already have a basis function located on them. The jump move is accepted with probability

$$\alpha(\pi, \pi') = \min\left\{1, \frac{p(\pi' \mid \mathcal{D}) r_m(\pi')}{p(\pi \mid \mathcal{D}) r_m(\pi) q(u)} \left| \frac{\partial(\pi')}{\partial(\pi, u)} \right| \right\}, \tag{3.9}$$

where $r_m(\pi)$ is the probability of choosing a jump of type m when the current state is π and $q(u)$ is the density function of u. The final term, a Jacobian, arises from the change of variables from (π, u) to π'. For our radial basis model, we can rewrite equation 3.9 as

$$\begin{aligned}\alpha(\{k, \mu, w\}, \{k', \mu', w'\}) = \min[1, &(\text{likelihood ratio}) \times (\text{prior ratio}) \\ &\times (\text{proposal ratio})].\end{aligned} \tag{3.10}$$

The Jacobian is not required because we are drawing our new location vector μ independent of the current parameters.

To illustrate this model, we will consider a birth step, from $\{k, \mu_k, w_k\}$ to $\{k+1, \mu_{k+1}, w_{k+1}\}$ (death steps are just an inversion of the following ratios). The prior ratio for a birth is

$$\frac{p(\mathrm{DF}_{k+1,\mu_{k+1}})[(k+1)!(N-k+1)!/N!]p(w_{k+1})}{p(\mathrm{DF}_{k,\mu_k})[k!(N-k)!/N!]p(w_k)}, \tag{3.11}$$

where $p(\mathrm{DF})$ and $p(w)$ are taken from equation 3.4, and the term $k!(N-k)!/N!$ represents the probability of choosing the k basis locations from the N data points.

The birth proposal ratio (PR) is given by

$$\begin{aligned}&\frac{\text{prob. of death move} \times \text{prob. proposing } w_k \times \text{prob. of deleting basis}}{\text{prob. of birth move} \times \text{prob. proposing } w_{k+1} \times \text{ prob. of creating basis}} \\ &= \frac{d_{k+1} p(w_k \mid \mathcal{D})/(k+1)}{b_k p(w_{k+1} \mid \mathcal{D})/(N-k)},\end{aligned} \tag{3.12}$$

where we have used the full conditionals of $p(w \mid \mathcal{D})$ as the proposal distribution for w (see section 3.5 for details). Finally, the likelihood ratio is taken from equation 3.3.

Note that when multiplying the prior and proposal ratios in equation 3.10, the terms involving factorials in equation 3.11 and the $(k+1)$ and $(N-k)$ terms in equation 3.12 cancel out, as will d_{k+1} and b_k when the number of basis functions is between 1 and $N-1$.

3.5 Reversible Jump Algorithm. The algorithm can be written in pseudocode as follows:

Starting with one RBF:

1. Draw the noise variance, σ^2, from its prior, gamma$(\sigma^{-2} \mid 10^{-3}, 10^{-3})$.
2. Draw the output coefficients w in a Gibbs sampling step. This uses the full conditionals of w given the data, $p(w \mid \mathcal{D}) = N(w \mid (\Psi^T\Psi + \lambda I)^{-1}\Psi y, \sigma^2\Psi^T\Psi)$, where λ is the prior precision for w and Ψ is the design matrix of outputs from the hidden layer of RBFs and polynomial terms. (See Smith & Roberts, 1993, for details of the Gibbs sampler.)
3. Iterate until convergence is assumed.[4]
    a. Draw a uniform random variable $u \sim U(0, 1)$.
    b. Propose the next state of the chain as follows:
        i. If u is less than b_k, then perform BIRTH step.
        ii. Else if u is less than $b_k + d_k$, perform DEATH step.
        iii. Else perform MOVE step.
    c. Redraw the coefficients w, as before.
    d. Draw a uniform random variable $u \sim U(0, 1)$.
        i. If $u < \alpha$, where α is from equation 3.10, then accept the proposed state.
        ii. Else set the next state to be the current state.
    e. Draw the noise variance, σ^2, in a Gibbs sampling step using the full conditionals $\sigma^{-2} =$ gamma $(\sigma^{-2} \mid 10^{-3} + N/2, 10^{-3} + \chi^2/2)$, where N is the number of data points and χ^2 is the sum of squared residuals for the current model.
    f. Repeat.

