

Evidence Evaluation for Bayesian Neural Networks Using Contour Monte Carlo

Faming Liang

fliang@stat.tamu.edu

Department of Statistics, Texas A&M University, College Station, TX 77843, U.S.A.

Bayesian neural networks play an increasingly important role in modeling and predicting nonlinear phenomena in scientific computing. In this article, we propose to use the contour Monte Carlo algorithm to evaluate evidence for Bayesian neural networks. In the new method, the evidence is dynamically learned for each of the models. Our numerical results show that the new method works well for both the regression and classification multilayer perceptrons. It often leads to an improved estimate, in terms of overall accuracy, for the evidence of multiple MLPs in comparison with the reversible-jump Markov chain Monte Carlo method and the gaussian approximation method. For the simulated data, it can identify the true models, and for the real data, it can produce results consistent with those published in the literature.

1 Introduction

The Bayesian evidence framework proposed in MacKay (1992a) provides a unified theoretical treatment of training, model selection, and prediction for Bayesian neural networks. It can be described as follows. Let H_1, \dots, H_m denote m models under consideration, where the term H denotes all the hypotheses and assumptions that are made in defining the model (e.g., network structure, specific noise model). In this article, the H_i 's are restricted to multiple layer perceptrons (MLPs) with different numbers of hidden units. Let ω_i be a vector that is the collection of parameters of the model H_i , including the connection weights and the parameters of the noise and prior distributions; $f(\omega_i|H_i)$ denote the prior distribution of ω_i ; D denote the training data; and $f(D|\omega_i, H_i)$ denote the likelihood for the model and parameters. According to Bayes' theorem, the posterior density of ω_i is then

$$f(\omega_i|D, H_i) = \frac{f(D|\omega_i, H_i) f(\omega_i|H_i)}{f(D|H_i)}, \quad (1.1)$$

where the denominator is

$$f(D|H_i) = \int f(D|\omega_i, H_i) f(\omega_i|H_i) d\omega_i. \quad (1.2)$$

When there are multiple models, $f(D|H_i)$ is the likelihood of the model H_i . Hence, it is called the evidence of the model.

The evidence can be used for model selection. The model of the largest evidence is selected as the most probable model. An advantage of this method is that it can make use of the full data for model training. In cross-validation, only part of the data can be used for model training. The Bayesian evidence framework also provides a natural interpretation for the regularization method, of which the regularization term is an analog of the prior of a Bayesian model. The evidence can also be used for prediction with a committee of networks (Perrone & Cooper, 1993). Let $P(H_i)$ denote the prior probability of the model H_i . The posterior probability of H_i is then

$$P(H_i|D) = \frac{f(D|H_i)P(H_i)}{f(D)}.$$

The point Bayesian prediction is given by

$$\begin{aligned} \bar{y} &= \sum_i P(H_i|D) \int y(x, \omega_i) f(\omega_i|H_i, D) d\omega_i \\ &= \frac{\sum_{i=1}^m f(D|H_i)P(H_i) \int y(x, \omega_i) f(\omega_i|H_i, D) d\omega_i}{\sum_{i=1}^m f(D|H_i)P(H_i)}. \end{aligned}$$

Refer to Bishop (1995) for further discussion on the Bayesian evidence framework.

Several methods have been proposed or used in the literature to evaluate the evidence for multiple models. The first method is the gaussian approximation method (MacKay, 1992a). In this method, the evidence is estimated by

$$\hat{f}(D|H_i) = f(D|\omega_i^{MP}, H_i) f(\omega_i^{MP}|H_i) (2\pi)^{d/2} \det^{-\frac{1}{2}} A, \quad (1.3)$$

where ω_i^{MP} is the MAP (maximum a posteriori) estimate of ω_i , d is the dimension of ω_i , and $A = -\nabla\nabla \log f(\omega_i|D, H_i)|_{\omega_i^{MP}}$ is the Hessian matrix evaluated at ω_i^{MP} . This estimate is often inaccurate due to the difficulties involved in the gaussian approximation and the determinant evaluation for the Hessian matrix. Walker (1969) showed that in the limit of an infinite training set, the posterior distribution is a mixture of gaussians, but with a finite training set, the approximation breaks down. The Hessian matrix is possibly singular or nearly singular for a network with redundant weights. In this case, the calculation of the determinant is unreliable. (See Thodberg, 1995, and Bishop, 1995, for the sophisticated methods of Hessian approximation.)

The second method is reversible-jump MCMC (RJMCMC; Green, 1995). Müller and Insua (1998) applied it to evaluate evidence for Bayesian neural networks. RJMCMC works by sampling the network structure and connection weights simultaneously from their joint posterior,

$$f(\omega, H|D) \propto f(D|\omega, H) f(\omega|H) P(H), \quad H \in \{H_1, \dots, H_m\}, \quad (1.4)$$

where the set $\{H_1, \dots, H_m\}$ contains all candidate models under consideration. The ratio of the evidence of the models H_i and H_j , $RE_{i,j} = P(D|H_i)/P(D|H_j)$ can be estimated by the ratio of their (relative) sampling frequencies, that is,

$$\widehat{RE}_{ij} = \frac{Rf_i}{Rf_j}, \quad 1 \leq i, \quad j \leq m,$$

where Rf_i and Rf_j denote the relative sampling frequencies of the models H_i and H_j , respectively. RJMCMC avoids the gaussian approximation, but it may converge very slowly for large networks. Müller and Insua (1998) note that even for one-hidden-layer regression networks, the weights corresponding to the connections from the hidden layer to the output layer should be integrated out explicitly from the joint posterior; otherwise, it would lead to a very slowly mixing Markov chain, rendering the scheme of little practical value. The slow convergence of RJMCMC is due to the inability of the Metropolis-Hastings algorithm (Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953; Hastings, 1970) in transition between different models. In RJMCMC, the sampler often suffers from some difficulty in transition between different models. Low-probability models are rarely visited. Unbalanced sample sizes of models often result in a poor estimate, in terms of overall accuracy, for the evidence of multiple models.

The third method is thermodynamic integration (Neal, 1993; Gelman & Meng, 1998). It calculates the ratio of the evidence of two models based on the overlap of the posterior distributions of the two models. Hence, it cannot be used directly to evaluate the ratio of evidence for two models with different dimensions. This method is not comparable to RJMCMC and the algorithm proposed in this article. The latter two methods can be used to simultaneously evaluate the evidence of multiple models with different dimensions.

In this article, we propose using the contour Monte Carlo algorithm (Liang, 2004) to evaluate evidence for Bayesian neural networks. As RJMCMC, this method avoids the evaluation of Hessian matrices. This method also overcomes the difficulty of RJMCMC in transition between different models. In CMC, the evidence is dynamically learned for each of the models. Weighted by the evidence, each of the models can be sampled from equally. CMC often results in an improved estimate, in terms of overall

accuracy, for the evidence of multiple models in comparison with RJMCMC and the gaussian approximation method. Numerical results show that the new method works well for both the regression and classification networks. For the simulated data, it can identify the true model, and for the real data, the results are consistent with those published in the literature.

This letter is organized as follows. In section 2, we describe briefly the contour Monte Carlo algorithm and illustrate it through a simple 1D example. In section 3, we show how the method can be used for evidence evaluation for regression MLPs. In section 4, we show how the method can be used for evidence evaluation for classification MLPs. Section 5 concludes with a brief discussion.

2 Evidence Evaluation Using Contour Monte Carlo

The contour Monte Carlo (CMC) algorithm, proposed recently by Liang (2004), can be regarded as a nontrivial generalization of the multicanonical algorithm (Berg & Neuhaus, 1991) and the Wang-Landau algorithm (Wang & Landau, 2001). In this section, we first review the algorithm in general, and then give the details of the implementation of the algorithm for evaluating evidence for Bayesian MLPs. The general review will facilitate the extension of the algorithm to the simultaneous use for network training and structure selection in section 5.

2.1 Contour Monte Carlo Algorithm. Suppose that we want to make an inference for the following Boltzmann distribution,

$$f(x) = \frac{1}{Z} \exp\{-U(x)/\tau\}, \quad x \in \mathcal{X},$$

where τ is the temperature, \mathcal{X} is the sample space, $U(x)$ is the energy function that corresponds to the negative log posterior in Bayesian neural networks, and $Z = \int_{\mathcal{X}} \exp\{-U(x)/\tau\} dx$ is the normalizing constant. Suppose that the sample space is partitioned into m disjoint subregions, E_1, \dots, E_m , according to some criterion chosen by the user, say, the energy function or some variable of interest. For example, the joint posterior distribution $f(\omega, H|D)$ given in equation 1.4 has the sample space

$$\mathcal{X} = \bigcup_{i=1}^m \mathcal{X}_i,$$

where $\mathcal{X}_i = \{x = (\omega_i, H_i) : f(\omega_i, H_i|D) > 0\}$ is the support of the marginal posterior $f(\omega_i|D, H_i)$. If we partition \mathcal{X} according to the index of models, then we have $E_i = \mathcal{X}_i$. The order of the subregions will not affect the performance of the algorithm.

