# Applications of Hybrid Monte Carlo to Bayesian Generalized Linear Models: Quasicomplete Separation and Neural Networks

Hemant ISHWARAN

The "leapfrog" hybrid Monte Carlo algorithm is a simple and effective MCMC method for fitting Bayesian generalized linear models with canonical link. The algorithm leads to large trajectories over the posterior and a rapidly mixing Markov chain, having superior performance over conventional methods in difficult problems like logistic regression with quasicomplete separation. This method offers a very attractive solution to this common problem, providing a method for identifying datasets that are quasicomplete separated, and for identifying the covariates that are at the root of the problem. The method is also quite successful in fitting generalized linear models in which the link function is extended to include a feedforward neural network. With a large number of hidden units, however, or when the dataset becomes large, the computations required in calculating the gradient in each trajectory can become very demanding. In this case, it is best to mix the algorithm with multivariate random walk Metropolis–Hastings. However, this entails very little additional programming work.

**Key Words:** Bayesian hierarchical models; Feedforward neural networks; Leapfrog algorithm; Markov chain Monte Carlo; Random walk Metropolis–Hastings.

## 1. INTRODUCTION

Quasicomplete separation is a common problem in logistic regression, occurring when a hyperplane separates part of the data into the two classification groups, with the remaining data lying along the hyperplane itself. Often what happens in practice, is that it is only a small subset of the data over which the two classification groups are perfectly separated, and usually this is due to the effect of a small subset of covariates. Unfortunately, as is well known, this kind of separation precludes the existence of the maximum likelihood estimator, and although perfectly valid inference is possible using conditional methods (Mehta and Patel 1995), valid inference is not possible based on maximum likelihood estimation applied to the full likelihood. The presence of quasicomplete separation is also a computational problem, because traditional iterative methods based on

Hemant Ishwaran is Associate Staff, Department of Biostatistics and Epidemiology/Wb4, Cleveland Clinic Foundation, 9500 Euclid Avenue, Cleveland, OH 44195 (E-mail: ishwaran@bio.ri.ccf.org).



*Journal of Computational and Graphical Statistics, Volume 8, Number 4, Pages 779–799*

maximizing the likelihood function will fail to converge. See Albert and Anderson (1984) and also Santner and Duffy (1986) for more details.

However, a Bayesian approach in this setting will not have the theoretical difficulties seen with maximum likelihood estimation, as long as we are careful to work with a proper prior. The difficulty, though, is in posterior exploration. With quasicomplete separation, groups of parameters can become highly correlated, and this can lead to slow mixing over the posterior by conventional Markov chain Monte Carlo (MCMC) methods. For example, methods such as random walk Metropolis–Hastings, or rejection based Gibbs sampling methods such as those used by Zeger and Karim (1991), Gilks and Wild (1992), and Dellaportas and Smith (1993) will all perform poorly in this setting.

A very promising MCMC method for handling problems like this is hybrid Monte Carlo, which has a long history of successful applications in physics. For example, Rossky, Doll, and Friedman (1978); Duane and Kogut (1986); Duane, Kennedy, Pendleton, and Roweth (1987); Horowitz (1991); and Neal (1994). Although variations of this method have been applied in physics for quite some time, the method seems to have been largely overlooked in statistics, excepting some recent applications to neural networks (Neal 1996; Gustafson 1997), and posterior exploration of covariance matrices (Daniels 1998).

One goal of this article is to study the performance of hybrid Monte Carlo in logistic regression problems with quasicomplete separation. The variation of hybrid Monte Carlo that we study is based on the "leapfrog" algorithm presented in Duane et al. (1987). As we will see, this method leads to a rapidly mixing Markov chain in this challenging problem, but in fact the method is also a very competitive and simple approach for fitting any Bayesian generalized linear model with a canonical link. The computationally most demanding aspect of the leapfrog algorithm is the repeated evaluation of the gradient of the log of the posterior. However, with a canonical link, we end up with a gradient with a simple expression that is relatively inexpensive to calculate.

A second goal of this article is to apply the leapfrog algorithm in Bayesian generalized linear models which have been extended to include a neural network link function. A neural network link offers a flexible adaptive method for exploring covariate effects beyond the usual canonical (linear predictor) model. The hybrid Monte Carlo method is an ideal candidate for posterior sampling in these models. Indeed, conventional methods such as random walk Metropolis–Hastings tend to do very poorly in neural network models (Neal 1996), while rejection sampling methods for generalized linear models (Gilks and Wild 1992; Dellaportas and Smith 1993) cannot even be applied because of the non-log-concavity introduced by the neural networks. Furthermore, adaptive rejection Metropolis sampling (Gilks, Best, and Tan 1995) will be difficult to implement because of the high dimensions involved and will mix slowly because of the high correlation between parameters.

Briefly, Section 2 formally introduces the Bayesian generalized linear model and Section 3 describes the leapfrog hybrid Monte Carlo algorithm (Duane et al. 1987), which is the method used in the quasicomplete separation example of Section 4 and the Poisson regression neural network example of Section 5. A discussion is given in Section 6 along with some computational strategies.

## 2. BAYESIAN GENERALIZED LINEAR MODELS

In classical generalized linear models (Nelder and Wedderburn 1972; McCullagh and Nelder 1989), we observe responses $Y_i$ and $K$-dimensional covariates $x_i$, where the conditional responses $(Y_i \mid \theta_i, \phi)$ are assumed to be independent random variables with the one parameter exponential family density,

$$f(y_i \mid \theta_i, \phi) = \exp\left[\frac{y_i\theta_i - \mu(\theta_i)}{a(\phi)} + c(y_i, \phi)\right], \quad i = 1, \ldots, n. \tag{2.1}$$

The classical model assumes that the mean $\mathbb{E}(Y_i) = \mu'(\theta_i)$ is related to the intercept $\beta_0$ and covariate parameter $\beta$ through $l\left(\mathbb{E}(Y_i)\right) = \beta_0 + \beta' x_i$, where $l : \mu'(\Theta) \to \mathbb{R}$ is a monotone differentiable link function, and $\Theta \neq \emptyset$ is the natural parameter space for $\theta_i$. The monotonicity of $l$ and $\mu'$ ensures invertibility, which allows the canonical parameter to be related to $(\beta_0, \beta)$ through the smooth invertible transformation $\theta_i = \tilde{\mu} l^{-1}(\beta_0 + \beta' x_i)$, where $\tilde{\mu}$ is the inverse for $\mu'$. In particular, the canonical link model corresponds to the case when $\tilde{\mu} l^{-1}(u) = u$ is the identify function, so that

$$\theta_i = \beta_0 + \beta' x_i. \tag{2.2}$$

The Bayesian generalized linear model follows by placing a hierarchical prior on the parameters $(\beta_0, \beta)$ in (2.1). A particularly convenient selection is to use normal priors with conjugate priors for hyperparameters

$$\begin{aligned}
(\beta_0 \mid b_0, \sigma_0) &\sim \mathrm{N}(b_0, \sigma_0) \\
(\beta \mid b, W) &\sim \mathrm{N}_K(b, W) \\
(b_0 \mid B_0) &\sim \mathrm{N}(0, B_0) \\
(b \mid B) &\sim \mathrm{N}_K(0, BI) \\
(\sigma_0^{-1} \mid s_1, s_2) &\sim \mathrm{gamma}(s_1, s_2) \\
(W^{-1} \mid V, v) &\sim \mathrm{Wishart}(V^{-1}, v).
\end{aligned} \tag{2.3}$$

The choice of constants in (2.3) are chosen to induce noninformative priors. For example, to encourage the priors for $\beta_0$ and $\beta$ to be flat we select $B_0$ and $B$ to be large constants (such as 100 or 1,000). The priors for $(\beta_0, \beta)$ are then completed using noninformative priors for $\sigma_0$ and $W$ as well. For example, by choosing $s_1 = .01$, $s_2 = 100$ and $V = 3I$, $v = K + 2$. The choice for $V$ and $v$ ensures that the prior mean for $W$ is $V$, which gives a reasonable trade-off between noninformativeness of $W$ and the amount of influence the data can have on its posterior.

Up to this point, it has been tacitly assumed that $\phi$, and hence, $a(\phi)$, is a known constant value. However, $\phi$ can also be introduced as a parameter into the generalized linear model as a method for modeling overdispersion (Mallick and Gelfand 1994), although we do not pursue this topic here.