The MOVE, BIRTH, and DEATH steps are simple. MOVE selects a basis function at random and resets its location vector to another datum drawn randomly from the data set. BIRTH adds another basis function at a randomly selected point in the data set that does not already contain one. DEATH selects just one basis at random and removes it.

The output of the algorithm is a Markov chain that has $p(k, w, \mu \mid \mathcal{D})$ as its stationary distribution. An initial portion of the chain is discarded to ensure convergence, and then every mth sample is used to make predictions using equations 1.2, 2.2, and 3.6.

[4] This usually involves inspection of statistics from the Markov chain such as mean values of sampled parameters; see Tierney (1994) for details and heuristics.

4 Performance on Some Test Sets

This section illustrates the accuracy of the Bayesian RBF networks on some standard test sets. All the simulations were run with a burn-in period of 5000 iterations of the reversible jump MCMC algorithm followed by 12,000 samples, of which every third is used in the calculations. The value of β in the gamma prior on the degrees of freedom is set to 3, and the α value is set to 10. This indicates a strong preference for simpler models. These values could be adjusted to each data set using an empirical Bayesian approach, which might well improve the accuracy of the networks. We chose not to do so in order to present a unified approach.

The prior for the output coefficients is kept vague with a log precision value $\log \lambda = -5$. Each run took between three and six hours on a DEC Alpha 500, depending on the size of the data set and the complexity of the problem.

4.1 Bivariate Test Functions. We tested the method on five nonlinear functions analyzed by Hwang, Lay, Maechler, Martin, and Schimert (1994) in a study of multilayer perceptrons and projection pursuit.

The five functions are as follows:

1. Simple interaction function

$$f^{(1)}(x_1, x_2) = 10.391[(x_1 - 0.4)(x_2 - 0.6) + 0.36].$$

2. Radial function

$$f^{(2)}(x_1, x_2) = 24.234[r^2(0.75 - r^2)]$$
$$\text{where } r^2 = (x_1 - 0.5)^2 + (x_2 - 0.5)^2.$$

3. Harmonic function

$$f^{(3)}(x_1, x_2) = 42.659[0.1 + \hat{x}_1(0.05 + \hat{x}_1^4 - 10\hat{x}_1^2\hat{x}_2^2 + 5\hat{x}_2^4)]$$
$$\text{where } \hat{x}_1 = x_1 - 0.5, \text{ and } \hat{x}_2 = x_2 - 0.5.$$

4. Additive function

$$f^{(4)}(x_1, x_2) = 1.3356\{1.5(1 - x_1) + e^{2x_1 - 1}\sin[3\pi(x_1 - 0.6)^2]$$
$$+ e^{3(x_2 - 0.5)}\sin[4\pi(x_2 - 0.9)^2]\}.$$

5. Complicated interaction function

$$f^{(5)}(x_1, x_2) = 1.9\{1.35 + e^{x_1}\sin[13(x_1 - 0.6)^2]e^{-x_2}\sin(7x_2)\}.$$

In accordance with Hwang et al.'s approach, 225 data points were generated on the unit square, and the response, y, was calculated by $y_i =$

Table 1: Results for Hwangs' Test Set.

Function	Classical Cubic	Bayesian Cubic	Classical Tps	Bayesian Tps
Simple	0.0067	0.0060 (11)	0.0094	0.0074 (13)
Radial	0.0049	0.0054 (15)	0.0053	0.0094 (17)
Harmonic	0.0320	0.0455 (41)	0.0932	0.0576 (49)
Additive	0.0210	0.0220 (33)	0.0240	0.0232 (32)
Complex	0.0369	0.0413 (33)	0.0380	0.0447 (41)

Note: The accuracy of the two methods differs by over 0.5% of the variance in only 3 of the 10 tests.

