Efficient Monte Carlo Methods for Conditional Logistic Regression

Cyrus R. MEHTA, Nitin R. PATEL, and Pralay SENCHAUDHURI

Exact inference for the logistic regression model is based on generating the permutation distribution of the sufficient statistics for the regression parameters of interest conditional on the sufficient statistics for the remaining (nuisance) parameters. Despite the availability of fast numerical algorithms for the exact computations, there are numerous instances where a data set is too large to be analyzed by the exact methods, yet too sparse or unbalanced for the maximum likelihood approach to be reliable. What is needed is a Monte Carlo alternative to the exact conditional approach which can bridge the gap between the exact and asymptotic methods of inference. The problem is technically hard because conventional Monte Carlo methods lead to massive rejection of samples that do not satisfy the linear integer constraints of the conditional distribution. We propose a network sampling approach to the Monte Carlo problem that eliminates rejection entirely. Its advantages over alternative saddlepoint and Markov Chain Monte Carlo approaches are also discussed.

KEY WORDS: Exact Logistic Regression; MCMC; Network Algorithms; Single Saddlepoint; Smart Monte Carlo.

1. INTRODUCTION

Logistic regression is a popular mathematical model for the analysis of binary data with widespread applicability in the physical, biomedical, and behavioral sciences. Parameter inference for this model is usually based on maximizing the unconditional likelihood function. For large well-balanced datasets or for datasets with only a few parameters, unconditional maximum likelihood inference is a satisfactory approach. However unconditional maximum likelihood inference can produce inconsistent point estimates, inaccurate p values, and inaccurate confidence intervals for small or unbalanced datasets and for datasets with a large number of parameters relative to the number of observations. Sometimes the method fails entirely as no estimates can be found that maximize the unconditional likelihood function. A methodologically sound alternative approach that has none of the aforementioned drawbacks is the exact conditional approach. Here one estimates the parameters of interest by computing the exact permutation distributions of their sufficient statistics, conditional on the observed values of the sufficient statistics for the remaining "nuisance" parameters. The major stumbling block to exact permutational inference has always been the heavy computational burden that it imposes. Important contributions to this computational problem have been made by Baglivo, Pagano, and Spino (1996), Hirji (1992), Hirji, Mehta, and Patel (1987, 1988), and Tritchler (1984). Despite the availability of fast numerical algorithms for the exact computations, there are numerous instances where a dataset is too large to be analyzed by the exact methods, yet too sparse or imbalanced for the maximum likelihood approach to be reliable. What is needed is a Monte Carlo alternative to the exact conditional approach that can bridge the gap between the exact and asymptotic methods of inference. The problem is technically difficult, because conventional Monte Carlo methods lead to massive rejection of samples that do not satisfy the constraints of the conditional distribution. We develop a network-based direct Monte Carlo sampling approach that eliminates rejection entirely. We then show how, as a byproduct of the network algorithm, one may compute single saddlepoint approximations to the exact permutation distributions. Finally, we discuss the advantages and limitations of direct Monte Carlo sampling relative to Monte Carlo sampling on Markov chains.

2. FORMULATION OF THE DIRECT MONTE CARLO SAMPLING PROBLEM

Let $\mathbf{Y} = (Y_1, Y_2, \ldots Y_g)$ be g independent binomial random variables, where Y_j represents the number of responses in n_j Bernoulli trials each having a response probability of π_j, and let $\mathbf{y}^* = (y_1^*, y_2^*, \ldots y_g^*)$ be the value of $\mathbf{Y}$ actually observed. Suppose that the response probabilities are specified by the logistic regression model

$$\log\left(\frac{\pi_j}{1-\pi_j}\right) = \boldsymbol{\lambda} a_j + \boldsymbol{\theta} w_j, \tag{1}$$

where $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots \lambda_c)$ and $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots \theta_d)$ are unknown model parameters and $\mathbf{a}_j = (a_{j1}, a_{j2}, \ldots a_{jc})'$ and $\mathbf{w}_j = (w_{j1}, w_{j2}, \ldots w_{jd})'$ are corresponding covariates. Then the likelihood function, or probability of the observed $\mathbf{y}^*$ given $(\boldsymbol{\lambda}, \boldsymbol{\theta})$, is