As mentioned in the introduction, the Bayesian generalized linear model is traditionally fit using rejection based Gibbs sampling methods such as in Zeger and Karim (1991), Gilks and Wild (1992), or Dellaportas and Smith (1993). However, normal approximation rejection sampling as in Zeger and Karim (1991) will not work properly in logistic

regression problems with quasicomplete separation because of the non-normality of the posterior and the singularity of the Fisher information matrix. Furthermore, univariate one coordinate at a time rejection sampling, as in Gilks and Wild (1992) or Dellaportas and Smith (1993), will lead to a slowly mixing chain, as will multivariate random walk Metropolis–Hastings, as we will see. Instead, we now introduce the hybrid Monte Carlo method as an alternative approach.

## 3. HYBRID MONTE CARLO

Hybrid Monte Carlo is an elaborate Markov chain Monte Carlo method designed to suppress the random walk nature exhibited in traditional Markov chain simulation methods (such as the random walk Metropolis–Hastings algorithm). The method is designed to promote rapid mixing of the Markov chain, and is especially suited to problems involving complex densities where exploration by a random walk may be too slow. Variations of this method have been studied in the physics literature for quite some time, in particular by Rossky, Doll, and Friedman (1978); Duane (1985); Duane and Kogut (1986); Duane et al. (1987); Kennedy (1990); as well as Horowitz (1991) and Neal (1994).

In this article we will concern ourselves with the "leapfrog" hybrid Monte Carlo algorithm outlined in Duane et al. (1987, equations 18–21) and which has been studied extensively by Neal (1996, chap. 3) in applications to Bayesian neural network models. The method applies to a $K$-dimensional random vector $U$ whose density $\pi$ with respect to Lebesgue measure is known up to a normalizing constant and is everywhere strictly positive and differentiable. The success of the method depends upon suppressing a random walk exploration of the density by using information about its shape to encourage movement towards higher probability regions. In particular, information about $\pi$ is incorporated through the $K$-dimensional gradient vector $\Delta(u) = \partial \log \pi(u)/\partial u$ for $\log \pi$ (which exists and is well defined by the positivity and differentiability assumed for $\pi$).

The hybrid Monte Carlo algorithm works by combining a stochastic step with a preset number of deterministic steps in a sophisticated discretization of Hamiltonian dynamics. In the special case where only one deterministic step is used, it is called the Langevin algorithm, which is a discrete time approximation to the Langevin diffusion process. In continuous time $t$, call $U_t$ a Langevin diffusion process for $\pi$ with variance $\delta^2$ if it satisfies the stochastic differential equation

$$dU_t = \delta \, dB_t + \frac{\delta^2}{2}\Delta(U_t)dt, \tag{3.1}$$

where $B_t$ is $K$-dimensional standard Brownian motion.

The hybrid algorithm works by augmenting the parameter of interest $U$ with a multivariate $K$-dimensional "momentum" variable, $Z$. In practice, however, it is unnecessary to keep track of the value of $Z$ from one step of the chain to the next. A physical interpretation for $Z$ and a more careful discussion of the discrete time approximations can be found in Duane et al. (1987), as well as Kennedy (1990) and Neal (1994; 1996, chap. 3). Here we merely indicate the steps involved in applying the hybrid algorithm. In particular, each step in the Markov chain simulation for $\pi$ involves executing the

following steps, where we suppose that $U_*$ is the current value in the Markov chain for $U$.

### 3.1 HYBRID MONTE CARLO ALGORITHM ("LEAPFROG" ALGORITHM)

1. Simulate $Z_* \sim \mathrm{N}_K(0, I)$. Let $U_0 = U_*$ and $Z_0 = Z_* + \delta\Delta(U_*)/2$.
2. For $l = 1, \ldots, L$, let

$$\begin{aligned} U_l &= U_{l-1} + \delta Z_{l-1} \\ Z_l &= Z_{l-1} + \delta_l \Delta(U_l), \end{aligned}$$

   where $\delta_l = \delta$ for $l < L$, otherwise $\delta_L = \delta/2$.
3. Move from the current value of $(U_*, Z_*)$ to $(U, Z)_{\text{new}}$, where

$$(U, Z)_{\text{new}} = \begin{cases} (U_L, Z_L) & \text{with probability} \quad p \\ (U_*, Z_*) & \text{with probability} \quad 1 - p, \end{cases}$$

   and

$$p = \min\left(\left[\frac{\pi(U_L)}{\pi(U_*)} \exp\left(-\frac{1}{2}(Z_L' Z_L - Z_*' Z_*)\right)\right], 1\right).$$

The algorithm is designed to generate an ergodic Markov chain with equilibrium distribution $\pi \otimes N_K(0, I)$, where $N_K(0, I)$ is the stationary distribution for the momentum variable $Z$. As mentioned earlier, the value for $Z$ is superfluous, although it plays a critical role in Step 1 by introducing a stochastic transition designed to make the chain irreducible and aperiodic. Step 2 provides the deterministic discrete time approximation to the Hamiltonian dynamics, and combined with the Metropolis acceptance rule in Step 3, ensures that detailed balance is satisfied.

The case where the number of leapfrog steps $L$ equals one is referred to as the Langevin algorithm, otherwise the algorithm is more generally referred to as the leapfrog algorithm. The Langevin algorithm is an example of Metropolis–Hastings. In this case, the proposal from the current step $U_*$ is towards a new point defined by

$$U_{\text{new}} = U_0 + \delta Z_0 = U_* + \delta\left(Z_* + \frac{\delta}{2}\Delta(U_*)\right) = U_* + \delta Z_* + \frac{\delta^2}{2}\Delta(U_*),$$

where $Z_* \sim \mathrm{N}_K(0, I)$. This is the discrete approximation to the diffusion (3.1).

Even with one leapfrog step, the Langevin algorithm already enjoys considerable advantages over the random walk Metropolis–Hastings algorithm (Roberts and Rosenthal 1995). As pointed out by Neal (1996, chap. 3), however, it is only when $L$ is reasonably large that the benefits of hybrid Monte Carlo can be fully exploited. In practice, $L$ should be chosen large enough so that the dynamics in Step 2 generate a candidate state $(U_L, Z_L)$ almost independent of the initial state $(U_*, Z_*)$. However, the candidate state should also be accepted with high probability, which involves tuning the step size $\delta$; with smaller steps resulting in higher acceptance rates, but with less exploration. Therefore,

in principle, $L$ and $\delta$ should be chosen large enough to promote a large trajectory, while still retaining a high acceptance rate. Often what occurs in practice is that, for a fixed $L$, there is some critical value for $\delta$, that when exceeded will result in very low acceptance rates. Selecting $\delta$ to equal this critical value generally leads to rapid exploration.

The leapfrog algorithm can also be extended to allow a different $\delta$ step size for each of the $K$ coordinates of $U$. This can be useful when there is substantial difference in the variances of the marginal distributions for $U$, which can occur in the Bayesian generalized linear model when covariates are measured on different scales. To reduce this problem, it is usually a very good idea to rescale all covariates so they have zero mean and variance one. This will also help in reducing correlation between parameters.

The extension is straightforward, and involves replacing the $\delta$ in the leapfrog algorithm with the $K \times K$ matrix $\delta D^{-1/2}$, for some diagonal matrix $D$ with diagonal entries $d_k^2$. In particular, the usual leapfrog algorithm corresponds to $D = I$, while the general case generates steps of size $\delta/d_k$ for $U$.

### 3.2 SIMULATIONS

To demonstrate the effectiveness of hybrid Monte Carlo, the leapfrog algorithm was applied to the stationary distribution $\pi$ defined by $U \sim N_K(0, \Sigma)$, where $\Sigma = (1-\rho)I + \rho 11'$ is an equicorrelation matrix with correlation $\rho$. Although it is easy to simulate $\pi$ directly, the two cases studied here—(1) $K = 10$, $\rho = .95$ and (2) $K = 40$, $\rho = .75$—are interesting because they pose challenging problems to conventional Markov chain simulation methods.