In the following, we let $\psi(x) = \exp\{-U(x)/\tau\}$, $I(x \in E_i)$ be the indicator function, indicating the subregion where the sample x belongs, and $g(E_i)$ be a weight associated with the subregion E_i . Suppose that $g(E_i)$ is known, and it happens to be the quantity $g(E_i) = \int_{E_i} \psi(x) dx$. If the Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970) is employed to simulate from the distribution,

$$\pi(x) = \frac{1}{m} \sum_{i=1}^m \frac{\psi(x)}{g(E_i)} I(x \in E_i), \quad (2.1)$$

then each of the subregions will have an equal sampling frequency. This implies that if the sample space is partitioned according to the index of models as discussed above, each of the models will be sampled from equally in a simulation from $\pi(x)$, and thus, enough samples can be easily collected for each of the models for an effective inference for $f(x)$. However, $g(E_i)$ is unknown and needs to be estimated prior to the simulation. CMC provides a method to estimate the $g(E_i)$'s. A run of CMC proceeds in several stages, as follows.

Let $\hat{g}^{(s,k)}(E_i)$ denote the estimate of $g(E_i)$ at the k th iteration of the s th stage of the simulation, $\mathbf{x}^{(s,k)}$ denote the sample obtained at the k th iteration of the s th stage of the simulation, and δ_s denote a modification factor for $\hat{g}^{(s,k)}(E_i)$'s at stage s . In the first stage ($s = 1$), the simulation starts with the initial estimates $\hat{g}^{(0,0)}(E_1) = \dots = \hat{g}^{(0,0)}(E_m) = 1$ and a random sample $\mathbf{x}^{(0,0)}$ ($k = 0$), and then iterates between the following two steps.

1. Sampling: Propose a new configuration \mathbf{x}^* in the neighborhood of $\mathbf{x}^{(s,k)}$ according to the proposal distribution $T(\mathbf{x}^{(s,k)} \rightarrow \mathbf{x}^*)$, where the proposal distribution $T(\cdot \rightarrow \cdot)$ can be chosen as in the Metropolis-Hastings algorithm.

Accept \mathbf{x}^* with the probability

$$\min \left\{ \frac{\hat{g}^{(s,k)}(E_{I_{\mathbf{x}^*}})}{\hat{g}^{(s,k)}(E_{I_{\mathbf{x}^{(s,k)}}})} \frac{\psi(\mathbf{x}^*)}{\psi(\mathbf{x}^{(s,k)})} \frac{T(\mathbf{x}^* \rightarrow \mathbf{x}^{(s,k)})}{T(\mathbf{x}^{(s,k)} \rightarrow \mathbf{x}^*)}, 1 \right\}, \quad (2.2)$$

where I_z is the index of the subregion where \mathbf{z} belongs. If it is accepted, set $\mathbf{x}^{(s,k+1)} = \mathbf{x}^*$. Otherwise, set $\mathbf{x}^{(s,k+1)} = \mathbf{x}^{(s,k)}$.

2. Weight updating: Set $\hat{g}^{(s,k+1)}(E_{I_{\mathbf{x}^{(s,k+1)}}}) = (1 + \delta_s) \hat{g}^{(s,k)}(E_{I_{\mathbf{x}^{(s,k+1)}}})$.

The algorithm will iterate until the relative sampling frequency of each of the subregions is stable; that is, the relative sampling frequencies fluctuate only within a small range. For example, we can define the following statistic,

$$S_k = \frac{1}{m} \sum_{i=1}^m \left| \frac{Rf_{i,(k+1)b}}{Rf_{i,kb}} - 1 \right|,$$

to measure the stability of the estimates, where b is the batch size and $Rf_{i,kb}$ is the relative sampling frequency of the subregion E_i calculated at the (kb) th iteration of the stage. Here we define $\frac{0}{0} = 1$ in S_k to accommodate the empty subregions of which the sampling frequencies are 0. Once the relative sampling frequencies are stable, we will resume the sample counter for each of the subregions, reduce δ_s to a smaller value, and proceed to the next stage of simulation. The estimates $\hat{g}^{(s,K_s)}(E_i)$'s will be passed on as initial values to the next stage; that is, we set $\hat{g}^{(s+1,0)}(E_i) = \hat{g}^{(s,K_s)}(E_i)$ for $i = 1, \dots, m$, where K_s denotes the total number of iterations performed during the s th stage.

At each iteration, the sampling step can be repeated a number of times. The repetition will usually improve the accuracy of the estimates $\hat{g}(E_i)$'s but should not affect their convergence. For simplicity, the sampling step is performed only once at each iteration in all simulations of this article. The convergence of the estimates is stated in the following theorem (the proof is presented in Liang, 2004).

Theorem 1. *As $\delta_s \rightarrow 0$ and $K_s \rightarrow \infty$, for any positive function $\psi(x)$ ($x \in \mathcal{X}$) with $\int_{\mathcal{X}} \psi(x)dx < \infty$, we have*

$$\hat{g}^{(s,k)}(E_i) \longrightarrow g(E_i), \quad \text{in probability} \tag{2.3}$$

for $i = 1, \dots, m$, where $g(E_i) = c \int_{E_i} \psi(x)dx$ and c is a constant that can be determined with an additional constraint on $g(E_i)$'s, for example, $\sum_{i=1}^m g(E_i)$ or a particular $g(E_i)$ is equal to a known number.

This theorem implies that $\hat{g}^{(s,K_s)}(E_i)$ is a consistent estimate of $g(E_i)$ as $\delta_s \rightarrow 0$ and $k \rightarrow \infty$. If δ_s has converged to 0 and $\hat{g}(E_i)$'s have converged to their true values, the relative sampling frequency of each subregion must be proportional to

$$\frac{\int_{E_i} \psi(x)dx}{g(E_i)} \propto 1, \quad \text{for } i = 1, \dots, m, \tag{2.4}$$

because of the acceptance rule, equation 2.2. It is inclusion of the factor $\hat{g}^{(s,k)}(E_{I_{\mathcal{X}^{(s,k)}}})/\hat{g}^{(s,k)}(E_{I_{\mathcal{X}^*}})$ in equation 2.2 that forces each of the subregions to be sampled from equally.

At each step of CMC, $\hat{g}^{(s,k)}(E_i)$'s are adjusted according to the status of the sampler. The adjustment penalizes the future visits to the overvisited subregions and awards those to the undervisited subregions. The δ_s works as the scale or step size of the adjustment. It should decrease monotonically with the simulation. The decreasing rate of δ_s may affect the convergence of the algorithm. As we know, the preceding stage aims at providing a good initial value for the following stage, or in other words, the following stage aims at fine-tuning the estimate obtained in the preceding stage.

Table 1: Unnormalized Mass Function of the 10-State Distribution.

x	1	2	3	4	5	6	7	8	9	10
$P(x)$	1	100	2	1	3	1	2	2000	10	1

Hence, the δ 's should be chosen such that the error of the estimate obtained in the preceding stage should be able to be corrected in the following stage with a reasonable number of iterations. In this article, δ_s decreases in the scheme $\delta_{s+1} = \sqrt{1 + \delta_s} - 1$. This scheme works well for all examples. Note that $\delta_{s+1} = \sqrt{1 + \delta_s} - 1 \approx 0.5\delta_s$ for small δ_s , but it decreases faster than $\delta_{s+1} = 0.5\delta_s$ for large δ_s . The algorithm will be run until $\delta_s < \delta_{end}$, where δ_{end} is usually a very small number, say, a number less than 10^{-6} .

CMC is a stochastic approximation process. Dividing the simulation into stages allows the effect of iterations and the modification factor to be considered separately. Liang (2003) calculated the mean and mean squared error of $\hat{g}^{(s,k)}(E_i)$ at each stage. As an estimator of $g(E_i)$, the bias of $\hat{g}^{(s,k)}(E_i)$ is on the order of $O(\delta_s)$ as $k \rightarrow \infty$, that is,

$$E\hat{g}^{(s,k+1)}(E_i) = g(E_i) + O(\delta_s), \quad \text{as } k \rightarrow \infty.$$

The mean squared error of $\hat{g}^{(s,k)}(E_i)$ is also on the order of $O(\delta_s)$, that is,

$$E(\hat{g}^{(s,k)}(E_i) - g(E_i))^2 \sim O(\delta_s), \quad \text{as } k \rightarrow \infty.$$

This will help us to choose an appropriate decreasing scheme for δ_s and the number of iterations. (See Liang, 2003, for further discussion on the issue.)

2.2 An Illustrative Example. We use this example to illustrate the CMC algorithm. The example consists of 10 states with the unnormalized mass function $P(x)$ as specified in Table 1. The transition matrix employed is a stochastic matrix of which each row is generated independently from Dirichlet(1, ..., 1).