$$\Pr\{\mathbf{Y} = \mathbf{y}^* | \boldsymbol{\lambda}, \boldsymbol{\theta}\} = \frac{\prod_{j=1}^{g} \binom{n_j}{y_j^*} \exp(\boldsymbol{\lambda} a_j y_j^* + \boldsymbol{\theta} w_j y_j^*)}{\prod_{j=1}^{g} [1 + \exp(\boldsymbol{\lambda} a_j + \boldsymbol{\theta} w_j)]^{n_j}}. \tag{2}$$

If we are interested in making inferences about $\boldsymbol{\theta}$ and regard $\boldsymbol{\lambda}$ as a nuisance parameter, then we may eliminate $\boldsymbol{\lambda}$ from the likelihood function by conditioning on its suffi-

Cyrus R. Mehta is President, Nitin R. Patel is Vice President, and Pralay Senchaudhuri is Director of Research and Development, Cytel Software Corporation, Cambridge, MA 02139 (E-mail: *mehta@cytel.com*). Patel is also Visiting Professor, Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA 02139.

Journal of the American Statistical Association
March 2000, Vol. 95, No. 449, Theory and Methods

cient statistic,

$$\mathbf{s}^* = \sum_{j=1}^{g} \mathbf{a}_j y_j^*. \tag{3}$$

Then the conditional likelihood function is

$$\Pr\{\mathbf{Y} = \mathbf{y}^* | \mathbf{s}^*, \boldsymbol{\theta}\} = \frac{\prod_{j=1}^{g} \binom{n_j}{y_j^*} \exp(\boldsymbol{\theta} w_j y_j^*)}{\sum_{\mathbf{y}\in\Gamma} \prod_{j=1}^{g} \binom{n_j}{y_j} \exp(\boldsymbol{\theta} w_j y_j)}, \tag{4}$$

where the summation in the denominator of (4) is over the conditional reference set

$$\Gamma = \left\{ \mathbf{y}\colon\ 0 \le y_j \le n_j, y_j \text{ integer } \forall j, \text{ and } \sum_{j=1}^{g} \mathbf{a}_j y_j = \mathbf{s}^* \right\}. \tag{5}$$

Let $T(\mathbf{Y})$ be any suitable test statistic for testing the null hypothesis

$$H_0\colon\ \boldsymbol{\theta} = \mathbf{0}.$$

Typical choices of $T(\mathbf{Y})$ include the score, likelihood ratio, and Wald statistics. Substituting $\boldsymbol{\theta} = \mathbf{0}$ in (4), an exact p value for testing H_0 is obtained by computing

$$\begin{aligned} p &\equiv \Pr(T(\mathbf{Y}) \ge T(\mathbf{y}^*) | \mathbf{s}^*, \mathbf{0}) \\ &= \frac{\sum_{\mathbf{y}\in\Gamma, T(\mathbf{y})\ge T(\mathbf{y}^*)} \prod_{j=1}^{g} \binom{n_j}{y_j}}{\sum_{\mathbf{y}\in\Gamma} \prod_{j=1}^{g} \binom{n_j}{y_j}}. \end{aligned} \tag{6}$$

Suppose that $\boldsymbol{\theta} = \theta$, a scalar quantity. In this case a sufficient statistic for θ is the score statistic

$$R = \sum_{j=1}^{g} w_j Y_j, \tag{7}$$

whose observed value is $r^* = \sum w_j y_j^*$. Inferences on θ are based on evaluating the two tail areas

$$\begin{aligned} F_\theta(r^*) &\equiv \Pr(R \le r^* | s^*, \theta) \\ &= \frac{\sum_{\mathbf{y}\in\Gamma, r\le r^*} \prod_{j=1}^{g} \binom{n_j}{y_j} e^{\theta r}}{\sum_{\mathbf{y}\in\Gamma} \prod_{j=1}^{g} \binom{n_j}{y_j} e^{\theta r}}, \end{aligned} \tag{8}$$

and

$$\begin{aligned} G_\theta(r^*) &\equiv \Pr(R \ge r^* | s^*, \theta) \\ &= \frac{\sum_{\mathbf{y}\in\Gamma, r\ge r^*} \prod_{j=1}^{g} \binom{n_j}{y_j} e^{\theta r}}{\sum_{\mathbf{y}\in\Gamma} \prod_{j=1}^{g} \binom{n_j}{y_j} e^{\theta r}}. \end{aligned} \tag{9}$$

