

CARNEGIE MELLON UNIVERSITY

MODEL SELECTION AND MODEL AVERAGING
FOR NEURAL NETWORKS

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

in

STATISTICS

By

HERBERT KUI HAN LEE III

Department of Statistics
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

January, 1999

Abstract

Neural networks are a useful statistical tool for nonparametric regression. In this thesis, I develop a methodology for doing nonparametric regression within the Bayesian framework. I address the problem of model selection and model averaging, including estimation of normalizing constants and searching of the model space in terms of both the optimal number of hidden nodes in the network as well as the best subset of explanatory variables. I demonstrate how to use a noninformative prior for a neural network, which is useful because of the difficulty in interpreting the parameters. I also prove the asymptotic consistency of the posterior for neural networks.

Keywords: Nonparametric regression, Bayesian statistics, Noninformative prior, Asymptotic consistency, Normalizing constants, Bayesian random searching, BARS

Acknowledgements

First, I need to thank my advisor, Larry Wasserman, without whom none of this would have been accomplished. Larry has been an ideal advisor. He was always willing to stop and help me with any problem. He always had an abundance of suggestions, and he helped me not lose sight of the big picture. His wisdom and patience are greatly appreciated.

I must also thank my parents, for they have always tried to push me to my potential, and have always been there when I needed them.

I would like to thank the members of my committee: Rob Kass, Chris Genovese, Mark Schervish, Valerie Ventura, and Andrew Moore. They have given me helpful suggestions and encouragement. The faculty as a whole have made this department a friendly and nurturing one. And I need to thank David Banks for helping convince me to return to graduate school.

I have discovered the secret to the operations of the department. It is the staff. I greatly appreciate all the help and moral support I have received from them, particularly Rose Krakovsky, Mari Alice McShane, Norene Mears, Heidi Sestrich, and Margie Smykla.

Last but not least, there have been my fellow students. Dan Cork has been an unending source of knowledge and support. His companionship, as well as his proofreading, is greatly appreciated. Kert Viele, Mark Fitzgerald, and Kevin Lynch have provided both statistical wisdom and non-statistical diversions to help keep me sane. Alix Gitelman has been a valuable compatriot and has helped me keep things in perspective, particularly during the crunch times. David “Soda” Algranati has provided much entertainment. And I would like to thank the many other students who have helped advance my education and helped make my time here enjoyable.

Contents

Abstract	ii
Acknowledgements	iii
1 Introduction	1
1.1 Overview	1
1.2 Nonparametric Regression	3
1.3 Neural Networks	6
1.3.1 Background	6
1.3.2 Fitting Algorithms	7
1.3.3 Fitting Algorithms in Bayesian Inference	8
1.4 Model Selection and Model Averaging	10
1.4.1 Overview of Model Selection	10
1.4.2 Selecting the Explanatory Variables	11
1.4.3 Selecting the Network Architecture	14
1.4.4 Simultaneous Variable and Size Selection	14
1.4.5 Model Averaging	15
1.5 Direction of the Rest of the Thesis	15
2 Priors for Neural Networks	16
2.1 Introduction	16
2.2 Some Proper Priors	18
2.2.1 The Müller and Rios Insua Model	19
2.2.2 Neal Model	21
2.2.3 MacKay Model	22

2.3	An Improper Prior	23
2.3.1	Specifying the Model	23
2.3.2	Fitting the Model	24
2.4	Theoretical Justification for the Prior Restrictions	25
2.4.1	Heuristic Explanation	25
2.4.2	Mathematical Details	27
2.5	Extensions of the model	30
2.5.1	Multivariate Response Variable	30
2.5.2	Categorical Response Variable (Classification)	30
2.5.3	Ordinal Response Variable	32
2.6	Examples	34
2.6.1	Motorcycle Accident Data	34
2.6.2	Iris Data	34
2.6.3	Social Attitudes Data	35
2.7	Discussion	38
3	Asymptotic Consistency	39
3.1	Introduction	39
3.2	Consistency	39
3.3	Asymptotic Consistency for Neural Networks	41
3.3.1	Sieve Asymptotics	41
3.3.2	The Number of Hidden Nodes as a Parameter	59
3.4	Discussion	61
4	Estimating the Normalizing Constant	63
4.1	Overview	63
4.2	Methods	65
4.2.1	Numerical Methods	65
4.2.2	Laplace Approximation	67
4.2.3	Importance Sampling	68
4.2.4	Reciprocal Importance Sampling	68
4.2.5	Bridge Sampling	69
4.2.6	Path Sampling	69
4.2.7	Density-Based Approximation	70

4.2.8	Partial Analytic Integration	70
4.2.9	BIC	71
4.3	Numerical Comparisons of the Methods	71
4.3.1	Ethanol Data	71
4.3.2	Simulated Data	76
4.4	Discussion	81
5	Model Selection and Model Averaging	82
5.1	Overview	82
5.1.1	Model Selection	82
5.1.2	Model Averaging	85
5.1.3	Searching the Model Space	86
5.1.4	Fitting the Models	87
5.2	Stepwise Algorithm	88
5.3	Occam's Window	90
5.4	Markov Chain Monte Carlo Model Composition	91
5.5	Bayesian Random Searching	92
5.6	Alternative Approaches	93
5.7	Examples	94
5.7.1	Ethanol Data	94
5.7.2	Simulated Data	96
5.7.3	Robot Arm Data	97
5.7.4	Ozone	101
5.8	Discussion	103
6	Conclusions	104
References		106

List of Tables

2.1	Social Attitudes Data	37
4.1	Normalizing Constant Estimates Based on the Full Posterior	74
4.2	Normalizing Constant Estimates Based on the Reduced Posterior	75
4.3	Numerical Estimates from BAYESPACK	76
4.4	BICs for Simulated Data	78
4.5	Normalizing Constant Approximations for Simulated Data	79
4.6	Normalizing Constant Approximations for Simulated Data	80
5.1	BICs for Ethanol Data from BARS	94
5.2	Comparison of Competing Methods on the Robot Arm Data	100
5.3	Comparison of Competing Methods on the Ozone Data	101
5.4	Comparison of Variable Selection on the Ozone Data	102

List of Figures

1.1	Pairwise Scatterplots for the Ozone Data	2
1.2	Estimated Smooths for the Ozone Data	3
1.3	Neural Network Model Diagram	7
2.1	A One-Node Function	17
2.2	Logistic Basis Functions	18
2.3	Fitted Function for Motorcycle Accident Data	35
2.4	Iris Data	36
4.1	(Log) Posterior Contours	66
4.2	Average Fitted Functions	72
4.3	Simulated Data	77
5.1	Fitted Function and Confidence Bands from Model Averaging for the Ethanol Data	95
5.2	A Six-Node Model for the Robot Arm Data	98
5.3	Test Data for the Robot Arm Problem	99

Chapter 1

Introduction

1.1 Overview

The goal of this thesis is to provide a framework for model selection in the realm of nonparametric regression and to study the asymptotic behavior of Bayesian methods for neural networks. Many standard techniques for nonparametric regression rely on ad hoc methods to determine the amount of smoothing and the subset of explanatory variables to include in the model. The Bayesian approach is a framework in which model selection is theoretically straightforward. It also provides a natural way to quantify the uncertainty of a fitted model. Neural networks have been shown to be able to approximate functions with arbitrarily good accuracy and hence are a good method for doing nonparametric regression. This thesis combines these two ideas, showing how to use neural networks to do a complete nonparametric regression, including the selection of the size of the network and the subset of explanatory variables. It establishes a methodology for doing nonparametric regression that includes all aspects of model selection and model uncertainty. Furthermore, it contains asymptotic results which show that the methods in this thesis have good frequentist properties, such as asymptotic consistency.

The key results of this thesis are the following: First, I demonstrate how to use a noninformative prior for neural networks which is simpler than other currently used priors. Second, I show that this noninformative prior, as well as many other more common priors, combines with the likelihood

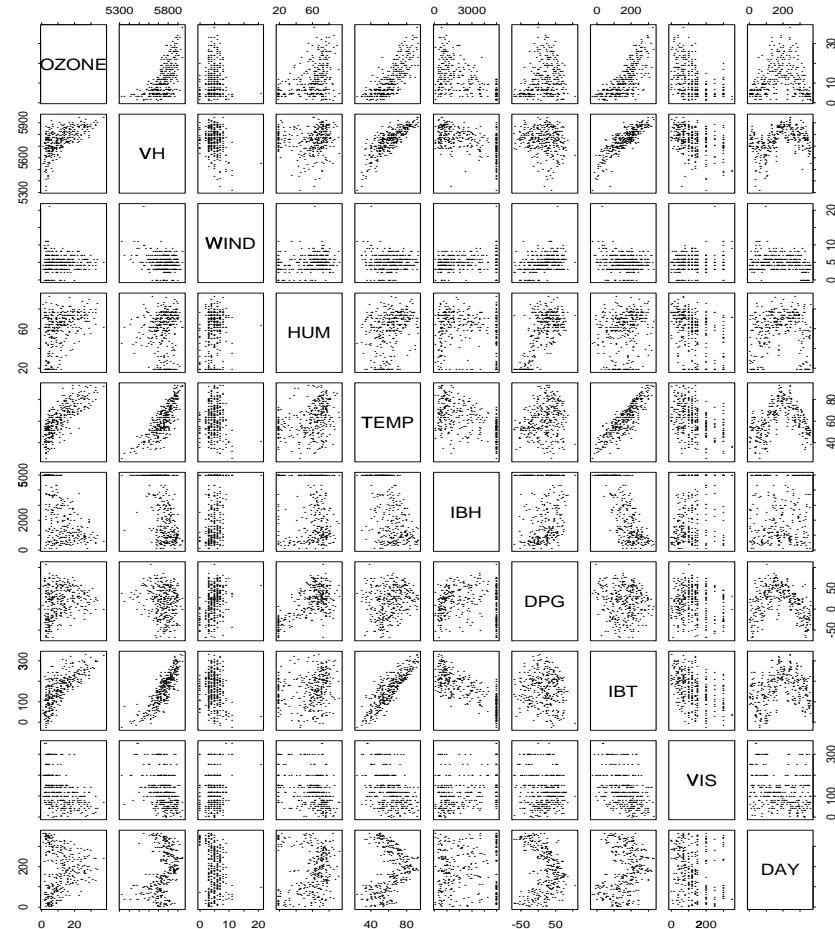


Figure 1.1: Pairwise Scatterplots for the Ozone Data

to produce an asymptotically consistent posterior. Third, I review many methods for estimating normalizing constants, an important issue in model selection, and draw conclusions about their reliability. Finally, I describe a methodology for model selection and model averaging using neural networks for nonparametric regression.

As an example of the type of problem to which this thesis applies, consider the Los Angeles ozone concentration data of Breiman and Friedman (1985). The data consist of the daily maximum ozone levels and eight other meteorological variables for 330 days of 1976. Pairwise scatterplots (Figure 1.1) show that most variables have a strong non-linear association with the ozone level, and that many variables are highly related to each other. Breiman and Friedman use these data as an

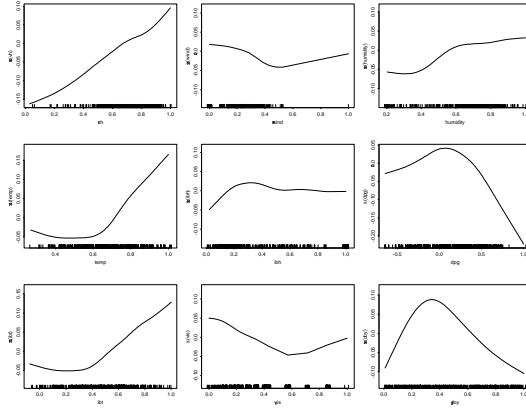


Figure 1.2: Estimated Smooths for the Ozone Data

example for their ACE algorithm, reporting that the best models seem to use only four or five of the explanatory variables. Hastie and Tibshirani (1990) use these data for comparing their additive model algorithms to several other regression algorithms. The estimated smoothing functions for all of the variables are shown in Figure 1.2. I fit these data with a neural network regression of the form

$$y_i = \sum_{j=0}^K \beta_j \Psi(\gamma_j' \mathbf{x}_i) + \varepsilon_i, \quad (1.1)$$

where K is the number of hidden nodes, the β_j 's are the weights of the basis functions, the γ_h 's are location and scale parameters (with h indexing subsets of covariates), $\varepsilon_i \stackrel{iid}{\sim} N(0, \sigma^2)$ is the error term, and Ψ is the logistic function:

$$\Psi(z) = \frac{1}{1 + \exp(-z)} \quad (1.2)$$

Some questions of interest are:

- Which explanatory variables (x) should be in the model?
- How many nodes (K) should be in the model?

1.2 Nonparametric Regression

Linear regression is a method of modeling data which uses a straight line in the one-dimensional case or a linear combination of the explanatory variables (a hyperplane) in the higher-dimensional

case. This model can be restrictive because it assumes that each explanatory variable is linearly related to the response variable. The idea of nonparametric regression is to use a more flexible class of response functions. Models are of the form

$$y_i = f(x_i) + \varepsilon_i, \quad (1.3)$$

where $f \in \mathcal{F}$, some class of regression functions, x are the explanatory variables, and ε_i is *iid* additive error with mean zero and constant variance. Sometimes normality of ε_i is assumed. The main distinction between the competing nonparametric methods is the class of functions, \mathcal{F} , to which f is assumed to belong.

Some of the basic nonparametric techniques are simple smoothing techniques. In one dimension, it is easy to smooth the data within bins, or by using a moving average. A more advanced approach is to use splines, piecewise polynomial smoothers. In higher dimensions the basic idea carries over, but the implementation needs to become more sophisticated. The Loess smoother (Cleveland, 1979) computes a weighted least-squares polynomial regression in a neighborhood of each point.

Some of the more complex techniques are forms of generalized additive models:

$$y_i = \sum_k f_k(x_i) + \varepsilon_i, \quad (1.4)$$

where certain additional assumptions are made about the form of f , or about which variables are included for each f_k . Hastie and Tibshirani (1990) look extensively at the univariate smooth f case. Projection Pursuit Regression (PPR) (Friedman and Stuetzle, 1981) considers the case

$$y_i = \sum_k f_k(\gamma_k^t x_i) + \varepsilon_i, \quad (1.5)$$

where f_k is an arbitrary smooth univariate function. A neural network can be viewed as a special case, where f_k is a fixed function, typically the logistic. Neural networks will be discussed extensively in the next section. Related techniques include ACE (Alternating Conditional Expectation) (Breiman and Friedman, 1985), AVAS (Additivity and Variance Stabilization) (Tibshirani, 1988), CART (Classification and Regression Trees, also called Recursive Partitioning Regression or RPR)

(Breiman et al., 1984), MARS (Multivariate Adaptive Regression Splines (Friedman, 1991), and wavelets (Donoho et al., 1995).

ACE (Alternating Conditional Expectation) (Breiman and Friedman, 1985) further generalizes the additive model by attempting to find a smooth transformation of both the response and explanatory variable so as to maximize the correlation between the transformed response and the sum of the transformed explanatory variables. Due to some known problems with ACE, Tibshirani (1988) created AVAS (Additivity and Variance Stabilization) as an improvement to ACE. AVAS forces the transformation of the response variable to be monotonic, and simultaneously tries to maximize correlation and stabilize the variance of the error terms.

Classification and Regression Trees (CART, also called Recursive Partitioning Regression or RPR) (Breiman et al., 1984) break the data into subsets and model the function as a constant over each subset. The subsets are chosen through a recursive algorithm which finds the locally best dichotomous division of a single subset. A generalization of CART is the Multivariate Adaptive Regression Spline (MARS) algorithm of Friedman (1991). Instead of fitting each subset with a constant function, each subset is fit with a function of the form $f_k = \Pi_p s_{kp}(x_{kp} - t_{kp})_+$ where s is a sign function (either +1 or -1) and t is a threshold.

A comparison study of the above methods was done by Banks et al. (1997). They ran a full factorial experiment over several different data models and model selection conditions. They found that there was no uniformly best method. Each method has one or more cases in which it is clearly superior to other methods, but it does poorly in other cases. There is very little theoretical work on comparing methods. Donoho and Johnson (1989) show that projection-based methods such as MARS, PPR, and neural networks should do well asymptotically on radially symmetric functions, while Loess is better for harmonic functions.

Another approach to nonparametric regression is to borrow ideas from density estimation. Kernel smoothing (Silverman, 1985) is essentially kernel density estimation applied to the regression setting. Recent Bayesian versions of kernel regression include that of Müller et al. (1996). A generalization of the kernel approach is to do Gaussian process regression. A Gaussian process is an infinite collection of random variables $\{Y(\mathbf{x})\}$, $\mathbf{x} \in X$, such that the distribution for the values of $Y(\cdot)$ evaluated over any finite set of points is multivariate Gaussian. A Gaussian process can be

specified uniquely by its mean function $E[Y(\mathbf{x})]$ and its covariance function $\text{Cov}(Y(\mathbf{x}_1), Y(\mathbf{x}_2))$. Neal (1996) shows that under certain conditions the limiting case of a neural network regression with arbitrarily many nodes is a Gaussian process regression.

1.3 Neural Networks

1.3.1 Background

Neural networks were originally created as an attempt to model the act of thinking by modeling neurons in a brain. Much of the early work in this area traces back to a paper by McCulloch and Pitts (1943) which introduced the idea of an activation function, although the authors used a threshold (indicator) function rather than the sigmoidal functions common today (as in Equation 1.2). Threshold activations were found to have severe limitations and thus sigmoidal activations became widely used instead (Anderson, 1982).

Much of the recent work on neural networks stems from a number of papers, including Funahashi (1989) and Hornik et al. (1989) that showed that neural networks are a way to approximate a function arbitrarily closely as the number of hidden nodes gets large. Neural networks work well with both smooth functions and those that have breakpoints in them, since a breakpoint can be modeled effectively with a single extra hidden node. Such discontinuities can be difficult for other nonparametric methods to handle.

I shall only consider feed-forward neural networks with one hidden layer of units with logistic activations and with one linear output unit, as these are sufficient for full approximation. The underlying idea of these neural networks is that they use a particular choice of functions to form a basis over the space of continuous functions. Sigmoid functions are used to form this basis. I shall be using the logistic function (Equation 1.2). The full model was given in Equation 1.1. A diagram of the model is shown in Figure 1.3. Each node in the layer of hidden units corresponds to a basis function (a Ψ_j in Equation 1.1). In practice only a finite number of hidden units are used, forming an approximation to the best regression function.

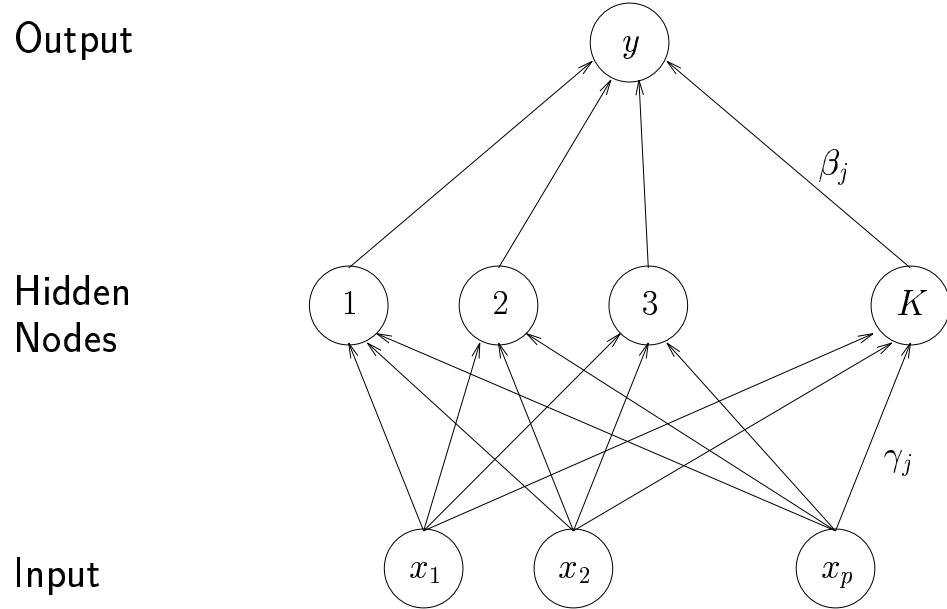


Figure 1.3: Neural Network Model Diagram

1.3.2 Fitting Algorithms

The standard method for fitting a neural network is *backpropagation*, which is essentially a gradient-descent algorithm. Each iteration involves only two calculations at each node, making it computationally efficient compared to most competing algorithms. It has apparently been discovered independently multiple times; it was made popular by Rumelhart et al. (1986).

To use the backpropagation algorithm, samples are taken from the data, and the weights of the network are adjusted in the direction of maximum gradient of the likelihood. The adjustments start from the final output node and are “propagated” backwards through the hidden nodes to the input nodes.

In practice, the backpropagation algorithm has several problems. First, it is only guaranteed to find a local maximum in the likelihood. While in many cases it will find the global maximum, it can easily get stuck in a local maximum which produces drastic over-smoothing of the data. Second, it can get stuck in “flat spots” where the gradient is computationally indistinguishable from zero (typically this occurs when the weights move the data to the tails of the activation function for a node). Finally, it can take a long time to converge. A good choice of step-size can help, although

the optimal choice can vary greatly between data sets and there is no good method of determining the right choice. The network fitting problem is still an open research area.

There have been many attempts to improve the backpropagation algorithm. One typical fix is to use *weight decay*, which is essentially a shrinkage method that includes a momentum term that is a multiple of the previous change in weights. It can improve both the speed and accuracy of a network. A further modification is *Quickprop*, developed by Fahlmann (1989). It uses a combination of ordinary backpropagation gradient descent and a local quadratic approximation involving both the current and previous error derivatives. In practice it is significantly faster than standard backpropagation while not significantly less accurate. Fahlmann and Lebiere (1989) have created another algorithm, *Cascade-Correlation*, which both selects a suitable number of nodes and finds the fitted values for the weights. While this method is extremely fast, it uses a very different network structure than I am considering (it only places one node in each hidden layer, and connects all previous layers to the current layer).

1.3.3 Fitting Algorithms in Bayesian Inference

There has been much less work done on fitting a neural network within the Bayesian framework. The Bayesian approach requires specifying a set of prior distributions for all of the weights in the network (and the variance of the error), then computing the posterior distributions for the weights using Bayes' Theorem. There are two main issues that need to be dealt with here. First, how does one specify a prior over parameters that have little or no interpretable meaning? Second, how does one efficiently approximate the posterior distribution when no analytic solution is known?

Often the choice of a prior is based on representing the user's prior beliefs. In the case of a neural network, it is difficult to provide any meaningful distribution for a particular weight since the correspondence between a weight and a feature of the data is not usually easily identifiable. Also, there is a certain non-identifiability inherent in the model in that the nodes can be substituted for each other; if all the inputs and outputs for a hidden node were switched with those of another node in the same layer, the model would be functionally equivalent, yet the individual parameters would be different. Because of this non-identifiability, all suggested priors are symmetric with respect to the ordering of the weights.

In Chapter 2 I shall review several priors which have been proposed. All of these are hierarchical in nature. My approach is to use a noninformative prior, and then enforce certain restrictions during the fitting process to ensure the propriety of the posterior.

The second issue raised in this section is how to find the posterior distribution. In some problems, a good choice of a conjugate prior can make it possible to analytically find the posterior distribution of the parameters. Neural networks, however, have no known conjugate priors. Some of the earlier work on Bayesian neural networks relied on fitting the network with backpropagation and then using a Gaussian approximation to get the posterior distribution (Buntine and Weigend, 1991). Modern Bayesian statistics has been aided by the introduction of Markov Chain Monte Carlo (MCMC) methods, which allow one to get an approximation of the posterior distribution by simulating random draws from the distribution. To get these draws, one simulates a Markov Chain with stationary distribution equal to the posterior distribution. This is relatively easy to do with Gibbs sampling by sampling from the conditional distribution for a parameter when all other parameters are known. At each step, a single (possibly multivariate) parameter is updated from its complete conditional where the current values of the other parameters are temporarily taken as the true values for the conditional distribution. Each of the other parameters is then updated in turn. In the case where a conditional distribution does not have a recognizable form and thus cannot be used for simulation, the Metropolis-Hastings algorithm can be used for that step. Let π be the conditional distribution of interest, let the current value for a parameter be w and generate a candidate value w^* from some convenient distribution. Then accept the candidate value with probability $\min\left(1, \frac{\pi(w^*)}{\pi(w)}\right)$ and otherwise keep the old value. This algorithm is due to Metropolis et al. (1953) and contains Gibbs sampling as a special case, which was introduced in a paper by Geman and Geman (1984). The underlying statistical theory of MCMC was solidified by Tierney (1994).

Nearly all modern methods for fitting Bayesian neural networks rely on MCMC. Müller and Rios Insua (1998b) use a multivariate version of MCMC and marginalize over some parameters to increase performance. Neal (1996) uses a hybrid Monte Carlo algorithm (Duane et al., 1987) which combines the Metropolis-Hastings algorithm with methods from dynamical simulation sampling techniques. MacKay (1992) combines MCMC with a local Gaussian approximation. I shall also use MCMC, with an additional restriction during a Metropolis step to prevent the posterior from

being improper.

1.4 Model Selection and Model Averaging

Model selection for a neural network entails both a selection of the structure of the network (in particular, how many hidden nodes to use) and a selection of which explanatory variables to use. The next subsection gives an introduction to the problem and addresses some philosophical aspects. The remaining subsections discuss variable selection methods, methods for selecting the size of the neural network, methods for attacking the combined problem, and model averaging.

1.4.1 Overview of Model Selection

The more hidden units that are used, the better the fit. With enough units, one can get a perfect fit of the data. Thus there is the need to balance two competing desires: to increase the number of units to improve the fit and to limit the number of units to decrease over-fitting and improve predictive performance. At the same time, we are also interested in choosing amongst competing explanatory variables. Just as in a multiple linear regression over a wide range of candidate explanatory variables, we would like to be able to use only the important variables and to leave the others out of the model. The additional challenge here is that we need to simultaneously choose our variables and choose the size of the network.

There are many competing ideas on how to choose the best model. Some are based on maximizing the likelihood subject to some penalty, such as the Akaike Information Criterion (Akaike, 1974) and the Bayesian Information Criterion (Schwarz, 1978). Another method is cross-validation (Stone, 1974). These are discussed in more detail in the next subsection.

In Bayesian inference, one can simply pick the model with highest posterior probability. The Bayesian approach also lends itself very nicely to prediction, in the guise of model averaging. In some cases, the analyst will find more than one model with high posterior probability, and these models will give different predictions (see, for example, the heart attack data in Raftery (1996)). Picking only a single model will grossly underestimate the variability of the estimate, since it ignores the fact that another model with significant posterior probability made a different prediction.

Instead, one should calculate predictions (or statistics thereof) by using a weighted average over all models in the search space, where the weights are the posterior probabilities of the models (Leamer, 1978). Let y be the response variable, D the data, and M_i the models of interest for $i \in \mathcal{I}$. Then the posterior predictive distribution of y is

$$P(y|D) = \sum_{i \in \mathcal{I}} P(y|D, M_i) P(M_i|D), \quad (1.6)$$

where $P(y|D, M_i)$ is the marginal posterior predictive density given a particular model (with all other parameters integrated out), and $P(M_i|D)$ is the posterior probability of model i .

From a philosophical standpoint, one might believe that for a given set of data, there is a true underlying model that involves a particular subset of the available explanatory variables. This is the approach of pure model selection. One envisions that if we had an infinite amount of data, we could determine exactly which variables belong in the model and which do not. As an example, some ecologists believe that there exists a model which determines the movements of manatee populations (Craig, 1997). Such an ecologist might collect data on all variables that were thought possibly relevant, and then the task of model selection would be to pick out which of these variables truly belong in the underlying model.

An alternative position to that of pure model selection is that of prediction. One may not believe that there exists an underlying true model, or that we could ever come close enough to the truth. For example, one might not believe that we could ever measure all of the variables that would influence manatee populations. In this case the best we can hope to do is to make accurate predictions. We may then want to resort to model averaging.

Some recent Bayesian approaches to the model selection problem for nonparametric regression include methods for splines (and other linear smoothers) (Sanil, 1998), CART (Chipman et al., 1998; Denison et al., 1998b), generalized additive models (Denison et al., 1998a; Smith and Kohn, 1996), MARS (Denison, 1997), and radial basis functions (Holmes and Mallick, 1998).

1.4.2 Selecting the Explanatory Variables

As in any model fitting problem, the more explanatory variables that are used, the smaller the error of the fitted function over the test data. However, using too many variables can lead to overfitting,

where the network is fitting the noise in the data as well as the signal, and the fitted function will perform poorly at prediction. To take an extreme case, if a variable is uncorrelated with the response variable, it should not be included in the model because while it may give a slight reduction in the error, it will not help predictive performance in the least. Another important case is when explanatory variables are highly correlated; a subset of these variables will typically give better predictive performance than all of them taken together because using all of them will lead to overfitting. Thus there is the need to balance the desire to use more variables for increased accuracy and the desire to limit the number of variables for increased predictive performance.

One method for variable selection is cross-validation (Stone, 1974). This involves breaking the data into sections (in the extreme case, each data point is a separate section). Then the model is fit using all but one section, and the error on the remaining section is found. This is repeated for all sections. The model producing the smallest sum of squared errors is deemed “best”.

Two other popular approaches are simple likelihood-based criteria. Akaike (1974) introduced the first, the Akaike Information Criterion (AIC), which is basically a penalized likelihood. It is defined as

$$AIC = L - p, \quad (1.7)$$

where L is the maximum of the log-likelihood and p is the number of parameters in the model. The optimal model is deemed to be the model with the largest AIC. Schwarz (1978) developed a similar criterion, the Bayesian Information Criterion (BIC), with a Bayesian motivation—it is an approximation to the log of a Bayes factor. Once again, the model with the largest BIC is selected. Let n be the sample size. Then the BIC is defined as

$$BIC = L - \frac{1}{2}p \log n. \quad (1.8)$$

The fully-Bayesian approach is to compare models by their posterior probabilities or through the use of Bayes factors. This requires a prior over the space of possible models, which could be taken as uniform if there is no prior information. Suppose we are comparing a set of models M_i for explaining the data, D . Denote the prior probability of model i by $P(M_i)$ and its posterior

probability by $P(M_i|D)$. Bayes' Theorem states that:

$$P(M_i|D) = \frac{P(D|M_i)P(M_i)}{\sum_j P(D|M_j)P(M_j)}. \quad (1.9)$$

Note that $P(D|M_i)$ involves marginalizing over the parameters in the model, which is an analytically intractable integral in the case of a neural network. Methods for approximating this integral will be discussed in Chapter 4.

The Bayes factor for comparing two models is the ratio of the posterior odds in favor of the first model to the prior odds in favor of the first model. The posterior odds in favor of model i are

$$\frac{P(M_i|D)}{P(M_j|D)} = \frac{P(D|M_i)}{P(D|M_j)} \frac{P(M_i)}{P(M_j)}, \quad (1.10)$$

and thus the Bayes factor is defined as

$$B_{ij} = \frac{P(D|M_i)}{P(D|M_j)}. \quad (1.11)$$

Kass and Raftery (1995) summarize the modern applications of Bayes factors in model selection.