To apply CMC to this example, we partitioned the sample space into three subregions: $E_1 = \{1, 2, 3\}$, $E_2 = \{4, 5, 6\}$, and $E_3 = \{7, 8, 9, 10\}$. The mass values contained in the subregions E_1 , E_2 , and E_3 are 103, 5, and 2013, respectively. This example mimics model selection problems, in which there is often a single model that dominates the others. CMC was run 50 times independently with $\psi(x) = P(x)$, $\delta_1 = 0.1$, $\delta_{end} = 10^{-6}$, $K_1 = 1000$ and $K_{s+1} = 1.5K_s$. The CPU time of each run is about 0.55 second on a 2.8 GHZ computer (all computations reported in this article were done on the same computer). Figure 1a shows the transition path of the subregions at the last stage of a run. It is easy to see that CMC results in a free random walk in the space of subregions. Figure 2 shows the relative errors of the

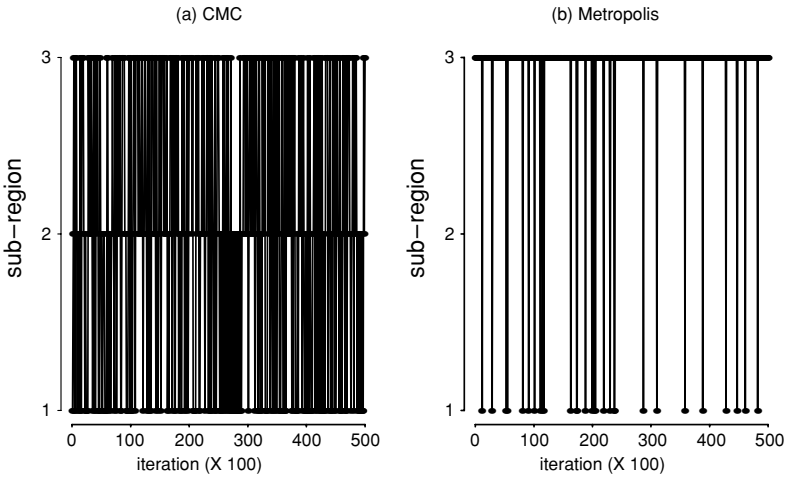


Figure 1: Transition path of the subregions at the last stage of a CMC run for the 10-state example. The samples are collected every 100 iterations in the simulation.

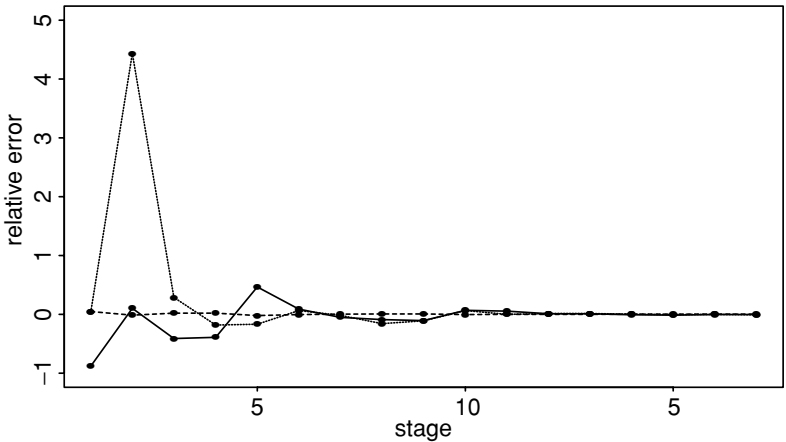


Figure 2: Plot of the relative errors of the estimates of $g(E_i)$'s versus the stages, where the solid line, dotted line, and dashed line correspond to the estimates of $g(E_1)$, $g(E_2)$, and $g(E_3)$, respectively.

estimates $g^{(s, K_s)}(E_i)$'s versus the stages where the relative error is defined as $\hat{g}(E_i)^{(s, K_s)} / g(E_i) - 1$. As the simulation evolves, the estimates are more and more accurate. The other computational results are summarized in Table 2. It shows that in CMC, each of the subregions is sampled equally likely,

Table 2: Computational Results for the 10-State Distribution.

Subregion	State	$g(E_i)$	CMC		Metropolis	
			$\hat{g}(E_i)$	Frequency (%)	$\hat{g}(E_i)$	Frequency (%)
E_1	1	103	103.108 (0.083)	0.320 (0.001)	102.882 (0.117)	0.047 (0.23e-3)
	2			32.313 (0.022)		4.710 (5.42e-3)
	3			0.642 (0.002)		0.094 (0.35e-3)
E_2	4	5	4.988 (0.005)	6.660 (0.010)	4.962 (0.019)	0.047 (0.37e-3)
	5			20.042 (0.023)		0.140 (0.72e-3)
	6			6.697 (0.012)		0.046 (0.30e-3)
E_3	7	2013	2012.904 (0.087)	0.031 (0.000)	2013.166 (0.123)	0.093 (0.36e-3)
	8			33.111 (0.025)		94.306 (6.03e-3)
	9			0.167 (0.001)		0.470 (1.12e-3)
	10			0.018 (0.001)		0.047 (0.33e-3)

Notes: The estimates $\hat{g}(E_i)$ are subject to the constraint $\sum_{i=1}^3 \hat{g}(E_i) = \sum_{i=1}^3 g(E_i) = 2021$. The numbers in the parentheses are the standard deviations of the preceding estimates.

regardless of the mass values contained in each subregion. The total mass contained in each subregion is estimated rather accurately by CMC. If each subregion is regarded as the sample space of a model, then the total mass contained in each subregion is the evidence of the model. Potentially, CMC serves as a good alternative of evidence evaluation for Bayesian MLPs.

For comparison, the Metropolis-Hastings algorithm was also applied to simulate from the distribution with the same transition matrix. Each run consists of $3.6e+6$ iterations and costs about the same CPU time as that of CMC. The algorithm was also run for 50 times independently. Figure 1b shows the transition path of the subregions in the first 50,000 iterations. The subregion E_2 was never visited in the plotted 500 iterations. Table 2 shows the estimates of the mass values and the sampling frequencies for each of the subregions. For this example, more than 94 percent of samples are drawn from E_3 in the Metropolis-Hastings runs, and the resulting estimates of $g(E_i)$'s are less accurate than those produced by CMC. Unbalanced sample sizes often result in an estimate with poor overall accuracy, although some components of the estimate corresponding to the oversampled subregions may be accurate.

2.3 Evidence Evaluation for Bayesian Neural Networks. Suppose that the training data D are modeled by a Bayesian MLP model. With slight abuse of notations, we denote a Bayesian MLP model with H hidden units by H . Our task here is to determine the "best" model by evaluating the evidence of m candidate models H_1, \dots, H_m . Furthermore, suppose

that the sample space of the posterior distribution $f(\omega, H|D)$ has been partitioned into m subregions with $E_1 = \{(\omega_1, H_1) : f(\omega_1, H_1|D) > 0\}$, \dots , $E_m = \{(\omega_m, H_m) : f(\omega_m, H_m|D) > 0\}$. For convenience in programming, the models are ordered in the number of hidden units in this article. The order of the models has no any effect on the convergence of the algorithm.

If CMC is run for this problem with the choice $\psi(\cdot) = P(D|\omega, H)P(\omega|H)$, theorem 1 implies that $\hat{g}^{(s,k)}(E_i)$ will converge to the evidence of the model H_i (up to a multiplication factor) in probability as $\delta_s \rightarrow 0$ and $K_s \rightarrow \infty$, that is,

$$\hat{g}^{(s,k)}(E_i) \rightarrow g(E_i) = c \int_{\mathcal{X}_i} P(D|\omega, H)P(\omega|H)d\omega, \quad \text{for } i = 1, \dots, m.$$

Thus, if $g(E_j) > 0$, $\hat{g}^{(s,k)}(E_i)/\hat{g}^{(s,k)}(E_j)$ will converge in probability to $g(E_i)/g(E_j)$, the ratio of the evidence of the models H_i and H_j .

Following is an implementation of CMC for evidence evaluation for Bayesian MLPs. Let $Q = (q_{ij})$ be a stochastic matrix, where q_{ij} denotes the proposal probability of the transition from the model H_i to H_j . In this article, Q is a triangular matrix with $q_{i,i-1} = q_{i,i} = q_{i,i+1} = \frac{1}{3}$ for $1 < i < m$, $q_{1,1} = q_{m,m} = \frac{2}{3}$, and $q_{1,2} = q_{m,m-1} = \frac{1}{3}$. Partition the sample space according to the index of models, initialize $\hat{g}^{(0,0)}(E_1) = \dots = \hat{g}^{(0,0)}(E_m) = 1$, and iterate between the following steps:

1. Draw a random number u from the uniform distribution $\text{Unif}(0,1)$.
2. If $u < q_{i,i-1}$, try a “death” move, which is to delete a hidden unit from the current model, that is, try to jump from model H_i to model H_{i-1} . If the move is accepted, set $i \leftarrow i - 1$.
3. If $u < q_{i,i-1} + q_{i,i}$, try a “within-subregion” move, which is to update the parameters of the current model, that is, try to draw a new sample from the posterior distribution $f(\omega_i|H_i, D)$ of model H_i .
4. Otherwise, try a “birth” move, which is to add a new hidden unit to the current model, that is, try to jump from model H_i to model H_{i+1} . If the move is accepted, set $i \leftarrow i + 1$.

The acceptance of the “birth,” “death,” and “within-subregion” moves are all guided by equation 2.2. For the within-subregion move, equation 2.6 is reduced to the conventional Metropolis-Hastings rule; the only difference is that the $\hat{g}^{(s,k)}$'s need to be updated here. For the sample space partition used in this article, the within-subregion move is also a within-model move, that is, trying to update $\omega^{(s,k)}$, the parameter vector of the current model to a new vector in the parameter space of the current model. In this article, $\omega^{(s,k)}$ is updated in blocks as follows. The weights on the connections fed to the same hidden or output unit are grouped into one block.