Random walk Metropolis–Hastings was used as a comparison, and was based on a simultaneous update of all parameters using a $\mathrm{N}_K(0, \epsilon^2 I)$ transition kernel. A simultaneous update is preferable to a coordinate-by-coordinate Gibbs sampling approach, which performs poorly here because of the much smaller width in the conditional distribution of each coordinate compared to its corresponding marginal. Most of the computational work in applying the leapfrog algorithm is usually spent in computing the gradient $\Delta$. Here it has a simple expression $\Delta(u) = -\Sigma^{-1}u$; but this is not always the case (see Equation (5.8) for the neural network model as comparison). In general, mixing over algorithms is a useful strategy for reducing this computational burden while retaining some of the advantage of hybrid Monte Carlo. Mixing is carried out at each iteration by selecting either the leapfrog algorithm with probability $p$ or some alternative cheaper algorithm(s) with probability $1-p$. Here we will investigate mixtures of 10% hybrid Monte Carlo with 90% Metropolis–Hastings.

Figure 1 contains plots of the first two coordinates of $U$ obtained from hybrid Monte Carlo, Metropolis, and from the mixture sampling. In all cases, zero was selected for the initial value of the chain. For $K = 10$ and $\rho = .95$, the hybrid Monte Carlo sampling used $L = 10$ leapfrog steps with an acceptance rate of roughly 70%. A burn in of 100 iterations was used before sampling every fifth iteration for 2,500 iterations. Even with this small number of iterations, we see that the chain has used large steps in exploring a substantial amount of the posterior (at least for the first two coordinates). To test convergence, the values $(U_m - \overline{U})'\widehat{\Sigma}^{-1}(U_m - \overline{U})$ were compared to a $\chi^2_K$ distribution

using the Kolmogorov–Smirnov distance ($\overline{U}$ and $\widehat{\Sigma}$ are the $K$-dimensional sample mean and variance matrix obtained from the simulation values $U_m$). The $p$ value from the test was .187, indicating convergence. Furthermore, the average first three autocorrelations for the sampled values $U_m$ were .31, .14, and .03.

In comparison, the Metropolis based exploration did more poorly, with parts of the posterior still left unexplored. This was based on a 1,000-iteration burn in, followed by

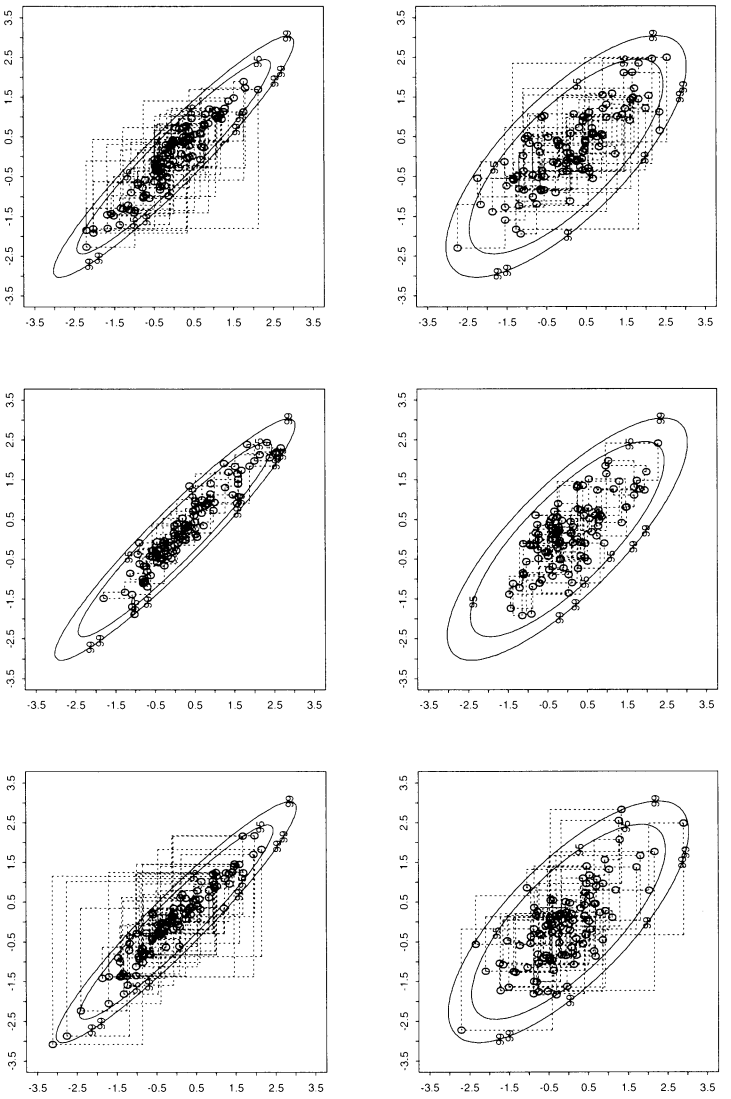

*Figure 1. Simulation history of the first two coordinates from 500 sampled points. For clarity, only each fifth iteration is plotted. Stationary distribution is $N_K(0, \Sigma)$, where $\Sigma = (1-\rho)I + \rho\, 11'$. Left column corresponds to $K = 10$, $\rho = .95$ while the right column is for $K = 40$ and $\rho = .75$. From top to bottom, simulation was via: (a) hybrid Monte Carlo; (b) Metropolis–Hastings; (c) 10% mixture of hybrid Monte Carlo and 90% Metropolis–Hastings. Note the superimposed theoretical 95% and 99% bivariate normal contours.*

a 31% acceptance rate over 25,000 iterations subsampled every 50th iteration (because each trajectory in the leapfrog algorithm involves 10 deterministic steps, the Metropolis exploration was based on 10 times as many iterations as a crude way of equating the overall computational time required in both methods). For Metropolis, the Kolmogorov–Smirnov test had a $p$ value of .56, but there was high autocorrelation even with the use of a 50 iteration lag (.93, .91, and .90). Mixing the Metropolis with hybrid Monte Carlo improved matters. The same parameters in each of the algorithms were used, with a burn in of 500 iterations followed by 10,000 iterations subsampled each 20th iteration. As we can see, the algorithm occasionally inherits a large step from the hybrid algorithm which helped in exploring the posterior and in reducing the autocorrelations (.61, .37, .19). Although these results are significantly better than Metropolis alone, they are less impressive than those observed with hybrid Monte Carlo. Given the simplicity of the gradient, hybrid Monte Carlo alone would have to be the preferred algorithm.

Similar results were seen for the case $K = 40$ and $\rho = .75$. Here the leapfrog algorithm used $L = 100$ steps with a 20-iteration burn in and a 76% acceptance rate over 500 samples. Metropolis–Hastings used a 2,000-iteration burn in followed by 30% acceptance over 50,000 iterations subsampled each 100th iteration (we used 100 times as many iterations to equate computational times). The mixture algorithm used a 200 iteration burn in followed by 5,000 iterations subsampled every 10th iteration.

## 4. HYBRID MONTE CARLO APPLIED TO THE GENERALIZED LINEAR MODEL

The leapfrog hybrid Monte Carlo algorithm is applied to the Bayesian generalized linear model within a Gibbs sampling framework. In this strategy we first update $(\beta_0, \beta)$ using the hybrid Monte Carlo method, followed by updates to all hyperparameters, which are straightforward to carry out thanks to our choice of conjugate priors.

The target density $\pi$ in the step for $(\beta, \beta_0 \mid b_0, b, \sigma_0, W)$ is proportional to

$$\pi(\beta_0 \mid b_0, \sigma_0)\pi(\beta \mid b, W) \prod_{i=1}^{n} \exp\left[\frac{y_i\theta_i - \mu(\theta_i)}{a(\phi)} + c(y_i, \phi)\right],$$

which is strictly positive and differentiable in the parameters by our choice of priors. To apply the hybrid algorithm we must calculate the $(K+1)$-dimensional gradient vector $\Delta = (\log \pi)'$ for $(\beta_0, \beta)$. With a canonical link (2.2), we get the following simple expressions

$$\begin{aligned} \Delta(\beta_0) &= \sum_{i=1}^{n} \tilde{y}_i - \frac{1}{\sigma_0}(\beta_0 - b_0) \\ \Delta(\beta) &= \sum_{i=1}^{n} x_i\tilde{y}_i - W^{-1}(\beta - b), \end{aligned} \tag{4.1}$$

where $\tilde{y}_i = y_i - \mu'(\theta_i)$. Notice that it will be convenient to keep track of $\theta = (\theta_1, \ldots, \theta_n)$ in the hybrid Monte Carlo update, since its value will be needed to compute the gradient defined by (4.1).