All of the above techniques rely on an exhaustive search to find the optimal set of variables. For a large set of candidate variables, this can be prohibitively computationally expensive for nonparametric techniques. Furthermore, models may be fit with iterative algorithms, and thus it is not only time-consuming, but for any given fitted model, the user is not guaranteed to have found the best fit. For example, a neural network fitting algorithm may converge to a local minimum rather than a global minimum. This complication doesn't arise in the standard model selection setting where it is assumed that the model can be reliably fit.

Two frequentist automated model selection techniques are Stepwise Regression and Leaps and Bounds (Furnival and Wilson, 1974). Two Bayesian approaches, Occam's Window and Markov Chain Monte Carlo Model Composition, are given in Raftery et al. (1997). They both use the Bayes factor (or an approximation to it) as a guide when moving through the space of models.

Foster and George (1997) take an entirely different approach. They include the choice of model as another parameter and add another level of hierarchy to the model and find the joint posterior. They use an Empirical Bayes approach to select hyperparameters and then a maximum likelihood approximation to get a model score function they call the *Empirical Bayes Criterion* (EBC).

1.4.3 Selecting the Network Architecture

Just as in the case of variable selection, a delicate balance exists in the selection of the size of the network. Too many hidden nodes leads to a model that overfits and predicts poorly. Too few nodes produces a model that over-smooths and fits poorly. Many of the methods for selecting the size of a network are borrowed directly from the problem of variable selection. Cross-validation, AIC, and BIC are all accepted methods for choosing the number of hidden nodes (Bishop, 1995).

There have been some modifications of the standard model selection criteria to specifically address the size selection problem. Moody (1992) proposes a revised way to count the number of parameters in a neural network based on the *effective* number of parameters, which is smaller than the actual number of parameters used in the model because of the high correlation between certain parameters within a neural network. The effective number of parameters can then be used in a generalization of the AIC. Murata et al. (1994) have developed another criterion, the Network Information Criterion (NIC) which is also an extension of the AIC based on a modified penalty for the number of parameters in the model.

Another approach to the node selection problem is a shrinkage-based one. Often referred to as *weight decay*, a penalty depending on the magnitude of the weights is added to the least-squares error term and this sum is minimized. The hope is that unimportant weights will be shrunk to zero, and thus some nodes can be eliminated. Weight decay is also sometimes used in the fitting algorithms to improve the speed and accuracy of convergence.

1.4.4 Simultaneous Variable and Size Selection

There is not much in the literature specifically on the problem of choosing both the size of the network and which inputs to use. Many of the previous methods can be used for both, or one could use a combination of two methods. One combined approach is a procedure called Automatic Relevance Determination (ARD) (MacKay, 1994; Neal, 1996). The main idea is that each input has its own hyperparameter which resembles a scale factor for the weights from that input to the hidden nodes. If that input is important for any of the hidden nodes, that hyperparameter will be large. If that input is of no use to the network, then that hyperparameter will be small.

My approach to this problem is to use the full Bayesian framework. By putting a prior over the space of all possible models, both in terms of variables and number of hidden nodes, I can estimate the posterior probabilities of all of the models (or a subset of them if the space is too large) and choose the model with highest posterior probability. In practice, I use an approximation to the posterior probabilities based on the BIC, which makes it much easier to compute.

1.4.5 Model Averaging

Model averaging was first advocated by Leamer (1978). By using the weighted average of the predictions given by each model under consideration, weighted by the posterior probabilities of the models, one can both improve predictive performance as well as more accurately estimate the error of the prediction. This is primarily useful in the case when there is no single model with high posterior probability. Raftery et al. (1997) give examples of significant decreases in prediction errors through the use of model averaging.

Other related ideas include bagging (Breiman, 1994) and bumping (Tibshirani and Knight, 1995). Both of these use bootstrap samples of the original data. Bagging calculates estimates on the bootstrap samples and then averages these estimates to obtain the bagged estimate. Bumping involves a minimization for each bootstrap sample, then another minimization over the estimates from each sample. Both methods are attempts to improve prediction and error estimation in the case where there is no unique dominant best model.

1.5 Direction of the Rest of the Thesis

In Chapter 2, I will discuss the details of fitting a neural network under the Bayesian approach. In particular, I show how to use a simple noninformative prior. In Chapter 3, I present some asymptotic consistency results for neural networks in the Bayesian context, with the models of Chapter 2 as a special case. In Chapter 4, I describe the problem of estimating normalizing constants, which is necessary for Bayesian model selection and model averaging. Finally in Chapter 5, I will discuss the implementation of model selection and model averaging in the nonparametric regression problem, and give several examples of the methods from this thesis.

Chapter 2

Priors for Neural Networks

2.1 Introduction

One approach for selecting a prior is to choose a prior that reflects one’s beliefs. Such a prior would typically be a proper prior that integrates to one. Another possible approach would be to use a noninformative (flat) prior that does not favor particular values of the parameter over other values. For example, one could use Lebesgue measure for a real-valued parameter or one could use Jeffreys’ prior (Jeffreys, 1961; Bernardo, 1979). Such priors are typically “improper” in that they do not have finite integrals. However, in many problems the posterior will still be proper. Thus an improper prior is a way to do Bayesian inference without having to specify a proper prior (in a sense, one is specifying that one’s beliefs are equally spread out over all possible values).

The improper prior approach is useful in the case of neural networks because many of the parameters are very difficult to interpret. The effect of the coefficients of the logistic basis functions (β) can be difficult to visualize because the logistic functions are nonlinear and can combine in unexpected ways. The coefficients inside the logistic function are even less interpretable. Let me start by explaining the interpretations of the parameters for a model with only one hidden node:

$$\hat{y}_i = \beta_0 + \frac{\beta_1}{1 + \exp(-\gamma_0 - \gamma_1 x_i)} . \quad (2.1)$$

Figure 2.1 shows this fitted function for $\beta_0 = 1$, $\beta_1 = 2$, $\gamma_0 = -6$, and $\gamma_1 = 10$ over x in the unit interval. In the one-hidden node case, the parameters are easily interpreted. β_0 represents the

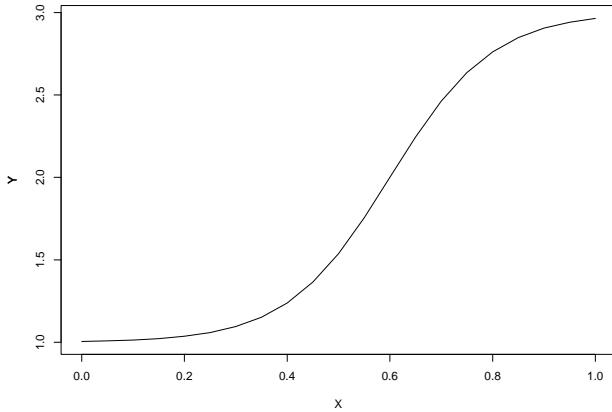


Figure 2.1: A One-Node Function

overall location of y , a sort of intercept, in that when the logistic function is close to zero (which it is here for values of x near or below zero), $y = \beta_0$. β_1 is the overall scale factor for y , in that the logistic function ranges from 0 to 1, so β_1 is like the range of y , which in this case is $3 - 1 = 2$. The γ parameters control the location and scale of the logistic function. The center of the logistic function occurs at $-\frac{\gamma_0}{\gamma_1}$, which in this case is 0.6. The larger the value of γ_1 , the steeper the increase in y as one moves away from the center. If γ_1 is positive, then the logistic rises from left to right. If γ_1 is negative, then the logistic will decrease as x increases.

This interpretation works fine for a single hidden node. For well-separated nodes (in terms of their centers and slopes, so that for any particular value of x , at most one node is near its center and the rest are far enough from their centers that a small change in x does not produce a noticeable change in the logistic function), one can continue to use this interpretation. Two nodes with centers near each other, but with opposite signs on γ_1 , can combine to produce a peak, similar in spirit to a kernel or a radial basis function. However, when the nodes effectively overlap, which is typically the case in a real problem, the parameters can no longer be interpreted as above.

For example, the left plot of Figure 2.2 results from maximum likelihood fitting of a two-node network to the motorcycle accident data of Silverman (1985). The x -axis is the time in milliseconds after the crash, and the y -axis the acceleration force on the head of the rider. The two-node model does a good job of capturing the main features of the data, although perhaps it over-smooths for

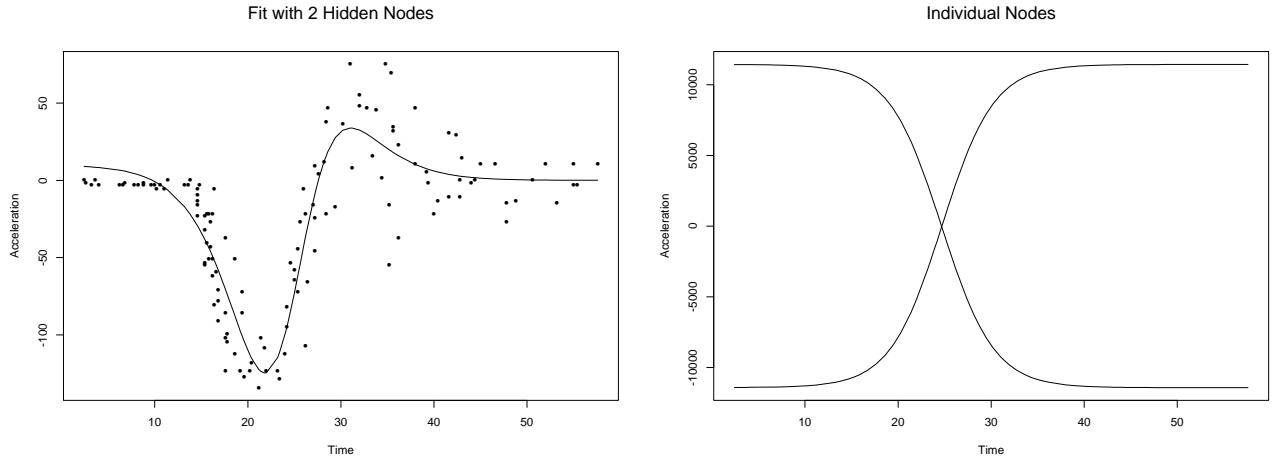


Figure 2.2: Logistic Basis Functions

small x values. Now keep in mind that this fit is achieved with only two-hidden nodes. It is impossible to use any of the above interpretations of the nodes for this fit, because there are more points of inflection in the fit than there are hidden nodes. The two individual nodes are shown in the right plot of the figure, and they have very similar centers and slopes, but combine to produce a highly non-linear fit. Looking at only the data, it is not at all clear that the particular basis functions of Figure 2.2 would work so well. In this case, one would not be likely to have any clear beliefs to specify in a prior. Hence a noninformative prior is a practical alternative.

In choosing an improper prior, one needs to be careful that it does produce a proper posterior. In this chapter, I will present an adjusted improper prior which turns out to be equivalent to a data-dependent prior. I will show that it guarantees a proper posterior, and that this adjusted prior is asymptotically equivalent to the original improper prior.

2.2 Some Proper Priors

Before presenting my choice of an improper prior, I review several proposed proper priors. There are three main approaches in the literature to choosing a prior, all of which involve hierarchical priors. The idea of a hierarchical model is to put a prior on the weights $p(w|\alpha)$ which depends on some other parameter α . One then puts a further prior (hyperprior) on α , $p(\alpha|\beta)$, where β could be

either another unknown parameter with its own hyperprior, or could just be a fixed parameter that needs to be chosen as part of the prior selection process.

All approaches start with the same basic model for the response y :

$$y_i = \beta_0 + \sum_{j=1}^k \beta_j \frac{1}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_{ih})} + \varepsilon_i, \quad (2.2)$$

$$\varepsilon_i \stackrel{iid}{\sim} N(0, \sigma^2). \quad (2.3)$$

I will give a brief description of each approach here, and then more detailed descriptions will follow in the subsequent subsections.

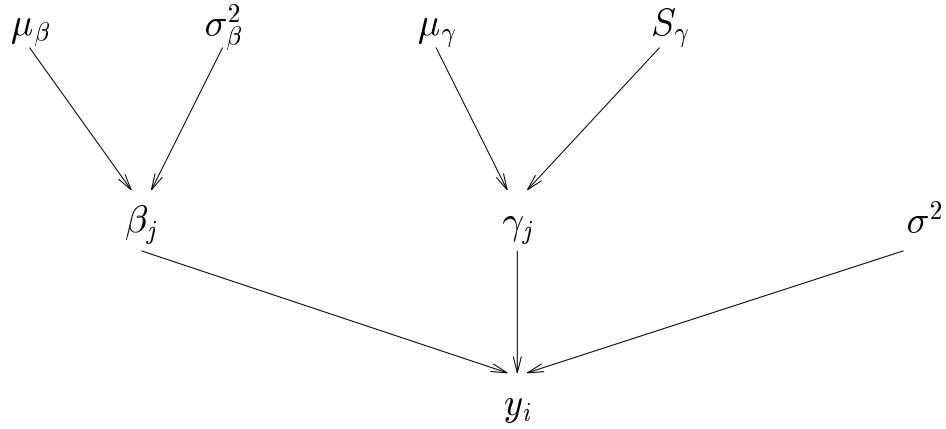
The first approach is to use an arbitrary prior of convenience with a large hierarchical structure to try to let the data determine as much as possible. Since the error in the model is assumed to be Gaussian, using a Gaussian prior over the weights provides some amount of conjugacy (at least for the output weights) and helps to simplify the computations. Müller and Rios Insua (1998b) suggest a three-stage hierarchical model that chooses the parameters at the top of the hierarchy to be of the same order of magnitude as the target data. The hierarchical structure is fairly simple, although many parameters are multivariate.

Neal (1996) suggests a four-stage model with arbitrary values for the parameters at the top of the hierarchy. The structure of the hierarchy is rather complex although all parameters are univariate.

MacKay (1992) starts by trying to use an improper prior in a simple two-stage model. To get around the impropriety of the posterior, he uses the data to choose fixed values for the hyperparameters. This is essentially a data-dependent prior for a one-stage model.

2.2.1 The Müller and Rios Insua Model

A directed acyclic graph (DAG) diagram of this model (Müller and Rios Insua, 1998b) is:



The distributions for parameters and hyperparameters are:

$$y_i \sim N \left(\sum_{j=0}^m \beta_j \Psi(\gamma'_j \mathbf{x}_i), \sigma^2 \right), \quad i = 1, \dots, N \quad (2.4)$$

$$\beta_j \sim N(\mu_\beta, \sigma_\beta^2), \quad j = 0, \dots, m \quad (2.5)$$

$$\gamma_j \sim N_p(\mu_\gamma, \mathbf{S}_\gamma), \quad j = 1, \dots, m \quad (2.6)$$

$$\mu_\beta \sim N(a_\beta, A_\beta) \quad (2.7)$$

$$\mu_\gamma \sim N_p(a_\gamma, \mathbf{A}_\gamma) \quad (2.8)$$

$$\sigma_\beta^2 \sim \Gamma^{-1} \left(\frac{c_\beta}{2}, \frac{C_\beta}{2} \right) \quad (2.9)$$

$$\mathbf{S}_\gamma \sim Wish^{-1}(c_\gamma, (c_\gamma \mathbf{C}_\gamma)^{-1}) \quad (2.10)$$

$$\sigma^2 \sim \Gamma^{-1} \left(\frac{s}{2}, \frac{S}{2} \right) \quad (2.11)$$

There are fixed hyperparameters that need to be specified. Müller and Rios Insua specify many of them to be of the same scale as the data. As some fitting algorithms work better when (or sometimes only when) the data have been re-scaled so that $|x_{ih}|, |y_i| \leq 1$ (cf. Ripley, 1993), one choice of starting hyperparameters is: $a_\beta = 0$, $A_\beta = 1$, $a_\gamma = \mathbf{0}$, $\mathbf{A}_\gamma = \mathbf{I}_p$, $c_\beta = 1$, $C_\beta = 1$, $c_\gamma = p + 1$, $\mathbf{C}_\gamma = \frac{1}{p+1} \mathbf{I}_p$, $s = 1$, $S = \frac{1}{10}$.

The two important devices that Müller and Rios Insua use to improve the rate of convergence of the chain in MCMC are blocking (using multivariate distributions) and marginalization. The multivariate aspect helps to take into account the complex correlation structure between the weights

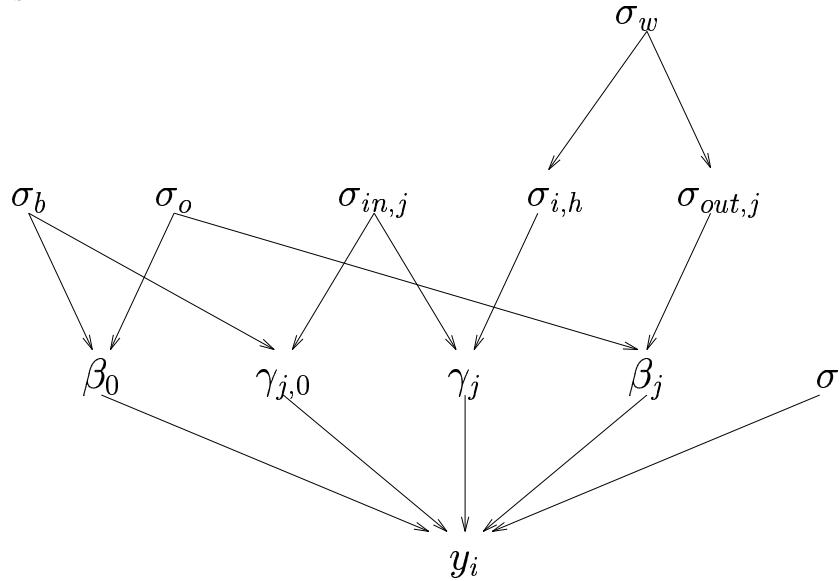
within the network, which can otherwise cause a chain to converge extremely slowly. The algorithm starts at an initial set of arbitrary values and then iterates through the following steps:

1. Given the current values for the rest of the parameters, replace β with a draw from a multivariate normal. Conditional on the other parameters, this step is equivalent to updating the coefficients in a multiple linear regression.
2. Given the current values for the hyperparameters but marginalizing over β , replace γ_j via a Metropolis step. A possible candidate distribution would be $N(\gamma_j, 0.01\mathbf{I}_p)$. Do this for $j = 1, \dots, m$.
3. Given the above parameters, update the hyperparameters from their complete conditional posterior distributions. All of these can be sampled directly, without Metropolis steps.

I initially worked with this model, but I discovered that it was more complex than necessary. The larger number of parameters made it difficult to evaluate Bayes factors. The model I use instead is in Section 2.3.

2.2.2 Neal Model

Neal's four-stage model (1996) contains more parameters than the previous model. A DAG diagram of the model is



Each of the original parameters (β and γ from Equation (2.2)) is treated as a univariate normal with mean zero and its own standard deviation. These standard deviations are the product of two hyperparameters, one for the originating node of the link in the graph, and one for the destination node. For example, the weight for the first input to the first hidden node, γ_{11} , has distribution $N(0, \sigma_{in,1}\sigma_{i,1})$, where $\sigma_{in,h}$ is the term for the links from the h th input and $\sigma_{i,j}$ is the term for the links into the j th hidden node; the weight from the first hidden node to the output (i.e. the regression coefficient), β_1 , has distribution $N(0, \sigma_{out,1}\sigma_o)$, where $\sigma_{out,j}$ is the term for the links from the j th hidden node, and σ_o is the term for links to the output node (here I only discuss the model for a univariate response; this model extends to a multivariate response by adding an additional output unit for each dimension of the response and using a collection of $\sigma_{o,h}$, one for each output unit).

For all of the new σ parameters and for the original σ of the error term, there is an inverse-gamma distribution. There is another set of hyperparameters that must be specified for the inverse-gamma priors on these σ parameters.

There are two technical notes on this model that should be mentioned. First, Neal uses hyperbolic tangent activation functions rather than logistic functions. These are essentially equivalent in terms of the neural network, the main difference being that their range is from -1 to 1 rather than 0 to 1 . The second note is that Neal also discusses using t distributions instead of normal distributions for the parameters.

2.2.3 MacKay Model

MacKay (1992) starts with the idea of using an improper prior, but knowing that the posterior would be improper, he uses the data to fix the hyperparameters at a single value. Under his model,

the distributions for parameters and hyperparameters are:

$$y_i \sim N\left(\sum_{j=0}^m \beta_j \Psi(\gamma'_j \mathbf{x}_i), \frac{1}{\nu}\right), \quad i = 1, \dots, N \quad (2.12)$$

$$\beta_j \sim N(0, \frac{1}{\alpha_1}), \quad j = 0, \dots, m \quad (2.13)$$

$$\gamma_j h \sim N_p(0, \frac{1}{\alpha_2}), \quad j = 1, \dots, m \quad h = 1, \dots, p \quad (2.14)$$

$$\gamma_j 0 \sim N_p(0, \frac{1}{\alpha_3}), \quad j = 1, \dots, m \quad (2.15)$$

$$\alpha_k \propto 1, \quad k = 1, 2, 3 \quad (2.16)$$

$$\nu \propto 1. \quad (2.17)$$

MacKay uses the data to find the posterior mode for the hyperparameters α and ν and then fixes them at their modes. He then fits the rest of the model using MCMC and uses a local quadratic approximation at the posterior mode to do model comparison.

Each of the three models I have just reviewed is fairly complicated. One has to deal with a hierarchy of parameters where the parameters are difficult to interpret, so that it is difficult to specify an informative prior. Rather than use one of these models, I sought a simpler model.

2.3 An Improper Prior

2.3.1 Specifying the Model

The model for the response y is

$$y_i = \beta_0 + \sum_{j=1}^k \beta_j \frac{1}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_{ih})} + \varepsilon_i, \quad (2.18)$$

$$\varepsilon_i \stackrel{iid}{\sim} N(0, \sigma^2). \quad (2.19)$$

Instead of having to specify a proper prior, we can use the noninformative improper prior

$$\pi(\beta, \gamma, \sigma^2) = (2\pi)^{-d/2} \frac{1}{\sigma^2}, \quad (2.20)$$

where d is the dimension of the model (number of parameters). The coefficient of $(2\pi)^{-d/2}$ is chosen for reasons explained later. This prior is proportional to the standard noninformative prior for linear regression.

To ensure that the posterior is proper, we need to put certain restrictions on the parameters during the MCMC fitting process. This idea has appeared in the literature for mixture models (Diebolt and Robert, 1994; Wasserman, 1998a). The key is to ensure that the logistic basis functions are linearly independent. To make this more formal, define

$$z_{ij} = \left[1 + \exp \left(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_{ih} \right) \right]^{-1} \quad (2.21)$$

and let Z be the matrix with elements (z_{ij}) . The fitting of the vector β is merely a multiple regression on the design matrix Z . We need a restriction on Z to ensure the linear independence of the columns of Z . A sufficient condition for linear independence is that the determinant of $Z^t Z$ is larger than some positive value C_n . We can let $C_n \rightarrow 0$ as $n \rightarrow \infty$ as long as $C_n > 0$ for all n . To ensure propriety of the posterior, we also need to bound the individual γ 's such that $|\gamma_{jh}| < D_n$ for all j, h , as will be explained later. This turns out to be equivalent to a data-dependent prior.

It is computationally useful to also bound the other parameters (namely $|\beta_j| < D_n$; σ does not need to be bounded). This can help with numerical stability in the MCMC fitting. It is also theoretically useful in that we can show that this prior satisfies the conditions for asymptotic consistency of the posterior of Theorem 3.3.1. For this theorem, we require $D_n = o(\exp(n^r))$ for all $r > 0$.

In many cases, using the coefficient of $(2\pi)^{-d/2}$ on Jeffreys' prior leads to the BIC being a order $O_p\left(\frac{1}{\sqrt{n}}\right)$ approximation to the log of the Bayes factor, instead of the usual $O_p(1)$ order approximation. In this case, I am not using Jeffreys' prior, but the coefficient is still a convenient and reasonable choice.

2.3.2 Fitting the Model

We use MCMC to fit the model. The full posterior is proportional to the likelihood times the prior:

$$f(\beta, \gamma, \sigma^2 | y) \propto (2\pi)^{-d/2} \frac{1}{\sigma^2} (2\pi\sigma^2)^{-n/2} \exp \left[-\frac{1}{2\sigma^2} \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^k \beta_j z_{ij} \right)^2 \right]. \quad (2.22)$$

The complete conditional distributions for β and σ^2 given the other parameters and the data are

$$\sigma^2|\gamma, y \sim Inv-\chi^2\left(n - k - 1, \frac{1}{n - k - 1} \sum_{i=1}^n \left(y_i - \hat{\beta}_0 - \sum_{j=1}^k \hat{\beta}_j z_{ij}\right)^2\right) \quad (2.23)$$

$$\beta|\gamma, \sigma^2, y \sim N\left((Z^t Z)^{-1} Z^t y, (Z^t Z)^{-1} \sigma^2\right), \quad (2.24)$$

where $\hat{\beta}$ are the fitted coefficients from a linear regression of y on Z . Thus the MCMC is as follows:

1. Start with arbitrary initial values for γ .
2. Compute Z .
3. Draw σ^2 via Equation (2.23).
4. Draw β via Equation (2.24).
5. For each j from $1, \dots, k$, do the following Metropolis step:
 - (a) Generate a candidate $\tilde{\gamma}_j \sim N(\gamma_j, 0.05^2)$.
 - (b) Re-compute Z with $\tilde{\gamma}_j$ and compute $|Z^t Z|$.
 - (c) If $|Z^t Z| > C_n$ and $|\gamma_{jh}| < D_n$ for all j, h , accept $\tilde{\gamma}_j$ with probability $\min\left(1, \frac{f(\beta, \tilde{\gamma}, \sigma^2 | y)}{f(\beta, \gamma, \sigma^2 | y)}\right)$; otherwise, reject the candidate and keep the current value of γ_j .
6. Repeat steps 2 though 5 for the required number of iterations.

The resulting draws will be a sample from the joint posterior distribution of the parameters. An explanation of why the determinant restriction keeps the posterior from being improper follows in the next section.

2.4 Theoretical Justification for the Prior Restrictions

2.4.1 Heuristic Explanation

To understand why the restrictions force propriety on the posterior, first look at a simplified scenario. Consider a univariate x of n observations. Instead of using logistic functions, suppose the basis

functions are indicator functions of the form

$$\Psi_j(x) = I_{\{x \leq a_j\}} \quad \text{or} \quad \Psi_j(x) = I_{\{x > a_j\}}. \quad (2.25)$$

Suppose for now that we are fitting only two hidden nodes and that we are not fitting an intercept. Then $z_j = (z_{1j}, \dots, z_{nj}) = (1, \dots, 1, 0, \dots, 0)$ or $(0, \dots, 0, 1, \dots, 1)$. Let

$$r_i = \sum_i z_{ij} \quad \text{for } i = 1, 2 \quad (2.26)$$

$$r_{12} = \sum_i z_{i1} z_{i2}. \quad (2.27)$$

Then the matrix of interest is

$$Z^t Z = \begin{bmatrix} \sum z_{i1}^2 & \sum z_{i1} z_{i2} \\ \sum z_{i1} z_{i2} & \sum z_{i2}^2 \end{bmatrix} = \begin{bmatrix} r_1 & r_{12} \\ r_{12} & r_2 \end{bmatrix}. \quad (2.28)$$

The determinant of this matrix is

$$|Z^t Z| = r_1 r_2 - r_{12}^2. \quad (2.29)$$

Then $|Z^t Z| > 0$ as long as neither $r_1 = 0$, $r_2 = 0$, nor $r_1 = r_2 = r_{12}$ —that is, if one of the indicator functions has no positive values, or if the two indicator functions produce identical output for the data x . The determinant will be positive if each indicator function has at least one positive value and if there is at least one data point where the values of the two indicator functions differ (i.e., there is a data point x_i that is between the thresholds of the two indicators). These are exactly the conditions that prevent linear dependence. If we choose a small enough C_n (for indicator functions, all we need is $0 < C_n < 1$), then the same conditions guarantee that $|Z^t Z| > C_n$.

Now suppose we add an intercept to the model. This adds a column of 1's to the Z matrix. The determinant now becomes

$$|Z^t Z| = \begin{vmatrix} r_1 & r_{12} & r_1 \\ r_{12} & r_2 & r_2 \\ r_1 & r_2 & n \end{vmatrix} = r_1 r_2 n + 2r_1 r_2 r_{12} - r_1^2 r_2 - r_1 r_2^2 - r_{12}^2 n. \quad (2.30)$$

This determinant will be zero if any of the following happen: $r_i = 0$, $r_i = n$, $r_1 = r_2 = r_{12}$, or $\{r_1 = n - r_2 \text{ and } r_{12} = 0\}$. Essentially we need to ensure that the thresholds for the indicators are

all separated by data points, and that no threshold occurs outside the range of the data. The same logic applies to larger datasets and more hidden nodes.

For indicator function hidden nodes, the requirement on the determinant also prevents impropriety in the posterior. Consider the case of trying to fit an indicator function where the parameter is the threshold. Suppose one tries to fit a threshold smaller than the smallest data point, x_{min} . Then the data do not provide any information for distinguishing between putting the threshold at $x_{min} - a$ and $x_{min} - b$ for any positive numbers a and b . Since this equivalence set has infinite mass under the prior, the posterior is improper. On the other hand, when fitting a threshold inside the range of the data, the data force propriety on the posterior.

The indicator functions relate to logistic functions in that indicator functions are a limiting case of logistic functions. Let

$$\Psi(x) = \frac{1}{1 + \exp(-\gamma_0 - \gamma_1 x)} \quad (2.31)$$

be a logistic function. If $\gamma_0, \gamma_1 \rightarrow \infty$ such that $\frac{\gamma_0}{\gamma_1} \rightarrow a$ for some constant a , then $\Psi(x) \rightarrow I_{\{x > -a\}}$. We would then need the determinant condition to guarantee linear independence. However, this is not the only condition we need to guarantee a proper posterior. The triangular region where $\gamma_0, \gamma_1 \rightarrow \infty$ such that $\frac{\gamma_0}{\gamma_1} \rightarrow a$ has infinite area. Over this region, as the parameters get large, the likelihood converges to some non-zero constant. Thus the posterior over this region alone is improper. For this reason, we also need to bound the individual parameters.

The logistic functions allow values between zero and one, so that two columns of $Z^t Z$ could be very similar, but not identical. In regression this is called multicollinearity. It is a case of near-linear dependence and causes instability in the parameter estimates. It is generally desirable to avoid this case, which we can do by requiring the determinant to be larger than some small positive number c rather than merely requiring it to be non-zero.

2.4.2 Mathematical Details

In this subsection, we provide a more rigorous justification for the prior. First we need some notation. Denote the likelihood by L_n . Let $\pi = \frac{1}{\sigma^2}$ be an improper prior. Let the modified prior

be $\pi_n = \pi I_{\Omega_n}$ where Ω_n is the set of parameter values such that $|Z^t Z| > C_n$, $|\gamma_{jh}| < D_n$, and $|\beta_j| < D_n$ for all j, h . Let C_n decrease to 0 (for example, $C_n = 1/n$) and let D_n increase with n (for example, $D_n = 10,000 + n$). Here one can clearly see how π_n is a data-dependent prior, as Ω_n depends upon x through Z .