This grouping method usually works well for networks with a small or moderate number of input and hidden units. For networks with a large number of input and hidden units, the group may need to break down further. Suppose that the connection weights have been grouped into blocks. One block is then selected at random, and the weights in the selected block are updated with a spherical proposal distribution. A direction is first generated uniformly, and then the radius is drawn from $N(0, \zeta_m^2)$, where ζ_m^2 is calibrated such that the acceptance rate of the CMC moves is about 0.2 to 0.4. This proposal is symmetric in the sense that the transition probability ratio $T(\omega^* \rightarrow \omega^{(s,k)})/T(\omega^{(s,k)} \rightarrow \omega^*) = 1$, where ω^* denotes the proposed parameter vector. It is easy to see that the within-subregion move is just a Metropolis-Hastings move in the above implementation. We note that the within-subregion move can be any MCMC move performed in a fixed dimensional space, say, the Gibbs sampler (Geman & Geman, 1984) or hybrid Monte Carlo (Neal, 1996).

In the birth move, a new hidden unit is proposed to add to the current model, and the weights on the new connections are drawn from $N(\mu_b, \zeta_b^2)$, where μ_b and ζ_b^2 are the sample mean and sample variance of the weights of the current model, respectively. The resulting transition probability ratio is

$$\frac{T(\omega^* \rightarrow \omega^{(s,k)})}{T(\omega^{(s,k)} \rightarrow \omega^*)} = \frac{q_{i+1,i}}{q_{i,i+1}} \frac{1}{i+1} \frac{1}{\prod_{j=1}^{\xi} \phi((\omega_j - \mu_b)/\zeta_b)}$$

where ξ is the number of newly added connections, $\omega_1, \dots, \omega_{\xi}$ denote the weights drawn for the new connections, and $\phi(z)$ is the density function of the standard normal random variable z .

In the death move, a hidden unit is selected at random, with the proposal to delete it from the model. Let μ_d and ζ_d^2 denote the sample mean and sample variance of the connection weights, except for those on the connections fed to or exiting from the selected hidden unit of the current model. The resulting transition probability ratio is

$$\frac{T(\omega^* \rightarrow \omega^{(s,k)})}{T(\omega^{(s,k)} \rightarrow \omega^*)} = \frac{q_{i-1,i}}{q_{i,i-1}} \frac{i}{1} \frac{\prod_{j=1}^{\xi} \phi((\omega_j - \mu_d)/\zeta_d)}{1}$$

where $\omega_1, \dots, \omega_{\xi}$ denote the connection weights on the connections fed to or exiting from the selected hidden unit.

In the above implementation of CMC, each of the candidate models is essentially trained by the Metropolis-Hastings algorithm, which is used in the within-subregion moves to draw samples from the posterior $f(\omega_i|H_i, D)$. Hence, the performance of CMC will depend on the effectiveness of the Metropolis-Hastings algorithm for the problems under consideration. In the current implementation, CMC may not work well for a set of large

MLPs due to the inability of the Metropolis-Hastings algorithm in sampling from the posterior distributions of large MLPs. This difficulty can be alleviated by the following techniques. The first is to replace the Metropolis-Hastings algorithm by an advanced MCMC algorithm that performs in a fixed dimensional space, say, hybrid Monte Carlo (Neal, 1996) or the Langevin algorithm (Grenander & Miller, 1994; Phillips & Smith, 1996). The second one is to integrate out as many parameters as possible for the posterior $f(\omega, H|D)$. The theoretical basis is Rao-Blackwell's theorem (Casella & Berger, 2002; Liu, 2001), which suggests that an analytical integration is helpful for reducing the variance of Monte Carlo computation. In the context of Bayesian neural networks, this is also suggested experimentally by Denison, Holmes, Mallick, and Smith (2002). In this letter, for regression MLPs, we integrate out some of the parameters, including the variance of the random errors and all the weights on the connections from the hidden units to the output unit; for classification MLPs, we cannot do so. Hence, sampling from the posterior of a classification MLP usually means a more difficult task than sampling from the posterior of a regression MLP, provided that other conditions, say, the network structure and the data amount, are all similar. The third one is to repartition the sample space according to the index of models and the energy function jointly. This is to reduce the dependence of CMC on the algorithm used in the within-subregion moves. This technique will be discussed at the end of this article.

3 Evidence Evaluation for Regression MLPs

We first consider the evidence evaluation for regression MLPs. Let $D = \{(x_{t1}, \dots, x_{tP}; y_t) : t = 1, \dots, n\}$ denote the training data, where x_1, \dots, x_P are explanatory variables and y is the response variable. Let the data be modeled by a one-hidden-layer MLP with P input units and H hidden units. The model can be written as

$$y_t = \beta_0 + \sum_{i=1}^H \beta_i \varphi \left(\gamma_{i0} + \sum_{j=1}^P x_{tj} \gamma_{ij} \right) + \epsilon_t, \quad t = 1, \dots, n, \quad (3.1)$$

where $\epsilon_t \sim N(0, \sigma^2)$, β_i 's and γ_{ij} 's denote the connection weights from the hidden units to the output unit and from the input units to the hidden units, respectively. Here, the bias unit is treated as a special input unit with a constant input, say, 1. The $\varphi(\cdot)$ is the activation function. It is set to the hyperbolic tangent function in this section. The hyperbolic tangent function is equivalent to the sigmoid function in modeling, but it often leads to a faster convergence in training than the sigmoid function.

To work with the evidence framework, we specify the following priors for the model. First, we assume that $H \sim \text{Unif}\{H_1, \dots, H_m\}$, that is, $P(H_i) = 1/m$ for $i = 1, \dots, m$. Independent of the prior $P(H)$, we assume

$$\begin{aligned} \sigma^2 &\sim \text{Inv-Gamma}(\nu, \eta), \\ \beta_i | \sigma^2 &\sim N(0, \tau_\beta \sigma^2), \quad i = 0, \dots, K \\ \gamma_{ij} | \sigma^2 &\sim N(0, \tau_\gamma \sigma^2), \quad i = 1, \dots, K, \quad j = 0, \dots, P, \end{aligned} \quad (3.2)$$

where ν , η , τ_β , and τ_γ are hyperparameters to be specified by the user. We note that a more sophisticated prior specification for the connection weights of MLPs is the automatic relevance determination (ARD) prior (MacKay, 1995; Neal, 1996), which performs a soft feature selection for MLPs. However, the prior, equation 3.2, has its own advantages. First, it leads to a closed expression for the marginal posterior of $\gamma = \{\gamma_{ij} : i = 1, \dots, K, j = 0, \dots, P\}$ by integrating out $\beta = (\beta_0, \dots, \beta_K)$ and σ^2 explicitly. Second, the prior, equation 3.2, is essentially noninformative if we restrict $\nu \leq 1$. If $\nu \leq 1$, it is easy to calculate that $E(\sigma^2) = \infty$, $\text{Var}(\sigma^2) = \infty$, and

$$\text{Var}(\beta_i) = E(\text{Var}(\beta_i | \sigma^2)) + \text{Var}(E(\beta_i | \sigma^2)) = \tau_\beta E(\sigma^2) = \infty,$$

and $\text{Var}(\gamma_{ij}) = \infty$. A vague and proper prior is usually preferred in Bayesian model selection problems as the evidence of the models can be sensitive to the prior. (See Kass & Raftery, 1995, for more discussion on the choice of priors.) With the above prior specification, we have the following posterior distribution,

$$\begin{aligned} f(\beta, \gamma, \sigma^2, H | D) &\propto \left(\frac{1}{2\pi\sigma^2} \right)^{\frac{n}{2}} \\ &\times \exp \left\{ -\frac{1}{2\sigma^2} \sum_{t=1}^n \left[y_t - \beta_0 - \sum_{i=1}^H \beta_i \varphi \left(\gamma_{i0} + \sum_{j=1}^P x_{tj} \gamma_{ij} \right) \right]^2 \right\} \\ &\times \left(\frac{1}{2\pi\tau_\beta\sigma^2} \right)^{\frac{H+1}{2}} \exp \left\{ -\frac{1}{2\tau_\beta\sigma^2} \sum_{i=0}^H \beta_i^2 \right\} \left(\frac{1}{2\pi\tau_\gamma\sigma^2} \right)^{\frac{(P+1)H}{2}} \\ &\exp \left\{ -\frac{1}{2\tau_\gamma\sigma^2} \sum_{i=1}^H \sum_{j=0}^P \gamma_{ij}^2 \right\} \frac{\eta^\nu}{\Gamma(\nu)} \frac{e^{-\eta/\sigma^2}}{(\sigma^2)^{\nu+1}} \frac{1}{m}. \end{aligned} \quad (3.3)$$

Integrating out β and σ^2 and taking the logarithm, we have

$$\begin{aligned}
 -\log f(\gamma, H|D) = & \text{Const} + \frac{H(P+1)}{2} \log(2\pi) + \frac{H+1}{2} \log(\tau_\beta) \\
 & + \frac{H(P+1)}{2} \log(\tau_\gamma) + \frac{1}{2} \log |\mathbf{B}| \\
 & - \log \Gamma \left(\frac{n}{2} + \frac{H(P+1)}{2} + \nu \right) \\
 & + \left(\frac{n}{2} + \frac{H(P+1)}{2} + \nu \right) \log \left(\eta + \frac{1}{2} \mathbf{y}' \mathbf{y} \right. \\
 & \left. - \frac{1}{2} \mathbf{y}' \mathbf{Z} \mathbf{B}^{-1} \mathbf{Z}' \mathbf{y} + \frac{1}{2\tau_\gamma} \sum_{i=1}^H \sum_{j=0}^P \gamma_{ij}^2 \right), \quad (3.4)
 \end{aligned}$$

where “Const” is the constant term independent of γ and H , $\mathbf{y} = (y_1, \dots, y_n)'$, $\mathbf{Z} = (z_1, \dots, z_n)'$ is a matrix with the t th row $\mathbf{z}'_t = [1, \varphi(\gamma_{10} + \sum_{j=1}^P x_{tj} \gamma_{1j}), \dots, \varphi(\gamma_{H0} + \sum_{j=1}^P x_{tj} \gamma_{Hj})]$, and $\mathbf{B} = \frac{1}{\tau_\beta} \mathbf{I}_{H+1} + \mathbf{Z}' \mathbf{Z}$. The \mathbf{I}_{H+1} denotes the identity matrix of order $H+1$.