Table 1. Study of Osteosarcoma. Classification of 46 patients into groups that are relapse free for at least three years (1), or less than three years (0). Covariates of interest are lymphocytic infiltration (high = 1), sex (male = 1), and osteoblastic pattern (yes = 1).

| *Relapse free time* | *Lymphocytic infiltration* | *Sex* | *Osteoblastic pattern* | *Frequency* |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 3 |
| 1 | 0 | 0 | 1 | 2 |
| 1 | 0 | 1 | 0 | 4 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 5 |
| 1 | 1 | 0 | 1 | 3 |
| 1 | 1 | 1 | 0 | 5 |
| 1 | 1 | 1 | 1 | 6 |
| 0 | 1 | 0 | 1 | 2 |
| 0 | 1 | 1 | 0 | 4 |
| 0 | 1 | 1 | 1 | 11 |

After completing the hybrid Monte Carlo update, we finish off by updating each hyperparameter. This completes one iteration in the Gibbs cycle. As mentioned earlier, the sampling for hyperparameters are straightforward because of conjugacy. Specifically, we need to perform the following simple simulations:

$$\begin{aligned}
(b_0 \mid \beta_0, \sigma_0, B_0) &\sim \mathrm{N}\left(B_0^* \frac{\beta_0}{\sigma_0}, B_0^*\right) \\
(b \mid \beta, W, B) &\sim \mathrm{N}_K(\Lambda W^{-1}\beta, \Lambda) \\
(\sigma_0^{-1} \mid \beta_0, b_0, s_1, s_2) &\sim \text{gamma}\left(s_1 + \frac{1}{2}, s_2^*\right) \\
(W^{-1} \mid \beta, b, V, v) &\sim \text{Wishart}((V^*)^{-1}, v+1),
\end{aligned}$$

where $B_0^* = (1/\sigma_0 + 1/B_0)^{-1}$, $\Lambda = (W^{-1} + I/B)^{-1}$, $s_2^* = (1/s_2 + (\beta_0 - b_0)^2/2)^{-1}$, and $V^* = V + (\beta - b)(\beta - b)'$.

## 4.1 LOGISTIC REGRESSION WITH QUASICOMPLETE SEPARATION

Goorin et al. (1987) studied 46 patients with nonmetastatic osteosarcoma in an attempt to quantify tumor recurrence times. Patients were classified into two groups consisting of those that experienced recurrence within three years and those that were recurrence free. Covariates used in predicting the dichotomized relapse free time were: sex, lymphocytic infiltration of the primary tumor and presence of osteoblastic pattern in the primary tumor. Table 1 contains the data of interest, which also appears in Mehta and Patel (1995).

As noted by Goorin et al. (1987), as well as by Mehta and Patel (1995), a classical logistic regression analysis of relapse free time will fail due to the quasicomplete separation present in the data. Indeed, if we look at Table 1 we see that all patients who are free of lymphocytic infiltration are also relapse free. Because of this, the data points are quasicompletely separated (Albert and Anderson 1984), with the quasicomplete separated points made up of those patients with lymphocytic infiltration (these are the points lying along the separating hyperplane). As mentioned in the introduction, this precludes the

Table 2. Median values or point estimates (in square brackets) and 95% credible or confidence intervals for parameters using logistic regression by maximum likelihood estimation (MLE), hybrid Monte Carlo (HMC), random walk Metropolis–Hastings (RWMH), and a 10% mixture of HMC with 90% RWMH (MIX). Parameter estimates for the MLE are based on the last iteration using Fisher's method of scoring, which fails to converge because of the quasi-complete separation of data points. The Wald confidence intervals are derived from the observed Fisher information, but are unavailable for lymphocytic infiltration because of the singularity of the information matrix. The last row in the table is obtained using exact conditional inference (ECI) (Mehta and Patel 1995). Because of the parameterization used, only the parameters for sex and osteoblastic pattern are directly comparable.

| *Method* | *Intercept* | *Lymphocytic infiltration* | *Sex* | *Osteoblastic pattern* |
|---|---|---|---|---|
| HMC | (3.9,[17.2],54.3) | (−52.3,[−14.4],−1.4) | (−4.9,[−1.9],.29) | (−3.4,[−1.3], .45) |
| RWMH | (4.4,[15.1],31.3) | (−28.7,[−12.7],−2.1) | (−4.3,[−1.9],−.005) | (−2.9,[−1.3],.20) |
| MIX | (4.4,[16.9],52.5) | (−50.5,[−14.5],−2.0) | (−4.8, [−1.9], .18) | (−3.4,[−1.3], .43) |
| MLE | (25.8,[27.7],29.6) | [−25.6] | (−3.4,[−1.6], .15) | (−2.7, [−1.2], .29) |
| ECI | · | · | (−4.0,[−1.5], .36) | (−2.9, [−1.2], .51) |

existence of the maximum likelihood estimator and, in particular, if $\beta_0$ and $\beta_1$ represent the intercept and lymphocytic infiltration parameters, then the likelihood is maximized as $\beta_0 \uparrow \infty$ along the line $\beta_0 + \beta_1 = c$, for some constant $c$. Indeed, this phenomenon can be observed by looking at Table 2, which contains the results from fitting the logistic regression model using the PROC GENMOD procedure in SAS 6.11 (SAS Institute Inc., 1996). Although the procedure failed to converge, we see that the parameter estimates for the intercept and lymphocytic infiltration parameters are large positive and large negative numbers, as expected.

There are no theoretical problems in implementing our Bayesian approach since we will select proper priors. As a benchmark comparison for the leapfrog algorithm we used multivariate random walk Metropolis–Hastings with a simultaneous update for all parameters. A mixture of the two algorithms was also tried, with 90% mixing for Metropolis and 10% mixing for hybrid Monte Carlo. Flat and noninformative priors were used for $(\beta_0, \beta, b_0, b, \sigma_0, W)$ by choosing $B_0 = B = 1,000$, $s_1 = .01$, $s_2 = 100$, $V = 3I$, and $v = K + 2 = 5$. Table 2 contains the parameter estimates and credible intervals based on the three simulation methods, while Figure 2 contains the chain histories for the parameters corresponding to the intercept term and the lymphocytic infiltration covariate. The linear relationship between parameter values for the intercept and the lymphocytic term is what creates problems for the maximum likelihood estimator, and as we can see from Figure 2, also challenges the Metropolis random walk exploration of the posterior (middle plot). The likelihood is highest along a straight line which, combined with the small sample size and choice of flat priors, attenuates the region around this line where the posterior will have high probability. This is what makes conventional random walk Markov chain exploration difficult, and by even mixing with only 10% hybrid Monte Carlo, we can see a vast improvement (bottom plot).

The top of Figure 2 corresponds to the hybrid Monte Carlo algorithm, which used a 1,000-iteration burn in before sampling the next 25,000 values every 10th iteration. The algorithm used $L = 100$ leapfrog steps with a step size of .15, which led to an overall acceptance rate of approximately 82%. As we can see, the large number of steps

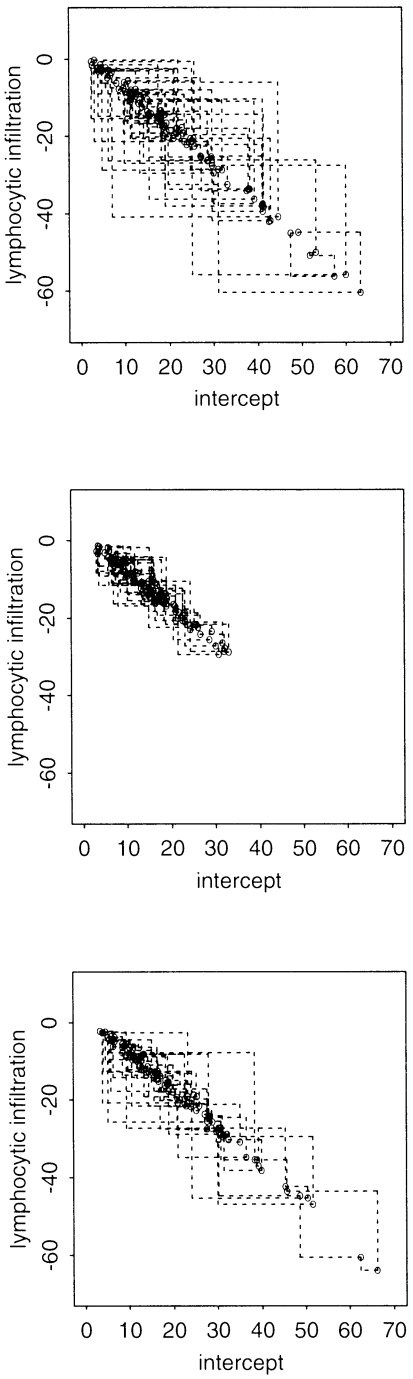


*Figure 2. History of sampled values for the intercept and lymphocytic infiltration parameters using Bayesian logistic regression. For clarity only each 25th iteration from 2,500 sampled values are plotted. Plots are for simulations using hybrid Monte Carlo (top), random walk Metropolis–Hastings (middle), and 10% mixture of hybrid Monte Carlo and 90% random walk Metropolis–Hastings (bottom).*