We need to show that the posterior is proper. First we will re-write the likelihood so that we can integrate out β . Writing the likelihood in vector notation gives us

$$L_n = f(\beta, \gamma, \sigma | y) \quad (2.32)$$

$$= (2\pi\sigma^2)^{-n/2} \exp \left[-\frac{1}{2\sigma^2} \sum_{i=1}^n \left(\sum_{j=0}^k \beta_j z_j - y_i \right)^2 \right] \quad (2.33)$$

$$= (2\pi\sigma^2)^{-n/2} \exp \left[-\frac{1}{2\sigma^2} (Z\beta - Y)^t (Z\beta - Y) \right]. \quad (2.34)$$

Now completing the square:

$$\begin{aligned} L_n &= (2\pi\sigma^2)^{-\frac{n}{2}} \exp \left[-\frac{1}{2\sigma^2} (\beta^t Z^t Z \beta - 2Y^t Z(Z^t Z)^{-1}(Z^t Z)\beta + Y^t Z(Z^t Z)^{-1}Z^t Y + \right. \\ &\quad \left. + Y^t Y - Y^t Z(Z^t Z)^{-1}Z^t Y) \right] \end{aligned} \quad (2.35)$$

$$\begin{aligned} &= (2\pi\sigma^2)^{-\frac{k+1}{2}} |Z^t Z|^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2\sigma^2} [\beta - (Z^t Z)^{-1}Z^t Y]^t (Z^t Z) [\beta - (Z^t Z)^{-1}Z^t Y] \right\} * \\ &\quad (2\pi\sigma^2)^{-\frac{n-(k+1)}{2}} |Z^t Z|^{\frac{1}{2}} \exp \left\{ -\frac{1}{2\sigma^2} [Y^t Y - \hat{Y}^t \hat{Y}] \right\} \end{aligned} \quad (2.36)$$

$$= f(\beta | \gamma, \sigma, y) f(\gamma, \sigma | y), \quad (2.37)$$

where \hat{Y} is the vector of fitted values, $\hat{Y} = E[Y | X]$. Note that $f(\beta | \gamma, \sigma, y)$ is a proper density as long as $|Z^t Z| > 0$, which is true over Ω_n . Denote by Γ_n the subspace of Ω_n that relates to all of the γ parameters, and let Υ_n be the subspace of Ω_n that relates to all of the β parameters. (Recall that this bound on the β parameters is for computational convenience and is necessary for the consistency of the posterior as shown in Theorem 3.3.1, but is not necessary for propriety of the posterior.)

We now show that the posterior is proper:

$$\int L_n \pi_n = \int_{\Omega_n} f(\beta|\gamma, \sigma, y) f(\gamma, \sigma|y) \left[(2\pi)^{-\frac{d}{2}} \frac{1}{\sigma^2} \right] d\beta d\sigma d\gamma \quad (2.38)$$

$$= (2\pi)^{-\frac{d}{2}} \int_{\Gamma_n} \int \left[\int_{\Upsilon_n} f(\beta|\gamma, \sigma, y) d\beta \right] \frac{1}{\sigma^2} f(\gamma, \sigma|y) d\sigma d\gamma \quad (2.39)$$

$$\leq (2\pi)^{-\frac{d}{2}} \int_{\Gamma_n} \int \left[\int f(\beta|\gamma, \sigma, y) d\beta \right] \frac{1}{\sigma^2} f(\gamma, \sigma|y) d\sigma d\gamma \quad (2.40)$$

$$= (2\pi)^{-\frac{d}{2}} \int_{\Gamma_n} \int \frac{1}{\sigma^2} f(\gamma, \sigma|y) d\sigma d\gamma \quad (2.41)$$

$$= (2\pi)^{-\frac{d}{2}} \int_{\Gamma_n} \int \frac{1}{\sigma^2} (2\pi\sigma^2)^{-\frac{n-(k+1)}{2}} |Z^t Z|^{\frac{1}{2}} \exp \left\{ -\frac{1}{2\sigma^2} [Y^t Y - \hat{Y}^t \hat{Y}] \right\} d\sigma d\gamma \quad (2.42)$$

$$\leq (2\pi)^{-\frac{d}{2}} \int_{\Gamma_n} \int \frac{1}{\sigma^2} (2\pi\sigma^2)^{-\frac{n-(k+1)}{2}} |Z^t Z|^{\frac{1}{2}} d\sigma d\gamma \quad (2.43)$$

$$= (2\pi)^{-\frac{n-(k+1)+d}{2}} (n-k+1)^{-1} \int_{\Gamma_n} |Z^t Z|^{\frac{1}{2}} d\gamma \quad (2.44)$$

The last integral is finite because Γ_n is a bounded set and the integrand is finite. Thus the posterior is proper.

In addition to showing that the adjusted prior leads to a proper posterior, it is also important to show that the adjusted prior is asymptotically equivalent to the original improper prior. We show this in both a global and local sense.

First, it is clear that, for any compact set κ ,

$$\int_{\kappa} |\pi_n - \pi| = \int_{\kappa} |\pi I_{\Omega_n} - \pi| \rightarrow 0 \text{ as } n \rightarrow 0 \quad (2.45)$$

because the true function must have a non-zero determinant (or else it would have one fewer node), and because for a large enough n , Ω_n will contain all elements of κ that satisfy the determinant condition. This equation says that, in the limit as the sample size grows, π_n converges to π on all compact sets. In this sense, the two priors are “asymptotically globally equivalent” (Wasserman, 1998a).

Next we show a condition of “asymptotic local equivalence”. It also relates to correct second-order frequentist coverage properties (Wasserman, 1998a). It essentially states that the original and adjusted priors have the same local properties (but the adjusted prior is better behaved in the tails).

Suppose θ_0 is the true value of the parameters, then for large n ,

$$\left| \frac{\partial \log \pi_n}{\partial \theta_0} - \frac{\partial \log \pi}{\partial \theta_0} \right| = O_p \left(\frac{1}{\sqrt{n}} \right) \rightarrow 0 \quad (2.46)$$

because if n is large enough, θ_0 will be contained in Ω_n .

2.5 Extensions of the model

In this section, I explain how to extend the above model for a continuous multivariate response variable, as well as for a univariate response variable which is either categorical or ordinal. A binary response variable can be thought of as a special case of either of these two, where the number of categories is two.

2.5.1 Multivariate Response Variable

To extend the above model to a multivariate y , we simply treat each dimension of y as a separate output, and add a set of connections from each hidden node to each of the dimensions of y . In this implementation, we assume that the error variance is the same for all dimensions, although this would be easily generalized to separate variances for each component. Denote the vector of the i th observation as y_i , so that the l th component (dimension) of y_i is denoted y_{il} . The model is now

$$y_{il} = \beta_{0l} + \sum_{j=1}^k \beta_{jl} \frac{1}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_{ih})} + \varepsilon_{il} \quad (2.47)$$

$$\varepsilon_{il} \stackrel{iid}{\sim} N(0, \sigma^2). \quad (2.48)$$

When fitting the model, we replace the single draw of the β vector in step 4 of the algorithm with a loop which draws a vector of β s, using Equation (2.24), for each dimension of y . Conditional on the values of the γ parameters, the vectors of β are independent, and so can be sampled separately in the MCMC process.

2.5.2 Categorical Response Variable (Classification)

The multivariate model can be easily extended to the problem of classification, that is, when one has a response variable which is categorical rather than numeric. In this implementation, I only deal

with a univariate categorical response. The trick is to use an alternate expression of the response variable, where instead of just letting the response variable be the category number, we let it be a vector of indicator variables, where each component corresponds to one of the possible categories. Thus if we are trying to classify an observation into one of q categories, we can think of the response as being a vector of length q with a one in the position corresponding to its true category and zeroes in all of the other positions.

First, some notation: let x_i be the vector of explanatory variables for case i . Let t_i be the response variable for case i , where the value of t_i is the category number to which case i belongs, $t_i \in \{1, \dots, q\}$. Let t_i^* be a vector in the alternate representation of the response (a vector of indicator variables), where the $t_{ig}^* = 1$ when $g = t_i$ and zero otherwise. Now let y_{ig} be latent variables corresponding to the t_{ig}^* , where the y_{ig} are the results of fitting the multivariate neural network model on the x_i .

Some neural network classification models attempt to model the probability that the observation falls into a particular category. In that case, one could have the model

$$P(t_i = g_o | y_i) = \frac{\exp(y_{ig_o})}{\sum_g \exp(y_{ig})}. \quad (2.49)$$

This is the common model in the computer science literature. However, this model has an entirely different error structure than the models of this thesis. Instead, I retain the Gaussian error structure on the y_i and deterministically set

$$t_i = \arg \max_{g \in \{1, \dots, q\}} y_{ig} \quad (2.50)$$

i.e., the fitted response is the category with the largest y_{ig} . This allows me to apply the results of this thesis to this problem directly.

To fit this model with MCMC, we sample y_i from a normal truncated such that $y_{it_i} > y_{ig}$ for $g \neq t_i$. We also set $y_{ig} = 0$ for all i , in order to ensure the model is well-specified (otherwise the location of all of the y_{ig} 's could be shifted by a constant without changing the model). The algorithm now becomes:

1. Start with arbitrary initial values for γ .
2. Compute Z .

3. Draw σ^2 via Equation (2.23).
4. For each $g \in \{1, \dots, q\}$, draw β_g via Equation (2.24).
5. For each $j \in \{1, \dots, k\}$, do the following Metropolis step:
 - (a) Generate a candidate $\tilde{\gamma}_j \sim N(\gamma_j, 0.05^2)$.
 - (b) Re-compute Z with $\tilde{\gamma}_j$ and compute $|Z^t Z|$.
 - (c) If $|Z^t Z| > C_n$ and $|\gamma_{jh}| < D_n$ for all j, h , accept $\tilde{\gamma}_j$ with probability $\min\left(1, \frac{f(\beta, \tilde{\gamma}, \sigma^2 | y)}{f(\beta, \gamma, \sigma^2 | y)}\right)$; otherwise reject the candidate and keep the current value of γ_j .
6. For each i and for each $g \in \{1, \dots, q\}$, draw y_{ig} from a normal truncated such that $y_{it_i} > y_{ig'}$ for $g' \neq t_i$.
7. Repeat steps 2 though 6 for the required number of iterations.

2.5.3 Ordinal Response Variable

Sometimes one will want to relate an ordinal variable to a set of covariates. Examples of such an ordinal variable include rankings, data which were originally continuous but have been collapsed into bins, and ratings from a survey where the choices are “excellent”, “good”, “fair”, and “poor”. In all of these cases, there is a clear ordering of the values of the response variable, but it is not clear the values should be mapped to integers. By treating the variable as ordinal, we do not impose any sort of distance metric between levels of the variable.

To use the neural network model in this chapter for regression on an ordinal variable, we need a few more parameters. As in the classification setting, let q be the number of levels of the response variable (the number of bins). Let x_i be the vector of explanatory variables for case i . Let t_i be the response variable for case i , where the value of t_i is the bin number of which case i belongs, $t_i \in \{1, \dots, q\}$. Now let y_i be latent variables corresponding to the t_i , where the y_i are the results of fitting the neural network model of this chapter on the x_i . We also need parameters for the cut-off values for y , which we call c_1, \dots, c_{q-1} . If $c_{h-1} < y_i \leq c_h$, then we say that x_i belongs to bin h , i.e., the fitted value is $\hat{t}_i = h$. For example, if $c_1 < y_i \leq c_2$, then the i th observation is assigned to

bin 2. I use a prior for the c 's which is flat, except that it is zero when $c_h \geq c_{h+j}$ for any positive j (i.e., the c 's must be in increasing order).

To fit this model with MCMC, we sample y_i from a normal truncated to the interval $[c_{h-1}, c_h]$, where $t_i = h$. We fix $c_1 = 0$ so that the model is well specified. We use a Metropolis step to sample each of c_2, \dots, c_{q-1} , although the acceptance probability will be one unless a c moves beyond a y_i , in which case it will be zero.

The algorithm now becomes:

1. Start with arbitrary initial values for γ , let $c_h = h - 1$, and draw y_i randomly from a uniform $[c_{h-1}, c_h]$ (where $c_0 = -1$ and $c_q = q - 1$).
2. Compute Z .
3. Draw σ^2 via Equation (2.23).
4. Draw β via Equation (2.24).
5. For each j from $1, \dots, k$, do the following Metropolis step:
 - (a) Generate a candidate $\tilde{\gamma}_j \sim N(\gamma_j, 0.05^2)$.
 - (b) Re-compute Z with $\tilde{\gamma}_j$ and compute $|Z^t Z|$.
 - (c) If $|Z^t Z| > C_n$ and $|\gamma_{jh}| < D_n$ for all j, h , accept $\tilde{\gamma}_j$ with probability $\min\left(1, \frac{f(\beta, \tilde{\gamma}, \sigma^2 | y)}{f(\beta, \gamma, \sigma^2 | y)}\right)$; otherwise reject the candidate and keep the current value of γ_j .
6. Draw y_i from a normal truncated to $[c_{h-1}, c_h]$, for each i .
7. Draw a candidate c_h from a uniform on $[c_h - .1, c_h + .1]$ and accept if no y_i changes bins. Do for $h \in \{2, \dots, q - 1\}$.
8. Repeat steps 2 though 7 for the required number of iterations.

2.6 Examples

2.6.1 Motorcycle Accident Data

The motorcycle accident data of Silverman (1985) provides a nice example of the smoothing abilities of a neural network nonparametric regression. Figure 2.3 shows the average of the fitted functions for 10,000 runs (after allowing 10,000 runs for burn-in). The dark line is the mean fitted function, and the dashed lines are bands of 95% posterior predictive probability (for the fitted value of y at a particular value of x). Four hidden nodes were used. The x -axis is the time in milliseconds after the crash, and the y -axis the acceleration force on the head of the rider. The neural network fitted function does a good job of smoothing the data, even with only four hidden nodes.

2.6.2 Iris Data

As an example of a classification problem, I present an analysis of the canonical iris data of Fisher (1936). I used only the subset of the data which is supplied as a built-in dataset in S-Plus. The dataset consists of 150 observations, 50 from each of three different species of iris: Setosa, Versicolor, and Virginica. For each observation there are four explanatory variables: sepal length and width, and petal length and width, all measured in centimeters. The goal is to use these four measurements to classify the flowers. Figure 2.4 shows both the true and fitted values. In the top row are the true classifications given by sepal measurements (on the left) and petal measurements (on the right). Using all four variables and 3 hidden nodes, the results of classification from a neural network are shown in the bottom row, with the fitted categories plotted by sepal measurements on the left and by petal measurements on the right. There were four misclassified cases, all of which were Versicolor (denoted by +) which were mistakenly classified as Virginica (denoted by \times). These four cases are marked with a square around the \times . Notice that these four cases occur in regions of overlap between the two species in both sets of measurements, so they are difficult cases to classify correctly from this data alone. The neural network model has done a good job of classification.

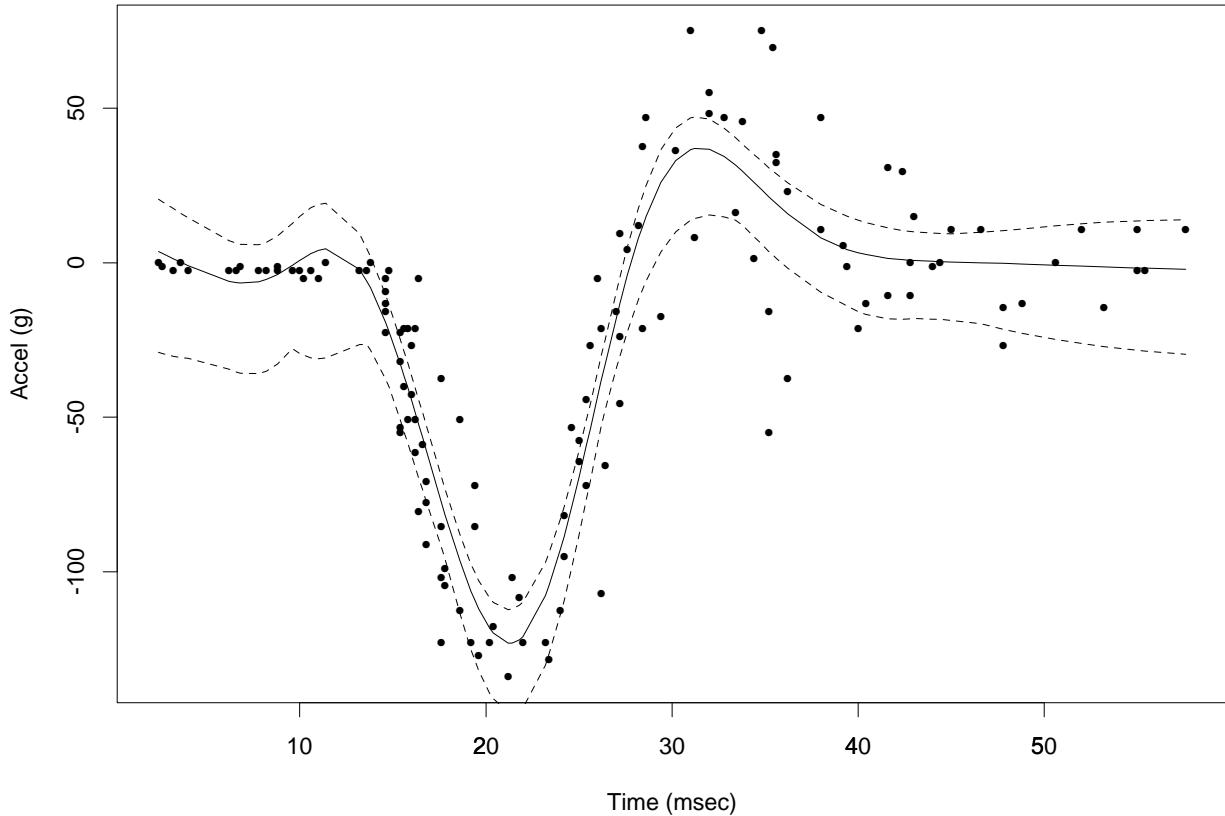


Figure 2.3: Fitted Function for Motorcycle Accident Data

2.6.3 Social Attitudes Data

Most datasets with ordinal response variables tend to have relatively few continuous explanatory variables and instead have mostly binary, ordinal, and categorical explanatory variables. Neural networks are designed for continuous explanatory variables, and may only operate to their full potential in that case. While they can handle indicator variables as inputs, there is no gain to a nonlinear transformation of an indicator variable, so there the neural network model may not work any better than much simpler models. Thus we should not expect beautiful fits from a neural network model if we give it mostly non-continuous explanatory variables.

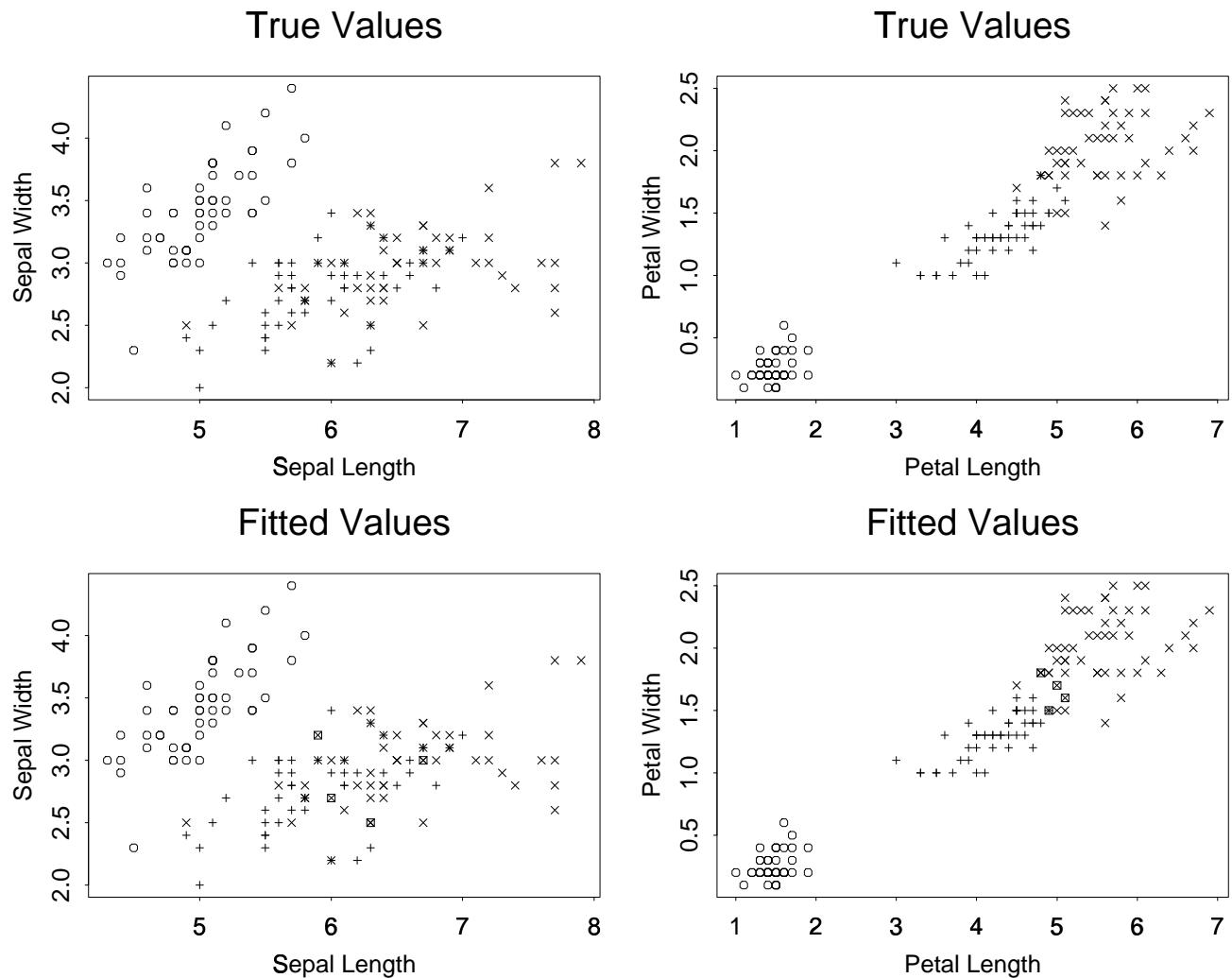


Figure 2.4: Iris Data

		Fitted Category				
		1	2	3	4	5
True Category	1	29	2	23	5	7
	2	6	4	19	4	3
	3	7	4	25	4	2
	4	4	3	12	4	10
	5	5	3	18	10	51

Table 2.1: Social Attitudes Data

For an example of an ordinal response variable, I fit the British Social Attitudes Survey data (Witherspoon, 1983). This data comes from a survey of households in Britain in 1983 of adults 18 or over living in private households. There was a series of yes/no questions asking whether the respondent supported or opposed a woman’s right to have an abortion under different circumstances. The respondents were then grouped into five ordered categories based on how many of the questions they answered “yes” to, with group 1 answering “yes” to the fewest questions, and group 5 answering “yes” to the most questions. The explanatory variables are political party affiliation (categorical), self-assessed social class (ordinal), gender (binary), age (continuous), and religion (categorical).

Table 2.1 shows the results of fitting a network with 9 hidden nodes to the data. As all of the explanatory variables besides age are not continuous, there is a high error rate for the fitted model. However, it is interesting to note that the model correctly fits most of the respondents in groups 1, 3, and 5, while incorrectly classifying most of the people in groups 2 and 4 into group 3. It appears that people who have more extreme views (those in groups 1 and 5, i.e., people strongly opposed the right to have an abortion or strongly in favor of it) are easier to classify than those with more intermediate views. Those people who support the right to an abortion in some, but not all, circumstances are difficult to distinguish by degree to which support this right. It appears that the model thinks that the data should really be classified into only three groups, those who are clearly opposed to the right to an abortion in nearly all circumstances, those who favor the right in nearly all circumstances, and those who favor the right in some circumstances but not in others. If the data were re-classified this way, the model would fit quite well.

2.7 Discussion

In this chapter, I have shown how to use an improper prior to do Bayesian inference for neural networks. This is useful because the parameters in a neural network are complex, so it is difficult to specify a proper prior that incorporates knowledge in a useful way. The standard methods of using hierarchical models produce a more complicated model that is more difficult to fit. The prior that I have proposed is much simpler to work with and allows the use of a model with many fewer parameters than a hierarchical model. This makes the MCMC process easier and faster.

Using restrictions during the Metropolis steps of the fitting process is equivalent to the use of a data-dependent prior. This adjusted prior produces a proper posterior, and is asymptotically equivalent to the original improper prior in both a global and a local sense.

The model works well in practice, and has many useful extensions. It can be adapted for use with multivariate responses, as well as binary, categorical, and ordinal response variables.

Chapter 3

Asymptotic Consistency

3.1 Introduction

This chapter discusses the large sample properties of neural network regression in the Bayesian setting. In this chapter we build upon the results of Funahashi (1989), Hornik et al. (1989), and others. These authors have shown that neural networks can approximate a function with arbitrary accuracy. However, they all use frequentist methods. Here we will show that the posterior is consistent for neural networks in the Bayesian setting. We show this consistency in two different paradigms. First, we take the sieve approach, where the number of hidden nodes grows with the sample size so that, asymptotically, we use an arbitrarily large number of hidden nodes. Second, we treat the number of hidden nodes as a parameter and show that the joint posterior is also consistent. Thus, we extend the theoretical justification of neural networks as universal approximators to the Bayesian approach.

3.2 Consistency

Given data $(X_1, Y_1), \dots, (X_n, Y_n)$ where X_i may be multivariate, let $f_o(x)$ be the underlying density of X . Let $E[Y|X = x] = g_o(x)$ be the true regression function of Y on X , and let $\hat{g}_n(x)$ be the estimated regression function.

Definition 1 \hat{g}_n is asymptotically consistent for g_o if

$$\int (\hat{g}_n(x) - g_o(x))^2 f_o(x) dx \xrightarrow{P} 0. \quad (3.1)$$

Another approach to consistency is to take it to mean that the posterior probability of every neighborhood of the true function tends to one as the sample size grows large. Doob (1949) showed that Bayesian methods are consistent almost surely with respect to the prior. However a prior that puts no mass near the true function will give no posterior mass near the true function. For example, suppose that $X \sim N(\theta, 1)$. If the prior is a point mass at any point, then the posterior will also be a point mass at the same point, which will be the correct answer with probability 1 with respect to the prior. However, the true answer could be any point, or any other distribution for θ , so this prior will give an incorrect posterior answer. The problem is that all other possibilities have probability zero under this particular prior. In general, one wants to avoid this possible problem and ensure that the choice of prior and estimator will be consistent over a large class of distributions.

In what follows, we will consider the consistency of density functions, and will later show that this translates into the above definition of consistency on the associated regression functions. Some of the ideas here and in the proof of Theorem 3.3.1 (in the next section) can be found in Barron et al. (1998) and Wasserman (1998b).

One possible choice of neighborhoods for density functions is the set of weak neighborhoods. Let $f_o(x, y)$ by the true underlying joint density function for x and y , and let $f_n(x, y)$ be the estimated density function. Let D_W be any distance that induces the weak topology, i.e. $D_W(f_n, f_o) \rightarrow 0$ if and only if $\int g(x) f_n(x) dx \rightarrow \int g(x) f_o(x) dx$ for every bounded, continuous function g . Then define a size ϵ weak neighborhood of f_o to be $\{f | D_W(f, f_o) < \epsilon\}$. It has been shown (Diaconis and Freedman, 1986) that having a prior which puts positive mass on weak neighborhoods of f_o does not guarantee that the posterior mass of every weak neighborhood of f_o will tend to 1.

Instead of weak neighborhoods, we use Hellinger neighborhoods when dealing with density functions. This topology is equivalent to that using L_1 neighborhoods. The Hellinger distance is defined as

$$D_H(f, f_o) = \sqrt{\iint \left(\sqrt{f(x, y)} - \sqrt{f_o(x, y)} \right)^2 dx dy}. \quad (3.2)$$

Definition 2 Suppose $(X_i, Y_i) \sim f_o$. The posterior is asymptotically consistent for f_o over Hellinger neighborhoods if

$$\text{for every } \epsilon > 0, \quad P(\{f; D_H(f, f_o) \leq \epsilon\} | (X_1, Y_1), \dots, (X_n, Y_n)) \xrightarrow{P} 1. \quad (3.3)$$

In the next section, Theorem 3.3.1 will show that the densities associated with neural networks are asymptotically consistent over Hellinger neighborhoods under certain conditions.

3.3 Asymptotic Consistency for Neural Networks

3.3.1 Sieve Asymptotics

A neural network is simply a linear combination of non-linear functions of the covariates. A popular choice of this non-linear function, also called an activation function, is the logistic function, $\Psi(z) = \frac{1}{1 + \exp(-z)}$. The network contains p inputs (explanatory variables) and k hidden nodes. At each hidden node, a weighted combination of the inputs (input weights γ_{jh}) is used as the argument for the logistic function. Finally a weighted sum of the outputs of the logistics is computed with output weights β_j . The resulting regression function is

$$E[Y_i | X_i = x_i] = \beta_o + \sum_{j=1}^k \beta_j \frac{1}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_{ih})}. \quad (3.4)$$

Here we will show the asymptotic consistency of the posterior for neural networks. We do this by estimating the joint density function $f(x, y)$, and relating it back to the conditional expectation. We take X to be uniformly distributed in $[0, 1]^p$ (since the explanatory variables are thought of as fixed, we can use any convenient distribution). Conditional on X , Y is normally distributed with mean determined by the neural network. We shall fix its standard deviation to be 1 for this calculation. Thus

$$Y | X = x \sim N \left(\beta_o + \sum_{j=1}^k \frac{\beta_j}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_h)}, 1 \right). \quad (3.5)$$

The number of input variables, p , will always be taken as fixed. The number of hidden nodes, k , will generally be allowed to grow with the sample size. Now a bit of notation needs to be introduced.

Let $f_o(x, y)$ be the true density. Define a family of neighborhoods

$$A_\epsilon = \{f; D_H(f, f_o) \leq \epsilon\} \quad \text{with } D_H(f, f_o) \text{ as defined in Equation (3.2).} \quad (3.6)$$

Let the number of hidden nodes grow such that $k_n \leq n^a$ for any $0 < a < 1$. Let \mathcal{F}_n be the set of all neural networks with each parameter less than C_n in absolute value, or equivalently, the subset of the parameter space:

$$|\gamma_{jh}| \leq C_n, \quad |\beta_j| \leq C_n \quad j = 0, \dots, k_n, \quad h = 0, \dots, p, \quad (3.7)$$

where C_n grows with n such that $C_n \leq \exp(n^{b-a})$ for any constant b such that $0 < a < b < 1$ with the a from the bound for k_n . For any $\gamma > 0$, let

$$K_\gamma = \left\{ f; E \left[\log \frac{f_o(x, y)}{f(x, y)} \right] \leq \gamma \right\}. \quad (3.8)$$

Denote the prior for f by $\pi_n(\cdot)$ and the posterior by $P(\cdot | X^n)$. Denote the predictive density by

$$\hat{f}_n(\cdot) = \int f(\cdot) dP(f | X^n). \quad (3.9)$$

The predictive density is the Bayes estimate of f . The key result is the following theorem.