We note that the posterior distribution is invariably a nonlinear and multimodal function. For example, the posterior is invariant with respect to arbitrary relabeling of hidden units and simultaneous changes of the sign of β_i and γ_{ij} 's with $j = 0, \dots, P$. Müller and Insua (1998) suggesting avoid the multimodality by imposing a relabeling constraint on the parameter space, say, $0 < \gamma_{10} < \dots < \gamma_{H0}$. In this article, we impose no constraint on the parameter space. In theory, this is straightforward and complete for computing the model evidence. Here we mention one point: the model evidence is invariant with respect to the relabeling constraint. This can be seen from the equation 1.2, which depends on only the values of the integrand, which is invariant with respect to the relabeling constraint.

3.1 Example 1: Simulated Data. We simulated y_1, \dots, y_n from equation 3.1 with $P = 2$, $H = 2$, $\gamma_1 = (\gamma_{10}, \gamma_{11}, \gamma_{12}) = (-0.5, 1, 3)$, $\gamma_2 = (\gamma_{20}, \gamma_{21}, \gamma_{22}) = (0.5, 2, 2)$, $\beta = (\beta_0, \beta_1, \beta_2) = (-1, 3, -1.5)$, $n = 30$, $\sigma = 0.2$, and $x_{i,j} \sim \text{Unif}(-2, 2)$ for $i = 1, \dots, n$ and $j = 1, 2$.

For this example, we consider the models with one to four hidden units, which are denoted by H_1, \dots, H_4 , respectively. In simulation, we set the hyperparameters $\nu = 1$, $\eta = 1$, and $\tau_\beta = \tau_\gamma = 100$ and set the CMC parameters $\rho = 0$, $\delta_1 = 0.01$, $\delta_{s+1} = \sqrt{\delta_s + 1} - 1$, $K_1 = 10^5$, and $K_{s+1} = 1.2K_s$. The CPU time cost by a single run is about 43 seconds. Figure 3 shows the path of the model transition at the last stage of the simulation with $\delta = 1.56 \times 10^{-4}$, where one point was plotted every 100 iterations. CMC leads to a free

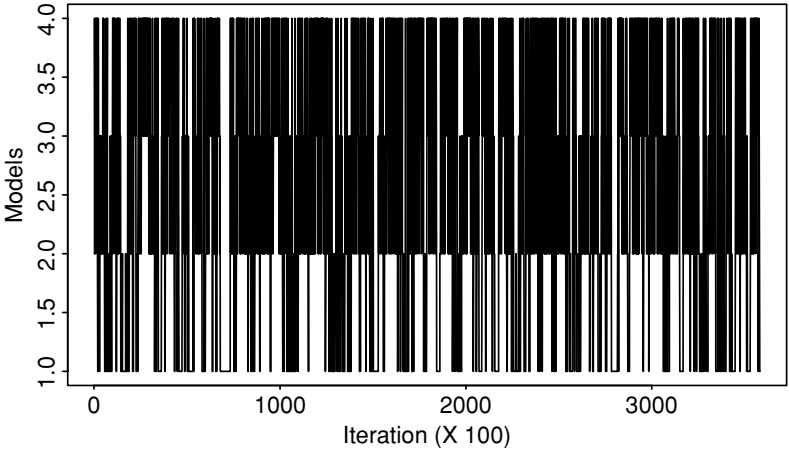


Figure 3: Transition path of the models in a CMC run for example 1. The samples are collected every 100 iterations in the simulation.

Table 3: Sensitivity Analysis for Hyperparameters for Example 1.

Model Setting	Frequency (%) (1,100)	Evidence (1,10)	Evidence (1,100)	Evidence (1,1000)	Evidence (0.1,100)
H_1	24.61 (0.26)	0.15 (0.01)	0.08 (0.01)	4.14 (0.99)	0.00 (0.00)
H_2	25.18 (0.13)	70.31 (0.43)	84.99 (0.39)	81.03 (1.87)	82.11 (0.36)
H_3	25.13 (0.08)	24.82 (0.34)	13.77 (0.34)	13.90 (1.01)	16.14 (0.32)
H_4	25.07 (0.15)	4.72 (0.10)	1.16 (0.05)	0.92 (0.09)	1.75 (0.05)

Notes: The choices of hyperparameters are shown in the second row, where (a, b) represents $\nu = \eta = a$ and $\tau_\beta = \tau_\gamma = b$. The Frequency (%) and Evidence columns show the relative sampling frequencies and the estimates of the evidence of the respective models. The numbers in parentheses are the standard deviations of the preceding estimates.

random walk in the model space. The model transition paths in the other stages are similar.

Later we repeated the run 10 times. In these runs, the constraint, $\sum_{i=1}^m \hat{g}(E_i) = 100$, was imposed on $\hat{g}(E_i)$'s to eliminate the unknown constant c of equation 2.3. Note that the same constraint was imposed on $\hat{g}(E_i)$'s in all simulations of this article. The estimates of the evidence (up to a multiplication factor) and the relative sampling frequencies of the models were reported in the columns "evidence (1,100)" and "Frequency (1,100)" of Table 3, respectively. It shows that the true model H_2 has been identified by CMC. To assess the sensitivity of the evidence to hyperparameters, we tried the choices $\nu = \eta = 1$ and $\tau_\beta = \tau_\gamma = 10$, $\nu = \eta = 1$ and $\tau_\beta = \tau_\gamma = 100$, and $\nu = \eta = 0.1$ and $\tau_\beta = \tau_\gamma = 100$. For each choice, CMC was run 10 times. The

Table 4: Sensitivity Analysis for the Sample Size for Example 1.

Sample Size	10	20	25	30	40	50
Error Rate	8	1	0	0	0	0
H_1	56.49 (7.78)	14.35 (7.60)	0.57 (0.39)	0.05 (0.05)	0.00 (0.00)	0.00 (0.00)
H_2	36.20 (6.18)	71.35 (6.19)	83.94 (1.34)	84.82 (0.85)	86.40 (0.74)	87.99 (0.42)
H_3	6.57 (1.45)	13.02 (1.52)	14.21 (1.12)	13.95 (0.75)	12.66 (0.66)	11.25 (0.38)
H_4	0.73 (0.18)	1.28 (0.20)	1.27 (0.20)	1.17 (0.11)	0.95 (0.08)	0.76 (0.04)

Notes: The second row shows the error rate (out of 10) that the true model could not be identified by CMC. The other rows show the estimates of the evidence of the respective models and the standard deviations (in parentheses) of the estimates.

computational results are also summarized in Table 3. Since CMC samples from each of the models equally, we omitted the relative sampling frequencies and report only the estimates of the evidence of the models. The results show that the evidence framework is rather robust to the variation of hyperparameters. The true model can be identified with all choices of hyperparameters shown in Table 3. Table 3 shows that the evidence framework tends to select a complex model as τ_β and τ_γ decrease. This is consistent with our intuition: a more complex model is needed to compensate the limitation for the effect of each connection weight. Relatively, the evidence framework is more robust to the variation of ν and η .

Next, we evaluated the sensitivity of the evidence framework to data amounts. We varied the sample size n : 10, 20, 25, 30, 40, and 50. For each value of n , 10 independent data sets were generated. For each of the data sets, CMC was run once with $\nu = \eta = 1$ and $\tau_\beta = \tau_\gamma = 100$. Table 4 shows the estimates of the evidence of the models, which are calculated by averaging over the 10 data sets. It shows that the evidence framework has a very good response to the data amount. Note that the total number of parameters of the true model is 10. With as few as 25 observations, the method has already been able to identify the true model for all data sets. As expected, the method tends to select a simpler model if the data information is not enough and tends to select the true model as the data information increases.

For comparison, the RJMCMC and gaussian approximation methods were also applied to the data sets of $n = 50$. RJMCMC was implemented as CMC except that the birth, death, and Metropolis moves are guided by the Metropolis-Hastings rule instead of equation 2.2. For each of the 10 data sets, RJMCMC was run with $1.7\text{e}+6$ iterations, which costs about the same CPU time as a run of CMC. These runs produce the following estimates for the evidence of the respective models: $H_1 : 0.00(0.00)$; $H_2 : 86.53(0.62)$; $H_3 : 12.51(0.53)$; and $H_4 : 0.96(0.09)$, where the numbers in the parentheses are the standard deviations of the preceding estimates. These estimates are quite consistent with those obtained by CMC, although the accuracy is slightly worse.