$f(x_{1i}, x_{2i}) + \epsilon_i$ where $f()$ is the true test function and ϵ_i is gaussian white noise drawn from a $N(0, 0.25^2)$ distribution. The test set comes from generating 10,000 data points on a 100 by 100 grid over the unit square—[(1/200, 1/200), (3/200,1/200), ..., (199/200,1/200),(1/200,3/200), ..., (199/200,199/200)].

Table 1 lists the results in units of fraction of variance unexplained (FVU), which is given by

$$\text{FVU} = \frac{E[\hat{f}(x) - f(x)]^2}{E[f(x) - \bar{f}]^2}, \tag{4.1}$$

where $\hat{f}(x)$ is the model's prediction, $f(x)$ is the true value of the function, and $\bar{f}$ is the mean of the true function over the test set.

We compare our method with a classical RBF network using 225 basis functions (one at every data point). The regularization term for the classical model was set using cross-validation. We tested both cubic and thin-plate splines (Tps) on the data. The mode of the number of basis functions in the MCMC chain is included in parentheses alongside the error value in Table 1. The mode of the chain gives some indication of the region of the marginalized $p(k)$ that the MCMC chain had converged to.

The modes in brackets give an indication that the model adjusts the dimension to the complexity of the problem. Both methods perform well on these data sets, accounting for over 95% of the variance in all but one test. The results are comparatively very similar, differing by more than 1% of the variance unexplained in only the harmonic test set. However, the Bayesian RBF models use substantially fewer basis functions.

4.2 Robot Arm Data Set. This next task compares the Bayesian RBF model with other Bayesian neural network approaches. The problem is to model the mapping of a two-dimensional "joint angle" (x_1, x_2) to the end

Table 2: Results for Robot Arm Data Set.

	Average Squared Error
Gaussian approximation method of MacKay	
Solution with highest evidence	0.00573
Solution with lowest test error	0.00557
Hybrid MCMC of Neal with 150 supertransitions	
Best over three runs	0.00554
Neal's method with 30 supertransitions	
Best over three runs	0.00557
Bayesian RBF	
Cubic	0.00379 (24)

arm position (y_1, y_2). The true relationship is given by:

$$y_1 = 2.0\cos(x_1) + 1.3\cos(x_1 + x_2) + \epsilon_1 \qquad (4.2)$$
$$y_2 = 2.0\sin(x_1) + 1.3\sin(x_1 + x_2) + \epsilon_2,$$

where $\epsilon_i \sim N(0, \sigma^2)$, $\sigma = 0.05$. The data set can be found at MacKay's web site.[5] Neal (1996) compares his Bayesian multilayer perceptron, computed using a hybrid MCMC (Duane, Kennedy, Pendleton, & Roweth, 1987), with that of MacKay, who chooses to approximate the integral in the predictive expectations using gaussians fitted at local modes. The results are presented in Table 2. Again the mode of the number of basis functions in the chain is given in parentheses next to the error value.

The Bayesian RBF network appears slightly more accurate than Bayesian multilayer perceptrons on this data set.

4.3 Sunspots. The Wolf sunspot time series has served as a benchmark data set for a number of statistical models (Weigend et al., 1992; Tong & Lim, 1980). The data represent 280 yearly averages of sunspots (dark patches on the sun) recorded between the years 1700 and 1979. Weigend trained a multilayer perceptron (MLP) using the records from 1700 to 1920. The model was evaluated on two test sets representing the years 1921–1955 and 1956–1979. The data sets are generated by lagging 12 years of values as inputs and using the next year's sunspot value as a target, that is, it is modeled as an autoregressive AR(12) process. To compare our Bayesian models with that of Weigend, we tested multiquadrics (Mq) with the basis parameter c set to 0.5. The results are presented in Table 3, in terms of FVU, alongside those of a classical model using 220 basis functions.

[5] http://wol.ra.phy.cam.ac.uk/mackay/.

Table 3: Results for Sunspot Data Set.

Method	Mode $p(k)$	Training Error, 1713–1920	First Test Set, 1921–1955	Second Test Set, 1956–1979
Weigend's MLP	N.A.	0.082	0.086	0.35
Bayesian Mq	54	0.078	0.089	0.249
Classical Mq	220	0.073	0.096	0.275

5 Summary

A Bayesian approach to RBFs has been presented where the dimension (number of basis functions) is one of the things that we do not know. Using a specially constructed Markov chain based on reversible jumps we are able to draw inferences on the dimension of the model. Predictions are made by averaging over many models of varying dimension and basis location. By examining the marginal distribution of the number of basis functions, we see that the networks automatically adjust their size to the complexity of the problem.