Theorem 3.3.1 Suppose that: (i) there exists an $r > 0$ and an N_1 such that $\pi_n(\mathcal{F}_n^c) < \exp(-nr)$, $\forall n \geq N_1$; (ii) for all $\gamma, r > 0$, there exists an N_2 such that $\pi_n(K_\gamma) \geq \exp(-nr)$, $\forall n \geq N_2$. Then $\forall \epsilon > 0$, the posterior is asymptotically consistent for f_o over Hellinger neighborhoods, i.e.

$$P(A_\epsilon | (X_1, Y_1), \dots, (X_n, Y_n)) \xrightarrow{P} 1. \quad (3.10)$$

Corollary 3.3.1 Let $g_o(x) = E_{f_o}[Y | X = x]$ be the true regression function, and let $\hat{g}_n(x) = E_{\hat{f}_n}[Y | X = x]$ be the regression function from the predictive density using a neural network. Then under the conditions of Theorem 3.3.1, \hat{g}_n is asymptotically consistent for g_o , i.e.

$$\int (\hat{g}_n(x) - g_o(x))^2 dx \xrightarrow{P} 0. \quad (3.11)$$

The proof of this theorem requires a number of steps. The basic idea is to consider the probability of the complement as a ratio of integrals. Let $R_n(f) = \frac{\prod_{i=1}^n f(x_i, y_i)}{\prod_{i=1}^n f_o(x_i, y_i)}$. Then

$$P(A_\epsilon^c | (X_1, Y_1), \dots, (X_n, Y_n)) = \frac{\int_{A_\epsilon^c} \prod_{i=1}^n f(x_i, y_i) d\pi_n(f)}{\int \prod_{i=1}^n f(x_i, y_i) d\pi_n(f)} = \frac{\int_{A_\epsilon^c} R_n(f) d\pi_n(f)}{\int R_n(f) d\pi_n(f)}. \quad (3.12)$$

One can show that the numerator is small relative to the denominator, i.e.,

$$P(A_\epsilon^c | (X_1, Y_1), \dots, (X_n, Y_n)) \xrightarrow{P} 0.$$

One step toward bounding the numerator involves bracketing entropy.

Definition 3 For any two functions l and u , define the bracket $[l, u]$ as the set of all functions f such that $l \leq f \leq u$. Let $\|\cdot\|$ be a metric. Define an ϵ -bracket as a bracket with $\|u - l\| < \epsilon$. Define a covering of a set of functions \mathcal{F}^* as a set of brackets $\{[l_1, u_1], \dots, [l_m, u_m]\}$ such that for each $f^* \in \mathcal{F}^*$, $f^* \in [l_j, u_j]$ for some $j \in \{1, \dots, m\}$. Define the bracketing number of a set of functions \mathcal{F}^* as the minimum number of ϵ -brackets needed in a covering of \mathcal{F}^* , and denote it by $N_{[]}(\epsilon, \mathcal{F}^*, \|\cdot\|)$. Finally, the bracketing entropy, denoted $H_{[]}()$, is the natural logarithm of the bracketing number (Pollard, 1991)

To find the bracketing entropy for a neural network, one can compute the covering number and use it as an upper bound. For now, consider the number of hidden nodes, k , to be fixed. Restrict the parameter space to \mathcal{F}_n (each parameter is bounded by C_n in absolute value). Then $\mathcal{F}_n \subset \mathbb{R}^d$ where $d = (p+2)k + 1$. Denote by $N(\epsilon, \mathcal{F}_n, \|\cdot\|)$ the covering number, i.e., the minimal number of balls of radius ϵ that are required to cover the set \mathcal{F}_n under the specified metric. Using L_∞ as a metric, to cover \mathcal{F}_n with balls of radius ϵ requires no more than $(\frac{C_n+1}{\epsilon})^d$ such balls:

$$N(\epsilon, \mathcal{F}_n, L_\infty) \leq \left(\frac{2C_n}{2\epsilon} + 1 \right)^d = \left(\frac{C_n + \epsilon}{\epsilon} \right)^d \quad (3.13)$$

$$\leq \left(\frac{C_n + 1}{\epsilon} \right)^d. \quad (3.14)$$

We now apply a theorem from van der Vaart and Wellner (1996) (Theorem 2.7.4, page 164) to bound the bracketing number.

Theorem 3.3.2 Let $s, t \in \mathcal{F}_n$, i.e., s and t are realizations of the parameter vector. Let $f_t(x, y) \in \mathcal{F}^*$ be a function of x and y with parameter vector equal to t . Suppose that

$$|f_t(x, y) - f_s(x, y)| \leq d^*(s, t)F(x, y) \quad (3.15)$$

for some metric d^* , some fixed function F , every s, t , and every (x, y) . Then for any norm $\|\cdot\|$,

$$N_{[]}(\epsilon \|F\|, \mathcal{F}^*, \|\cdot\|) \leq N(\epsilon, \mathcal{F}_n, d^*). \quad (3.16)$$

Since we are interested in computing the Hellinger bracketing entropy for neural networks, we need to use the L_2 norm on the square roots of the density functions, f . The L_∞ covering number was computed above, so here $d^* = L_\infty$. The version of the theorem we are interested in is:

$$\text{If } |\sqrt{f_t(x, y)} - \sqrt{f_s(x, y)}| \leq d^*(s, t)F(x, y) \text{ for some } F, \quad (3.17)$$

$$\text{then } N_{[]} (2\epsilon \|F\|_2, \mathcal{F}^*, \|\cdot\|_2) \leq N(\epsilon, \mathcal{F}_n, d^*). \quad (3.18)$$

To show that the condition holds true, apply the Fundamental Theorem of Integral Calculus. For particular vectors s and t , let $g(u) = \sqrt{f_{((1-u)s+ut)}(x, y)}$. Let $\omega_i = (1-u)s_i + ut_i$ and denote the space of ω_i by Ω_i . Then,

$$\sqrt{f_t(x, y)} - \sqrt{f_s(x, y)} = \int_0^1 \frac{dg}{du} du \quad (3.19)$$

$$= \int_0^1 \sum_{i=1}^d \frac{\partial g}{\partial \omega_i} \frac{\partial \omega_i}{\partial u} du \quad (3.20)$$

$$= \int_0^1 \sum_{i=1}^d \frac{\partial g}{\partial \omega_i} (t_i - s_i) du \quad (3.21)$$

$$= \sum_{i=1}^d (t_i - s_i) \int_0^1 \frac{\partial g}{\partial \omega_i} du \quad (3.22)$$

$$\leq \sum_{i=1}^d \sup_i |t_i - s_i| \int_0^1 \sup_{\omega_i \in \Omega_i} \left| \frac{\partial g}{\partial \omega_i} \right| du \quad (3.23)$$

$$= \sup_i |t_i - s_i| \sum_{i=1}^d \sup_{\omega_i \in \Omega_i} \left| \frac{\partial g}{\partial \omega_i} \right| \int_0^1 du \quad (3.24)$$

$$\leq \sup_i |t_i - s_i| \sum_{i=1}^d \sup_i \left[\sup_{\omega_i \in \Omega_i} \left| \frac{\partial g}{\partial \omega_i} \right| \right] \quad (3.25)$$

$$= d \sup_i |t_i - s_i| \sup_i \left[\sup_{\omega_i \in \Omega_i} \left| \frac{\partial g}{\partial \omega_i} \right| \right] \quad (3.26)$$

$$= \|t - s\|_\infty F(x, y), \quad (3.27)$$

where $F(x, y) = d \sup_i \left[\sup_{\omega_i \in \Omega_i} \left| \frac{\partial g}{\partial \omega_i} \right| \right]$. Here $\frac{\partial g}{\partial \omega_i}$ is the partial derivative of \sqrt{f} with respect to the i th parameter. Recall that $f(x, y) = f(y|x)f(x)$ where $f(x) = 1$ since $X \sim U[0, 1]$ and $f(y|x)$

is normal with mean determined by the neural network and variance 1. Thus we have

$$\begin{aligned}
\sqrt{f(x, y)} &= (2\pi)^{-1/4} \exp \left[-\frac{1}{4} \left(y - \beta_o - \sum_{j=1}^k \frac{\beta_j}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_h)} \right)^2 \right], \\
\frac{\partial \sqrt{f}}{\partial \beta_o} &= \frac{1}{2} \left(y - \beta_o - \sum_{j=1}^k \frac{\beta_j}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_h)} \right) \sqrt{f(x, y)}, \\
\frac{\partial \sqrt{f}}{\partial \beta_c} &= \frac{1}{2} \cdot \frac{1}{1 + \exp(-\gamma_{co} - \sum_{h=1}^p \gamma_{ch} x_h)} \left(y - \beta_o - \sum_{j=1}^k \frac{\beta_j}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_h)} \right) \sqrt{f(x, y)} \\
&= \frac{1}{1 + \exp(-\gamma_{co} - \sum_{h=1}^p \gamma_{ch} x_h)} \frac{\partial \sqrt{f}}{\partial \beta_o} \quad c = 1, \dots, k, \\
\left| \frac{\partial \sqrt{f}}{\partial \beta_c} \right| &\leq \left| \frac{\partial \sqrt{f}}{\partial \beta_o} \right| \quad c = 1, \dots, k, \\
\frac{\partial \sqrt{f}}{\partial \gamma_{co}} &= -\frac{\beta_c \exp(-\gamma_{co} - \sum_{h=1}^p \gamma_{ch} x_h)}{(1 + \exp[-\gamma_{co} - \sum_{h=1}^p \gamma_{ch} x_h])^2} \frac{\partial \sqrt{f}}{\partial \beta_o} \quad c = 1, \dots, k, \\
\left| \frac{\partial \sqrt{f}}{\partial \gamma_{co}} \right| &\leq \left| C_n \frac{\partial \sqrt{f}}{\partial \beta_o} \right| \quad c = 1, \dots, k, \\
\frac{\partial \sqrt{f}}{\partial \gamma_{cd}} &= -\frac{x_d \beta_c \exp(-\gamma_{co} - \sum_{h=1}^p \gamma_{ch} x_h)}{(1 + \exp[-\gamma_{co} - \sum_{h=1}^p \gamma_{ch} x_h])^2} \frac{\partial \sqrt{f}}{\partial \beta_o} \quad c = 1, \dots, k, d = 1, \dots, p, \\
\left| \frac{\partial \sqrt{f}}{\partial \gamma_{cd}} \right| &\leq \left| x_d C_n \frac{\partial \sqrt{f}}{\partial \beta_o} \right| \quad c = 1, \dots, k, d = 1, \dots, p.
\end{aligned}$$

Now notice that $\frac{d}{dz} \left(z \exp \left(-\frac{z^2}{4} \right) \right) = \left(1 - \frac{z^2}{2} \right) \exp \left(-\frac{z^2}{4} \right)$. So $\left[z \exp \left(-\frac{z^2}{4} \right) \right]$ has a minimum of $-\sqrt{2} e^{-1/2}$ at $-\sqrt{2}$ and a maximum of $\sqrt{2} e^{-1/2}$ at $+\sqrt{2}$. Thus $\left| \frac{\partial \sqrt{f}}{\partial \beta_o} \right| \leq (8\pi e^2)^{-1/4}$, $\left| \frac{\partial \sqrt{f}}{\partial \gamma_{co}} \right| \leq C_n (8\pi e^2)^{-1/4}$, and $\left| \frac{\partial \sqrt{f}}{\partial \gamma_{cd}} \right| \leq x_d C_n (8\pi e^2)^{-1/4}$. Recalling that $X \sim U[0, 1]$, it is apparent that $\left| \frac{\partial \sqrt{f}}{\partial \gamma_{cd}} \right| \leq C_n (8\pi e^2)^{-1/4}$. Thus $\left| \frac{\partial g}{\partial \omega_i} \right| \leq C_n (8\pi e^2)^{-1/4}$ for all i which gives us $F(x, y) = dC_n (8\pi e^2)^{-1/4}$ for Equations (3.16) and (3.27). For further mathematical convenience, we will use $F(x, y) = \frac{dC_n}{2} > dC_n (8\pi e^2)^{-1/4}$.

Now apply Theorem 3.3.2 with $F = \frac{dC_n}{2}$ and L_2 for a norm on the square roots of the densities (i.e. Hellinger distance) to get that

$$N_{[]} (2\epsilon \|F\|_2, \mathcal{F}^*, \|\cdot\|_2) \leq N(\epsilon, \mathcal{F}_n, d^*) \quad \text{and hence} \tag{3.28}$$

$$N_{[]} (\epsilon, \mathcal{F}^*, \|\cdot\|_2) \leq \left(\frac{dC_n(C_n + 1)}{\epsilon} \right)^d \leq \left(\frac{d(C_n^*)^2}{\epsilon} \right)^d \quad \text{where } C_n^* = C_n + 1. \tag{3.29}$$

For notational convenience, the star will be dropped in future calculations. Having established this bound for a fixed k , we can now let k_n grow such that $k_n \leq n^a$ for any constant $0 < a < 1$. Denote by $H_{[]}(\cdot) = \log N_{[]}(\cdot)$ the Hellinger bracketing entropy.

Lemma 3.3.1 Suppose $H_{[]}(\cdot) \leq \log \left[\left(\frac{C_n^2 d_n}{u} \right)^{d_n} \right]$, $d_n = (p+2)k_n + 1$, $k_n \leq n^a$ and $C_n \leq \exp(n^{b-a})$ for $0 < a < b < 1$. Then for any fixed constants $c, \epsilon > 0$, and for all sufficiently large n , $\int_0^\epsilon \sqrt{H_{[]}(\cdot)} du \leq c\sqrt{n}\epsilon^2$.

Proof.

$$\int_0^\epsilon \sqrt{H_{[]}(\cdot)} du \leq \int_0^\epsilon \sqrt{\log \left[\left(\frac{C_n^2 d_n}{u} \right)^{d_n} \right]} du \quad \text{by Equation (3.29)} \quad (3.30)$$

$$= \sqrt{d_n} \int_0^\epsilon \sqrt{\log \frac{C_n^2 d_n}{u}} du \quad (3.31)$$

$$= \sqrt{\frac{d_n}{2}} \int_{\infty}^{\sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}}} -C_n^2 d_n v^2 e^{-v^2/2} dv \quad \text{with } v = \sqrt{2 \log \frac{C_n^2 d_n}{u}} \quad (3.32)$$

$$= C_n^2 d_n \sqrt{\frac{d_n}{2}} \int_{\sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}}}^{\infty} v^2 e^{-v^2/2} dv \quad \text{now integrate by parts} \quad (3.33)$$

$$= C_n^2 d_n \sqrt{\frac{d_n}{2}} \left[-ve^{-v^2/2} \Big|_{\sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}}}^{\infty} + \int_{\sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}}}^{\infty} e^{-v^2/2} dv \right] \quad (3.34)$$

$$= C_n^2 d_n \sqrt{\frac{d_n}{2}} \left[\frac{\epsilon}{C_n^2 d_n} \sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}} + \sqrt{2\pi} \int_{\sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}}}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-v^2/2} dv \right] \quad (3.35)$$

$$\leq C_n^2 d_n \sqrt{\frac{d_n}{2}} \left[\frac{\epsilon}{C_n^2 d_n} \sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}} + \sqrt{2\pi} \frac{\phi \left(\sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}} \right)}{\sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}}} \right] \quad \text{by Mill's Ratio} \quad (3.36)$$

$$= C_n^2 d_n \sqrt{\frac{d_n}{2}} \left[\frac{\epsilon}{C_n^2 d_n} \sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}} + \frac{\sqrt{2\pi}}{\sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}}} \frac{1}{\sqrt{2\pi}} \frac{\epsilon}{C_n^2 d_n} \right] \quad (3.37)$$

$$= \epsilon \sqrt{\frac{d_n}{2}} \sqrt{2 \log \frac{C_n^2 d_n}{\epsilon}} \left[1 + \frac{1}{2 \log \frac{C_n^2 d_n}{\epsilon}} \right] \quad (3.38)$$

$$\leq \epsilon \sqrt{d_n} \sqrt{\log \frac{C_n^2 d_n}{\epsilon}} [1 + 1] \quad (3.39)$$

$$= 2\epsilon \sqrt{d_n} \sqrt{\log C_n^2 + \log d_n - \log \epsilon}. \quad (3.40)$$

Now substitute in $d_n \leq (p+2)n^a + 1$, $C_n \leq \exp(n^{b-a})$ to get that

$$\int_0^\epsilon \sqrt{H_{[]}^{}(u)} du \leq 2\epsilon \sqrt{(p+2)n^a + 1} \sqrt{2n^{b-a} + \log((p+2)n^a + 1) - \log \epsilon}. \quad (3.41)$$

Since $0 < a < b < 1$, there exists a γ such that $a < \gamma < b$ and $b - a < 1 - \gamma$. This follows from the fact that since $0 < a < b < 1$, there must exist a $\delta > 0$ such that $a + \delta < b$ and $b + \delta < 1$. Now let $\gamma = a + \delta$ to see that $b - a = b + \delta - (a + \delta) < 1 - (a + \delta) = 1 - \gamma$.

$$\text{Hence, } \frac{1}{\sqrt{n}} \int_0^\epsilon \sqrt{H_{[]}^{}(u)} du \leq \quad (3.42)$$

$$2\epsilon \sqrt{n^{-\gamma}} \sqrt{(p+2)n^a + 1} \sqrt{n^{-(1-\gamma)}} \sqrt{2n^{b-a} + \log((p+2)n^a + 1) - \log \epsilon} = (3.43)$$

$$2\epsilon \sqrt{(p+2)n^{-(\gamma-a)} + n^{-\gamma}} \sqrt{2n^{-(1-\gamma)-(b-a)} + n^{-(1-\gamma)} \log((p+2)n^a + 1) - n^{-(1-\gamma)} \log \epsilon} (3.44)$$

$$\rightarrow 0 \quad \text{as } n \rightarrow \infty. \quad (3.45)$$

$$\text{Thus for } c, \epsilon > 0 \text{ and sufficiently large } n, \quad \frac{1}{\sqrt{n}} \int_0^\epsilon \sqrt{H_{[]}^{}(u)} du \leq c\epsilon^2. \quad (3.46)$$

And so $\int_0^\epsilon \sqrt{H_{[]}^{}(u)} du \leq c\sqrt{n}\epsilon^2$. ■

Wong and Shen (1995) showed the following result:

Theorem 3.3.3 Let $R_n(f) = \prod_{i=1}^n \frac{f(x_i, y_i)}{f_o(x_i, y_i)}$. There exist $c_1, c_2, c_3, c_4 > 0$ such that for any $\epsilon > 0$,

$$\text{if } \int_{\epsilon^2/2^8}^{\sqrt{2}\epsilon} \sqrt{H_{[]}^{}(u/c_3)} du \leq c_4 \sqrt{n}\epsilon^2, \quad (3.47)$$

$$\text{then } P^* \left(\sup_{f \in A_\epsilon^c \cap \mathcal{F}_n} R_n(f) \geq \exp(-c_1 n \epsilon^2) \right) \leq 4 \exp(-c_2 n \epsilon^2). \quad (3.48)$$

Corollary 3.3.2 Under the conditions of Theorem 3.3.3, $\sup_{f \in A_\epsilon^c \cap \mathcal{F}_n} R_n(f) \leq 4 \exp(-c_2 n \epsilon^2)$ a.s. for sufficiently large n .

Proof. From Lemma 3.3.1 we have that $\int_{\epsilon^2/2^8}^\epsilon \sqrt{H_{[]}^{}(u)} du \leq \int_0^\epsilon \sqrt{H_{[]}^{}(u)} du \leq c_4 \sqrt{n}\epsilon^2$.

Absorb the constant c_3 into the constant C_n of Lemma 3.3.1 and substitute $\sqrt{2}\epsilon$ for ϵ to get that

$$\int_{\epsilon^2/2^8}^{\sqrt{2}\epsilon} \sqrt{H_{[]}^{}(u/c_3)} du \leq 2c_4 \sqrt{n}\epsilon^2. \text{ Absorbing the 2 into } c_4 \text{ gives the conditions for Theorem 3.3.3,}$$

which implies that $P^* \left(\sup_{f \in A_\epsilon^c \cap \mathcal{F}_n} R_n(f) \geq \exp(-c_1 n \epsilon^2) \right) \leq 4 \exp(-c_2 n \epsilon^2)$. Now apply the first Borel-Cantelli Lemma. ■

We can now bound the numerator of Equation (3.12).

Lemma 3.3.2 *Let $R_n(f) = \prod_{i=1}^n \frac{f(x_i, y_i)}{f_o(x_i, y_i)}$ be the ratio of likelihoods under neural network f and the true neural network f_o . Let \mathcal{F}_n be as in Equation (3.7). Suppose that $\int_0^\epsilon \sqrt{H_{[]}^{}(u)} du \leq c\sqrt{n}\epsilon^2$ for all $\epsilon > 0$. If there exists a constant $r > 0$ such that \mathcal{F}_n satisfies $\pi_n(\mathcal{F}_n^c) < \exp(-nr) \forall n \geq N$, then there exists a constant c_2 such that $\int_{A_\epsilon^c} R_n(f) d\pi_n(f) < \exp(-\frac{nr}{2}) + \exp(-nc_2\epsilon^2)$ except on a set of probability tending to zero.*

Proof. Decompose the integral into two parts:

$$\int_{A_\epsilon^c} R_n(f) d\pi_n(f) = \int_{A_\epsilon^c \cap \mathcal{F}_n^c} R_n(f) d\pi_n(f) + \int_{A_\epsilon^c \cap \mathcal{F}_n} R_n(f) d\pi_n(f) \quad (3.49)$$

$$\leq \int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) + \int_{A_\epsilon^c \cap \mathcal{F}_n} R_n(f) d\pi_n(f). \quad (3.50)$$

The second integral is bounded as follows:

$$\int_{A_\epsilon^c \cap \mathcal{F}_n} R_n(f) d\pi_n(f) \leq \sup_{A_\epsilon^c \cap \mathcal{F}_n} R_n(f) \pi_n(A_\epsilon^c \cap \mathcal{F}_n) \quad (3.51)$$

$$\leq \sup_{A_\epsilon^c \cap \mathcal{F}_n} R_n(f) \quad (3.52)$$

$$\leq \exp(-nc_2\epsilon^2) \text{ a.s. by Corollary 3.3.2.} \quad (3.53)$$

Now let $c > 0$. Denote the sample space by \mathcal{X}^n with observed data x^n , and denote the true distribution function for the data by P_o^n . Let λ be Lebesgue measure.

$$P \left(\int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) > c \right) \leq \frac{1}{c} E \left[\int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) \right] \quad \text{by Markov's Inequality} \quad (3.54)$$

$$= \frac{1}{c} \int_{\mathcal{X}^n} \int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) dP_o^n \quad (3.55)$$

$$= \frac{1}{c} \int_{\mathcal{X}^n} \int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) \frac{dP_o^n}{d\lambda} d\lambda(x) \quad (3.56)$$

$$= \frac{1}{c} \int_{\mathcal{X}^n} \int_{\mathcal{F}_n^c} \frac{f(x^n)}{f_o(x^n)} d\pi_n(f) f_o(x^n) d\lambda(x^n) \quad (3.57)$$

$$= \frac{1}{c} \int_{\mathcal{X}^n} \int_{\mathcal{F}_n^c} f(x^n) d\pi_n(f) d\lambda(x^n) \quad (3.58)$$

$$= \frac{1}{c} \int_{\mathcal{F}_n^c} \left[\int_{\mathcal{X}^n} f(x^n) d\lambda(x^n) \right] d\pi_n(f) \text{ by Fubini's Theorem} \quad (3.59)$$

$$= \frac{1}{c} \int_{\mathcal{F}_n^c} d\pi_n(f) \quad (3.60)$$

$$= \frac{1}{c} \pi_n(\mathcal{F}_n^c) \quad (3.61)$$

$$< \frac{1}{c} \exp(-nr) \quad \text{for } n \geq N. \quad (3.62)$$

Now let $c = \exp(-\frac{nr}{2})$. Then

$$P \left(\int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) > \exp(-\frac{nr}{2}) \right) < \exp(-\frac{nr}{2}). \quad (3.63)$$

Thus $P \left(\int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) > \exp(-\frac{nr}{2}) \right) \xrightarrow{P} 0$ and so $\int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) < \exp(-\frac{nr}{2})$ except on a set with probability tending to zero. Combining the two parts of the integral gives

$$\int_{A_\epsilon^c} R_n(f) d\pi_n(f) \leq \int_{\mathcal{F}_n^c} R_n(f) d\pi_n(f) + \int_{A_\epsilon^c \cap \mathcal{F}_n} R_n(f) d\pi_n(f) \quad (3.64)$$

$$< \exp(-\frac{nr}{2}) + \exp(-nc_2\epsilon^2). \quad (3.65)$$

Where the last inequality holds except on a set with probability tending to zero. ■

Next a lower bound is found for the denominator of Equation (3.12).

Lemma 3.3.3 *Let p and q be any two density functions. Denote the Kullback-Leibler distance between p and q by $D_K(p, q)$. Then*

$$E_p \left| \log \frac{p}{q} \right| \leq D_K(p, q) + \frac{2}{e}. \quad (3.66)$$

Proof. Let $A = \left\{ x; \log \frac{p}{q} > 0 \right\}$. Then

$$E_p \left| \log \frac{p}{q} \right| = \int_A p \log \frac{p}{q} - \int_{A^c} p \log \frac{p}{q} + \int_{A^c} p \log \frac{p}{q} - \int_{A^c} p \log \frac{p}{q} \quad (3.67)$$

$$= \int p \log \frac{p}{q} - 2 \int_{A^c} p \log \frac{p}{q} \quad (3.68)$$

$$= D_K(p, q) + 2 \int_{A^c} q \frac{\log \frac{q}{p}}{p}. \quad (3.69)$$

Now note that for $z > 1$ (or indeed for $z > 0$), $\frac{\log z}{z}$ is maximized at $z = e$. So for $z > 1$, $\frac{\log z}{z} \leq \frac{1}{e}$.

So when $\log \frac{q}{p} < 0$, then $\frac{q}{p} > 1$ and thus $\frac{\log \frac{q}{p}}{p} \leq \frac{1}{e}$.

$$E_p \left| \log \frac{p}{q} \right| \leq D_K(p, q) + \frac{2}{e} \int_{A^c} q \leq D_K(p, q) + \frac{2}{e}. \quad (3.70)$$
■

Lemma 3.3.4 Let $R_n(f) = \frac{f(x_n, y_n)}{f_o(x_n, y_n)}$. For any given $\gamma > 0$, define $K_\gamma = \{f; D_K(f_o, f) \leq \gamma\}$. Let $\delta > 0$. Suppose that for all $\gamma, r > 0$, $\exists N$ s.t. $\pi_n(K_\gamma) \geq \exp(-nr) \forall n \geq N$. Then for large n , $\int R_n(f) d\pi_n(f) > e^{-n\delta}$ except on a set of probability tending to zero.

Proof. This proof is due to A. Barron (personal communication).

Let $\lambda > 0$ and define $\gamma = \delta/\lambda$. Let $m(x_n) = \int f(x_n, y_n) d\pi_n(f)$.

First we show that $D_K(f_o, m) \leq nr + n\delta/\lambda$, and then we will show that $P(R_n(f) \leq e^{-n\delta}) \xrightarrow{P} 0$.

$$D_K(f_o, m) = E \left[\log \frac{f_o}{\int f d\pi_n(f)} \right] \quad (3.71)$$

$$\leq E \left[\log \frac{f_o}{\int_{K_\gamma} f d\pi_n(f)} \right] \quad (3.72)$$

$$= E \left[\log \frac{f_o}{\pi_n(K_\gamma) \frac{\int_{K_\gamma} f d\pi_n(f)}{\pi_n(K_\gamma)}} \right] \quad (3.73)$$

$$= \log \frac{1}{\pi_n(K_\gamma)} + D_K \left(f_o, \frac{\int_{K_\gamma} f d\pi_n(f)}{\pi_n(K_\gamma)} \right) \quad (3.74)$$

$$= -\log \pi_n(K_\gamma) + D_K \left(f_o, \int f(x^n) d\pi_n(f|K_\gamma) \right) \quad (3.75)$$

$$\leq -\log \pi_n(K_\gamma) + \int_{K_\gamma} D_K(f_o, f) d\pi_n(f|K_\gamma) \text{ by Jensen's Inequality} \quad (3.76)$$

$$\leq -\log \pi_n(K_\gamma) + n\gamma \text{ by definition of } K_\gamma \quad (3.77)$$

$$\leq -\log e^{-nr} + \frac{n\delta}{\lambda} \quad (3.78)$$

$$\leq nr + \frac{n\delta}{\lambda}. \quad (3.79)$$

$$\text{So } P(R_n(f) \leq e^{-n\delta}) = P \left(\frac{m(x^n)}{f_o(x^n)} \leq e^{-n\delta} \right) \quad (3.80)$$

$$= P \left(\log \frac{f_o(x^n)}{m(x^n)} \geq n\delta \right) \quad (3.81)$$

$$\leq P \left(\log \left| \frac{f_o(x^n)}{m(x^n)} \right| \geq n\delta \right) \quad (3.82)$$

$$\leq \frac{1}{n\delta} E \left[\log \left| \frac{f_o(x^n)}{m(x^n)} \right| \right] \text{ by Markov's Inequality} \quad (3.83)$$

$$\leq \frac{1}{n\delta} \left[D_K(f_o, m) + \frac{2}{e} \right] \text{ by Lemma 3.3.3} \quad (3.84)$$

$$\leq \frac{1}{n\delta} \left[nr + \frac{n\delta}{\lambda} + \frac{2}{e} \right] \quad \text{by Equation (3.79)} \quad (3.85)$$

$$= \frac{r}{\delta} + \frac{1}{\lambda} + \frac{2}{n\delta e}, \quad (3.86)$$

$$\text{and hence } \lim_n P(R_n(f) \leq e^{-n\delta}) \leq \frac{r}{\delta} + \frac{1}{\lambda}. \quad (3.87)$$

Since the last equation holds true for all $r > 0$,

$$\lim_n P(R_n(f) \leq e^{-n\delta}) \leq \frac{1}{\lambda}. \quad (3.88)$$

And this holds true for all $\lambda > 0$, so

$$P(R_n(f) \leq e^{-n\delta}) \xrightarrow{P} 0. \quad (3.89)$$

Thus, we can conclude that for all $\delta > 0$ and for sufficiently large n , $\int R_n(f) d\pi_n(f) > e^{-n\delta}$ except on a set of probability tending to 0. ■

Now we have done the hard work for the proof of Theorem 3.3.1.

Proof of Theorem 3.3.1:

By Lemma 3.3.2, $\int_{A_\epsilon^c} R_n(f) d\pi_n(f) < \exp(-\frac{nr}{2}) + \exp(-nc_2\epsilon^2)$ for sufficiently large n .