$L$ and high acceptance rate allowed the Markov chain to use large steps in exploring the posterior, especially in the more difficult lower right region. In contrast, the random walk Metropolis algorithm led to much less movement in the chain, with very little of the lower right region of the posterior being explored. This was based on a 25,000-iteration burn in, with subsampling every 100th iteration over 250,000 iterations. The algorithm was based on a multivariate normal transition kernel with mean zero and standard deviation .35, leading to an overall acceptance rate of about 29%. As just mentioned, mixing the Metropolis with hybrid Monte Carlo substantially improved matters. Mixing was based on the same parameters in each of the algorithms, while using a burn in of 2,500 iterations followed by 75,000 iterations subsampled each 30th iteration.

Even with the large number of iterations used, the Metropolis–Hastings algorithm has yet to fully explore the posterior. In particular, this leads to narrower credible intervals for parameters than those based on hybrid Monte Carlo. For example, the Metropolis-based exploration found sex to be nearly significant at the 5% level, while for hybrid Monte Carlo gender was not significant. Furthermore, the intervals obtained by hybrid Monte Carlo agree much more closely to the confidence intervals obtained using exact conditional inference (Mehta and Patel 1995). Exact conditional inference is a method for providing valid frequentist confidence intervals and tests even when the data contain a quasicomplete separation.

Another consequence of limited posterior exploration is that it fails to alert us to the problem of quasicomplete separation. With hybrid Monte Carlo, we observe a non-normally distributed posterior and a broad range of posterior values for the intercept and lymphocytic infiltration parameters. These are clear signals of the separation, but with Metropolis, even after 250,000 iterations, the posterior still appears to be bivariate normal (Figure 2), and in practice things would probably be worse because we would not have used so many iterations. However, by adding 10% hybrid mixing we get pretty much the same exploration as with hybrid Monte Carlo alone. This did, however, require more iterations, and given the simplicity of the gradient, the leapfrog algorithm seems to be the best choice here.

# 5. FEEDFORWARD NEURAL NETWORKS

## 5.1 NEURAL NETWORK LINK FUNCTIONS

Let $\psi : \mathbb{R} \to \Theta$ be some prespecified monotone differentiable function and let $g_j$ be a real valued function. Define

$$\theta_i = \alpha_0 + \psi\left(\sum_{j=1}^{J} \alpha_j g_j(\beta_{0,j} + \beta_j' x_i)\right), \quad \text{for} \quad i = 1, \ldots, n. \tag{5.1}$$

The representation (5.1) extends the generalized linear model (2.1) described by canonical parameters $\theta_i = \tilde{\mu} l^{-1}(\beta_0 + \beta' x_i)$, by introducing, on the right-hand side, what is commonly referred to as a feedforward neural network with one hidden layer and $J$ hidden units. In neural network parlance, the covariates $x_i$ in (5.1) are referred to as input signals, parameters $(\alpha_j, \beta_j)$ are weights, $\alpha_0$ and $\beta_{0,j}$ are referred to as bias terms,

while the canonical parameters $\theta_i$ are called output units. In principle, networks like (5.1) can be fit for any number of hidden units $J$, and for any choice of possibly different "activation functions" $g_j$. However, in this article we will only consider the class of logistic activation functions,

$$g_j(u) = g(u) = \big(1 + \exp(-u)\big)^{-1}, \quad \text{for} \quad j = 1, \ldots, J,$$

where the number of hidden units $J$ will be determined by data exploration. The logistic activation is a popular choice in neural network models, and for us it has the added appeal of a convenient closed-form expression for its derivative. This will be a useful feature when we apply the leapfrog hybrid Monte Carlo algorithm.

The representation (5.1) also describes the usual link function as a special case. For example, let $\alpha_0 = 0$, $J = 1$, $\alpha_1 = 1$, $g_1(u) = u$, and $\psi = \tilde{\mu} l^{-1}$. In the case of a canonical link, $\psi(u) = \tilde{\mu} l^{-1}(u) = u$ and

$$\theta_i = \beta_{0,1} + \beta_1' x_i.$$

Neural networks like (5.1) generate a rich class of functions, even for the case of similar activation functions. Indeed, by employing logistic activations, it is possible to uniformly approximate any continuous function on a compact set by using a large enough number of hidden units (Cybenko 1989; Funahashi 1989; Hornik Stinchcombe, and White 1989). However, the theoretical problem with neural networks is not in their richness, but rather with their lack of identification in parameters. Recently, Sussmann (1992), as well as Hwang and Ding (1997), showed that feedforward networks similar to (5.1) are only identified up to sign equivalences. Two parameters, $(\beta_{0,j}, \beta_j)$ and $(\beta_{0,j'}, \beta_{j'})$, are called sign equivalent if either $(\beta_{0,j}, \beta_j) = (\beta_{0,j'}, \beta_{j'})$ or $(\beta_{0,j}, \beta_j) = -(\beta_{0,j'}, \beta_{j'})$. Therefore, because of the complexity of the network (5.1), and the possibility of permutation and sign indeterminacies, exploring the posterior becomes challenging for conventional Markov chain Monte Carlo methods. This is the reason we explore the use of hybrid Monte Carlo.

## 5.2 HIERARCHICAL PRIORS

The hierarchical prior structure for $(\alpha_j, \beta_{0,j}, \beta_j)$ in the neural network model (5.1) will employ normal priors with conjugate priors for hyperparameters. Hierarchical priors like these have been used with great success in neural network models (MacKay 1992; Neal 1996) and furthermore will greatly simplify our Gibbs sampling. Following the same style as in (2.3), we select

$$\begin{aligned}
(\alpha_j \mid a_j, \tau_j) &\sim \mathrm{N}(a_j, \tau_j) \\
(\beta_{0,j} \mid b_{0,j}, \sigma_{0,j}) &\sim \mathrm{N}(b_{0,j}, \sigma_{0,j}) \\
(\beta_j \mid b_j, W_j) &\sim \mathrm{N}_K(b_j, W_j) \\
(a_j \mid A_j) &\sim \mathrm{N}(0, A_j) \\
(b_{0,j} \mid B_{0,j}) &\sim \mathrm{N}(0, B_{0,j}) \\
(b_j \mid B_j) &\sim \mathrm{N}_K(0, B_j I)
\end{aligned}$$

$$\begin{aligned}(\tau_j^{-1} \mid t_1, t_2) &\sim \text{gamma}(t_1, t_2)\\ (\sigma_{0,j}^{-1} \mid s_1, s_2) &\sim \text{gamma}(s_1, s_2)\\ (W_j^{-1} \mid V_j, v_j) &\sim \text{Wishart}(V_j^{-1}, v_j).\end{aligned} \tag{5.2}$$

To ensure that $\theta_i \in \Theta$ we must also select a suitable prior for $\alpha_0$. In some settings, $\Theta = \mathbb{R}^+$, or $\mathbb{R}^-$, but the majority of applications are for $\Theta = \mathbb{R}$, such as in the binomial or Poisson generalized linear model. When $\Theta = \mathbb{R}$, a convenient choice for $\alpha_0$ is

$$(\alpha_0 \mid \tau_0) \sim \text{N}(0, \tau_0), \tag{5.3}$$

where $\tau_0$ is some reasonably large value, such as 5.