We have chosen to keep λ, the prior precision (regularization parameter) for the output coefficients, fixed at a small value and mix over the dimension and position of the basis functions. Alternately, it is straightforward to fix the dimension and mix over the regularization parameter by placing a hyperprior on λ or allowing both dimension and regularization to vary. An advantage of the reported method is that smaller models result from clamping the prior precision to a small value, and so computational savings are made during the simulations.

Other amendments to the approach could be to include sampling from the input dimension during the jump steps. This would be appropriate if we had the knowledge to suggest that some covariates might be irrelevant. The marginal densities, giving the posterior probability of each covariate, are readily obtainable from the final chain.

Appendix

Consider a model $\mathcal{S}$ that smooths target values $\mathbf{y}$ given input values $\mathbf{x}$. This action is said to be linear if $\mathcal{S}(a\mathbf{y_1} + b\mathbf{y_2} \mid \mathbf{x}) = a\mathcal{S}(\mathbf{y_1} \mid \mathbf{x}) + b\mathcal{S}(\mathbf{y_2} \mid \mathbf{x})$ for any constants a and b (Hastie & Tibshirani, 1990). As such, classical RBF networks are linear smoothers. The model's output $\hat{f}$ at points $x_1, \ldots, x_n$ can be written in matrix notation as

$$\hat{\mathbf{f}} = \mathbf{S}\mathbf{y},$$

where $\mathbf{S}$ is an $n \times n$ smoother matrix.

To see this, let Ψ denote the design matrix given by the output of the basis functions (hidden layer) at the data points $x_1, \ldots, x_n$, and let w denote the final layer output coefficients for the network. Then we can write the least-squares approximation as

$$w = (\Psi^{\mathbf{T}}\Psi + \lambda I)^{-1}\Psi^{\mathbf{T}}\mathbf{y}, \tag{A.1}$$

where λ is the regularization parameter (or prior precision in a Bayesian setting). Point forecasts are made by $\hat{\mathbf{f}} = \Psi w$, which from equation A.1 can be written as

$$\hat{\mathbf{f}} = \Psi(\Psi^{\mathbf{T}}\Psi + \lambda I)^{-1}\Psi^{\mathbf{T}}\mathbf{y},$$

and hence

$$\mathbf{S} = \Psi(\Psi^{\mathbf{T}}\Psi + \lambda I)^{-1}\Psi^{\mathbf{T}}.$$

In accordance with Hastie and Tibshirani (1990), we define the DF of our RBF networks:

$$\mathrm{DF} = tr(\mathbf{S}). \tag{A.2}$$

This is the sum of the eigenvalues of S, which gives a measure of the amount of fitting that S, the expected smoothing matrix of our model, achieves.

Acknowledgments

We acknowledge the helpful comments of D. Denison and two anonymous referees regarding this work. C. C. H. was assisted by an EPSRC research award and sponsored by the Water Research Centre, Swindon, United Kingdom.

References

Bernardo, J., & Smith, A. (1994). *Bayesian theory*. New York: Wiley.

Besag, J., Green, P., Higdon, D., & Mengersen, K. (1996). Bayesian computation and stochastic systems. *Stat. Sci., 10*, 3–66.

Bishop, C. (1995). *Neural networks for pattern recognition*. New York: Oxford University Press.

Broomhead, D. S., & Lowe, D. (1988). Multivariate functional interpolation and adaptive networks. *Complex Systems, 2*, 321–355.

Denison, D., Mallick, B., & Smith, A. F. M. (in press). Bayesian M.A.R.S. *Statistics and Computing*.

Draper, D. (1995). Assessment and propagation of model uncertainty (with discussion). *J. Royal Stat. Soc. B, 57*, 45–98.

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

Franke, R. (1982). Scattered data interpolation: Tests of some methods. *Mathematics of Computation, 38*, 181–200.