By Lemma 3.3.4, $\int R_n(f) d\pi_n(f) \geq e^{-n\delta}$ except on a set of probability tending to 0.

$$P(A_\epsilon^c | (x_1, y_1), \dots, (x_n, y_n)) = \frac{\int_{A_\epsilon^c} R_n(f) d\pi_n(f)}{\int R_n(f) d\pi_n(f)} \quad (3.90)$$

$$< \frac{\exp(-\frac{nr}{2}) + \exp(-nc_2\epsilon^2)}{e^{-n\delta}} \quad (3.91)$$

$$= \exp\left(-n\left[\frac{r}{2} - \delta\right]\right) + \exp(-ne^2[c_2 - \delta]). \quad (3.92)$$

Now choose δ such that both $\frac{r}{2} - \delta > \xi$ and $c_2 - \delta > \xi$ where $\xi > 0$. Then

$$P(A_\epsilon^c | (x_1, y_1), \dots, (x_n, y_n)) \leq e^{-n\xi} + e^{-ne^2\xi}. \quad (3.93)$$

Thus for sufficiently large n ,

$$P(A_\epsilon^c | (x_1, y_1), \dots, (x_n, y_n)) \xrightarrow{P} 0. \quad (3.94)$$

■

Theorem 3.3.1 implies that the Hellinger distance $D_H(f_o, f) \xrightarrow{P} 0$, where f is a random draw from the posterior. What is left to show is that this also holds true for the predictive density, \hat{f}_n , and that the predictive regression function, $\hat{g}_n(x) = E_{\hat{f}_n}[Y|X]$, converges in mean square to the true regression function,

$$g_o(x) = E_{f_o}[Y|X] = \beta_o + \sum_{j=1}^{k_n} \frac{\beta_j}{1 + \exp(-\gamma_{j0} - \sum_{h=1}^p \gamma_{jh} x_h)}. \quad (3.95)$$

Proof of Corollary 3.3.1: First we show that $D_H(f_o, \hat{f}_n) \xrightarrow{P} 0$. For any $\epsilon > 0$:

$$D_H(f_o, \hat{f}_n) \leq \int D_H(f_o, f) d\pi_n(f|X^n) \quad \text{by Jensen's inequality} \quad (3.96)$$

$$= \int_{A_\epsilon} D_H(f_o, f) d\pi_n(f|X^n) + \int_{A_\epsilon^c} D_H(f_o, f) d\pi_n(f|X^n) \quad (3.97)$$

$$\leq \int_{A_\epsilon} \epsilon d\pi_n(f|X^n) + \int_{A_\epsilon^c} D_H(f_o, f) d\pi_n(f|X^n) \quad (3.98)$$

$$\leq \epsilon + \int_{A_\epsilon^c} D_H(f_o, f) d\pi_n(f|X^n). \quad (3.99)$$

The second term goes to zero in probability by Theorem 3.3.1, and ϵ is arbitrary, so $D_H(f_o, \hat{f}_n) \xrightarrow{P} 0$.

The Hellinger distance between f_o and \hat{f}_n is:

$$D_H(f_o, \hat{f}_n) = \left(\iint \left[\sqrt{\hat{f}_n(x, y)} - \sqrt{f_o(x, y)} \right]^2 dy dx \right)^{1/2} \quad (3.100)$$

$$= \left(\iint \frac{1}{\sqrt{2\pi}} \left[\exp\left(-\frac{1}{4}(y - \hat{g}_n(x))^2\right) - \exp\left(-\frac{1}{4}(y - g_o(x))^2\right) \right]^2 dy dx \right)^{1/2} \quad (3.101)$$

$$= \left(\iint \frac{1}{\sqrt{2\pi}} \left[\exp\left(-\frac{1}{2}(y - \hat{g}_n(x))^2\right) + \exp\left(-\frac{1}{2}(y - g_o(x))^2\right) - 2 \exp\left(-\frac{1}{4}[(y - \hat{g}_n(x))^2 + (y - g_o(x))^2]\right) \right] dy dx \right)^{1/2} \quad (3.102)$$

$$= \left(2 - 2 \iint \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{4}[2y^2 - 2y(\hat{g}_n(x) + g_o(x)) + \hat{g}_n(x)^2 + g_o(x)^2]\right) dy dx \right)^{1/2} \quad (3.103)$$

$$= \left(2 - 2 \iint \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(y - \frac{\hat{g}_n(x) + g_o(x)}{2}\right)^2\right] \exp\left[-\frac{1}{8}(\hat{g}_n(x) - g_o(x))^2\right] dy dx \right)^{1/2} \quad (3.104)$$

$$= \left(2 - 2 \int \exp\left[-\frac{1}{8}(\hat{g}_n(x) - g_o(x))^2\right] dx \right)^{1/2}. \quad (3.105)$$

Since $D_H(\hat{f}_n, f_o) \xrightarrow{P} 0$, we also get that $\int \exp\left[-(\hat{g}_n(x) - g_o(x))^2\right] dx \xrightarrow{P} 1$. We will now show that this implies that $(\hat{g}_n(x) - g_o(x))^2 \rightarrow 0$ a.s. on a set, Ω , with probability tending to one, and hence $\int (\hat{g}_n(x) - g_o(x))^2 dx \xrightarrow{P} 0$.

Suppose by way of contradiction that $(\hat{g}_n(x) - g_o(x))^2$ does not converge a.s. to 0 on Ω . Then there exists an $\epsilon > 0$ and a subsequence $\hat{g}_{n_i}(x)$ such that $(\hat{g}_{n_i}(x) - g_o(x))^2 > \epsilon$ on a set A where $P(A) > 0$. Now decompose our integral:

$$\int \exp\left[-(\hat{g}_n(x) - g_o(x))^2\right] dx = \int_A \exp\left[-(\hat{g}_n(x) - g_o(x))^2\right] dx + \int_{A^c} \exp\left[-(\hat{g}_n(x) - g_o(x))^2\right] dx \quad (3.106)$$

$$\leq P(A)e^{-\epsilon} + P(A^c) < 1. \quad (3.107)$$

The inequality is strict because $\epsilon > 0$ and $P(A) > 0$. But a strict inequality is a contradiction since the integral converges in probability to one. Thus $(\hat{g}_n(x) - g_o(x))^2 \rightarrow 0$ a.s. on Ω . Now apply Scheffé's Theorem to get that $\int (\hat{g}_n(x) - g_o(x))^2 dx \rightarrow 0$ a.s. on Ω and hence

$$\int (\hat{g}_n(x) - g_o(x))^2 dx \xrightarrow{P} 0. \quad (3.108)$$

■

We will now show that Theorem 3.3.1 is relevant, in that some standard choices of priors and reasonable conditions on the true regression function will satisfy the conditions of the theorem.

Theorem 3.3.4 *Let $\{\pi_n\}$ be a sequence of priors for the regression parameters, where for each n , π_n is an independent normal with mean zero and standard deviation σ for each of the parameters in the neural network. Suppose that the true regression function g_o is either continuous or square integrable. Then $\int (\hat{g}_n(x) - g_o(x))^2 dx \xrightarrow{P} 0$.*

The proof of this theorem requires some results about the approximation abilities of neural networks.

Theorem 3.3.5 *(Funahashi, 1989) Let $x \in \mathcal{X}$ be the independent variables on a compact space \mathcal{X} . If the true regression function, $g_o(x)$, is continuous, then for any given $\epsilon > 0$, there exists a neural network regression function, $g(x)$, such that*

$$\sup_{x \in \mathcal{X}} |g(x) - g_o(x)| < \epsilon. \quad (3.109)$$

Theorem 3.3.6 (Hornik et al., 1989) If μ is a probability measure on $[0, 1]^p$, then for every $\epsilon > 0$ and every g_o such that $\int g_o^2(x) d\mu(x) < \infty$, there exists a neural network g such that

$$\|g - g_o\|_2 = \sqrt{\int (g(x) - g_o(x))^2 d\mu(x)} < \epsilon. \quad (3.110)$$

We will also need the following lemma. We will continue to work with $x \in \mathcal{X} = [0, 1]^p$.

Lemma 3.3.5 Suppose that g is a neural network regression with parameters $(\theta_1, \dots, \theta_d)$, and let \tilde{g} be another neural network with parameters $(\tilde{\theta}_1, \dots, \tilde{\theta}_{\tilde{d}_n})$. Define $\theta_i = 0$ for $i > d$ and $\tilde{\theta}_j = 0$ for $j > \tilde{d}_n$. Suppose that the number of nodes of g is k , and that the number of nodes of \tilde{g} is \tilde{k}_n where $\tilde{k}_n = O(n^a)$ for some a , $0 < a < 1$. Let

$$M_\delta = \left\{ \tilde{g} \mid |\theta_i - \tilde{\theta}_i| \leq \delta \quad i = 1, 2, \dots \right\}. \quad (3.111)$$

Then for any $\tilde{g} \in M_\delta$ and for sufficiently large n ,

$$\sup_{x \in \mathcal{X}} (\tilde{g}(x) - g(x))^2 \leq (5n^a)^2 \delta^2. \quad (3.112)$$

Proof.

$$\text{Denote } g(x) = \beta_o + \sum_{j=1}^k \frac{\beta_j}{1 + \exp(-\gamma'_j x)}, \quad \tilde{g}(x) = \tilde{\beta}_o + \sum_{j=1}^{\tilde{k}_n} \frac{\tilde{\beta}_j}{1 + \exp(-\tilde{\gamma}'_j x)}. \quad (3.113)$$

$$\text{Then for any } x \in [0, 1]^p, \quad |\tilde{\gamma}'_j x - \gamma'_j x| = \left| \sum_{h=1}^p (\tilde{\gamma}_{jh} - \gamma_{jh}) x_h \right| \leq \sum_h |\tilde{\gamma}_{jh} - \gamma_{jh}| \leq p\delta. \quad (3.114)$$

Consider the following difference:

$$\left| \tilde{\beta}_j (1 + \exp(-\gamma'_j x)) - \beta_j (1 + \exp(-\tilde{\gamma}'_j x)) \right| = \left| \tilde{\beta}_j - \beta_j + \tilde{\beta}_j \exp(-\gamma'_j x) - \beta_j \exp(-\tilde{\gamma}'_j x) \right|. \quad (3.115)$$

Let $u = -\gamma'_j x$. Then $-\tilde{\gamma}'_j x = u + \xi$ for some ξ such that $|\xi| \leq p\delta$. Also note that $e^\xi < 1 + 2|\xi|$ for

$\xi < 1$. Now we have that

$$\left| \tilde{\beta}_j (1 + \exp(-\gamma'_j x)) - \beta_j (1 + \exp(-\tilde{\gamma}'_j x)) \right| \leq |\tilde{\beta}_j - \beta_j| + |\tilde{\beta}_j e^u - \beta_j e^{u+\xi}| \quad (3.116)$$

$$\leq \delta + e^u |\tilde{\beta}_j - \beta_j e^\xi| \quad (3.117)$$

$$\leq \delta + e^u |\tilde{\beta}_j - \beta_j (1 + 2|\xi|)| \quad (3.118)$$

$$\leq \delta + e^u [|\tilde{\beta}_j - \beta_j| + |2\xi\beta_j|] \quad (3.119)$$

$$\leq \delta + (\delta + 2p\delta |\beta_j|) e^u. \quad (3.120)$$

Relating this back to the neural network basis functions:

$$\left| \frac{\tilde{\beta}_j}{1 + \exp(-\tilde{\gamma}'_j x)} - \frac{\beta_j}{1 + \exp(-\gamma'_j x)} \right| = \frac{|\tilde{\beta}_j (1 + \exp(-\gamma'_j x)) - \beta_j (1 + \exp(-\tilde{\gamma}'_j x))|}{(1 + \exp(-\tilde{\gamma}'_j x))(1 + \exp(-\gamma'_j x))} \quad (3.121)$$

$$\leq \frac{\delta + (\delta + 2p\delta |\beta_j|) e^u}{(1 + \exp(-\tilde{\gamma}'_j x))(1 + e^u)} \quad (3.122)$$

$$\leq \delta + (\delta + 2p\delta |\beta_j|) = 2\delta (1 + p |\beta_j|). \quad (3.123)$$

For simplicity in notation, denote

$$\Gamma_j = \sup_{x \in \mathcal{X}} \left| \frac{\tilde{\beta}_j}{1 + \exp(-\tilde{\gamma}'_j x)} - \frac{\beta_j}{1 + \exp(-\gamma'_j x)} \right|. \quad (3.124)$$

Looking at the difference in the sums:

$$\left| \sum_{j=1}^{\tilde{k}_n} \frac{\tilde{\beta}_j}{1 + \exp(-\tilde{\gamma}'_j x)} - \frac{\beta_j}{1 + \exp(-\gamma'_j x)} \right| \leq \sum_j \Gamma_j \leq 2\delta \sum_j (1 + p |\beta_j|) \quad (3.125)$$

$$\leq 2\delta \left(n^a + \sum_j p |\beta_j| \right) \leq 4\delta n^a \text{ for large } n. \quad (3.126)$$

Then for any $x \in \mathcal{X}$:

$$(\tilde{g}(x) - g(x))^2 = \left(\tilde{\beta}_o + \sum_{j=1}^{\tilde{k}_n} \frac{\tilde{\beta}_j}{1 + \exp(-\tilde{\gamma}'_j x)} - \beta_o - \sum_{j=1}^k \frac{\beta_j}{1 + \exp(-\gamma'_j x)} \right)^2 \quad (3.127)$$

$$\leq (\tilde{\beta}_o - \beta_o)^2 + 2 |\tilde{\beta}_o - \beta_o| \sum_j \Gamma_j + \left(\sum_j \Gamma_j \right)^2 \quad (3.128)$$

$$\leq \delta^2 + 8\delta^2 n^a + 16\delta^2 n^{2a} = (1 + 4n^a)^2 \delta^2 \quad (3.129)$$

$$\leq (5n^a)^2 \delta^2. \quad (3.130)$$

■

Proof of Theorem 3.3.4: First we show that condition (i) of Theorem 3.3.1 holds.

$$\pi_n(\mathcal{F}_n^c) = \int_{\mathcal{F}_n^c} \pi_n(\theta) d\theta \quad (3.131)$$

$$\leq \sum_{i=1}^{d_n} 2 \int_{C_n}^{\infty} \phi\left(\frac{\theta_i}{\sigma}\right) d\theta_i \quad (3.132)$$

$$= d_n \left[2\sigma \int_{C_n/\sigma}^{\infty} \phi(\tau) d\tau \right] \quad (3.133)$$

$$\leq d_n \left[\frac{2\sigma\phi\left(\frac{C_n}{\sigma}\right)}{\frac{C_n}{\sigma}} \right] \quad \text{Mill's Ratio} \quad (3.134)$$

$$= d_n \left[\frac{2\sigma^2}{C_n} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{C_n^2}{2\sigma^2}\right) \right]. \quad (3.135)$$

Now take $C_n = e^{n^{b-a}}$, $0 < a < b < 1$:

$$\pi_n(\mathcal{F}_n^c) \leq d_n \left(\sigma^2 \sqrt{\frac{2}{\pi}} \right) \exp\left(-n^{b-a}\right) \exp\left(-\frac{1}{2\sigma^2} e^{2n^{b-a}}\right) \quad (3.136)$$

$$= \exp\left(-\left[n^{b-a} - \log\left(d_n \sigma^2 \sqrt{\frac{2}{\pi}}\right)\right]\right) \exp\left(-\frac{1}{2\sigma^2} e^{2n^{b-a}}\right) \quad (3.137)$$

$$d_n = (p+2)n^a + 1 < (p+3)n^a$$

$$< \exp\left(-\left[n^{b-a} - \log\left((p+3)\sigma^2 \sqrt{\frac{2}{\pi}}\right) - a \log n\right]\right) \exp\left(-\frac{1}{2\sigma^2} e^{2n^{b-a}}\right) \quad (3.138)$$

$$\leq \exp\left(-\left[\frac{1}{2}n^{b-a}\right]\right) \exp\left(-\frac{1}{2\sigma^2} e^{2n^{b-a}}\right) \quad \text{for large } n \quad (3.139)$$

$$\leq \exp\left(-\frac{1}{2\sigma^2} e^{2n^{b-a}}\right). \quad (3.140)$$

For sufficiently large n , $e^{2n^{b-a}} > n$, and so $\pi_n(\mathcal{F}_n^c) < \exp(-nr)$ where $r = \frac{1}{2\sigma^2}$.

Next we need to show that condition (ii) holds. We do this by first finding a neighborhood (M_δ) of a close approximating neural network, and then showing that this neighborhood has sufficiently large prior probability. Let g_o be the true regression function, and suppose that g_o is continuous. For any $\gamma > 0$, choose $\epsilon = \sqrt{\frac{\gamma}{2}}$ in Theorem 3.3.5 and let g be a neural network such that $\sup_{x \in \mathcal{X}} |g(x) - g_o(x)| < \epsilon$. Let $\delta = \frac{\epsilon}{5n^a} = \sqrt{\frac{\gamma}{50}}n^{-a}$ for Lemma 3.3.5. The following calculation will show that for any $\tilde{g} \in M_\delta$, $D_K(f_o, \tilde{g}) \leq \gamma$, i.e. $M_\delta \subset K_\gamma$.

$$D_K(f_o, \tilde{f}) = \iint f_o(x, y) \log \frac{f_o(x, y)}{\tilde{f}(x, y)} dy dx \quad (3.141)$$

$$= \frac{1}{2} \iint [(y - \tilde{g}(x))^2 - (y - g_o(x))^2] f_o(y|x) f_o(x) dy dx \quad (3.142)$$

$$= \frac{1}{2} \iint [-2y\tilde{g}(x) + \tilde{g}(x)^2 + 2yg_o(x) - g_o(x)^2] f_o(y|x) f_o(x) dy dx \quad (3.143)$$

$$= \frac{1}{2} \int (\tilde{g}(x) - g_o(x))^2 f_o(x) dx \quad (3.144)$$

$$= \frac{1}{2} \int (\tilde{g}(x) - g(x) + g(x) - g_o(x))^2 f_o(x) dx \quad (3.145)$$

$$\leq \frac{1}{2} \int \left[\sup_{x \in \mathcal{X}} (\tilde{g}(x) - g(x))^2 + \sup_{x \in \mathcal{X}} (g(x) - g_o(x))^2 + 2 \sup_{x \in \mathcal{X}} |\tilde{g}(x) - g(x)| \sup_{x \in \mathcal{X}} |g(x) - g_o(x)| \right] f_o(x) dx \quad (3.146)$$

$$< \frac{1}{2} \int [\epsilon^2 + \epsilon^2 + 2\epsilon^2] f_o(x) dx \quad \text{by Theorem 3.3.5 and Lemma 3.3.5} \quad (3.147)$$

$$= 2\epsilon^2 = \gamma. \quad (3.148)$$

Finally we show that for all $\gamma, r > 0$, there exists an N_r s.t. $\pi_n(K_\gamma) \geq \exp(-nr) \forall n \geq N_r$.

$$\pi_n(K_\gamma) \geq \pi_n(M_\delta) \quad (3.149)$$

$$= \prod_{i=1}^{\tilde{d}_n} \int_{\theta_i-\delta}^{\theta_i+\delta} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}u^2\right) du \quad (3.150)$$

$$\geq \prod_{i=1}^{\tilde{d}_n} 2\delta \inf_{u \in [\theta_i-\delta, \theta_i+\delta]} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}u^2\right) \quad (3.151)$$

$$\geq \prod_{i=1}^{\tilde{d}_n} 2\delta \inf_{u \in [\theta_i-1, \theta_i+1]} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}u^2\right) \quad (3.152)$$

$$= \prod_{i=1}^{\tilde{d}_n} \delta \sqrt{\frac{2}{\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\zeta_i\right), \quad \zeta_i = \max((\theta_i - 1)^2, (\theta_i + 1)^2) \quad (3.153)$$

$$\geq \left(\delta \sqrt{\frac{2}{\pi\sigma^2}} \right)^{\tilde{d}_n} \exp\left(-\frac{1}{2\sigma^2}\zeta \tilde{d}_n\right), \quad \zeta = \max_i(\zeta_1, \dots, \zeta_{\tilde{d}_n}) \quad (3.154)$$

$$= \left(\sqrt{\frac{\gamma}{50}} n^{-a} \sqrt{\frac{2}{\pi\sigma^2}} \right)^{\tilde{d}_n} \exp\left(-\frac{1}{2\sigma^2}\zeta\tilde{d}_n\right) \quad (3.155)$$

$$= \exp\left(-\tilde{d}_n \left[a \log n - \log \sqrt{\frac{\gamma}{25\pi\sigma^2}} \right] - \frac{1}{2\sigma^2}\zeta\tilde{d}_n\right) \quad (3.156)$$

$$\geq \exp\left(-\left[2a \log n + \frac{1}{2\sigma^2}\zeta\right]\tilde{d}_n\right), \quad \text{for large } n \quad (3.157)$$

$$\geq \exp\left(-\left[2a \log n + \frac{\zeta}{2\sigma^2}\right](p+3)n^a\right), \quad \tilde{d}_n = (p+2)\tilde{k}_n + 1 \leq (p+2)n^a + 1 \leq (p+3)n^a \quad (3.158)$$

$$\geq \exp(-nr), \quad \text{for any } r \text{ and for all } n \geq N_r \text{ for some } N_r. \quad (3.159)$$

One can use a similar argument to show that a neural network can approximate any L_2 function arbitrarily closely. Note that for any L_2 function h , $\|h\|_2 \geq \|h\|_1$, so

$$\text{if } \left(\int (g(x) - g_o(x))^2 d\mu(x) \right)^{1/2} < \epsilon, \quad \text{then} \quad \int |g(x) - g_o(x)| d\mu(x) < \epsilon. \quad (3.160)$$

Equations (3.144) through (3.148) now become:

$$D_K(f_o, \tilde{f}) = \frac{1}{2} \int (\tilde{g}(x) - g_o(x))^2 f_o(x) dx \quad (3.161)$$

$$= \frac{1}{2} \int (\tilde{g}(x) - g(x) + g(x) - g_o(x))^2 f_o(x) dx \quad (3.162)$$

$$\leq \frac{1}{2} \left[\int_{x \in \mathcal{X}} \sup(\tilde{g}(x) - g(x))^2 f_o(x) dx + \int (g(x) - g_o(x))^2 f_o(x) dx \right. \\ \left. + 2 \sup_{x \in \mathcal{X}} |\tilde{g}(x) - g(x)| \int |g(x) - g_o(x)| f_o(x) dx \right] \quad (3.163)$$

$$< \frac{1}{2} [\epsilon^2 + \epsilon^2 + 2\epsilon^2] \quad \text{by Theorem 3.3.6 and Lemma 3.3.5} \quad (3.164)$$

$$= 2\epsilon^2 = \gamma. \quad (3.165)$$

■

Theorem 3.3.7 *Let the prior for the regression parameters be the noninformative prior of Chapter 2, i.e., $\pi_n = \frac{1}{\sigma^2}$ over the region where $|Z^t Z| > C_n$, $|\gamma_{jh}| < D_n$ and $|\beta_j| < D_n$, for all j and h . Let $C_n \rightarrow 0$, with $C_n > 0$ for all n , and let $D_n \rightarrow \infty$ such that $D_n = o(\exp(n^r))$ for all $r > 0$. Suppose that the true regression function g_o is either continuous or square integrable. Then $\int (\hat{g}_n(x) - g_o(x))^2 dx \xrightarrow{P} 0$.*

Proof. Clearly the conditions of Theorem 3.3.1 hold. The rest of the logic from the previous proof (of Theorem 3.3.4) now gives the result. \blacksquare

3.3.2 The Number of Hidden Nodes as a Parameter

In the previous section, we let the number of hidden nodes, k_n , increase to infinity as a function of n . A different approach is to treat the number of hidden nodes as another parameter in the model and to specify a prior distribution for it. We will show that this approach also leads to an asymptotically consistent posterior.

Let $\lambda_i = P(k = i)$ be the prior probability that the number of hidden nodes is i , $\sum \lambda_i = 1$. Let π_i be the prior for the parameters of the regression equation, given that $k = i$. The joint prior for all of the parameters is $\sum_i \lambda_i \pi_i$. We now extend the result of Theorem 3.3.1.

Theorem 3.3.8 Suppose that: (i) there exists a sequence $r_i > 0$ and a sequence N_i such that for each i , $\pi_i(\mathcal{F}_n^c) < \exp(-nr_i)$ for all $n \geq N_i$; (ii) for all $\gamma, r > 0$, there exists an I and a sequence M_i such that for any $i \geq I$, $\pi_i(K_\gamma) \geq \exp(-nr)$ for all $n \geq M_i$; (iii) B_n is a bound which grows with n such that for all $r > 0$, there exists a $q > 1$ and an N such that $\sum_{i=B_n+1}^{\infty} \lambda_i < \exp(-n^q r)$ for $n \geq N$; (iv) for all i , $\lambda_i > 0$. Then for all $\epsilon > 0$,

$$P(A_\epsilon | (x_1, y_1), \dots, (x_n, y_n)) \xrightarrow{P} 1. \quad (3.166)$$

Proof. We will now show that it is still the case that $P(A_\epsilon^c | (x_1, y_1), \dots, (x_n, y_n)) \xrightarrow{P} 0$, by showing that the conditions of Theorem 3.3.1 hold for the combined prior π .

For the first condition of Theorem 3.3.1 let

$$\mathcal{F}_n^c = \bigcup_{i=0}^{\infty} \mathcal{G}_i^c \quad (3.167)$$

where \mathcal{G}_i is the set of all neural networks with i nodes and with each parameter less than C_n in

absolute value, $C_n \leq \exp(n^b)$, $0 < b < 1$. Now we have that

$$\pi(\mathcal{F}_n^c) = \sum_{i=0}^{\infty} \lambda_i \pi_i(\mathcal{G}_i^c) \quad (3.168)$$

$$\leq \sum_{i=0}^{B_n} \lambda_i \pi_i(\mathcal{G}_i^c) + \sum_{i=B_n+1}^{\infty} \lambda_i \quad (3.169)$$

$$\leq \sum_{i=0}^{B_n} \lambda_i \exp(-nr_i) + \sum_{i=B_n+1}^{\infty} \lambda_i \quad \text{by assumption (i)} \quad (3.170)$$

Letting $r^* = \min\{r_0, r_1, \dots, r_{B_n}\}$ gives

$$\pi(\mathcal{F}_n^c) \leq \exp(-nr^*) \sum_{i=0}^{B_n} \lambda_i + \sum_{i=B_n+1}^{\infty} \lambda_i \quad (3.171)$$

$$\leq \exp(-nr^*) + \sum_{i=B_n+1}^{\infty} \lambda_i. \quad (3.172)$$

By assumption (iii), $\sum_{i=B_n+1}^{\infty} \lambda_i < \exp(-n^q r^*)$ for sufficiently large n , where $q > 1$. If we now let $r = r^*/2$, then $\pi(\mathcal{F}_n^c) < \exp(-nr)$ for sufficiently large n .

Now we check the second condition of Theorem 3.3.1. For any $\gamma > 0$,

$$\pi(K_\gamma) = \sum_{i=0}^{\infty} \lambda_i \pi_i(K_\gamma) \quad (3.173)$$

$$\geq \lambda_I \pi_I(K_\gamma), \quad \text{where } I \text{ is from assumption (ii)} \quad (3.174)$$

$$\geq \lambda_I e^{-nr^*}, \quad (3.175)$$

for any r^* for sufficiently large n by assumption (ii). Since I is a constant, λ_I does not depend on n and is positive by assumption (iv). Since r^* is arbitrary, $\pi(K_\gamma) \geq e^{-nr}$ for any r for sufficiently large n . Thus the conditions of Theorem 3.3.1 hold, and so do its conclusions. ■

We can also extend the rest of the results from the preceding section in the following theorem.

Theorem 3.3.9 *Let π_i be an independent normal with mean zero and standard deviation σ for each of the i parameters. Let the prior for k be a geometric with parameter p . Let $B_n = O(n^q)$ for any $q > 1$. Suppose that the true regression function g_o is either continuous or square integrable. Then $\int (\hat{g}_n(x) - g_o(x))^2 dx \xrightarrow{P} 0$.*

Proof. We merely need to show the four conditions of Theorem 3.3.8, and then the result follows directly from that theorem and the proof of Corollary 3.3.1. Following the ideas from the proof of Theorem 3.3.4, we will show that the conditions of Theorem 3.3.8 do hold true.

Condition (i) follows from the proof of Theorem 3.3.4, Equations (3.131) through (3.140), because for any given i , $i < n^a$ for $a > 0$ for sufficiently large n .

For condition (ii), first we adapt lemma 3.3.5 to the case of a fixed number of hidden nodes i : Let $\tilde{g} \in M_\delta$ where \tilde{g} has i hidden nodes, and let g have i^* hidden nodes. Then for sufficiently large n , $\sup_{x \in \mathcal{X}} (\tilde{g}(x) - g(x))^2 \leq C^2 \delta^2$, where $C = 5[(p+2) \max\{i, i^*\} + 1]$. Now if we let I be the number of hidden nodes required by Theorems 3.3.5 or 3.3.6, it is still true that $M_\delta \subset K_\gamma$ (c.f. Equations (3.141) through (3.148)). Clearly $\pi_i(M_\delta) \geq \exp(-nr)$, and thus condition (ii) holds true.

Clearly condition (iv) is true for a geometric prior. For condition 3:

$$P(K > B_n) = \sum_{i=B_n+1}^{\infty} p(1-p)^i \quad (3.176)$$

$$= \frac{p(1-p)^{B_n+1}}{p} = (1-p)^{B_n+1} \quad (3.177)$$

$$= \exp[(B_n + 1) \log(1-p)] \quad \text{now let } B_n = n^q \quad (3.178)$$

$$= \exp[-n^q(-\log(1-p))]. \quad (3.179)$$

Since $q > 1$, there exists a q^* such that $1 < q^* < q$. Then for any $r > 0$, $P(k > B_n) \leq \exp[-n^{q^*}r]$ for sufficiently large n . ■

Note that Theorem 3.3.9 also holds true for a Poisson prior for k with $B_n = O(\exp(n^q))$ for $q > 1$. Let the parameter for the Poisson distribution be ξ . We can show condition (iii) easily by using Markov's inequality:

$$P(K \geq B_n + 1) \leq \frac{E[K]}{B_n + 1} = \frac{\xi}{B_n + 1} \leq \frac{\xi}{B_n} = \xi \exp(-n^q) < \exp(-n^{q^*}r). \quad (3.180)$$

3.4 Discussion

We have shown that using a neural network to estimate any continuous or square integrable function will be arbitrarily accurate with probability tending to one, given enough data. This is an important

result in that neural networks are in widespread use for data analysis today.

One should note that the theorems of the previous section show that the posterior is consistent both in the case that the number of hidden nodes grows to infinity, and in the case that the number of hidden nodes is a parameter which is estimated from the data. Thus one could either use a large number of hidden nodes to more simply get a good approximation, or one let the data drive the number of hidden nodes in the model. In the latter case, model selection and model averaging become important issues, and these will be discussed in a later chapter.

The results of this chapter are also further generalizable. The particular choice of an independent normal prior is sufficient, but not necessary. Clearly one could use many other priors, including hierarchical priors. One need only to ensure that they satisfy the two conditions of the main theorem. In the case where the number of nodes is also a parameter, one could use other priors for the number of nodes.

One could also adapt the results to a larger class of true regression functions. The choices of continuous and square integrable functions were made because they are a rich enough class to contain nearly all the functions in which we might be interested. There were also readily available results from the computer science literature on the asymptotic approximation abilities of neural networks for these classes of functions.

Finally, one need not take the regression error to have variance one, but could instead take $f(Y|X = x) \sim N(g(x), \sigma^2)$. This adds a nuisance parameter to the problem, but should not affect the main structure of the proof.

Chapter 4

Estimating the Normalizing Constant

4.1 Overview

To compute the Bayes factor for comparing two models, one needs the normalizing constant of the posterior. Thus this chapter is relevant to this thesis primarily because of the following chapter on model selection and model averaging. However, the problem of estimating the normalizing constant for a posterior is a difficult problem in its own right.

Denote the marginal posterior density of the data for a given model as m :

$$m = \int L(\theta|y)\pi(\theta)d\theta = \int h(\theta)d\theta, \quad (4.1)$$

where L is the likelihood under the model, π is the prior, θ is the vector of parameters, and $h(\cdot) = L(\cdot)\pi(\cdot)$, the product of the likelihood and the prior. Recall that the posterior probability of model M_i can be written as:

$$P(M_i|D) = \frac{m_i P(M_i)}{\sum_j m_j P(M_j)}, \quad (4.2)$$

where D is the data and m_i is the normalizing constant of Equation (4.1). Thus one needs to estimate the m_i in order to compare posterior probabilities. In this chapter, I will review a number of different approximation methods, and I will try them on neural network regressions on two datasets. The methods have difficulties even on these simple datasets, and I will give some possible reasons for their problems.