The choice of constants in (5.2) will depend upon the model of interest. As mentioned in Section 3, when fitting the canonical model (2.2) we like to work with noninformative priors for $(\beta_{0,1}, \beta_1)$. However, in the general feedforward network (5.1), it is generally a good strategy to avoid using very flat priors. For example, in a network with many hidden units $J$, it is important to choose a prior for $\alpha_j$ that will suppress large values, otherwise $\theta_i$ can become very large, which can lead to very large or very small mean values $\mathbb{E}(Y_i) = \mu'(\theta_i)$ (in a Poisson regression $\mathbb{E}(Y_i) = \exp(\theta_i)$). One good strategy is to select $A_j = 1/J$, $t_1 = .1$, and $t_2 = 10$. It is also best to select a reasonably flat prior for $\beta_j$, for example, let $B_j = 3$. Furthermore, to avoid having the logistic functions peak at their 0 and 1 limit values, the prior for $\beta_{0,j}$ should not be too flat; a good choice is to let $B_{0,j} = 1$. Like the canonical link case, we use noninformative priors for $\sigma_{0,j}$ and $W_j$ by choosing $s_1 = .01$, $s_2 = 100$, and $V_j = 3I$, $v_j = K + 2$.

### 5.3 Hybrid Monte Carlo for Neural Network Links

As in the canonical link model, the leapfrog hybrid Monte Carlo algorithm is applied to the generalized neural network model (5.1) within a Gibbs sampling framework. In this strategy we first update the conditional parameter for $\alpha_0$, which is then followed by an update for the remaining conditional parameters in the lower hierarchy of (5.2). The updates for the lower level parameters in (5.2) use the hybrid Monte Carlo method, which can also be applied in the update for $\alpha_0$ when its prior is strictly positive and differentiable. We will present the case for $\alpha_0$ specified by (5.3) in the following discussion, otherwise another method such as Metropolis–Hastings will have to be used. After completing these updates, we then update all hyperparameters.

Let $\alpha = (\alpha_1, \ldots, \alpha_J)'$, $\beta_0 = (\beta_{0,1}, \ldots, \beta_{0,J})'$, and $\beta = (\beta_1, \ldots, \beta_J)'$. Upgrade the notation for hyperparameters in the same way. For example, let $W = (W_1, \ldots, W_J)'$ be the matrix of stacked inverse Wishart matrices. The hybrid Gibbs sampler starts by first simulating

$$(\alpha_0 \mid \alpha, \beta_0, \beta, \tau_0), \tag{5.4}$$

$$(\alpha \mid \alpha_0, \beta_0, \beta, a, \tau), \tag{5.5}$$

$$(\beta_0 \mid \alpha_0, \alpha, \beta, b_0, \sigma_0), \tag{5.6}$$

and

$$(\beta \mid \alpha_0, \alpha, \beta_0, b, W). \tag{5.7}$$

Each of the four steps are completed using hybrid Monte Carlo, with each step in (5.5)–(5.7) involving a simultaneous update for all parameters. In theory, the four steps could be combined into one large step involving a simultaneous update for all parameters. In practice, however, this is likely to be computationally infeasible because of the large number of gradients involved and the strong possibility of numerical overflow. This same problem may also occur in the step for $\beta$ when there are a large number of hidden units $J$. To avoid this, the step can be broken into simultaneous updates for each of the $J$ different $\beta_j$ parameters.

The target density $\pi$ in each of the steps (5.4)–(5.7) is proportional to

$$\pi(\alpha_0 \mid \tau_0)\pi(\alpha \mid a, \tau)\pi(\beta_0 \mid b_0, \sigma_0)\pi(\beta \mid b, W) \prod_{i=1}^{n} \exp\left[\frac{y_i\theta_i - \mu(\theta_i)}{a(\phi)} + c(y_i, \phi)\right],$$

which, as in the canonical link case, is strictly positive and differentiable in the parameters by our choice of priors, and because of the smoothness of the neural network (5.1). To apply the hybrid algorithm we must calculate the gradient vector $\Delta = (\log \pi)'$ in each of the steps (5.4)–(5.7). This gives a 1, $J$, $J$ and $(J \times K)$-dimensional vector for $\Delta(\alpha_0)$, $\Delta(\alpha)$, $\Delta(\beta_0)$ and $\Delta(\beta)$, respectively. Letting $\tilde{y}_i = y_i - \mu'(\theta_i)$, as before, these are defined by

$$\begin{aligned}
\Delta(\alpha_0) &= \sum_{i=1}^{n} \tilde{y}_i - \frac{\alpha_0}{\tau_0} \qquad (5.8)\\
\Delta(\alpha_j) &= \sum_{i=1}^{n} \tilde{y}_i g_j(\beta_{0,j} + \beta_j' x_i)\psi'\left(\sum_{j=1}^{J} \alpha_j g_j(\beta_{0,j} + \beta_j' x_i)\right) - \frac{1}{\tau_j}(\alpha_j - a_j)\\
\Delta(\beta_{0,j}) &= \alpha_j \sum_{i=1}^{n} \tilde{y}_i g_j'(\beta_{0,j} + \beta_j' x_i)\psi'\left(\sum_{j=1}^{J} \alpha_j g_j(\beta_{0,j} + \beta_j' x_i)\right) - \frac{1}{\sigma_{0,j}}(\beta_{0,j} - b_{0,j})\\
\Delta(\beta_j) &= \alpha_j \sum_{i=1}^{n} x_i \tilde{y}_i g_j'(\beta_{0,j} + \beta_j' x_i)\psi'\left(\sum_{j=1}^{J} \alpha_j g_j(\beta_{0,j} + \beta_j' x_i)\right) - W_j^{-1}(\beta_j - b_j),
\end{aligned}$$

for $j = 1, \ldots, J$.

After completing the hybrid Monte Carlo updates (5.4)–(5.7), we complete one iteration in the Gibbs cycle by sampling hyperparameters. As in the canonical link case, this is straightforward because of conjugacy:

$$\begin{aligned}
(a_j \mid \alpha_j, \tau_j, A_j) &\sim \mathrm{N}(A_j^* \frac{\alpha_j}{\tau_j}, A_j^*)\\
(b_{0,j} \mid \beta_{0,j}, \sigma_{0,j}, B_{0,j}) &\sim \mathrm{N}(B_{0,j}^* \frac{\beta_{0,j}}{\sigma_{0,j}}, B_{0,j}^*)\\
(b_j \mid \beta_j, W_j, B_j) &\sim \mathrm{N}_K(\Lambda W_j^{-1}\beta_j, \Lambda)
\end{aligned}$$

$$
\begin{aligned}
(\tau_j^{-1} \mid \alpha_j, a_j, t_1, t_2) &\sim \text{gamma}(t_1 + \frac{1}{2}, t_2^*) \\
(\sigma_{0,j}^{-1} \mid \beta_{0,j}, b_{0,j}, s_1, s_2) &\sim \text{gamma}(s_1 + \frac{1}{2}, s_2^*) \\
(W_j^{-1} \mid \beta_j, b_j, V_j, v_j) &\sim \text{Wishart}((V_j^*)^{-1}, v_j + 1),
\end{aligned}
$$

where $A_j^* = (1/\tau_j + 1/A_j)^{-1}$, $B_{0,j}^* = (1/\sigma_{0,j} + 1/B_{0,j})^{-1}$, $\Lambda = (W_j^{-1} + I/B_j)^{-1}$, $t_2^* = (1/t_2 + (\alpha_j - a_j)^2/2)^{-1}$, $s_2^* = (1/s_2 + (\beta_{0,j} - b_{0,j})^2/2)^{-1}$, and $V_j^* = V_j + (\beta_j - b_j)(\beta_j - b_j)'$.

## 5.4 Poisson Regression With Neural Network Link

Here we revisit the Poisson regression analysis of nesting horseshoe crabs in Agresti (1996, chap. 4), which is based on the data from the larger study of Brockmann (1996). Each female horseshoe crab in the study had one male crab companion, and in total there were 173 male-female couples in the data considered here (Agresti 1996, tab. 4.2). In some cases, however, a couple also attracted unattached male crabs, called satellites, who competed with the attached male for fertilization of eggs laid by the female. The number of satellite males for each female is assumed to be a Poisson random variable with a mean which depends upon features of the female crab.