Genest, C., & Zidek, J. V. (1986). Combining probability distributions: A critique and annotated bibliography. *Statistical Science, 1*, 114–148.

Girosi, F. (1994). Regularization theory, radial basis functions and networks. In V. Cherkassky, F. Friedman, & H. Wechsler (Eds.), *From statistics to neural networks*. Berlin: Springer-Verlag.

Girosi, F., Jones, M., & Poggio, T. (1995). Regularization theory and neural networks. *Neural Computation, 7*, 219–269.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika, 82*, 711–732.

Hastie, T. J., & Tibshirani, R. J. (1990). *Generalised additive models*. London: Chapman & Hall.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrica, 57*, 97–109.

Hwang, J., Lay, S., Maechler, R., Martin, D., & Schimert, J. (1994). Regression modeling in back-propagation and projection pursuit learning. *IEEE Trans. Neural Networks, 5*, 342–353.

Jacobs, R. A. (1995). Methods for combining experts' probability assessments. *Neural Computation, 7*, 867–888.

Key, J. (1996). *Studies of a simulation approach to Bayesian model comparison*. Unpublished Ph.D. dissertation, Department of Mathematics, Imperial College, London.

Lowe, D. (1995). On the use of nonlocal and nonpositive definite basis functions in radial basis function networks. In *Proc. 4th Int. Conf. Artificial Neural Networks* (pp. 206–210).

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

MacKay, D. (1992b). A practical Bayesian framework for backpropagation networks. *Neural Computation, 4*, 448–472.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equations of state calculations by fast computing machines. *J. Chem. Phys., 21*, 1087–1091.

Min, C., & Zellner, A. (1993). Bayesian and non-Bayesian methods for combining forecasts with applications to forecasting international growth rates. *J. Econometrics, 56*, 89–118.

Neal, R. (1996). *Bayesian learning for neural networks*. Berlin: Springer-Verlag.

Platt, J. (1991). A resource-allocating network for function interpolation. *Neural Computation, 3*, 213–225.

Powell, M. J. D. (1987). Radial basis functions for multivariate interpolation: A review. In J. C. Mason & M. G. Cox (Eds.), *Algorithms of approximation* (pp. 143–167). Oxford: Clarendon Press.

Richardson, S., & Green, P. J. (1997). On Bayesian analysis of mixtures with an unknown number of components. *J. R. Statist. Soc. B, 59*, 731–792.

Ripley, B. (1996). *Pattern recognition and neural networks*. Cambridge: Cambridge University Press.

Roberts, S., & Tarassenko, L. (1994). A probabilistic resource allocating network for novelty detection. *Neural Computation, 6*, 270–284.

Smith, A. F. M., & Roberts, G. O. (1993). Bayesian computation via Gibbs sampler and related Markov chain Monte Carlo methods. *J. R. Statist. Soc. B, 55*, 2–24.

Tierney, L. (1994). Markov chains for exploring posterior distributions. *Ann. Statist., 22*, 1701–1762.

Tong, H., & Lim, K. S. (1980). Threshold autoregression, limit cycles and cyclical data. *J. R. Statist. Soc. B, 42*, 245–292.

Vapnik, V., Golowich, S. E., & Smola, A. (1997). Support vector method for function approximation, regression estimation and signal processing. In M. Mozer, M. I. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems, 9* (pp. 281–287). Cambridge, MA: MIT Press.

Wahba, G. (1990). *Spline models for observational data*. Philadelphia: SIAM.

Weigend, A., Huberman, B. A., & Rumelhart, D. E. (1992). Predicting sunspots and exchange rates with connectionist networks. In C. Casdagli & S. Eubank (Eds.), *Nonlinear modeling and forecasting*. Reading, MA: Addison-Wesley.

Wolpert, D. H. (1992). Stacked generalization. *Neural Computation, 5*, 241–259.

Yingwei, L., Sundararajan, N., & Saratchandran, P. (1997). A sequential learning scheme for function approximation using minimal radial basis function neural networks. *Neural Computation, 9*, 461–478.

Received March 26, 1997; accepted October 7, 1997.