In the gaussian approximation method, the evidence, equation 1.3, is calculated for an MLP model as follows (Bishop, 1995). Let $E_D = \frac{1}{2} \sum_{t=1}^n [y_t - \beta_0 - \sum_{i=1}^K \beta_i \varphi(\gamma_{i0} + \sum_{j=1}^P x_{tj} \gamma_{ij})]^2$; $E_w = \frac{1}{2} [\sum_{i=0}^K \beta_i^2 + \sum_{i=1}^K \sum_{j=1}^P \gamma_{ij}^2]$; $(\beta^{MP}, \gamma^{MP})$ be the minimizer of $\zeta E_D + \lambda E_w$, that is, $(\beta^{MP}, \gamma^{MP}) = \arg \min_{\beta, \gamma} \zeta E_D + \lambda E_w$; $B = \zeta \nabla \nabla E_D$ be the Hessian matrix of the training error function E_D , d be the order of B , v_1, \dots, v_d be the eigenvalues of B , and $A = B + \lambda I$. The log evidence of an MLP model with H hidden units is approximately

$$\begin{aligned} \log f(D|H) = & -\lambda^{MP} E_w^{MP} - \zeta^{MP} E_D^{MP} - \frac{1}{2} \log \det(A) + \frac{d}{2} \log \lambda^{MP} \\ & + \frac{n}{2} \log \zeta^{MP} + \log H! + 2 \log H + \frac{1}{2} \log \left(\frac{2}{v} \right) \\ & + \frac{1}{2} \log \left(\frac{2}{n-v} \right), \end{aligned} \quad (3.5)$$

where E_w^{MP} and E_D^{MP} are respective values of E_w and E_D evaluated at $(\beta^{MP}, \gamma^{MP})$, and λ^{MP} and ζ^{MP} are respective choices of λ and ζ that maximize the log evidence. The estimates produced by the equation 3.5 for the evidence of the respective models are $H_1 : 0.00(0.00)$; $H_2 : 86.25(2.57)$; $H_3 : 10.23(1.50)$; and $H_4 : 3.52(1.98)$, where the numbers in the parentheses are the standard deviations of the preceding estimates. These estimates are not very accurate in comparison with those produced by CMC and RJMCMC. This is consistent with the findings of other authors (Bishop, 1995); the estimates produced by the gaussian approximation method are often not very accurate.

3.2 Example 2: Ischemic Heart Disease. The ischemic heart disease data (Kutner, Nachtsheim, & Neter, 2004) were collected by a health insurance plan and provide information concerning 788 subscribers who made claims resulting from coronary heart disease. The response variable is the natural logarithm of the total cost of services provided, and the predictors include the number of interventions or procedures carried out, the number of tracked drugs prescribed, the number of other diseases that the subscriber had during period, and the number of other complications that arose during heart disease treatment.

Kutner et al. (2004) used the data set to illustrate MLPs as a nonlinear regression model. In this article, we use it to compare CMC, RJMCMC, and the gaussian approximation method for evidence evaluation for Bayesian MLPs. For this example, we consider the models H_1, \dots, H_7 , which have one to seven hidden units, respectively. The first 600 observations were used for model building. CMC was run five times with $v = \eta = 1$, $\tau_\beta = \tau_\gamma = 100$, $K_1 = 60000$, $K_{s+1} = 1.2K_s$, $\delta_1 = 0.01$, $\delta_{s+1} = \sqrt{\delta_s + 1} - 1$, and $\delta_{end} = 5 \times$

Table 5: Model Selection for Example 2.

Models	CMC	RJMCMC	Gaussian Approximation
H_1	0.0047 (0.0044)	0.0008 (0.0008)	0.0061
H_2	92.5505 (0.7151)	93.2279 (1.6667)	96.2062
H_3	7.1970 (0.6787)	6.5628 (1.5878)	3.5778
H_4	0.2425 (0.0495)	0.2038 (0.0769)	0.2042
H_5	0.0053 (0.0017)	0.0041 (0.0024)	0.0046
H_6	0.0001 (0.0000)	0.0004 (0.0002)	0.0006
H_7	0.0000 (0.0000)	0.0001 (0.0000)	0.0006

Notes: H_i denotes the MLP model with i hidden units. The numbers in columns 2 and 3 are, respectively, the estimates of the evidence and the standard deviations of the estimates.

Table 6: Training and Prediction Errors for Example 2.

Models	Training Error	Prediction Error
H_1	0.3975 (0.0008)	0.2794 (0.0014)
H_2	0.3771 (0.0032)	0.2744 (0.0033)
H_3	0.3650 (0.0011)	0.2890 (0.0032)
H_4	0.3609 (0.0013)	0.2904 (0.0022)
H_5	0.3595 (0.0015)	0.2866 (0.0049)
H_6	0.3581 (0.0017)	0.2778 (0.0038)
H_7	0.3570 (0.0006)	0.2800 (0.0023)

Note: The numbers in columns 2 and 3 are, respectively, the training/prediction errors and their standard deviations.

10^{-5} . The CPU time cost by each run is about 850 seconds. The computational results are summarized in Table 5.

For comparison, RJMCMC and the gaussian approximation method were also applied to this example. RJMCMC was run five times independently. Each run consists of $2e+6$ iterations and costs about the same CPU time as that of CMC. The gaussian approximation method evaluates the evidence of the models with equation 3.5. The results are shown in Table 5.

Table 5 shows that the estimates produced by CMC, RJMCMC, and the gaussian approximation method are consistent: all identify model H_2 as the most probable model. We also compared the generalization ability of models H_1, \dots, H_7 . Each of the models was simulated with the Metropolis moves for $1.1e+5$ iterations, where the first 10,000 iterations were discarded for the burn-in process and the remaining iterations were used for prediction. Table 6 shows the training and prediction errors produced by the models in five runs. The model H_2 produces the smallest prediction error. The larger evidence models tend to produce smaller prediction errors, although the

Table 7: Model Selection for Example 3.

Model	H_1	H_2	H_3	H_4	H_5
Evidence	27.53 (1.91)	60.76 (1.59)	10.64 (0.30)	1.00 (0.03)	0.07 (0.00)
Notes	Best BIC	Best AIC _c			

Notes: The H_i denotes the model with i hidden units. The estimates of the evidence and their standard deviations (in parentheses) are computed by averaging over 10 independent runs.

anticorrelation is weak. This example provides an experimental judgment for the importance of evidence evaluation for Bayesian MLPs.

3.3 Example 3: Airline Data. The data set is Series G of Box and Jenkins (1970), the monthly totals (in 100,000s) of international airline passengers from January 1949 through December 1960, for a total of 144 observations. Faraway and Chatfield (1998) analyzed the data using MLPs with various hidden units and various input patterns. The “best” model that they identified with the BIC and AIC_c criteria (Sugiura, 1978; Hurvich & Tsai, 1989) are shown in Table 7.

For comparison, CMC was applied to this example. As in Faraway and Chatfield (1998), the first 11 years of observations (132 observations) were used for model building, and $(y_{t-13}, y_{t-2}, y_{t-1})$ was used as the input pattern. In modeling time series data, the input pattern of a MLP can generally be chosen according to the partial autocorrelation graph of the data, as suggested by Chatfield (2001). CMC was run 10 times with $\nu = \eta = 1$ and $\tau_\beta = \tau_\gamma = 100$, $K_1 = 2.4 \times 10^5$, $K_{s+1} = 1.1K_s$, $\delta_1 = 10^{-4}$, $\delta_{s+1} = \sqrt{\delta_s + 1} - 1$, and $\delta_{end} = 2 \times 10^{-5}$. Each run costs about 115 seconds CPU time. The computational results in Table 7 coincide with those obtained with the information criteria (Faraway & Chatfield, 1998). The two largest evidence models are the best AIC_c model and the best BIC model, respectively.

4 Evidence Evaluation for Classification MLPs

Let $D = \{(x_{t1}, \dots, x_{tP}; y_t) : t = 1, \dots, n\}$ denote the training data, where y is the response variable that takes values in a finite set $\{0, 1, \dots, q - 1\}$ only. Here, $q \geq 2$ denotes the number of classes of the data. Let the data be modeled by a one-hidden-layer MLP with P input units, H hidden units, and Q output units, where $Q = 1$ if $q = 2$ and $Q = q$ otherwise. If $q = 2$, the model can be written as

$$y_t = \begin{cases} 1 & \text{with probability } p_t, \\ 0 & \text{with probability } 1 - p_t, \end{cases} \quad (4.1)$$

where

$$p_t = \varphi' \left(\beta_0 + \sum_{i=1}^H \beta_i \varphi \left(\gamma_{i0} + \sum_{j=1}^P x_{tj} \gamma_{ij} \right) \right), \quad t = 1, \dots, n, \quad (4.2)$$

where $\beta = (\beta_0, \dots, \beta_H)$ and $\gamma = \{\gamma_{ij} : i = 1, \dots, H, j = 0, \dots, P\}$ denote the connection weights as for the regression MLPs, and $\varphi(\cdot)$ and $\varphi'(\cdot)$ denote the activation functions for the hidden units and the output unit, respectively. In this section, $\varphi(\cdot)$ and $\varphi'(\cdot)$ are both the sigmoid function for all examples. If $q > 2$, we have $y_t = l - 1$ with the probability p_{tl} for $1 \leq l \leq q$, where

$$p_{tl} = \frac{\exp(z_{tl})}{\sum_{j=1}^q \exp(z_{tj})}, \quad (4.3)$$

and

$$z_{tl} = \beta_{li} + \sum_{i=1}^H \beta_{li} \varphi \left(\gamma_{i0} + \sum_{j=1}^P x_{tj} \gamma_{ij} \right),$$

where β_{li} denotes the weight on the connection from the i th hidden unit to the l th output unit, and γ_{ij} denotes the weight on the connection from the j th input unit to the i th hidden unit as before.