The covariates considered are the carapace width of the female in centimeters, as well as the females' color, which belonged to one of the four categories: medium light, medium, medium dark, and dark in color. Color is a proxy for age, with older crabs tending to be darker. As indicated by Agresti (1996), satellites are attracted to younger females, and therefore it is reasonable to introduce color as a quantitative linear effect in the model. For convenience, the width measurement was standardized to have zero mean and variance one.

The effect of width and age of the female on the mean number of satellites was studied using the neural network model (5.1) with $J = 10$ logistic activations and by selecting $\psi(x) = x$. The selection for the prior was based on the strategy outlined in Section 5.2 with model fitting via the Gibbs sampler described in Section 5.3. Because of the relatively high number of hidden units, the hybrid step (5.7) for $\beta$ was broken into simultaneous updates, one for each of the 10 different $\beta_j$ parameters.

The choice for the number of hidden units, $J = 10$, was determined through informal data exploration, with the final choice representing a compromise between flexibility and avoiding overfitting the model. For example, the median value for the log-likelihood with 10 hidden units was $-452.7$, which was almost identical to the value $-451.5$ obtained by fitting the same model, but with $J = 20$ hidden units. The goodness-of-fit measured by $\sum_i (y_i - \mu_i)^2/\mu_i$, was also similar between the two models, with median values 250.8 and 251.6 for the models with $J = 10$ and $J = 20$ units, respectively. Furthermore, an inspection of residuals revealed no important differences between the two models. A more formal model selection mechanism could be based on the Bayes factor (e.g., Raftery 1996) or through automatic relevance determination (ARD) (Neal 1996, chap. 4).

Figures 3 and 4 contain the predicted values and confidence intervals computed

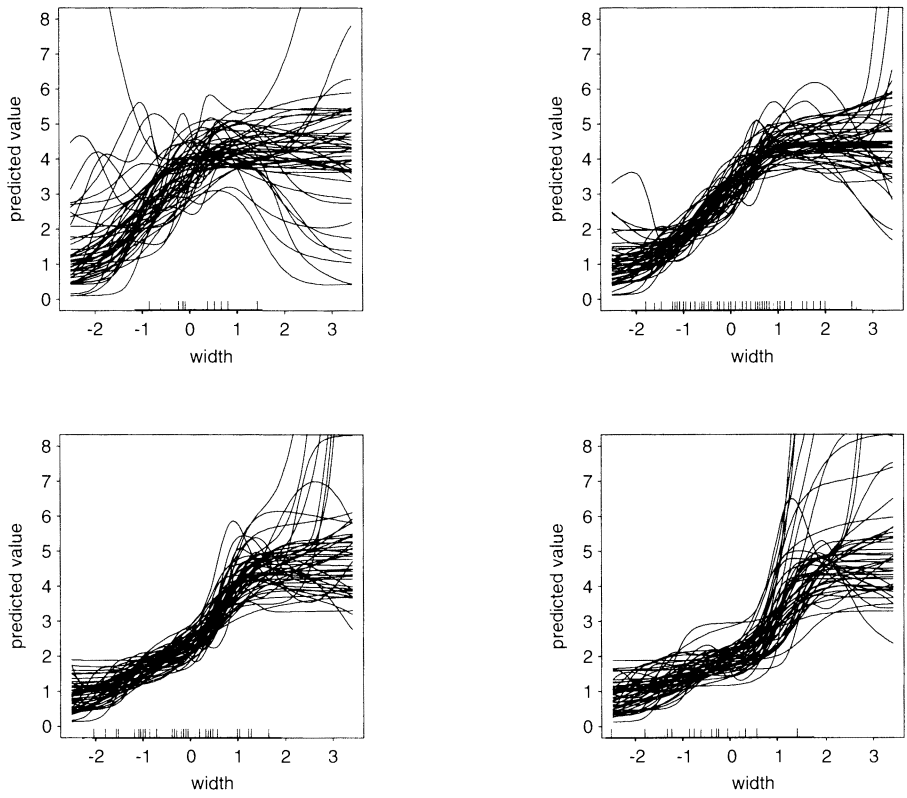


*Figure 3. Predicted values from Poisson regression using 10 hidden units with logistic activations. Values are computed from each 50th iteration of the 2,500 sampled parameter values. From left to right, top to bottom, plots are for crabs that are medium light, medium, medium dark, and dark in color.*

from the output of the Gibbs sampler. The output consisted of 2,500 sampled values, which were subsampled each fourth iteration following an initial 1,000 iteration burn-in, and were based on acceptance rates in the range of 85%–95% for the different updates using $L = 50$ leapfrog steps. The run time per iteration was noticeably longer than for the logistic regression example of Section 4. This is not surprising given the complexity of the gradients (5.8), and in fact with a larger dataset and more hidden units, the computational requirements would soon become too demanding. This point will be addressed in the discussion.

The first and fourth plot in each of Figures 3 and 4 correspond to the medium light and dark female crabs. The larger variability seen in these plots is due to the small number of crabs with these color types (the breakdown was 6.9%, 54.9%, 25.4%, and 12.7% for medium light, medium, medium dark, and dark crabs, respectively). Nevertheless, the four plots suggest that satellite males are attracted to larger females across all color types. This relationship appears to be linear, although there seems to be some kind of thresholding effect for large width values. This should be compared to the predicted values from maximum likelihood estimation, which with a log-link, generates predicted values that appear quadratic in width, especially for medium and medium dark crabs

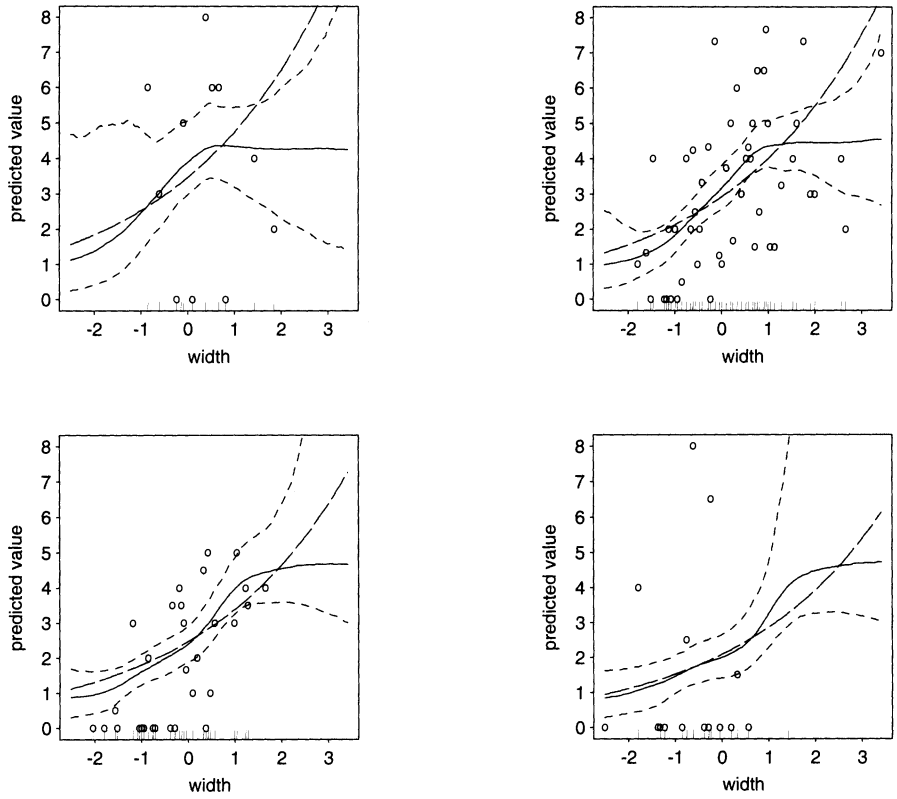


*Figure 4. The 5%–50%–95% quantiles for predicted values as in Figure 3, but calculated from the complete 2,500 sampled parameter values. Circles indicate average observed values, while heavy dashed lines are predicted values using maximum likelihood estimation with a log-link. As in Figure 3, plots from left to right, top to bottom, are for crabs that are medium light, medium, medium dark, and dark in color.*

(which make up most of the data). This illustrates the limitation in using a prespecified link function.

The plots also show, as expected, that satellites are more strongly attracted to the younger females, which can be seen by comparing predicted values at fixed widths across the four colors (and hence age groups). For example, the median predicted value at the baseline average width was approximately 4, 3, 2, and 1.5 for the four different age groups.