There have been many proposals on how to estimate these analytically intractable integrals. The methods fall primarily into two categories: numerical methods and sample-based methods. Numerical methods attempt to estimate the integral directly from the unnormalized density function and include such methods as quadrature and Monte Carlo integration. Sample-based methods rely on the fact that we can use Markov Chain Monte Carlo (MCMC) to get a sample from the posterior, and then use this sample to estimate the integral of Equation (4.1). These methods include Laplace approximations and importance sampling. Finally, the Bayesian Information Criterion (BIC) can be used to directly approximate ratios of normalizing constants (such as Bayes factors), and hence posterior probabilities.

This integration problem is difficult for several reasons. First, the integral is typically of moderately high dimension, such that evaluating the function over a grid is infeasible. In the case of a neural network, there are two additional problems. First, symmetries in the model (both in the likelihood and the prior) induce multimodality in the posterior. Second, the nonlinearity of the model results in irregularly shaped contours, which are far from the Gaussian contours that are assumed by many standard integration techniques.

As a simple example of a symmetry producing multiple equivalent peaks in the posterior, consider a neural network with two nodes. Now consider swapping the values of the parameters for the two nodes (essentially just switching the order of the nodes). The likelihood remains the same (as does the posterior for all priors which treat the nodes symmetrically, which mine do). Thus the posterior will have two symmetric peaks which correspond to the two orders of the two nodes. For k nodes, there are $k!$ such permutations.

There are several possible solutions to this problem. One could force the MCMC sample to visit all modes with equal probability, jumping between modes (Nobile, 1994). Alternatively, one could break the symmetry by using a prior which is not symmetric with respect to the ordering of the modes. Instead, my approach is to let the MCMC sample wander unrestricted (in practice, primarily around a single mode or two) and then “fold” the sample onto a single mode by re-ordering the parameters of the sample. In particular, I require that $\gamma_{11} < \gamma_{21} < \dots < \gamma_{k1}$, i.e., the slope parameter inside the logistic function of the first input variable must be increasing with respect to the node number. For the problem of estimating the area under the function, the problem

is equivalent to that of estimating the area under one of these modes and multiplying by $k!$ because of the symmetry. Thus I reflect the MCMC sample so that all samples appear to be from the same mode. This reflection technique eliminates this source of multimodality and also does not put any restrictions on the MCMC sample. An alternative approach would be to restrict the MCMC sample to a single mode, but this would hamper the mixing of the chain. My approach allows the normal mixing, but still removes the problem of multimodality by taking advantage of the symmetry in the posterior.

The second area of difficulty that is particularly relevant for neural networks is the irregular shape of the contours of the posterior. As an example of how irregular these contours can be, even in a simple problem, Figure 4.1 shows some of the contours of the log of the posterior from a two-node neural network, after integrating out all but the four γ parameters inside of the logistic functions of the two nodes. The four plots are different views of the contours for the log-posterior as a function of γ_{10} and γ_{21} , i.e., the location parameter of one node vs. the slope parameter of the other node. Note that the first three plots are local views in the neighborhood of the mode, and the last plot is a global view. In the upper left plot are simply the contours of the posterior, where the levels are chosen to be the maximum minus the .5, .95, and .99 quantiles of a chi-square with four degrees of freedom (e.g. 87.4, 81.2, and 77.4, where the maximum is at 90.7). Notice how the distribution is bimodal, with each peak being a different non-elliptical shape. The upper right plot is an image plot, where the highest levels are darkest, and the lower levels are lighter. The lower left plot is a perspective plot, also showing only those parts of the log-posterior which are no more than the .5 quantile of a χ^2_4 away from the maximum. The lower right plot shows a global view of the log-posterior.

4.2 Methods

4.2.1 Numerical Methods

A large number of numerical methods exist for approximating integrals. They fall into two main groups: quadrature methods and Monte Carlo methods. An attempt to use the quadrature methods in the IMSL software routines package (Fortran code) failed, as they consistently reported a failure

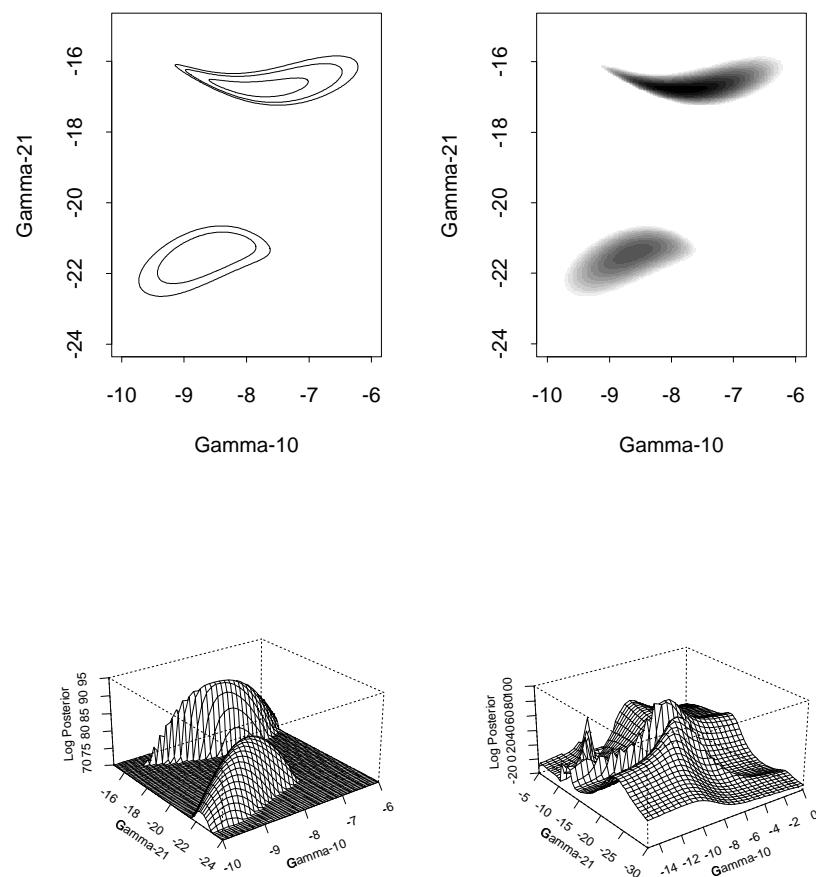


Figure 4.1: (Log) Posterior Contours

to converge. BAYESPACK (Genz and Kass, 1997) is a package of numerical integration routines designed specifically for posterior densities. It contains several Monte Carlo methods for estimating normalizing constants. These procedures were generally more successful than the IMSL routines.

4.2.2 Laplace Approximation

The Laplace approximation is based on a normal approximation to the posterior density (Tierney and Kadane, 1986). The Laplace approximation is:

$$\hat{m}_L = \frac{h(\hat{\theta})}{\phi(\hat{\theta}; \hat{\theta}, \hat{\Sigma})}, \quad (4.3)$$

where $\hat{\theta}$ is the posterior mode and $\hat{\Sigma}$ is the negative of the inverse of the observed Hessian of the log-posterior. The error is of order $O_p(n^{-1})$ in the sense that $m = \hat{m}_L(1 + O_p(n^{-1}))$. As a computational shortcut, $\hat{\Sigma}$ can be further approximated with the posterior variance-covariance matrix of the parameters.

The Laplace approximation assumes that the posterior is unimodal, which is certainly not the case for a neural network where the posterior is highly multi-modal because of the non-identifiability and symmetry of the parameters. Furthermore, there are ridges in the posterior due to the high correlation of the parameters. To improve the approximation, DiCiccio et al. (1995) suggest using only those points from the MCMC simulation in a neighborhood of the posterior mode, which they call the volume-corrected estimator. Let

$$B = \left\{ \theta \mid (\theta - \hat{\theta})^t \hat{\Sigma}^{-1} (\theta - \hat{\theta}) < \delta \right\} \quad (4.4)$$

for δ chosen such that $\Phi(B) = \int_B \phi(\theta; \hat{\theta}, \hat{\Sigma}) d\theta = \alpha$ for an appropriate choice of α , the fraction of the theoretical normal distribution in B . Also, define $P(B) = \int_B p(\theta|y) d\theta$ to be the theoretical fraction of the posterior contained in B , and let \hat{P} be the sample estimate of $P(B)$, i.e. the fraction of the MCMC sample in B . DiCiccio et al. show that the volume-corrected Laplace estimate is

$$\hat{m}_L^* = \frac{h(\hat{\theta})}{\phi(\hat{\theta}; \hat{\theta}, \hat{\Sigma})} \frac{\alpha}{\hat{P}}. \quad (4.5)$$

They also show that the order of the error for this estimator is an improvement over that of the original Laplace estimator if $mn^{-(4+p)/2} \rightarrow \infty$, where m is the simulation size, n is the sample size, and p is the dimension of the parameter space.

4.2.3 Importance Sampling

Importance sampling is a well-established method for estimating posterior quantities (Geweke, 1989). It uses a random sample from a distribution Q with density q to do Monte Carlo integration. In particular, draw $\tilde{\theta}_1, \dots, \tilde{\theta}_M$ from Q to get the estimate

$$\hat{m}_I = \frac{1}{M} \sum_i \frac{h(\tilde{\theta}_i)}{q(\tilde{\theta}_i)}. \quad (4.6)$$

The choice of Q can be problematic, since one wants to pick Q such that q is similar to h , and q has tails as thick or thicker than h . DiCiccio et al. (1995) suggest using the posterior to help choose Q . A simple choice is to use a normal distribution with mean $\hat{\theta}$ and variance-covariance matrix $\hat{\Sigma}$.

The importance sampler works best when the posterior is similar in shape to the approximating density q . For a neural network, this is unlikely to be true over the whole space. DiCiccio et al. give a volume-corrected estimator which only uses the points in the neighborhood B of Equation 4.4. This should work better for a neural network because the effect of the tails of the distribution is removed. The locally-restricted approximation is

$$\hat{m}_I^* = \frac{\frac{1}{M} \sum_i h(\tilde{\theta}_i) I_B(\tilde{\theta}_i) / q(\tilde{\theta}_i)}{\frac{1}{M} \sum_i I_B(\tilde{\theta}_i)} \quad (4.7)$$

where $I_B(\cdot)$ is the indicator function for the set B .

4.2.4 Reciprocal Importance Sampling

One drawback of importance sampling is that it requires drawing a second random sample (although typically from a simpler distribution than the posterior). One could instead use the original MCMC sample directly. The reciprocal importance sampler was introduced by Gelfand and Dey (1994):

$$\hat{m}_R = \left\{ \frac{1}{M} \sum_i \frac{q(\theta_i)}{h(\theta_i)} \right\}^{-1}, \quad (4.8)$$

where $q(\cdot)$ is an arbitrary density. The choice of q equal to the prior density leads to the harmonic mean estimator of Newton and Raftery (1994). Analogously to the importance sampler, one wants to pick q to have tails as thin or thinner than h . Failure to have appropriately-sized tails on q can lead to very high variance in the estimates.

Again, there is a local version of the reciprocal importance sampler. DiCiccio et al. define it as

$$\hat{m}_R^* = \alpha \left\{ \frac{1}{M} \sum_i \frac{q(\theta_i) I_B(\theta_i)}{h(\theta_i)} \right\}^{-1}. \quad (4.9)$$

By restricting the sample to a local area (away from the tails), one hopes to avoid the high variance that is often observed in the reciprocal importance sampler estimates.

4.2.5 Bridge Sampling

Bridge sampling (Meng and Wong, 1993) is another method that can be applied to our integral. Meng and Wong show that, for any function $a(\cdot)$ satisfying certain regularity conditions,

$$m = \frac{\int h(\theta) a(\theta) q(\theta) d\theta}{\int q(\theta) a(\theta) p(\theta|y) d\theta}, \quad (4.10)$$

where $q(\cdot)$ is the normal approximation to $p(\theta|y)$. Using our MCMC sample $\theta_1, \dots, \theta_N$ from the posterior, and also drawing a sample $\tilde{\theta}_1, \dots, \tilde{\theta}_M$ from q , we can combine these into Equation 4.10 to get the estimate

$$\hat{m}_B = \frac{\frac{1}{M} \sum_i h(\tilde{\theta}_i) a(\tilde{\theta}_i)}{\frac{1}{N} \sum_j q(\theta_j) a(\theta_j)}. \quad (4.11)$$

Meng and Wong show that the optimal choice of $a(\cdot)$ is

$$a(\theta) = \left(\frac{Nh(\theta)}{m} + Mq(\theta) \right)^{-1}. \quad (4.12)$$

As this choice of $a(\cdot)$ depends on m , it requires an iterative solution between evaluating \hat{m}_B and $a(\theta)$. Bridge sampling contains the previous sample-based methods as special cases. In my experience, the iterative bridge sampler does not always converge, and thus is less useful.

4.2.6 Path Sampling

Instead of using a single density between the sampling function and the function to be integrated (the “bridge” function), path sampling aims to create a continuous path of functions which connects the simpler sampling function to the more complex function to be integrated (Gelman and Meng, 1996). If q is our sampling function and h is our intractable function, then we index the path with

δ to get the path $q^*(\theta|\delta) = q^{1-\delta}(\theta)h^\delta(\theta)$. An estimate of the integral can be found using this path. However, this requires samples from the functions along the path, which are significantly more difficult to sample from in the case of neural networks. One would need to use Metropolis steps on all parameters, which would not be computationally feasible since one needs to generate a single sample from many different densities along the path. Hence I did not pursue this method.

4.2.7 Density-Based Approximation

Another approach is based on estimating the posterior density from the results of the MCMC simulation. Since the posterior density is

$$p(\theta|y) = \frac{L(\theta)\pi(\theta)}{m} = \frac{h(\theta)}{m}, \quad (4.13)$$

we can rewrite this to get

$$m = \frac{h(\theta)}{p(\theta|y)} = \frac{h(\theta_0)}{p(\theta_0|y)} \quad (4.14)$$

for any point θ_0 . If one could do a density estimation at θ_0 , then the above equation is simple. However, the posterior for a neural network is not unimodal and contains ridges of high posterior probability making density estimation a difficult task. The density estimation step requires user adjustment of tuning parameters, which is difficult to do in higher dimensions. Because of this added user difficulty, it would not be practical to use this method for many different models during a model selection process, so this method was not pursued further.

4.2.8 Partial Analytic Integration

As the model is a hierarchical model, there is some conditional independence amongst the parameters. It is indeed possible to analytically integrate out some of the parameters, in particular the output coefficients and the variance (the β 's and σ^2), leaving a “reduced” model. In theory, exact integration should be more accurate than using any of the above approximations for our integral.

For the importance sampler, the reduced model greatly simplifies the calculations as it removes the need to generate random draws for the variance parameter σ^2 . Thus I only use the importance

sampler and the bridge sampler for the reduced model, not for the full model. For the other approximations, I calculate estimates using both the full and reduced models.

4.2.9 BIC

The Bayesian Information Criterion (BIC) (Schwarz, 1978):

$$BIC = L - \frac{1}{2}p \log n, \quad (4.15)$$

where L is the maximum of the log-likelihood, p is the number of parameters in the model, and n is the sample size. The BIC is an approximation of the log of the Bayes factor for comparing the model to the null model (that with only an intercept term). In many cases it can be shown that the BIC minus the log of the Bayes factor is $O_p(1)$. Thus we can approximate the posterior probability of model i with

$$p(M_i | D) \approx \frac{e^{BIC_i}}{\sum_j e^{BIC_j}}, \quad (4.16)$$

where BIC_i is the BIC for the model M_i . This approximation is simpler than the others in that it only requires the maximum likelihood estimates of the parameters rather than a sample from the posterior distribution.

4.3 Numerical Comparisons of the Methods

In this section I will describe the results of comparing these methods on two different datasets. The first is a one-variable dataset that looks approximately quadratic, implying two or maybe three nodes are necessary. The second dataset is a simulated example containing two variables of interest and two noise variables, and requiring three nodes.

4.3.1 Ethanol Data

The first dataset is the ethanol data of Brinkman (1981). The data come from an automobile engine experiment where ethanol was burned as a fuel. This dataset contains 88 observations for one explanatory and one response variable, so it is a simple test of the approximation methods for

comparing different numbers of hidden nodes in the model. The explanatory variable of interest is the equivalence ratio at which the engine was run, a measure of the level of ethanol in the fuel (which is composed of ethanol and air). The response variable is the normalized concentration of nitrogen oxide emissions (NO and NO₂). The data have been scaled to the unit square (which improves computational stability).

The network was fit using the MCMC algorithm described in Chapter 2 with 25,000 burn-in runs and 100,000 runs for two, three, and four-node networks, with five repetitions for each number of nodes. Figure 4.2 shows the data and typical mean fitted functions for the models (i.e., the fitted values averaged over all of the MCMC samples). The two and three-node fitted functions are very similar, and both appear to fit quite well. The four-node fitted function displays some possible overfitting for the smaller values of the equivalence ratio, indicating visually that we would prefer a two or three-node model. Since the fits appear so similar, I expect that the various approximation methods will produce results either preferring the two-node model (for parsimony), or treating the two and three-node models approximately equally. One might expect that the four-node model will not be preferred to the smaller models because of the overfitting.

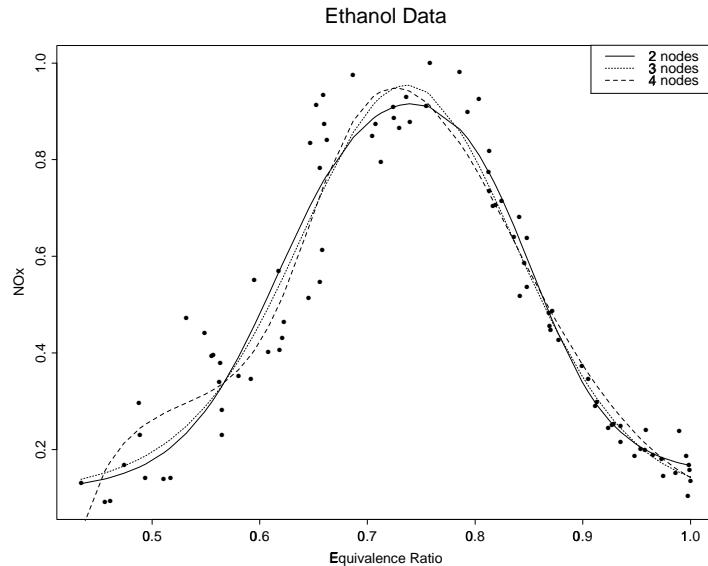


Figure 4.2: Average Fitted Functions

Despite the fact that the fitted curves appear to fit very well in all cases, and that this is one of the

simplest possible cases in that there is only one input variable and only two to four hidden nodes, the approximation methods give vastly different answers. Since the correct answer is unknown, we can't compare these results to the "true" value. However, we can at least compare them to the visual results of Figure 4.2.

Table 4.1 shows the estimates of the normalizing constants using Laplace approximation and Reciprocal Importance Sampling (RIS) on the full posterior (all parameters), and the volume-corrected versions of these two methods. Networks with two, three, and four hidden nodes are shown, and there are five repetitions for each combination of number of nodes and method. Also included in the last column is the exponentiated BIC, which should differ from the other methods only by a constant (i.e. the ratios of models should be equal for the BIC and for other methods). As one can clearly see from the table, the various methods do not agree at all. The Laplace estimates are larger, especially for the larger models. The BIC clearly picks out the two-node model, while the Laplace methods prefers the four-node model, and the RIS gives roughly equal credence to the three and four-node models. Since the four-node model seems to be over-fitting, this is evidence that we should either stick to the BIC, or that all of these methods are failing. One possible thing that may be going wrong here is that one of the parameters in the model is the variance of the error, which has a conditional posterior distribution which is inverse-gamma, not normal. This may cause problems with the sampling methods that expect normal posteriors. Another likely source of error is the irregularly shaped contours. As the parameters are highly correlated, the posteriors become harder to approximate with a normal density. This problem is aggravated as the number of nodes increases, hence the deteriorating performance of the Laplace approximation. Also note that the non-BIC methods are not very consistent across different repetitions, meaning that they are very sensitive to the MCMC sample, especially for the larger models. The sample size may also be too small for good performance of these methods.

Table 4.2 shows estimates of the normalizing constants based on a reduced posterior, where the output coefficients and the final variance parameter have been analytically integrated out. These estimates should be better than those in the previous table because there are fewer dimensions being estimated. The methods included are the Laplace approximation, the reciprocal importance sampler (RIS), the importance sample (Imp Samp), and the bridge sampler, as well as the volume-corrected

	Laplace	Laplace-VC	RIS	RIS-VC	BIC
2 Nodes	2.38e+36	2.60e+37	1.81e+36	9.03e+34	1.24e+35
	1.39e+35	2.49e+35	1.75e+35	8.77e+33	1.14e+35
	3.83e+35	9.13e+35	4.74e+35	2.37e+34	1.35e+35
	2.42e+35	6.53e+35	2.88e+35	1.44e+34	1.37e+35
	5.04e+36	6.56e+36	9.49e+35	4.75e+34	1.33e+35
3 Nodes	6.19e+44	2.77e+44	2.82e+42	1.41e+41	1.72e+32
	2.08e+46	2.76e+46	8.13e+43	4.06e+42	4.29e+32
	1.09e+47	4.91e+46	3.36e+42	1.68e+41	1.55e+32
	6.96e+45	4.98e+45	4.20e+43	2.10e+42	6.86e+32
	4.43e+45	4.12e+45	5.79e+43	2.90e+42	3.83e+32
4 Nodes	9.42e+47	5.22e+47	3.23e+42	1.62e+41	7.05e+29
	3.36e+48	1.38e+49	4.23e+43	2.11e+42	2.08e+30
	2.15e+55	2.40e+55	4.99e+47	2.50e+46	3.18e+30
	1.17e+44	5.40e+44	1.49e+41	7.44e+39	5.29e+30
	2.60e+47	9.08e+48	7.16e+44	3.58e+43	1.04e+32

Table 4.1: Normalizing Constant Estimates Based on the Full Posterior

versions of the first three. Again, there are three possible numbers of hidden nodes for the network, and five repetitions of each combination of nodes and method. The five runs are the same as in the previous table, so the results are directly comparable across the two tables.

In this table, the methods produce much more similar results, and they are also more consistent for different MCMC runs (repetitions). The volume correction plays a larger role in the larger models, where it appears to shrink the estimates a little. The Laplace approximations still seem to favor the four-node model, although not by as much as before, and the two-node model is favored over the three-node model. The reciprocal importance sampler estimates generally prefer the two-node model. The importance sampler favors the four-node model. The bridge sample supports either the two-node model, or both the two and four-node models, although it is not clear why this might be so.

Since this is a real data set and the true answer is not known, it was hoped that the BAYESPACK software (Genz and Kass, 1997) would provide a “gold standard”. However this package’s numerical integration routines did not seem to be any more reliable than the above methods. In fact, because the routines sample the height of the function in a neighborhood of the mode, but do not have

	Laplace	Laplace-VC	RIS	RIS-VC	Imp Samp	Imp Samp-VC	Bridge
2 Nodes	1.33e+38	1.24e+39	8.01e+38	4.28e+37	4.43e+38	1.23e+39	9.32e+37
	1.41e+38	3.37e+38	3.80e+38	2.33e+37	7.05e+38	3.60e+38	1.83e+38
	1.26e+38	1.61e+38	1.99e+38	1.17e+37	4.84e+38	1.69e+38	1.82e+38
	1.32e+38	3.19e+38	3.31e+38	1.91e+37	4.64e+38	3.40e+38	1.96e+38
	2.91e+38	6.97e+38	5.70e+38	3.18e+37	4.77e+38	6.40e+38	2.05e+38
3 Nodes	1.47e+38	1.67e+39	8.40e+37	4.22e+36	1.14e+39	3.65e+38	1.39e+35
	1.49e+37	6.59e+38	1.26e+38	6.31e+36	1.51e+38	8.73e+37	1.55e+35
	1.31e+40	4.35e+40	1.02e+39	5.11e+37	6.89e+39	1.92e+39	8.19e+36
	1.75e+38	2.29e+39	1.25e+38	6.26e+36	7.01e+38	2.26e+38	1.92e+35
	1.94e+37	1.69e+38	4.94e+36	2.47e+35	2.46e+38	2.80e+37	3.85e+35
4 Nodes	9.44e+37	1.06e+39	5.36e+36	2.68e+35	3.08e+40	3.70e+38	3.30e+36
	1.02e+39	4.34e+39	9.57e+37	4.79e+36	1.61e+41	2.08e+39	1.31e+37
	3.43e+40	3.40e+40	5.22e+37	2.61e+36	4.82e+41	3.68e+39	1.28e+38
	1.79e+38	2.37e+39	3.48e+37	1.74e+36	2.08e+41	4.50e+39	3.59e+36
	2.77e+41	6.77e+41	2.01e+40	1.00e+39	1.94e+42	1.60e+41	3.92e+38

Table 4.2: Normalizing Constant Estimates Based on the Reduced Posterior

the additional information about the contours from an MCMC sample (e.g., a posterior variance-covariance matrix for the parameters), they end up sampling points that fall off of the peaks because the shape of the contours is so irregular. Table 4.3 shows the results of six different numerical methods in estimating the normalizing constant, using the reduced posterior. Of note is that the routines did not work at all for the four-node model, because they always tried to evaluate the posterior at points too far off the peaks (because they didn't sample along the contours, as the MCMC methods did). The posterior then becomes numerically unstable and causes the methods to fail. This also appears to be the case for the stochastic radial-spherical method on the two-node model. Even for the cases where BAYESPACK did find a numerical answer, there was considerable disagreement between the methods, especially for the three-node model. Thus, this package was not useful in establishing a standard. It had as much trouble as the MCMC-based methods in determining the answer. Since I encountered this much trouble in this simple case (one variable, two to four nodes), I did not attempt to use this package in the next example.

	2 Nodes	3 Nodes
Monte Carlo Integration	2.9e+40	3.6e+45
Subregion Adaptive Integration	3.5e+39	5.9e+43
Mixed Spherical-Radial	1.9e+40	2.5e+43
Gauss-Hermite Integration	8.1e+39	9.5e+41
Modified Gauss-Hermite	8.1e+39	4.3e+46
Stochastic Radial-Spherical	NA	5.4e+43

Table 4.3: Numerical Estimates from BAYESPACK

4.3.2 Simulated Data

The simple ethanol data example of the previous section produced fairly muddy results from the various approximation methods. To further compare the methods, I used a simulated data set so that the true answer is known. The data consist of 120 observations. There are four explanatory variables, uniformly distributed over the four-dimensional hyper-cube, and one response variable, which is a function of only the first two explanatory variables plus some small random noise:

$$Y = \frac{1}{1 + \exp(6 - 18X_1)} - \frac{1}{1 + \exp(12 - 18X_1)} + \frac{1}{1 + \exp(15 - 15X_1 - 15X_2)} - 0.905 + \varepsilon, \quad (4.17)$$

where $\varepsilon \sim N(0, 0.01^2)$. A graph of the surface for the first two explanatory variables is shown in Figure 4.3. The graph shows that there are two things going on in the data. First, there is a “bump” along the X_1 axis, so that Y is higher for X_1 values in the middle than on the ends. Also in the data is a “wave” that starts out with low Y values in the front corner, and rises to high Y values in the rear corner, so that Y increases with the sum of X_1 and X_2 . Thus the model can be fit perfectly (without the noise) with three hidden nodes and only the first two explanatory variables. Note that interaction of X_1 and X_2 makes it difficult to fit this model with a simple additive model.

To test out the normalizing constant approximation methods, I ran a full factorial design. Models were fit with two, three, and four hidden nodes (where three is the correct number). Each possible combination of explanatory variables (from one to four at a time) was used, and each approximation method was attempted on each combination of explanatory variables and nodes. In many cases, numerical problems prevented the actual implementation of the approximations. For example, when I

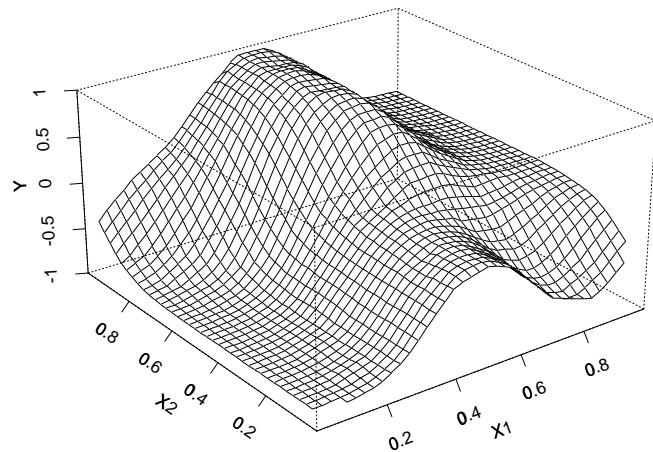


Figure 4.3: Simulated Data

fit the network using only one of the noise explanatory variables and left out the two important variables, the fitted surface was flat, and the likelihood was also flat. This leads to a sample covariance matrix with many near-zero entries which cannot be inverted, at least by standard matrix software (e.g., IMSL). So while the BIC was computed for all models, the other approximation methods are displayed for only a subset of the models, because it was not possible to use all of the approximation methods for some of the models that produced a very poor fit. In practice, this is not a problem because a numerical failure can be viewed as evidence that the model is poor. There is another case in which the computer sometimes faces numerical problems: when the model contains more hidden nodes than necessary. Sometimes superfluous nodes will turn out to be duplicates of other nodes in

Variables	2 Nodes	3 Nodes	4 Nodes
1 2 3 4	3.38e-25	1.63e-27	3.27e-30
	4.61e-38	4.31e-41	6.13e-44
	1.38e-46	1.49e-49	1.70e-52
	1.05e-45	3.22e-49	4.03e-51
1 2	2.74e+23	2.77e+155	3.14e+151
1 3	1.71e-26	3.37e-30	5.68e-33
1 4	5.29e-27	8.92e-31	1.10e-34
2 3	6.23e-39	1.01e-42	5.47e-46
2 4	9.60e-39	1.91e-41	5.57e-46
3 4	1.06e-47	6.52e-52	4.21e-56
1 2 3	5.24e+21	2.68e+152	2.31e+147
1 2 4	3.36e+22	3.17e+152	4.94e+147
1 3 4	7.65e-29	6.45e-34	4.69e-38
2 3 4	1.36e-40	2.16e-45	4.46e-50
1 2 3 4	6.09e+20	2.75e+149	1.81e+143

Table 4.4: BICs for Simulated Data

the sense that they are linearly dependent on other nodes; if we think of the nodes as basis functions, then a superfluous node is in the subspace of the previous nodes. Because of this linear dependence, we again face numerical problems in inverting matrices. Again, this is not a practical problem, in that this model would be inferior to the one without the superfluous node, so we are not losing any models of high posterior probability.