For the classification MLPs, we have the following mutually independent priors,

$$\begin{aligned} H &\sim \text{Unif}\{H_1, \dots, H_m\}, \\ \beta_i &\sim N(0, \sigma_\beta^2), \quad i = 0, \dots, H, \\ \gamma_{ij} &\sim N(0, \sigma_\gamma^2), \quad i = 1, \dots, H, \quad j = 0, \dots, P, \end{aligned} \quad (4.4)$$

where σ_β^2 and σ_γ^2 are hyperparameters to be specified by the user. With prior 4.4, we have the negative log posterior (up to an additive constant) for the case $q = 2$,

$$\begin{aligned} -\log f(\beta, \gamma, H|D) &= \text{Const} + \frac{1 + H(P + 2)}{2} \log(2\pi) + \frac{H + 1}{2} \log \sigma_\beta^2 \\ &\quad + \frac{H(P + 1)}{2} \log \sigma_\gamma^2 + \frac{1}{2\sigma_\beta^2} \sum_{i=0}^H \beta_i^2 \\ &\quad + \frac{1}{2\sigma_\gamma^2} \sum_{i=1}^H \sum_{j=1}^P \gamma_{ij}^2 - \sum_{t=1}^n y_t \log p_t \\ &\quad - \sum_{t=1}^n (1 - y_t) \log(1 - p_t). \end{aligned} \quad (4.5)$$

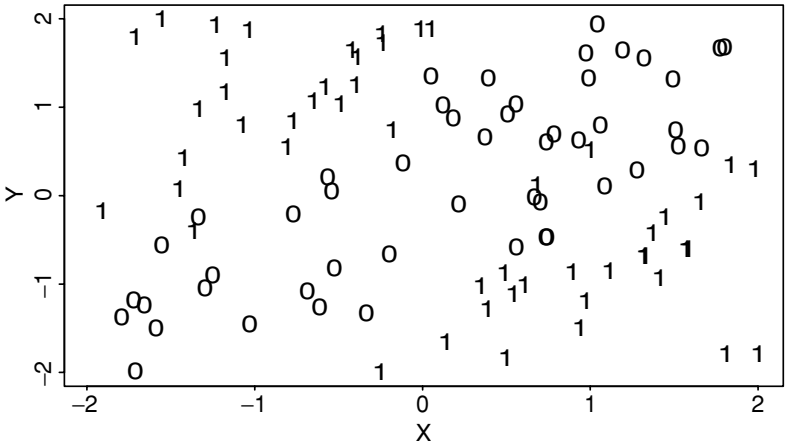


Figure 4: One training data set of example 4, where 0 and 1 denote the respective observations of the two classes.

For the case $q > 2$, we have

$$\begin{aligned}
 -\log f(\beta, \gamma, H|D) = & \text{Const} + \frac{1 + H(P + 2)}{2} \log(2\pi) + \frac{H + 1}{2} \log \sigma_\beta^2 \\
 & + \frac{H(P + 1)}{2} \log \sigma_\gamma^2 + \frac{1}{2\sigma_\beta^2} \sum_{i=0}^H \beta_i^2 \\
 & + \frac{1}{2\sigma_\gamma^2} \sum_{i=1}^H \sum_{j=1}^P \gamma_{ij}^2 - \sum_{t=1}^n \sum_{l=1}^q y_{tl} \log p_{tl}, \tag{4.6}
 \end{aligned}$$

where $y_{tl} = 1$ if $y_t = l - 1$ and 0 otherwise. The following examples illustrate the use of CMC for classification MLPs.

4.1 Example 4: Simulated Data. We simulated y_1, \dots, y_n from equations 4.1 and 4.2 with $P = 2, H = 2, \gamma_1 = (\gamma_{10}, \gamma_{11}, \gamma_{12}) = (22.12, -17.59, 17.15), \gamma_2 = (\gamma_{20}, \gamma_{21}, \gamma_{22}) = (6.00, 4.49, -3.75), \beta = (\beta_0, \beta_1, \beta_2) = (18.38, -10.66, -10.50), n = 100$, and $x_{ij} \sim \text{Unif}(-2, 2)$ for $i = 1, \dots, n$ and $j = 1, 2$. Figure 4 shows one data set generated with the above setting.

To eliminate the effect of randomness of the data generation process, we generated 10 data sets independently. For each data set, CMC was run once with $\nu = \eta = 1, \sigma_\beta^2 = \sigma_\gamma^2 = 100, \delta_1 = 0.1, \delta_{s+1} = \sqrt{\delta_s + 1} - 1, \delta_{\text{end}} = 10^{-5}, K_1 = 10^4$, and $K_{s+1} = 1.5K_s$. Each run costs about 13.5 minutes of CPU time. Here we took more iterations in each run to get accurate estimates for the evidence of the models. In fact, with only 10 s CPU time (about 1% of the CPU time reported above), the correct order of evidence of the

Table 8: Sensitivity Analysis for Hyperparameters for Example 4.

Model Setting	Frequency (%) (1,100)	Evidence (1,20)	Evidence (1,100)	Evidence (1,500)	Evidence (0.1,100)
H_1	25.77 (1.05)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
H_2	25.00 (0.31)	54.39 (1.20)	62.66 (2.05)	62.37 (3.08)	63.12 (2.36)
H_3	24.74 (0.38)	34.03 (0.60)	29.53 (1.33)	29.46 (2.24)	29.07 (1.48)
H_4	24.49 (0.44)	11.58 (0.63)	7.81 (0.75)	8.17 (1.31)	7.81 (0.92)

Notes: The choices of hyperparameters are shown in the second row, where (a, b) represents $\nu = \eta = a$ and $\tau_\beta = \tau_\gamma = b$. The Frequency (%) and Evidence columns show the relative sampling frequencies and the estimates of the evidence of the respective models. The numbers in parentheses are the standard deviations of the preceding estimates.

models has been identified by CMC. The computational results in Table 8 show that the true model is identified by CMC. The equal relative sampling frequency of each model implies that CMC works well for classification MLPs.

To assess the sensitivity of the evidence framework to hyperparameters, we tried the choices $\nu = \eta = 1$ and $\sigma_\beta^2 = \sigma_\gamma^2 = 20$, $\nu = \eta = 1$ and $\sigma_\beta^2 = \sigma_\gamma^2 = 500$, and $\nu = \eta = 0.1$ and $\sigma_\beta^2 = \sigma_\gamma^2 = 100$. For each choice, CMC was run once for each of the data sets. The computational results are summarized in Table 8. The true model can be identified by CMC with each of the above parameter settings. From the data generation process, we know the setting $\sigma_\beta^2 = \sigma_\gamma^2 = 20$ is inappropriate for this example. Even with this setting, the true model can still be identified by CMC. Of course, here the data contain enough information to correct the inappropriate prior.

4.2 Example 5: Ripley Data. This is a synthetic problem from Ripley (1994). It consists of two input features, two classes, and 250 training patterns. For this data set, we considered the models with two to six hidden units, which are denoted by H_2, \dots, H_6 , respectively. In simulation, we set $\nu = 1 = \eta = 1$, $\sigma_\beta^2 = \sigma_\gamma^2 = 100$; $K_1 = 10^4$, $K_{i+1} = 1.5K_i$, $\delta_1 = 0.1$, $\delta_{i+1} = \sqrt{\delta_i + 1} - 1$, and $\delta_{end} = 10^{-5}$. The simulation was repeated 10 times independently. The computational results (see Table 9) show that the highest evidence model is H_3 . This is different from the result of Penny and Roberts (1999), where H_4 was the largest evidence model. We note that Penny and Roberts adopted the ARD prior for the connection weights and estimated the model evidence with the gaussian approximation method. We also note that the evidence values reported in Penny and Roberts have large variance, and the evidence of H_3 and H_4 is not significantly different. Later, CMC was run with $\sigma_\beta^2 = \sigma_\gamma^2 = 20$. With this setting, the largest evidence model is identified as H_4 . The results are also reported in Table 9.

Table 9: Model Selection for the Ripley Data and the Irises Data.

Data	Model Evidence					
	H_1	H_2	H_3	H_4	H_5	H_6
Ripley ^a	—	6.32 (1.43)	40.74 (1.79)	34.66 (1.48)	14.44 (1.06)	3.84 (0.38)
Ripley ^b	—	3.95 (0.88)	28.14 (0.98)	37.15 (0.63)	22.28 (0.66)	8.49 (0.33)
Irises	0.02 (0.01)	61.41 (0.48)	29.11 (0.31)	7.92 (0.15)	1.54 (0.04)	—

Notes: The H_i denotes the model with i hidden units. The estimates of the evidence and their standard deviations (in parentheses) are computed by averaging over 10 independent runs.

^a Results with $\sigma_\beta^2 = \sigma_\gamma^2 = 100$.