## 6. DISCUSSION

The leapfrog hybrid Monte Carlo method (Duane et al. 1987) is an effective method for fitting the Bayesian generalized linear model with canonical link. The simplicity of the gradient in the case of a canonical link makes it easy to implement the algorithm and reduces the amount of computation required in each trajectory. Conventional methods like random walk Metropolis–Hastings require less computation per iteration, but the additional cost of using hybrid Monte Carlo is more than offset by the rapid mixing of

the Markov chain. The benefit of rapid mixing is especially pronounced when the model contains parameters that are highly correlated. The example of Section 4 illustrates this point. There we considered logistic regression with only three covariates, but where two parameters were highly correlated due to quasicomplete separation of the data. Random walk Metropolis and rejection sampling based algorithms perform very poorly in this example, while hybrid Monte Carlo rapidly explores the posterior. Even in such a low-dimensional problem, conventional MCMC methods have difficulty identifying quasicomplete separation, and the problem only becomes worse when there is quasicomplete separation with a large number of covariates. This is the more common scenario, and we cannot rely on conventional methods for dealing with this problem. The hybrid Monte Carlo method, however, works very effectively on these problems, even with up to 20–30 covariates (in the author's experience). To identify quasicomplete separation with a large number of covariates, the strategy is to first standardize covariates (see point 1 below) and then run the leapfrog algorithm, looking for parameters whose posteriors have large standard deviations with a wide range of values in relation to other parameters. These are usually the problematic covariates, and are a signal that quasicomplete separation exists.

There are several things to keep in mind when implementing the leapfrog hybrid Monte Carlo algorithm:

1. It is usually a good idea to standardize all covariates to have zero mean and variance one. This avoids the problem of having to select different step sizes $\delta$ for each covariate, and furthermore, it will reduce the correlation between parameters, thus encouraging faster mixing.
2. Selecting the step size $\delta$ and number of leapfrog steps $L$ in a trajectory can sometimes be tricky when the sampler first starts. Furthermore, these values usually need to be retuned after a few hundred iterations because the likelihood surface and the gradient can sometimes change dramatically as the Markov chain moves over the parameter space. Therefore, it is a good idea to start off the sampler by using random walk Metropolis–Hastings for the first several hundred iterations. This does not entail very much additional programming work because the leapfrog algorithm contains a Metropolis component which can easily be modified into the required Metropolis–Hastings step.
3. The gradient for the neural network link model is fairly complex, and with a large number of hidden units and/or a large dataset, the computations required for a single trajectory can become quite expensive. When this happens, it is a good idea to mix hybrid Monte Carlo with a simpler MCMC method, like random walk Metropolis–Hastings. This reduces the computational burden and we occasionally benefit by a large step from hybrid Monte Carlo. As mentioned in point 2, there is very little additional programming needed to introduce a Metropolis–Hastings mixing step.

## ACKNOWLEDGMENTS

This research was supported by grant funds from the Natural Sciences and Engineering Research Council of Canada. The author is grateful to the associate editor and three referees whose comments were very constructive and helpful in improving the original manuscript.

*[Received June 1998. Revised November 1998.]*

## REFERENCES


Agresti, A. (1996), *An Introduction to Categorical Data Analysis*, New York: Wiley.

Albert, A., and Anderson, J. A. (1984), "On the Existence of Maximum Likelihood Estimates in Logistic Regression Models," *Biometrika*, 71, 1–10.

Brockmann, H. J. (1996), "Satellite Male Groups in Horseshoe Crabs, *Limulus polyphemus*," *Ethology*, 102, 1–21.

Cybenko, G. (1989), "Approximations by Superpositions of a Sigmoidal Function," *Mathematics of Control Signals, and Systems*, 2, 303–314.

Daniels, M. J. (1998), "Computing Posterior Distributions for Covariance Matrices," in *Computing Science*, (vol. 30), *Proceedings of the 30th Symposium on the Interface*, ed. S. Weisberg, pp. 192–196.

Dellaportas, P., and Smith, A. F. M. (1993), "Bayesian Inference for Generalized Linear and Proportional Hazards Models via Gibbs Sampling," *Applied Statistics*, 42, 443–459.

Duane, S. (1985), "Stochastic Quantization Versus the Microcanonical Ensemble: Getting the Best of Both Worlds," *Nuclear Physics B*, 257, 652–662.

Duane, S., and Kogut, J. B. (1986), "The Theory of Hybrid Stochastic Algorithms," *Nuclear Physics B*, 275, 398–420.

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

Funahashi, K. (1989), "On the Approximate Realization of Continuous Mappings by Neural Networks," *Neural Networks*, 2, 183–192.

Gilks, W. R., and Wild, P. (1992), "Adaptive Rejection Sampling for Gibbs Sampling," *Applied Statistics*, 41, 337–348.

Gilks, W. R., Best, N., and Tan, K. K. C. (1995), "Adaptive Rejection Metropolis Sampling within Gibbs Sampling," *Applied Statistics*, 44, 455–472.

Goorin, A. M., Perez-Atayde, A., Gebhardt, M., Andersen, J. W., Wilkinson, R. H., Delorey, M. J., Watts, H., Link, M., Jaffe, N., Frei III, E., and Abelson, H. T. (1987), "Weekly High-Dose Methotrexate and Doxorubicin for Osteosarcoma: The Dana-Farber Cancer Institute/The Children's Hospital—Study III," *Journal of Clinical Oncology*, 5, 1178–1184.

Gustafson, P. (1997), "Large Bayesian Hierarchical Analysis of Multivariate Survival Data," *Biometrika*, 83, 230–242.

Hornik, K., Stinchcombe, M., and White, H. (1989), "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, 2, 359–366.

Horowitz, A. M. (1991), "A Generalized Guided Monte Carlo Algorithm," *Physics Letters B*, 268, 247–252.

Hwang, J. T., and Ding A. A. (1997), "Prediction Intervals for Artificial Neural Networks," *Journal of the American Statistical Association*, 92, 748–757.

Kennedy, A. D. (1990), "The Theory of Hybrid Stochastic Algorithms," in *Probabilistic Methods in Quantum Field Theory and Quantum Gravity*, eds. P. H. Damgaard et al., New York: Plenum Press, pp. 209–223.

Mallick, B. K., and Gelfand, A. E. (1994), "Generalized Linear Models With Unknown Link Functions," *Biometrika*, 81, 237–245.

McCullagh, P., and Nelder, J. A. (1989), *Generalized Linear Models* (2nd ed), London: Chapman and Hall.

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

Mehta, C. R., and Patel, N. R. (1995), "Exact Logistic Regression: Theory and Examples," *Statistics in Medicine*, 14, 2143–2160.

Neal, R. M. (1994), "An Improved Acceptance Procedure for the Hybrid Monte Carlo Algorithm," *Journal of Computational Physics*, 111, 194–203

——— (1996), *Bayesian Learning for Neural Networks*, New York: Springer-Verlag.

Nelder, J. A., and Wedderburn, R. W. M. (1972), "Generalized Linear Models," *Journal of the Royal Statistical Society*, Ser. A, 135, 370–384

Raftery, A. E. (1996) "Approximate Bayes Factors and Accounting for Model Uncertainty in Generalized Linear Models," *Biometrika*, 83, 251–266.

Roberts, G. O., and Rosenthal, J. S. (1995), "Optimal Scaling of Discrete Approximations to Langevin Diffusions," University of Cambridge Research Report 95-11.

Rossky, P. J., Doll, J. D., and Friedman, H. L. (1978), "Brownian Dynamics as Smart Monte Carlo Simulation," *Journal of Chemical Physics*, 69, 4628–4633.

Santner, T. J., and Duffy, D. E. (1986), "A Note on A. Albert and J. A. Anderson's Conditions for the Existence of Maximum Likelihood Estimates in Logistic Regression Models," *Biometrika*, 73, 755–758.

SAS Institute, Inc. (1996), *SAS/STAT Software: Changes and Enhancements Through Release 6.11*, Cary, NC: SAS Institute Inc.

Sussmann, H. J. (1992), "Uniqueness of the Weights for Minimal Feedforward Nets With a Given Input-Output Map," *Neural Networks*, 5, 589–593.

Zeger, S. L., and Karim, M. R. (1991), "Generalized Linear Models With Random Effects: A Gibbs Sampling Approach," *Journal of the American Statistical Association*, 86, 79–86.