As long as we can fit the model, we can compute the BIC. Table 4.4 shows the BIC for each of the models in this simulation. The columns are the number of hidden nodes, and the rows are the combinations of explanatory variables, with the numbers indicating that that variable is present (e.g. 124 means that X_1 , X_2 , and X_4 were included in the model, and that X_3 was not). The BIC for the true model is shown in boldface type. The results in this table are quite remarkable. The BIC very clearly picks out the important models—those that include both X_1 and X_2 , and those with three or four nodes. Two hidden nodes are clearly not enough. And for a given set of variables, the three-node models dominate the four-node models, showing that only three nodes are necessary. For the models which do not include both X_1 and X_2 , the BIC is extremely small and decreases as the number of hidden nodes increases (which makes sense because we are adding nodes without

Two Hidden Nodes				
	Laplace	Laplace-VC	RIS	RIS-VC
1 2	1.53e+25	3.34e+25	0	2.27e+23
1 2 3	1.05e+29	2.07e+28	1.64e+16	2.06e+15
1 2 4	1.47e+24	1.35e+24	9.27e+22	4.63e+21
1 2 3 4	3.62e+24	1.43e+24	1.75e+22	8.74e+20

Three Hidden Nodes				
	Laplace	Laplace-VC	RIS	RIS-VC
1 2	5.70e+149	4.69e+149	1.24e+147	6.02e+145
1 2 3	4.62e+145	5.33e+145	6.69e+141	3.35e+140
1 2 4	1.48e+146	3.90e+146	1.97e+142	9.86e+140
1 2 3 4	1.51e+138	3.41e+138	7.50e+132	3.75e+131

Four Hidden Nodes				
	Laplace	Laplace-VC	RIS	RIS-VC
1 2	1.10e+159	6.56e+158	9.67e+154	4.84e+153
1 2 3	1.07e+146	7.97e+146	5.36e+139	2.68e+138
1 2 4	4.32e+146	3.56e+147	2.16e+140	1.08e+139
1 2 3 4	6.73e+135	6.10e+136	2.98e+127	1.49e+126

Table 4.5: Normalizing Constant Approximations for Simulated Data

any benefit in fit). The BIC has done its job.

Table 4.5 shows the normalizing constant estimates using the methods based on the full posterior. Again, the estimates for the true model are shown in boldface type. The methods only produced estimates for models including both X_1 and X_2 , as the covariance matrix for the parameters was typically not invertible in the other cases. The three and four-node models are clearly preferred to the two-node models, and the model with only two variables is clearly preferred to the larger models. Of note is that both the Laplace approximation and the reciprocal importance sample give significantly higher posterior probability to the four-node, two-variable model than the three-node, two-variable model (which is the correct model). One possible explanation for the confusion of these methods is the irregular contours of the posterior. Using the full set of parameters means a lot of highly correlated parameters and non-normally shaped contours. An upcoming table will present approximations based on the reduced posterior, and this simpler estimation problem proves more tractable for the approximation methods.

One other item of note is the zero entry for the reciprocal importance sampler for the two-node,

Two Hidden Nodes							
	Laplace	Lapl-VC	RIS	RIS-VC	Imp Samp	I. S.-VC	Bridge
12	1.21e+28	1.37e+28	0.0157	1.91e+26	1.55e+27	2.26e+27	5.10e+08
123	1.89e+30	3.21e+29	6.16e+16	5.44e+16	1.26e+26	2.12e+26	8.57e+10
12 4	1.10e+27	8.88e+26	5.85e+26	2.93e+25	6.10e+26	6.14e+26	5.77e+26
1234	1.88e+27	6.14e+26	7.05e+25	3.53e+24	5.11e+25	4.03e+25	1.68e+25
Three Hidden Nodes							
	Laplace	Lapl-VC	RIS	RIS-VC	Imp Samp	I. S.-VC	Bridge
12	1.14e+152	1.17e+152	1.24e+151	6.18e+149	7.31e+149	1.08e+150	3.72e+149
123	7.78e+147	2.61e+148	3.03e+146	1.51e+145	5.37e+145	1.42e+146	4.95e+145
12 4	2.83e+148	3.70e+148	2.75e+146	1.38e+145	2.62e+146	1.09e+146	1.12e+146
1234	8.45e+141	2.46e+142	3.88e+138	1.94e+137	1.52e+142	2.51e+142	3.64e+141
Four Hidden Nodes							
	Laplace	Lapl-VC	RIS	RIS-VC	Imp Samp	I. S.-VC	Bridge
12	6.03e+149	1.46e+151	7.29e+147	3.65e+146	1.25e+150	7.27e+149	1.41e+146
123	5.80e+141	1.02e+143	2.90e+138	1.45e+137	5.98e+145	5.88e+143	6.69e+141
12 4	7.46e+141	5.19e+142	4.86e+137	2.43e+136	3.88e+145	3.52e+143	1.09e+142
1234	5.97e+137	1.17e+139	1.49e+132	7.45e+130	3.60e+139	2.23e+139	8.76e+128

Table 4.6: Normalizing Constant Approximations for Simulated Data

two-variable model. This is an example of the instability of this method, as a problem in the tails of the distribution can cause catastrophic failure of the method. The local (volume-corrected) version, however, eliminates this tail problem and produces an estimate closer to the Laplace estimates.

Table 4.6 shows the normalizing constant estimates using the methods based on the reduced (partially analytically integrated) posterior. Again, the smallest two-node model presents some challenges to the methods, and the reciprocal importance sampler demonstrates its instability. Also noteworthy is the discrepancy between the bridge sampler estimate and the other estimates. Overall, the methods show a significant disagreement about the value of the normalizing constant for a particular model. However, they do generally agree that, for a given number of nodes, the only two important variables are X_1 and X_2 . And they agree that three is the right number of nodes in that two nodes are clearly insufficient, while four nodes do not significantly improve the fit. In this sense, all of the methods do find the correct answer.

4.4 Discussion

While all of the methods performed well in the relatively straightforward conditions of the simulated data (Section 4.3.2), there is a breakdown in the consensus on the simple real data example (the ethanol data of Section 4.3.1). It is not clear why the various MCMC-based approximations perform so differently from the BIC in this very simple example. I had expected that using a large (100,000) MCMC sample from the posterior along with one of the above methods would produce a better estimate of the normalizing constant than the BIC. While most methods have theoretical results showing that they are asymptotically better approximations than the BIC, this is asymptotic with respect to the sample size, so it may be that the samples in this chapter are not large enough for the asymptotics to hold. In this case, the BIC may be a more useful approximation for standard sample sizes. Furthermore, the BIC only requires computation of the maximum likelihood, rather than an MCMC sample from the posterior. In this sense, the BIC is both easier to compute and more stable, in that we are eliminating the additional variability induced by the MCMC sample. Henceforth, I will use the BIC for all model selection and model averaging purposes.

It is not clear that any of the methods are getting the correct answer. The multimodality and the irregularity of the posterior contours appear to cause problems for all of the methods. Of the available methods, the BIC seems best able to deal with these problems.

Note that even if an approximation method estimates the normalizing constant poorly, it may still be useful for model selection, as long as the rank order of the estimates of the normalizing constant are correct. In the case where the order of two models is incorrect, it is still acceptable to use the method if the two incorrectly ordered models give very similar predictions. Thus we can still do model selection even if we have difficulties estimating the normalizing constant.

Chapter 5

Model Selection and Model Averaging

5.1 Overview

5.1.1 Model Selection

When using neural networks for nonparametric regression, we are faced with two related problems of model selection: we need to choose the subset of the explanatory variables to include in the model, and we need to choose the number of hidden nodes to use in the model. These choices are important because as we include more variables and/or more hidden nodes, the fit will improve. However, at some point, adding more variables or nodes will result in over-fitting: the fit will be better for the observed data, but will result in worse predictive performance on other data. The variance for prediction is a sum of two components: the square of the bias (error between the fitted regression function and the true regression function) and the variance of the model. As we add more variables or nodes, the bias decreases, but the variance increases. Thus we need to find the right trade-off between reducing variance and reducing bias.

The approach of model selection involves picking a single best model. Philosophically, this might arise because one feels that there is a single true underlying model, and the goal of the data analysis is to identify that model. In some areas of applications, this approach may be justified. In other areas, one may doubt that an analyst could ever find the exact true model; for example, one could never measure every relevant variable and include all of them in the dataset. However, we still

might do model selection in order to find the closest approximation to the true model. For practical purposes, we may not need the exact true model, but we can find a model using our data that we feel is sufficiently close to this true model.

In the Bayesian framework, the problem of model selection is straightforward. One need merely put a prior over the space of models, then compute the posterior probabilities of all of the models and choose the model with highest posterior probability. Suppose we are comparing a set of models M_i for explaining the data, Y . Denote the prior probability of model i by $P(M_i)$ and its posterior probability by $P(M_i|Y)$. Bayes' Theorem states that:

$$P(M_i|Y) = \frac{P(Y|M_i)P(M_i)}{\sum_j P(Y|M_j)P(M_j)}, \quad (5.1)$$

where the sum is over all models. Conceptually, this process is very simple. In practice, the term $P(D|M_i)$ involves marginalizing over the parameters in the model. For a neural network, this is an analytically intractable integral. Methods for estimating this integral were reviewed in Chapter 4. In that chapter, I concluded that the BIC (Bayesian Information Criterion) (Schwarz, 1978) may be the most reliable way of estimating this quantity. Recall that the BIC for model M_i is defined as

$$BIC_i = L_i - \frac{1}{2}p_i \log n, \quad (5.2)$$

where L_i is the maximum of the log-likelihood, n is the sample size, and p_i is the number of parameters in model M_i . For the prior over the space of models, I use the noninformative prior that puts equal mass on each model, i.e. $P(M_i) = P(M_j)$ for all i and j . The BIC approximation then becomes

$$P(M_i|Y) \approx \frac{P(Y|M_i)}{\sum_j P(Y|M_j)} \approx \frac{e^{BIC_i}}{\sum_j e^{BIC_j}}. \quad (5.3)$$

Note that the Bayesian approach automatically takes care of the balance between improving fit and not overfitting, because adding additional variables or nodes that do not sufficiently improve the fit will dilute the posterior, causing a lower posterior probability for the model. This approach, in addition to being conceptually straightforward, also has the advantage that it can be used simultaneously on both the problem of choosing a subset of explanatory variables, and on the problem of choosing the number of hidden nodes for the network. Furthermore, in many cases, the BIC has been shown

to be asymptotically consistent for choosing the true model. While this has not yet been shown for neural networks, Keribin (1997) has recently shown it to be true for mixture models, which have some similarities to neural networks. So it seems plausible that it would be true for neural networks as well. Because of all of the advantages listed above, I use this approach in the rest of the thesis. For completeness, I shall now review some alternative approaches.

An alternative Bayesian-motivated approach is Automatic Relevance Detection (ARD) (MacKay, 1994; Neal, 1996). ARD utilizes an additional layer of hyperparameters in the model, where each explanatory variable has an associated hyperparameter that relates to the magnitude of the parameters associated with that explanatory variable. A prior is placed over the hyperparameters, and the full posterior for the model is computed. If any of the variable hyperparameters turns out to be small, it indicates that that variable has little effect on the model, and could be dropped. A variable with a larger effect would necessarily have a larger associated hyperparameter. There are two major drawbacks to this approach. First, one needs to decide how large is large, and how small a hyperparameter would have to be in order to cause the variable to be dropped from the model. Second, one needs to specify a prior for the hyperparameters, and the choice of prior is linked to the determination of “large” for a hyperparameter.

There are many methods for model selection, although they tend to treat the problem of selecting explanatory variables and the problem of choosing the size of the network as two separate problems, rather than dealing with them simultaneously. The two main types of methods for variable selection are cross-validation methods and likelihood-based criteria. A major drawback of many of these methods is that they were largely developed as ad hoc methods, and do not give a sound theoretical justification for why they would work. Some theoretical results have been proven about these methods, although they generally do tend to overfit the model.

Cross-validation (Stone, 1974) divides the data into equally-sized bins (sometimes as small as one observation per bin). The models under consideration are then fit using all but one bin of data, and the error on the excluded bin under each of the models is computed. This process is repeated for all of the bins, and the model with the smallest sum of squared errors is chosen as the best model. It is hoped that by leaving out a part of the data, one can get some measure of the predictive power of the model, thus reducing the tendency to overfit. Extensions to the idea of cross-validation have

produced a large family of related methods. In some specific cases, cross-validation has been shown to be asymptotically optimal (Stone, 1982), although it tends to overfit in practice.

The BIC, discussed earlier (Equation (5.2)), is a likelihood-based method. Another likelihood-based method is the AIC (Akaike Information Criterion). The AIC (Akaike, 1974) is simply the maximum of the log-likelihood minus the number of parameters in the model. The model with the largest AIC is deemed best. Thus, it attempts to balance an improvement in fit (measured with the log-likelihood) with the size of the model. Compared to the BIC, the AIC has a smaller penalty for model size. The BIC has a Bayesian motivation, in that it is an approximation to the log of the Bayes factor for comparing that model to the null model. In many classes of models, the BIC has been shown to be asymptotically consistent for choosing the true model (if one assumes that such a thing exists), and this implies that the AIC must be overfitting, since it uses a smaller penalty.

Frequentist approaches for selecting the number of hidden nodes include the methods listed above for model selection as well as a shrinkage-based approach called weight decay. Weight decay involves an additional parameter, which translates into a penalty term associated with the size of the weights. This penalty is added into the least-squares error term and this sum is then minimized to find the best model. The idea is for the unnecessary weights to be shrunk towards zero, so that these extra hidden nodes can be removed.

5.1.2 Model Averaging

An alternative to selecting a single model is model averaging (Leamer, 1978; Raftery and Madigan, 1994). One might be more interested in optimizing prediction than in finding a single best model. Or one might not subscribe to any of the philosophical arguments for using a single model, and may just want to try to use as full a model as possible. Using a single model may underestimate the prediction error, in that the uncertainty associated with choosing that model is not included in the final model. Model averaging uses a weighted average of all of the models to find predicted values. The terms in the average are weighted by their posterior probabilities. Let y be the response variable, D the data, and M_i the models of interest for $i \in \mathcal{I}$. Then the posterior predictive distribution of y is

$$P(y|D) = \sum_{i \in \mathcal{I}} P(y|D, M_i) P(M_i|D), \quad (5.4)$$

where $P(y|D, M_i)$ is the marginal posterior predictive density given a particular model (with all other parameters integrated out), and $P(M_i|D)$ is the posterior probability of model i .

When more than one model has a relatively high posterior probability, it is advantageous to use model averaging for prediction rather than simply using the single best model for prediction. Some examples of model averaging resulting in dramatic decreases in prediction errors are given in Raftery et al. (1997). Model averaging can also help when models with high posterior probability give conflicting advice, warning the user that the uncertainty associated with those predictions is much larger than one might expect if only a single model were used. Raftery (1996) gives a medical example where the predictions vary greatly between two models which both have similarly high posterior probability.

While model averaging fits naturally into the Bayesian framework, it is harder to develop a Frequentist analog. Some attempts to account for model uncertainty without the Bayesian framework include bagging (Breiman, 1994) and bumping (Tibshirani and Knight, 1995). Both of these methods use bootstrap samples of the original data to try to deal with the uncertainty in the selection of the final model used for prediction.

One drawback of the model averaging approach is that the final “model” is an average of multiple models. The result of model averaging can be difficult to interpret, since it is not a single model. In the case where the ability to interpret the final model is important, model averaging should not be used. In practice, neural networks are normally difficult to interpret on their own, so this is not much of a concern in the context of this thesis.

5.1.3 Searching the Model Space

The above methods for model selection and model averaging assume that one can compute the posterior probabilities for all of the models of interest. In practice, the space of possible models may be too large to do a complete enumeration of all of the models. For example, for p explanatory variables, there are 2^p possible subsets of variables that could be used for regression; now multiply by the number of hidden nodes that one might want to consider to get an even larger number of models over which one would need to find all of the fitted models and posterior probabilities. This is typically not feasible for a complicated model like a neural network where fitting a single model

may take some time. Thus there is a need for an automated technique to search through the model space.

Two traditional search methods are stepwise regression and Leaps and Bounds (Furnival and Wilson, 1974). In the next section, I will describe an implementation of a stepwise algorithm. Some Bayesian methods are Occam's Window (Raftery and Madigan, 1994), Markov Chain Monte Carlo Model Composition (Raftery et al., 1997), and a related technique, Bayesian Random Searching (Lee, 1996), all of which I will describe in later sections. These Bayesian methods all use approximations to Bayes factors for moving between models in the search space.

A final note about searching the model space needs to be made in the context of neural networks. In many statistical applications, one can fit the model with reasonable confidence (e.g. linear regression). However, fitting a neural network, in either a frequentist or Bayesian framework, involves the use of an iterative algorithm which could find a local maximum rather than a global maximum. One may want to keep in mind that a model visited during a search algorithm may not necessarily be fit correctly.

5.1.4 Fitting the Models

In the implementations of all of the search strategies, I use the BIC as the criterion for moving between models. Thus I do not need to take a fully Bayesian approach at this step and do not need to use MCMC to find the posteriors for each of the models under consideration. Instead, I can use standard numerical maximization methods to find the maximum likelihood estimates of the parameters and use those to find the BIC. This saves much time in computation, and given a good algorithm for finding the MLEs, could be more reliable than depending on convergence of an MCMC run.

I have implemented in SAS, using the macro language facilities, all of the search strategies I describe below. The core of the programs is code from Sarle (1994), of the SAS Institute, that uses proc NLP to find the maximum likelihood estimates of the parameters in a neural network. These estimates are all that is necessary to compute the BIC.

Once a subset of model with high posterior probability is found and those probabilities are calculated (or estimated), a more fully Bayesian approach would then call for each model of high

probability to be fit using MCMC so that one can find the posterior distribution of the parameters, conditional on that model. These full posteriors then can be used for prediction, either from a single model or from an average of models.

5.2 Stepwise Algorithm

Stepwise algorithms for finding a best model have existed in the linear model literature for some time. The basic idea is to move through the model space by taking steps consisting of adding or deleting a single variable. At each step, the process can be moving forward (adding variables) or backward (removing variables). A greedy algorithm picks the single best variable to add (remove) at each step by maximizing some criterion over the possible models that would result from adding (removing) a variable. The algorithm continues moving in the same direction until no more variables can be added (removed). The algorithm then reverses direction and attempts to remove (add) variables. This continues until the algorithm switches directions in two consecutive steps, indicating that no better models are one step away from the current model, or from any previous model visited, and so the algorithm ends.

I have implemented a stepwise algorithm in SAS. I use the BIC as the criterion for moving between models, only moving to models with a larger BIC. I also consider adding (removing) a hidden node instead of adding another explanatory variable. The basic algorithm is as follows:

1. Find the fit and BIC for the starting model (typically the null model with no parameters, or a model with all of the parameters).
2. Generate candidate models:
 - (a) If moving forward, fit all of the models with one more parameter than the current model, as well as the model with one more hidden node than the current model.
 - (b) If moving backward, fit all of the models with one fewer parameter than the current model, as well as the model with one fewer hidden node than the current model.
3. Compute the BICs for each of these candidate models. If any of the candidates has a BIC that

is larger than the BIC of the current model, then choose the candidate model with the largest BIC and make that the new current model.

4. If a better candidate model was found, then return to step 2 and continue moving in the same direction.
5. If a better candidate model was not found (all candidate models had smaller BICs than the current model), then return to step 2, but switch directions.
6. If the direction is switched in two consecutive passes, then no better models exist with either one more or one fewer variable or node, so terminate the algorithm.

Stepwise algorithms are admittedly ad hoc and may not find the model with highest posterior probability. However, they often work well in practice, and are relatively simple to implement, so they are worth trying. They tend to run into trouble when the effects of variables are correlated, so that an important interaction between two variables may not be found when the algorithm is only allowed to move in steps of one variable at a time.

Once the stepwise algorithm has been run, the user now has a subset of models that hopefully includes all of the models with the highest posterior probabilities. If one is doing model selection, then the final model of the stepwise search will be considered the best model. If one is doing model averaging, then the posterior probabilities of the models in the subset are calculated using the BICs. In either case, a more fully Bayesian analysis would then use MCMC to find the posterior for the parameters of each of the models of high posterior probability.

In practice, I have found that a particular run of the stepwise algorithm will find only a local maximum for one of the models, and so it may get stuck in a sub-optimal model. Since this results from the fitting algorithm being stuck in a local maximum, rather than in a true local maximum of the BIC, I usually fit each candidate model five times, and use the fit with the highest BIC as the candidate in the stepwise algorithm.

5.3 Occam's Window

Occam's Window (Raftery and Madigan, 1994) narrows the model space by only keeping models with high posterior probability in the set of models under consideration. In addition, the algorithm employs the principle of Occam's Razor, in that if two models have equal posterior probability, then the simpler model is to be preferred to the larger model. These two ideas form the basis of the algorithm.

The details of which models are kept under consideration and which models are excluded from the final set of models are:

1. Exclude from consideration any model with posterior probability less than $1/c$ times that of the model with highest posterior probability. $c = 20$ is suggested as a guideline, which is compared to the standard 0.05 cutoff for a p -value.
2. When comparing submodels of the current model, exclude all submodels of any submodels which have been excluded by the first rule.
3. When comparing supermodels of the current model, exclude all supermodels of any supermodels which do not have higher posterior probability than the current model.

Note that the log of the ratio of posterior probabilities can be approximated by a difference of BICs. Second, note that stronger evidence is required to exclude a smaller model than is required to exclude a larger model. This asymmetry is due to the application of Occam's Razor.

The algorithm is run as a pair of algorithms, once in an upward direction, once in a downward direction. These directions can be done in either order, so that one can start with a small (or null) model, grow that model with the up algorithm to get a set of models under consideration, then run the down algorithm on each of the models found from the up algorithm to get a final set of models. Or the algorithm could start with a large model on which the down algorithm is run. The subset of models found by the down algorithm would then be fed into the up algorithm to find the final set of models with high posterior probability.

This algorithm has a similar problem to the stepwise algorithm, in that when a candidate model is considered, one does not know if the MLE has been accurately found, or if the maximization

algorithm has merely found a local maximum. So I also recommend fitting each candidate model multiple times (I use five) and using the fit with the best BIC when decided if the model should be kept or excluded.

5.4 Markov Chain Monte Carlo Model Composition

The idea behind Markov Chain Monte Carlo Model Composition (MC³) (Raftery et al., 1997) is to create a Markov Chain with state space equal to the set of models under consideration and equilibrium distribution equal to the posterior probabilities of the models. Thus if we simulate this chain, the proportion of time that the chain spends visiting each model is a simulation-consistent estimate of the posterior probability of that model. To create such a chain, let the transition probabilities between two models be as follows:

1. The probability of moving from the current model to a model which differs by two or more parameters (differing by inclusion or exclusion) is zero.
2. The probability of moving from the current model to one which differs by exactly one variable or node (either one more or one less) is $\frac{1}{r} \min\left\{1, \frac{Pr(M'|D)}{Pr(M|D)}\right\}$, where r is the number of parameters being considered (i.e., the dimension of the search space), $Pr(M'|D)$ is the posterior probability of the model being moved to, and $Pr(M|D)$ is the posterior distribution of the model being moved from.
3. Otherwise, the chain stays in its current state.

In my implementation in SAS, I use the BIC to approximate the model posterior probabilities. When fitting the candidate model, I first fit is as if I have not done so before. I then compare the BIC for this fit to all previous times this model has been fit, and use the largest BIC over all times this model has been fit.

This algorithm has a feature which is potentially both a drawback and a benefit: it may visit the same model repeatedly, and it will fit the model as if it had not been visited before. On one hand, this is a drawback, because it may be a waste of computing power to re-compute the model after it has already been fit before. On the other hand, when dealing with iterative fitting algorithms that

might not find the optimal fit every time, MC³ has a chance to find a better fit on a second visit, so it is more tolerant of fitting difficulties. In this way, it partly avoids the optimal fit problem faced by the stepwise and Occam's Window algorithms.

5.5 Bayesian Random Searching

While MC³ may be simulation-consistent, there have not been any studies done on how long the simulation needs to run in order to reach its equilibrium state. Furthermore, the BICs of the models visited are computed, but not used directly in the final estimation of posterior probabilities. At some level, it seems wasteful to throw this information away and rely solely on the steady state properties of the chain. Instead, one could use the same Markov Chain simulation, but keep a record of all of the BICs for the models visited. To get the posterior probability of each model, one would just use the BIC if the model was visited, and exclude the model if it was not visited. In practice, this may be more accurate than MC³ because it does not rely on any of the steady state properties of the chain. The chain is merely used as a mechanism for effectively searching through the large model space, hopefully finding all of the models with high posterior probability. I used this method successfully in an earlier work (Lee, 1996), and shall refer to it as Bayesian Random Searching (BARS). This approach is similar to that of Chipman, George, and McCulloch (1998) in their implementation of Bayesian CART.

In practice, BARS seems to work well because it retains the advantages of the MCMC-based searching without relying on asymptotic approximations to the posterior probabilities. We get the BIC directly, and we have multiple chances to fit the models. These multiple visits help ensure that we can find the optimal fits of the individual models and are fairly comparing the models.

Note that BARS is applicable to a wide variety of problems, and is not just limited to neural network problems. The idea of searching the model space using a Markov Chain with transition probabilities based on the BICs of the models could be applied to many other model selection problems.

5.6 Alternative Approaches

I am not currently aware of any alternative methods that search the model space over both subsets of explanatory variables and over networks with different numbers of hidden nodes. I should, however, mention two approaches that address at least parts of the problem.

MacKay (1994) and Neal (1996) use Automatic Relevance Detection as a substitute for model averaging over the space of subsets of explanatory variables. However, they do not fully integrate this with an attempt to compute the posterior probabilities of models with different numbers of hidden nodes. While MacKay does use a Gaussian approximation for the normalizing constants (which he calls the “evidence”), he also sets aside part of the data as a test set, and trains the network on the remaining data. He then finds the error on the test set, and can choose the model with the number of nodes that minimizes the error on the test set. Neal advocates using a large number of hidden nodes and not doing any model selection over the number of nodes. He asserts that when using MCMC to fit the model, the chain will mix much better if there are more nodes than the minimum necessary for a decent fit. He proposes choosing a prior that will ensure enough shrinkage to a smoother function, so that the model does not overfit the data. This approach also avoids any attempt to estimate posterior probabilities of models.

Müller and Rios Insua (1998a) use a reversible-jump mechanism in their MCMC to move over the space of models with different numbers of hidden nodes. The reversible-jump chain does directly give estimates of the probabilities of the models, and it has the advantage of being an automated method of searching the space. However, they do not address the issue of selecting over subsets of explanatory variables. I have done some experimentation with reversible-jump methods, but I have found it very difficult to get the chain to mix between models. Since the parameters are all highly correlated with each other, adding or removing a hidden node can drastically affect all of the other parameters. This can make it difficult to generate a candidate model with any realistic chance of being accepted in the Metropolis step. This problem becomes even more difficult for the case of moving between explanatory variables.

Nodes	BIC	Posterior Probability
2	81.0	0.596
3	80.3	0.291
4	79.2	0.096
5	77.4	0.017
6	70.8	2.4e-5
7	67.3	6.9e-7
8	66.2	2.3e-7

Table 5.1: BICs for Ethanol Data from BARS

5.7 Examples

5.7.1 Ethanol Data

I tried each of the above search methods on the ethanol data (Brinkman, 1981) that I introduced in Chapter 4. This dataset contains only one explanatory variable, but it is not clear what the optimal number of hidden nodes is. In Chapter 4, the BIC called for using only two hidden nodes. Now I test the search methods of this chapter on this dataset, to see if they all come to the same conclusion.

The methods all agreed that the two-node model is the model with highest posterior probability, and could find this model whether they were started with a small or large number of hidden nodes. I will use BARS as the standard, as it seems to have the best performance. The BICs of the models visited and the associated estimated posterior probabilities are shown in Table 5.7.1. Note that the estimates in this table are from one run of the BARS algorithm. Because the neural network is fit with an iterative procedure from a random starting position, it is possible that a different fit would be found with a different start, and so the fitting algorithm also may have a local minimum problem. Thus, a repeat of this analysis may result in slightly different BIC values for the models in the table. This fitting problem is an argument in favor of the MCMC-based methods, in that they can be told to re-fit the model each time it is considered, and this may help find the best fit of each model.

Figure 5.1 shows the fitted function from model averaging using the probabilities from Table 5.7.1. The Bayesian framework allows direct estimates of the uncertainty in the fit. In the figure, the dark line is the fitted function, and the dashed lines are confidence bands. The confidence bands

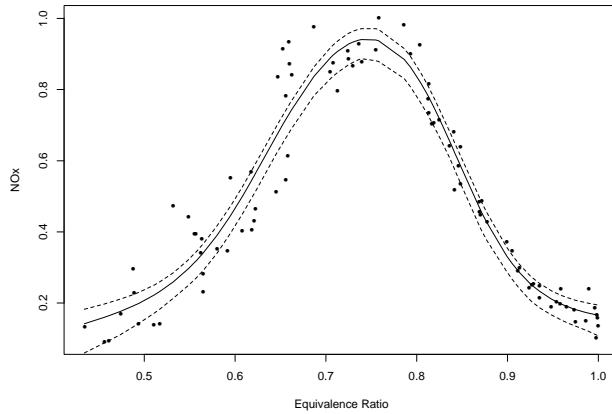


Figure 5.1: Fitted Function and Confidence Bands from Model Averaging for the Ethanol Data

were created by finding, for each observed data point, an interval (with respect to the y -axis) with 95% posterior predictive probability, and connecting the endpoints of these intervals with a curve.

BARS found that the two-node model has over half of the posterior probability. It also found that the three-node model has about three-tenths of the posterior probability, which is higher than was found in the previous chapter on normalizing constants. It now appears that the fitted values in Chapter 4 were only from a local maximum in the likelihood. However, the two-node model is still the dominant model. The posterior probabilities of the models decrease with the size of the model.

MC³ (using the same Markov Chain as BARS) found that the two-node model was the highest posterior probability model, spending 53% of its time on that model. It estimated the posterior probability of the three-node model as 29.1% and that of the four-node model as 13.6%. So it appears that MC³ and BARS give similar results for this dataset.

The stepwise algorithm would typically find the maximum BIC for the two-node model. In most cases, it would find that the three-node model was the second-best model, but it would not find a BIC as high as the 80.3 found by BARS. It typically found a BIC of 78.8, which is directly comparable to the values in Chapter 4. Thus it appears that it is difficult to find the slightly higher BIC model that BARS did manage to find. As a result, stepwise would normally estimate the posterior probability of the two-node model to be much closer to one.

Occam's Window excludes the three-node model from consideration, because the two-node

model is both more parsimonious and has a higher BIC. Thus it does not matter to Occam's Window whether it finds the optimal or the close-to-optimal three-node model, because both are excluded. The result of running Occam's Window is that the two-node model is the only one returned.

I should point out that, in running the algorithms a number of times on this same dataset, I sometimes found that both the stepwise and Occam's Window procedures would get stuck in local maxima. In particular, fitting the four-node model caused some difficulties, so that if I started these algorithms on a larger model, sometimes they would find that the five-node model was significantly better than the four-node model, only because it has more trouble fitting the four-node model. As a result, they would return the five-node model as the model with highest posterior probability.

This innocent-looking example has only one explanatory variable and apparently only needs two hidden nodes. However, it has managed to cause quite a number of problems, in both the current chapter (local maxima in the BIC) and in the previous chapter (estimating normalizing constants). It does underscore the importance of choosing a good starting model for the search algorithms, of fitting models multiple times to better ensure optimal fitting, and of running them multiple times from different starting positions to ensure adequate exploration of the model space.