^b Results with $\sigma_\beta^2 = \sigma_\gamma^2 = 20$.

4.3 Example 6. Irises Data. The irises data (Fisher, 1936) are classified into three categories: setosa, versicolor, and virginica. Each category has 50 samples. Each sample possesses four attributes: sepal length, sepal width, petal length, and petal width. A subset of 90 samples was chosen at random for training. For this data set, we considered the models with one to five hidden units. In simulation, we set $\nu = 1$, $\eta = 1$, $\sigma_\beta^2 = \sigma_\gamma^2 = 100$; $K_1 = 5 \times 10^4$, $K_{s+1} = 1.5K_s$, $\delta_1 = 0.01$, $\delta_{i+1} = \sqrt{\delta_i + 1} - 1$, and $\delta_{end} = 10^{-6}$. The simulation was repeated 10 times independently. The computational results summarized in Table 9 show that the highest evidence model is H_2 , of which the total number of connection weights (including biases) is 13. This coincides with that obtained with the method of structure learning with forgetting (Ishikawa, 2000).

5 Discussion

In summary, we have provided a new method for evidence evaluation for Bayesian neural networks. The simulated examples show that the true model can be identified by the method with an appropriate prior setting and enough training data. We also assessed the sensitivity of the evidence framework to hyperparameters. The numerical results show that the evidence framework is rather robust to choices of hyperparameters.

Since many authors, including MacKay (1992b), Thodberg (1995), and Penny and Roberts (1999), have reported the empirical correlation on the model evidence and generalization error, we skip this part in the article. Now it is widely believed that there is a weak (anti) correlation between evidence and generalization error (MacKay, 1992b). (For more discussion on the issue, refer to Bishop, 1996.)

In simulation from equation 1.4, if there are too many models under consideration, we may evaluate them part by part. For example, we first work on the models H_1, \dots, H_{m_1} and then work on the models $H_{m_1}, H_{m_1+1}, \dots, H_m$.

The ratios of the evidence of all the models can be adjusted according to the overlap model H_{m_1} . This method is extremely useful when some models are significantly different from others in structures. In this case, we can group the models with similar structures together to accelerate the simulation process. Note that CMC can be used only to simultaneously estimate the evidence of the multiple models for which the parameter have reasonable overlaps, for example, a sequence of nested models. It cannot compare the evidence for a sequence of models that are completely different.

In this article, we considered only the problem of selecting the number of hidden units for MLPs. Extensions to other structure selection problems are immediate, say, the selection of input patterns, the selection of noise models, or outlier detection.

In this article, the model is trained (withing model moves) with the Metropolis-Hastings algorithm, and the model transition (between model moves) is guided by the CMC rule. In the following we present one extension of CMC, where both the within- and between-model moves are guided by the CMC rule. Suppose that the sample space \mathcal{X} is partitioned according to the model and the energy function jointly,

$$\mathcal{X} = \bigcup_{i=1}^m \bigcup_{j=1}^{v_i} E_{ij},$$

where $\bigcup_{j=1}^{v_i} E_{ij} = \mathcal{X}_i$ is a partition of \mathcal{X}_i made according to the energy function. Without loss of generality, we assume that $E_{i,1}, \dots, E_{i,v_i}$ are arranged in ascending order of energy, that is, for any $w \in E_{i,j_1}$ and $w' \in E_{i,j_2}$, if $j_1 < j_2$ then $U_i(w) < U_i(w')$, where $U_i(\cdot)$ denotes the energy function of the model H_i . As $\delta_s \rightarrow 0$ and $K_s \rightarrow \infty$, Theorem 1 implies that

$$\widehat{RE}_{i_1, i_2} = \frac{\sum_{j=1}^{v_{i_1}} \hat{g}(E_{i_1, j})}{\sum_{j=1}^{v_{i_2}} \hat{g}(E_{i_2, j})}, \quad (5.1)$$

forms a consistent estimator of the ratio of the evidence of the models H_{i_1} and H_{i_2} .

The CMC algorithm can easily overcome high energy barriers of the energy landscape; the above algorithm has taken advantage of this advance in both network training and structure selection. It will be useful for hard or big networks, for which the conventional Metropolis-Hastings algorithm cannot sample efficiently from their posterior distributions.

Acknowledgments

I thank the editor, the associate editor, and two anonymous referees for their constructive comments and suggestions that led to significant improvement of this article.

References

- Berg, B. A., & Neuhaus, T. (1991). Multicanonical algorithms for 1st order phase-transitions. *Physics Letters B*, *267*, 249–253.
- Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford: Oxford University Press.
- Box, G. E. P., & Jenkins, G. M. (1970). *Time series analysis, forecast and control*. San Francisco: Holden Day.
- Casella, G., & Berger, R. L. (2002). *Statistical inference* (2nd ed.). Duxbury, MA: Thomson Learning.
- Chatfield, C. (2001). *Time-series forecasting*. London: Chapman & Hall.
- Denison, D., Holmes, C., Mallick, B., & Smith, A. F. M. (2002). *Bayesian methods for nonlinear classification and regression*. New York: Wiley.
- Faraway, J., & Chatfield, C. (1998). Time series forecasting with neural networks: A comparative study using the airline data. *Appl. Statist.*, *47*, 231–250.
- Fisher, R. A. (1936). The use of multiple measurements in taxonomic problem. *Annals of Eugenics*, *7*, 179–188.
- Gelman, A., & Meng, X. L. (1998). Simulating normalizing constants: From importance sampling to bridge sampling to path sampling. *Statistical Science*, *13*, 163–185.
- Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* *6*, 721–741.
- Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, *57*, 97–109.
- Grenander, U., & Miller, M. (1994). Representations of knowledge in complex systems (with discussion). *J. Roy. Statist. Soc. B*, *56*, 549–603.
- Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, *57*, 97–109.
- Hurvich, C. M., & Tsai, C. L. (1989). Regression and time series model selection in small samples. *Biometrika*, *76*, 297–307.
- Ishikawa, M. (2000). Structural learning and rule discovery. In I. Cloeth & J. M. Zurada (Eds.), *Knowledge-based neurocomputing* (pp. 153–206). Cambridge, MA: MIT Press.
- Kass, R. E., & Raftery, A. E. (1995). Bayes factors. *J. Amer. Statist. Assoc.*, *90*, 773–795.
- Kutner, M. H., Nachtsheim, C. J., & Neter, J. (2004). *Applied regression models* (4th ed.). New York: McGraw-Hill.
- Liang, F. (2003). *Contour Monte Carlo: Theoretical results, practical considerations, and applications* (Tech. Rep.). College Station: Department of Statistics, Texas A&M University.
- Liang, F. (2004). Annealing contour Monte Carlo for structure optimization in an off-lattice protein model. *Journal of Chemical Physics*, *120*, 6756–6763.
- Liu, J. S. (2001). *Monte Carlo strategies in scientific computing*. New York: Springer.
- MacKay, D. J. C. (1992a). The evidence framework applied to classification problems. *Neural Computation*, *4*, 720–736.
- MacKay, D. J. C. (1992b). A practical Bayesian framework for back-propagation networks. *Neural Computation*, *4*, 448–472.

- MacKay, D. J. C. (1995). Bayesian non-linear modeling for the 1993 energy prediction competition. In G. Heidbreder (Ed.), *Maximum entropy and Bayesian methods, Santa Barbara 1993*. Dordrecht: Kluwer.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, *21*, 1087–1091.
- Müller, P., & Insua, D. R. (1998). Issues in Bayesian analysis of neural network models. *Neural Computation*, *10*, 749–770.
- Neal, R. M. (1993). *Probabilistic inference using Markov chain Monte Carlo methods* (Tech. Rep. No. CRG-TR-93-1). Toronto: Department of Computer Science, University of Toronto.
- Neal, R. M. (1996). *Bayesian learning for neural networks*. New York: Springer.
- Penny, W. D., & Roberts, S. J. (1999). Bayesian neural networks for classification: How useful is the evidence framework? *Neural Networks*, *12*, 877–892.
- Perrone, M. P., & Cooper, L. N. (1993). When networks disagree: Ensemble methods for hybrid neural networks. In R. J. Mammone (Ed.), *Artificial neural networks for speech and vision* (pp. 126–142). London: Chapman & Hall.
- Phillips, D. B., & Smith, A. F. M. (1996). Bayesian model comparison via jump diffusions. In W. R. Gilks, S. T. Richardson, & D. J. Spiegelhalter (Eds.), *Markov chain Monte Carlo in practice* (pp. 215–240). London: Chapman & Hall.
- Ripley, B. D. (1994). Neural networks and related methods for classification. *Journal of the Royal Statistical Society, B*, *4*, 409–456.
- Sugiura, N. (1978). Further analysis of the data by Akaike's information criterion and the finite corrections. *Communications in Statistics—Theory and Methods*, *7*, 13–26.
- Thodberg, H. H. (1995). A review of Bayesian neural networks with an application to near infrared spectroscopy. *IEEE Transactions on Neural Networks*, *7*, 56–72.
- Walker, A. M. (1969). On the asymptotic behavior of posterior distributions. *Journal of the Royal Statistical Society B*, *31*, 80–88.
- Wang, F., & Landau, D. P. (2001). Efficient, multiple-range random walk algorithm to calculate the density of states. *Physical Review Letters*, *86*, 2050–2053.

Received January 7, 2004; accepted November 1, 2004.