5.7.2 Simulated Data

The simulated data that I introduced in Chapter 4 provides a handy test set, because we know the right answer. The dataset contains four explanatory variables, of which only the first two are relevant. The true regression function uses three hidden nodes. The test is to see if search strategies can all find the true function as the model of highest posterior probability.

The results were similar to that of the ethanol data. For starting points not too far from the true model, all of the search methods found the true model to be the only one with high posterior probability. The BIC of the three-node model with the first two explanatory variables is 359, while that of either the four-node model with the first two variables is 357, and that of either three-node model with the first two variables and one of the second two is 352. All other BICs are even smaller. These results are fairly consistent across all of the search algorithms for any reasonable choice of starting model. Occam's Window returns the two-variable, three-node model as the only model worth considering because it is a submodel of all of the other reasonable good models.

5.7.3 Robot Arm Data

A dataset commonly used in Bayesian neural network papers is the robot arm data of MacKay (1992). The idea of the data is to model the relationship between the two joint angles of the arm (x_1 and x_2) and the resulting arm position in Cartesian coordinates (denoted by y_1 and y_2). In this case, the data were actually simulated from the true functions and Gaussian noise was added. The true model is

$$y_1 = 2.0 \cos(x_1) + 1.3 \cos(x_1 + x_2) + \varepsilon_1 \quad (5.5)$$

$$y_2 = 2.0 \sin(x_1) + 1.3 \sin(x_1 + x_2) + \varepsilon_2, \quad (5.6)$$

where $\varepsilon_i \stackrel{iid}{\sim} N(0, 0.05^2)$. The values for x_1 were originally generated uniformly from the intervals $[-1.932, -0.453]$ and $[+0.453, +1.932]$, and the values for x_2 were generated independently from a uniform on $[0.534, 3.142]$.

The data are divided into two groups: a training set of 200 observations, and a test set of 200 observations. The models are fit using only the first 200 observations, and then the models can be validated on the other 200 observations. Since we have the true function, we could also generate additional data from the true model and test the fitted models on much larger sets of data. Indeed, I did generate an additional 1,000 observations from the above distributions on which I could test my fitted model.

All of the model search algorithms agree that the model that includes both explanatory variables and uses six hidden nodes is the model with highest posterior probability, and that it has nearly all of the posterior probability. The BIC of the seven-node model is about 4 smaller, which gives the six-node model about 98% of the posterior probability, and the seven-node model has the other 2%.

I then ran the MCMC code for 20,000 burn-in iterations and 20,000 draws from the posterior distribution. To get fitted values, I averaged the fitted values over each of the 20,000 draws from the posterior. Plots of the original data, the fitted values, and the residuals are shown in Figure 5.2. In the figure, the left column is y_1 and the right column is y_2 . The top row is a plot of the observed data in the training set, with y on the vertical axis and x_1 and x_2 on the bottom axes. The center row shows the fitted values on the vertical axis. The bottom row shows the residuals on the vertical axis. In all cases, the interpolation function of S-Plus has been used to help create the picture, which does

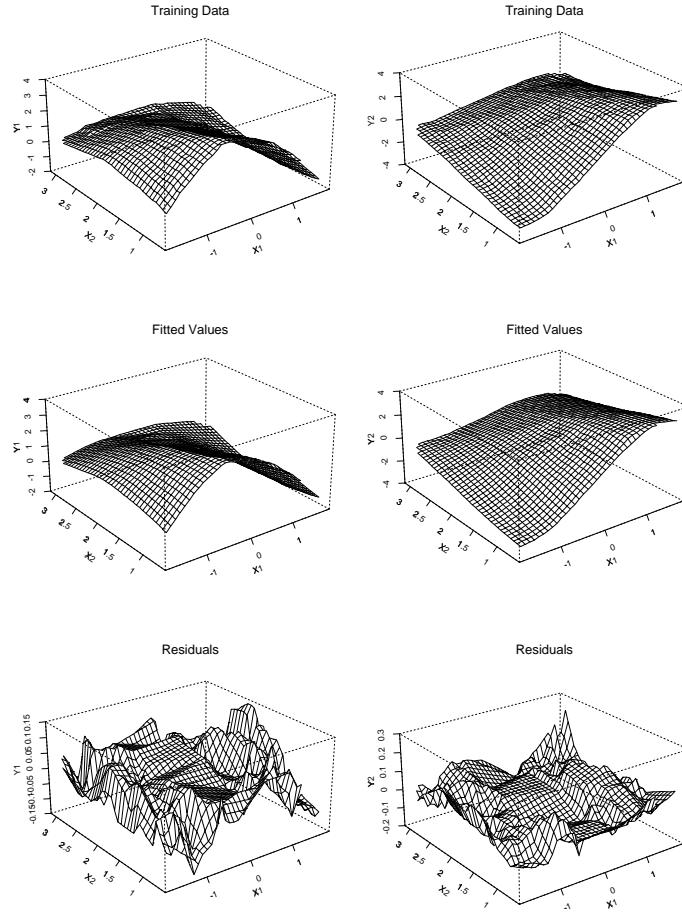


Figure 5.2: A Six-Node Model for the Robot Arm Data

involve some small amount of additional smoothing, but is the best way to get a viewable image. Notice that the residuals are reasonably scattered.

I then used the MCMC output to fit the model on the 200 cases in the test data set. The theoretical optimum value for the mean squared error (MSE) is 0.005, and my model actually managed to have an MSE of 0.005007. This is certainly an excellent fit! Plots of the test data, the fitted values, and the residuals are shown in Figure 5.3. There does seem to be some small problem with the residuals for large values of x_1 , in that for y_1 the residuals are unusually negative for large values of x_1 and small values of x_2 , and that for y_2 the residuals are unusually large for large values of x_1 and x_2 . However, the small mean square error shows that this model does indeed fit quite well. In order to

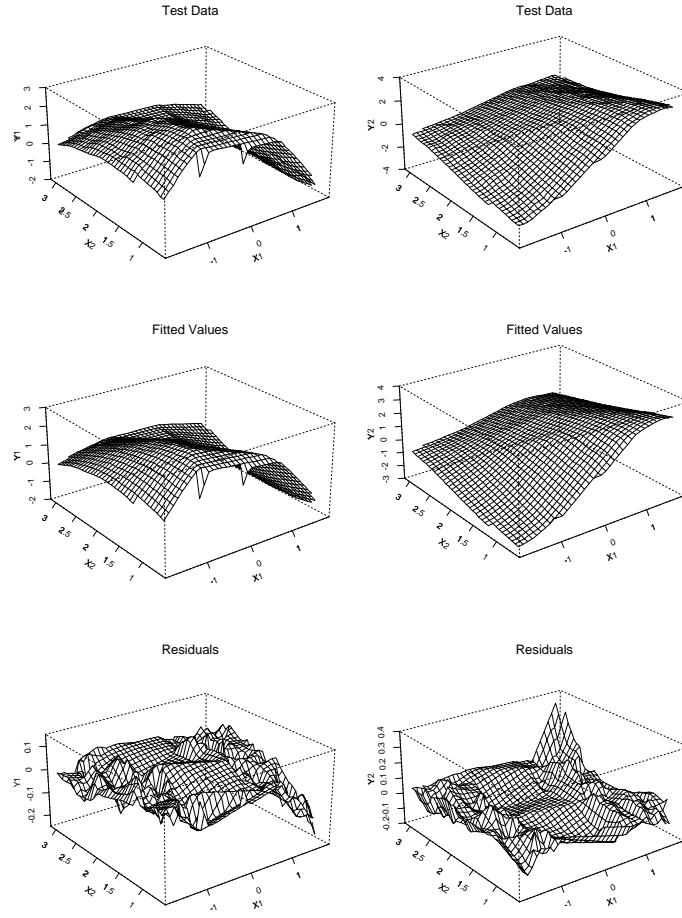


Figure 5.3: Test Data for the Robot Arm Problem

see if the fit on this particular set of test data is just a lucky break, I also generated a new test set of 1,000 observations from the original distribution and fit the model to these data. Indeed, the MSE rose to 0.00565. This is still a very good fit, and it will turn out to be comparable to the fits achieved by competing methods on the test data.

This dataset has also been analyzed by several others in the Bayesian neural network literature, in particular MacKay (1992), Neal (1996), and Müller and Rios Insua (1998b). Table 5.7.3 shows the MSE achieved on the test data by the methods of each of the above competing models. MacKay (1992) uses what he calls the “evidence” for selecting the number of nodes. This evidence is a Gaussian approximation of a Bayes factor. Neither his model with highest evidence (which has ten

Method	Mean Square Error on Test Data
MacKay, Gaussian Approximation	
Solution with highest evidence	0.00573
Solution with lowest test error	0.00557
Neal, Hybrid Monte Carlo	
16-Node Model	0.00557
16 Nodes with ARD	0.00549
Müller and Insua	
Posterior Predictions	0.00620
My model from this thesis	
Posterior Predictions	0.00501

Table 5.2: Comparison of Competing Methods on the Robot Arm Data

nodes), or even his model with lowest MSE on the test data (which has nine nodes), fit as well as the model from this thesis. Neal (1996) achieves similar results to the best model of MacKay by using his hybrid Monte Carlo methods and averaging over his runs after removing a burn-in period. Neal uses a network with sixteen hidden nodes, under the paradigm that choosing a good prior will balance the overfitting, and so one should use more nodes than necessary so the MCMC mixes well. Neal also tries applying his model selection technique of Automatic Relevance Determination (ARD), which is meant primarily to select explanatory variables by adding a hyperparameter to control the magnitude of the parameters for the explanatory variables inside the logistic functions (see Section 1.4.4). Neal demonstrates the viability of ARD by adding extra “noise” explanatory variables and then fitting a sixteen-node network to the data. In doing so, he manages to improve his fit a little in the sense that his MSE is smaller (see Table 5.7.3). In all cases, Neal is not that concerned with finding an optimal number of hidden nodes to use in his model. Finally, Müller and Rios Insua (1998b) applied their reversible-jump MCMC model to these data. They found the posterior distribution on the number of nodes to be the probabilities 0.30, 0.53, 0.15, and 0.02 for six, seven, eight, and nine nodes respectively. Using the full posterior (averaged over all model sizes) gave a set of predictions on the test data that had MSE 0.0062.

Compared to the above methods, my methods have found a model which is both more parsimonious (having fewer nodes) as well as better fitting (in that its MSE on the test data is smaller).

5.7.4 Ozone

The model search methods were applied to the groundlevel ozone data of Breiman and Friedman (1985) that was introduced in Chapter 1. There are 330 observations of the response variable, groundlevel ozone (as a pollutant), and nine explanatory variables: VH, the altitude at which the pressure is 500 millibars; WIND, the wind speed (mph) at Los Angeles International Airport (LAX); HUM, the humidity (%) at LAX; TEMP, the temperature (degrees F) at Sandburg Air Force Base; IBH, the temperature inversion base height (feet); DPG, the pressure gradient (mm Hg) from LAX to Daggert; IBT, the inversion base temperature (degrees F) at LAX; VIS, the visibility (miles) at LAX; and DAY, the day of the year.

All of the algorithms generally found that the model with nearly all of the posterior probability was one with three nodes and five variables (VH, HUM, DPG, IBT, DAY) having BIC 264. The next best models were one with six nodes and five variables (HUM, DPG, IBT, VIS, DAY) having BIC 260, and one with three nodes and three variables (HUM, IBT, DAY) having BIC 259. It did happen that sometimes the stepwise algorithm and the Occam's Window algorithm would get stuck in a local maximum of the BIC and not find the global maximum, underscoring the importance of running those algorithms with several different starting models. The MCMC-based algorithms (BARS and MC³) did not have this problem and could typically find the global maximum.

Method	R^2
ACE, 9 variables	0.82
ACE, 4 variables	0.78
GAM	0.80
TURBO	0.80
Box-Tidwell	0.82
Neural Network	0.79

Table 5.3: Comparison of Competing Methods on the Ozone Data

This dataset has been analyzed by several others in the nonparametric regression literature, and so it is useful for comparing the methods of this thesis to other nonparametric regression techniques.

Breiman and Friedman (1985) used this dataset in their paper on Alternating Conditional Expectation (ACE). As a goodness-of-fit measure, they used the estimated multiple correlation coefficient, R^2 . They fit the model both using all nine explanatory variables, as well as a subset of only four that were chosen via a stepwise algorithm (the four are TEMP, IBH, DPG, and VIS). The comparison of the R^2 's is shown in Table 5.7.4. Hastie and Tibshirani (1984) fit a Generalized Additive Model (GAM) to the data. Friedman and Silverman (1989) fit the data using TURBO. In the discussion of the previous paper, Hawkins (1989) fit the data with linear regression after using Box-Tidwell style transformations on the variables. For comparison, I include the result of my model with three nodes and five explanatory variables. Table 5.7.4 shows that all of the above methods have similar goodness-of-fit to the data. All of the methods do manage to find a reasonable fit, but none is clearly better than the others.

Method	VH	WIND	HUM	TEMP	IBH	DPG	IBT	VIS	DAY
Stepwise ACE				X	X	X		X	
Stepwise GAM	X	X	X	X	X	X		X	X
TURBO	X			X	X	X		X	X
BRUTO				X	X	X		X	X
Optimal Neural Net (3 Nodes)	X			X		X	X		X
Second-best NN (6 Nodes)				X		X	X	X	X
Third-best NN (3 Nodes)				X			X		X

Table 5.4: Comparison of Variable Selection on the Ozone Data

Aside from the ACE model with only four variables, the other models in Table 5.7.4 all use more explanatory variables than does the neural network, and are thus less parsimonious and subject to increased prediction error. Hastie and Tibshirani (1990) do a comparison of several methods on these data in terms of variable selection. In addition to some of the above methods, they also include a stepwise algorithm for their GAM models, as well as a response to TURBO which they call BRUTO, which is meant to do automatic variable selection and smoothing parameter selection. Table 5.7.4 shows the variables chosen by the models in each of these methods. It is interesting to note that the methods seriously disagree on which variables to select. Partly, this may be because the variables are highly correlated with each other, so that different subsets may give similar predictions.

However, TURBO and BRUTO are largely in agreement with each other. And the three neural network models have similar choices of variables, although these are very different from those of TURBO and BRUTO. At the very least, it does seem clear that some variable selection is necessary because of the high level of correlation between the explanatory variables, even if there is dissent about which subset is optimal.

5.8 Discussion

Computing posterior probabilities for a collection of neural network models is a very difficult task. In this chapter, I have presented a method for searching the model space and finding models with high posterior probability. My approach allows one to simultaneously search across subsets of explanatory variables and across networks of different sizes, which the competing methods do not fully do. The BIC provides a reasonable approximation for the posterior probabilities. The BARS approach provides a useful method for finding the models of highest posterior probability in the expansive model space, although one can usually achieve similar results by running several stepwise algorithms from different starting models. Tests on several canonical examples show that my approach performs well when compared to other methods. I can often achieve a similar goodness-of-fit while using a smaller, more parsimonious model.

The methods of this chapter allow one to more fully account for uncertainty. Model averaging can help account for the uncertainty associated with choosing a model. Since the model space searching techniques do not actually visit all possible models, there is still some uncertainty still unaccounted for due to the unvisited models, although this is typically unavoidable due to the vast size of the model space; one hopes that the search does find all important models, so that the additional uncertainty is negligible. Finally, it is useful to note that the Bayesian framework allows easy computation of further measures of uncertainty. For example, it is straightforward to get confidence bands for the fitted regression function, as shown in Figure 5.1.

Chapter 6

Conclusions

This thesis provides a complete methodology for Bayesian nonparametric regression using neural networks. It describes how to use a noninformative prior, and shows that this prior and many other reasonable priors are asymptotically consistent for the posterior. It describes how to search the model space for models of high posterior probability using Bayesian Random Searching (BARS) as well as several other methods. Then one can either use the single model of highest posterior probability, or average over the posterior distribution of models. The examples in Chapter 5 show that the models that result from the methods in this thesis fit the data well and perform well when compared to competing methods. These methods also allow one to more fully account for uncertainty, and the Bayesian framework allows straightforward uncertainty calculations.

This thesis contributes to the body of statistical research in several ways. It provides the first implementation of a fully noninformative prior for neural networks. It is the first detailed proof of the asymptotic consistency of neural networks for the posterior. It surveys many methods for estimating integrals (normalizing constants), and finds that the BIC is one of the most robust. Finally, it contributes to the field of model selection and model averaging. BARS may exist by another name in the literature, but I am not familiar with published applications of it. Little model selection work has been done for models that are fit iteratively, where one is not guaranteed to have fit the model correctly, and so one may have further trouble searching the model space because of this uncertainty. Model selection for Bayesian neural networks has been incomplete in that no one else

uses the full posterior for selecting both the number of hidden nodes and the subset of explanatory variables. This thesis fills in that gap.

While the work in this thesis feels largely complete, there are some directions for future research. One obvious extension to this work is to apply the model selection results to models for binary, categorical, or ordinal explanatory variables. This may involve either the models of Chapter 2 or models with alternative probability distributions (e.g., multinomial instead of Gaussian error). A major theoretic result that would bolster this thesis would be the proof that the BIC minus the log of the Bayes factor for comparing that model to the null model is $O_p(1)$ and that the BIC is asymptotically consistent for selecting the true model. This has been shown true in many other cases, but not yet for neural networks. A proof of the approximation abilities of the BIC would be useful in further justifying its use for the model search process. Finally, there are some computational loose ends. Chapter 4 seems somewhat incomplete in that I could not find a method for approximating the normalizing constants that was more accurate than the BIC, which is not a high-order approximation. Perhaps more work in this area could produce better numerical integration and approximation techniques. Also, the current implementations of the model searching algorithms are far from maximally efficient. There is much room for improvement in terms of computational speed. In particular, there may be ways to allocate how much time one should spend trying to fit each particular model in that models with potentially high posterior probability should be given more attention than those that have exceedingly small posterior probability (because they don't fit well, perhaps because they have left out a necessary explanatory variable). Because of the uncertainty in fitting an individual model, it is sometimes hard to tell if the model is bad because of a poor fitting job or because the model is actually bad. Knowing which models one should pay more attention to fitting carefully (and thus consuming more computing time) would help the efficiency of the algorithms.

References

- Akaike, H. (1974). “A New Look at Statistical Model Identification.” *IEEE Transactions on Automatic Control*, AU–19, 716–722.
- Anderson, J. A. (1982). “Logistic Discrimination.” In *Classification, Pattern Recognition and Reduction of Dimensionality*, eds. P. R. Krishnaiah and L. N. Kanal, vol. 2 of *Handbook of Statistics*, 169–191. Amsterdam: North Holland.
- Banks, D. L., Maxion, R. A., and Olszewski, R. T. (1997). “Comparing Methods for Multivariate Nonparametric Regression.” Submitted to *Journal of Computational and Graphical Statistics*.
- Barron, A., Schervish, M. J., and Wasserman, L. (1998). “The Consistency of Posterior Distributions in Nonparametric Problems.” *The Annals of Statistics*. To appear.
- Bernardo, J. M. (1979). “Reference Posterior Distributions for Bayesian Inference (with discussion).” *Journal of the Royal Statistical Society B*, 41, 113–147.
- Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford: Clarendon Press.
- Breiman, L. (1994). “Bagging Predictors.” Tech. Rep. 421, University of California, Berkeley, Department of Statistics.
- Breiman, L. and Friedman, J. H. (1985). “Estimating Optimal Transformations for Multiple Regression and Correlation.” *Journal of the American Statistical Association*, 80, 580–619.
- Breiman, L., Friedman, J. H., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Belmont, CA: Wadsworth.

- Brinkman, N. D. (1981). "Ethanol Fuel—A Single-Cylinder Engine Study of Efficiency and Exhaust Emissions." *SAE Transactions*, 90, 810345, 1410–1424.
- Buntine, W. L. and Weigend, A. S. (1991). "Bayesian Back-Propagation." *Complex Systems*, 5, 603–643.
- Chipman, H., George, E., and McCulloch, R. (1998). "Bayesian CART Model Search (with discussion)." *Journal of the American Statistical Association*, 93, 935–960.
- Cleveland, W. S. (1979). "Robust Locally-Weighted Regression and Smoothing Scatterplots." *Journal of the American Statistical Association*, 74, 829–836.
- Craig, B. (1997). "Manatee: Selecting From Among Several Models." Presented at 1997 Joint Statistical Meetings, Anaheim, CA.
- Denison, D., Mallick, B., and Smith, A. (1998a). "Automatic Bayesian Curve Fitting." *Journal of the Royal Statistical Society B*, 60, 333–350.
- (1998b). "A Bayesian CART Algorithm." *Biometrika*, 85, 363–377.
- Denison, D. G. (1997). "Simulation Based Bayesian Nonparametric Regression Methods." Ph.D. thesis, Imperial College, London.
- Annals of Statistics, 14, 1–26.

DiCiccio, T. J., Kass, R. E., Raftery, A., and Wasserman, L. (1995). "Computing Bayes Factors." Tech. Rep. 630, Carnegie Mellon University, Department of Statistics.

Diebolt, J. and Robert, C. (1994). "Estimation of Finite Mixture Distributions Through Bayesian Sampling." *Journal of the Royal Statistical Society B*, 56, 363–375.

Donoho, D. L. and Johnstone, I. (1989). "Projection Based Approximations and a Duality with Kernel Methods." *Annals of Statistics*, 17, 58–106.

- Donoho, D. L., Johnstone, I. M., Kerkyacharian, G., and Picard, D. (1995). “Wavelet Shrinkage: Asymptopia?” *Journal of the Royal Statistical Society B*, 57, 301–369.
- Doob, J. L. (1949). “Application of the Theory of Martingales.” In *Le Calcul des Probabilités et ses Applications*, 23–27. Colloque Internationaux du Centre National de la Recherche Scientifique, Paris.
- Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). “Hybrid Monte Carlo.” *Physics Letters B*, 195, 216–222.
- Fahlmann, S. E. (1989). “Faster Learning Variations on Back-Propagation: An Empirical Study.” In *Proceedings of the 1988 Connectionist Models Summer School, Pittsburgh*, eds. D. Touretzky, G. Hinton, and T. Sejnowski, 38–51. Denver: Morgan Kaufmann.
- Fahlmann, S. E. and Lebiere, C. (1989). “The Cascade-Correlation Learning Architecture.” In *Advances in Neural Information Processing Systems 2*, ed. D. Touretzky, 524–532. Denver: Morgan Kaufmann.
- Fisher, R. A. (1936). “The Use of Multiple Measurements in Taxonomic Problems.” *Annals of Eugenics*, 7, 179–188.
- Foster, D. P. and George, E. I. (1997). “Empirical Bayes Variable Selection.” The University of Pennsylvania and The University of Texas at Austin, unpublished.
- Friedman, J. H. (1991). “Multivariate Adaptive Regression Splines.” *The Annals of Statistics*, 19, 1–141.
- Friedman, J. H. and Silverman, B. W. (1989). “Flexible Parsimonious Smoothing and Additive Modelling (with discussion).” *Technometrics*, 31, 3–39.
- Friedman, J. H. and Stuetzle, W. (1981). “Projection Pursuit Regression.” *Journal of the American Statistical Association*, 76, 817–823.
- Funahashi, K. (1989). “On the Approximate Realization of Continuous Mappings by Neural Networks.” *Neural Networks*, 2, 3, 183–192.

- Furnival, G. M. and Wilson, R. W. J. (1974). "Regression by Leaps and Bounds." *Technometrics*, 16, 499–511.
- Gelfand, A. E. and Dey, D. K. (1994). "Bayesian Model Choice: Asymptotics and Exact Calculations." *Journal of the Royal Statistical Society B*, 56, 501–514.
- Gelman, A. and Meng, X.-L. (1996). "Simulating Normalizing Constants: From Importance Sampling to Bridge Sampling to Path Sampling." Columbia University and University of Chicago, unpublished.
- Geman, S. and Geman, D. (1984). "Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6, 721–741.
- Genz, A. and Kass, R. E. (1997). "Subregion-adaptive Integration of Functions Having a Dominant Peak." *Journal of Computational and Graphical Statistics*, 6, 92–111.
- Geweke, J. (1989). "Bayesian Inference in Econometric Models Using Monte Carlo Integration." *Econometrica*, 57, 1317–1340.
- Hastie, T. and Tibshirani, R. (1984). "Generalized Additive Models." Tech. Rep. 98, Stanford University, Department of Statistics.
- (1990). *Generalized Additive Models*. London: Chapman and Hall.
- Hawkins, D. (1989). "Discussion of 'Flexible Parsimonious Smoothing and Additive Modelling' by J. Friedman and B. Silverman." *Technometrics*, 31, 3–39.
- Holmes, C. and Mallick, B. (1998). "Bayesian Radial Basis Functions of Variable Dimension." *Neural Computation*, 10, 1217–1233.
- Hornik, K., Stinchcombe, M., and White, H. (1989). "Multilayer Feedforward Networks are Universal Approximators." *Neural Networks*, 2, 5, 359–366.
- Jeffreys, H. (1961). *Theory of Probability*. Third edition ed. New York: Oxford University Press.

- Kass, R. E. and Raftery, A. E. (1995). “Bayes Factors.” *Journal of the American Statistical Association*, 90, 430, 773–795.
- Keribin, C. (1997). “Consistent Estimation of the Order of Mixture Models.” Tech. rep., Université d’Evry-Val d’Essonne, Laboratoire Analyse et Probabilité.
- Leamer, E. E. (1978). *Specification Searches: Ad Hoc Inference with Nonexperimental Data*. New York: Wiley.
- Lee, H. K. H. (1996). “Model Selection for Consumer Loan Application Data.” Tech. Rep. 650, Carnegie Mellon University, Department of Statistics.
- MacKay, D. J. C. (1992). “Bayesian Methods for Adaptive Methods.” Ph.D. thesis, California Institute of Technology.
- (1994). “Bayesian Non-Linear Modeling for the Energy Prediction Competition.” *ASHRAE Transactions*, 100, pt. 2, 1053–1062.
- McCulloch, W. S. and Pitts, W. (1943). “A Logical Calculus of the Ideas Immanent in Nervous Activity.” *Bulletin of Mathematical Biophysics*, 5, 115–133.
- Meng, X. L. and Wong, W. H. (1993). “Simulating Ratios of Normalizing Constants Via a Simple Identity: a Theoretical Exploration.” Tech. Rep. 365, University of Chicago, Department of Statistics.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). “Equations of State Calculations by Fast Computing Machine.” *Journal of Chemical Physics*, 21, 1087–1091.
- Moody, J. E. (1992). “The Effective Number of Parameters: An Analysis of Generalization and Regularization in Nonlinear Learning Systems.” In *Advances in Neural Information Processing Systems 4*, eds. J. E. Moody, S. J. Hanson, and R. P. Lippmann. San Mateo, CA: Morgan Kaufmann.

- Müller, P., Erkanli, A., and West, M. (1996). “Bayesian Curve Fitting Using Multivariate Normal Mixtures.” *Biometrika*, 83, 1, 67–79.
- Müller, P. and Rios Insua, D. (1998a). “Feedforward Neural Networks for Nonparametric Regression.” In *Practical Nonparametric and Semiparametric Bayesian Statistics*, eds. D. Dey, P. Müller, and D. Sinha. New York: Springer-Verlag.
- (1998b). “Issues in Bayesian Analysis of Neural Network Models.” *Neural Computation*, 10, 571–592.
- Murata, N., Yoshizawa, S., and Amari, S. (1994). “Network Information Criterion—Determining the Number of Hidden Units for an Artificial Neural Network Model.” *IEEE Transactions on Neural Networks*, 5, 6, 865–871.
- Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. New York: Springer.
- Newton, M. and Raftery, A. (1994). “Approximate Bayesian Inference by the Weighted Likelihood Bootstrap (with discussion).” *Journal of the Royal Statistical Society B*, 56, 3–48.
- Nobile, A. (1994). “Bayesian Analysis of Finite Mixture Distributions.” Ph.D. thesis, Carnegie Mellon University, Department of Statistics.
- Pollard, D. (1991). “Bracketing Methods in Statistics and Econometrics.” In *Nonparametric and Semiparametric Methods in Econometrics and Statistics: Proceedings of the Fifth International Symposium in Econometric Theory and Econometrics*, eds. W. A. Barnett, J. Powell, and G. E. Tauchen, 337–355. Cambridge, UK: Cambridge University Press.
- Raftery, A. (1996). “Approximate Bayes Factors and Accounting for Model Uncertainty in Generalized Linear Models.” *Biometrika*, 83, 251–266.
- Raftery, A. E. and Madigan, D. (1994). “Model Selection and Accounting for Model Uncertainty in Graphical Models Using Occam’s Window.” *Journal of the American Statistical Association*, 89, 1535–1546.

- Raftery, A. E., Madigan, D., and Hoeting, J. A. (1997). "Bayesian Model Averaging for Linear Regression Models." *Journal of the American Statistical Association*, 437, 179–191.
- Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). "Learning Internal Representations by Error Propagation." In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, eds. D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, vol. 1, 318–362. Cambridge, MA: MIT Press.
- Sanil, A. (1998). "An Approach for Selection of Nonparametric Regression Methods." Ph.D. thesis, Carnegie Mellon University, Department of Statistics.
- Sarle, W. S. (1994). "Neural Network Implementation in SAS Software." In *Proceedings of the Nineteenth Annual SAS Users Group International Conference*, 38–51. SAS Institute, Cary, NC.
- Schwarz, G. (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, 6, 2, 461–464.
- Silverman, B. W. (1985). "Some Aspects of the Spline Smoothing Approach to Non-Parametric Curve Fitting." *Journal of the Royal Statistical Society B*, 47, 1–52.
- Smith, M. and Kohn, R. (1996). "Nonparametric Regression using Bayesian Variable Selection." *Journal of Econometrics*, 75, 317–343.
- Stone, C. J. (1982). "Optimal Rates of Convergence for Nonparametric Regression." *Annals of Statistics*, 10, 1040–1053.
- Stone, M. (1974). "Cross-validatory Choice and Assessment of Statistical Predictions." *Journal of the Royal Statistical Society B*, 36, 111–147.
- Tibshirani, R. (1988). "Estimating Transformations for Regression Via Additivity and Variance Stabilization." *Journal of the American Statistical Association*, 83, 394–405.
- Tibshirani, R. and Knight, K. (1995). "Model Search and Inference by Bootstrap 'Bumping'." Tech. rep., University of Toronto, Department of Statistics.

- Tierney, L. (1994). "Markov Chains for Exploring Posterior Distributions." *Annals of Statistics*, 22, 1701–1762.
- Tierney, L. and Kadane, J. (1986). "Accurate Approximations for Posterior Moments and Marginal Densities." *Journal of the American Statistical Association*, 81, 82–86.
- van der Vaart, A. W. and Wellner, J. A. (1996). *Weak Convergence and Empirical Processes*. New York: Springer.
- Wasserman, L. (1998a). "Asymptotic Inference for Mixture Models Using Data Dependent Priors." Tech. Rep. 677, Carnegie Mellon University, Department of Statistics.
- (1998b). "Asymptotic Properties of Nonparametric Bayesian Procedures." In *Practical Nonparametric and Semiparametric Bayesian Statistics*, eds. D. Dey, P. Mueller, and D. Sinha. Springer Lecture Notes. To appear.
- Witherspoon, S. (1983). "British Social Attitudes." Tech. rep., Social and Community Planning Research, London.
- Wong, W. H. and Shen, X. (1995). "Probability Inequalities for Likelihood Ratios and Convergence Rates of Sieve MLEs." *Annals of Statistics*, 23, 339–362.