Substituting $\theta = 0$ in these expressions yields exact one-sided p values for H_0. An exact $100 \times (1-\gamma)\%$ confidence interval for θ, $(\underline{\theta}, \bar{\theta})$, is obtained by solving $G_{\underline{\theta}}(r^*) = \gamma/2$ and $F_{\bar{\theta}}(r^*) = \gamma/2$ for $\underline{\theta}$ and $\bar{\theta}$.

In this article we present a Monte Carlo algorithm for evaluating (6) to any desired level of accuracy by repeatedly sampling vectors $\mathbf{y} \in \Gamma$ with probability proportional to $\prod_{j=1}^{g} \binom{n_j}{y_j}$. Suppose that we sample N times and that in m of these samples, $T(\mathbf{y}) \ge T(\mathbf{y}^*)$. Then an unbiased estimate of p is $\hat{p} = m/N$, and its variance is estimated by $\sqrt{\hat{p}(1-\hat{p})/N}$. With slight modification, the sampling algorithm can also be used to sample vectors $\mathbf{y} \in \Gamma$ with probability proportional to $\prod_{j=1}^{g} \binom{n_j}{y_j} e^{\theta r}$ for any value of θ. This enables us to evaluate (8) and (9) and thereby obtain the confidence interval $(\underline{\theta}, \bar{\theta})$.

3. NETWORK ALGORITHM FOR DIRECT MONTE CARLO SAMPLING

It is the requirement $\mathbf{y} \in \Gamma$ that makes the sampling problem so difficult. The naive approach is to ignore this constraint during the sampling phase, and to reject all sampled sequences that fail to satisfy the constraint. The main advantage of sampling with rejection is that it is simple to implement and consumes no memory resources on the computer. However, one can easily show that the rejection method is so inefficient as to be unworkable in practice. As an example, consider a logistic regression model with two binary covariates and a constant term:

$$\log \frac{\pi_j}{1-\pi_j} = \lambda_1 + \lambda_2 a_{2j} + \theta w_j.$$

If we are interested in testing the null hypothesis that $\theta = 0$ based on a sample of size n, then we must generate binary sequences, $\{y_1, y_2, \ldots y_n\}$, satisfying the two constraints $\sum_{j=1}^{n} y_j = s_1$ and $\sum_{j=1}^{n} a_{2j} y_j = s_2$, so as to eliminate λ_1 and λ_2 from the likelihood function. Suppose that $s_1 = s_2 = 8$ and $a_{2j} = 1$ for $j \le 10$ and 0 otherwise. Then there are only $10!/8!2! = 45$ ways to select the binary y_j's so as to satisfy the two constraints. On the other hand, there are 2^n ways to select a binary sequence of size n. Thus the acceptance rate is $45/(2^n)$. For $n = 30$, one would require 10^{10} unconstrained samples to pick up 400 acceptable ones. On a Pentium class PC it would take years to estimate a single p value! Moreover, the situation is much worse when there are more than two covariates or the covariates are not binary.

An approach to solving the Monte Carlo sampling problem is to store the reference set Γ as a network of nodes connected by arcs. These nodes and arcs define distinct paths that are in one-to-one correspondence with the elements of Γ. The problem of sampling a sequence $\mathbf{y} \in \Gamma$ with probability proportional to $\prod_{j=1}^{g} \binom{n_j}{y_j} e^{\theta r}$ is then converted into the equivalent problem of traversing a path through the network with that same probability.

3.1 Network Representation for a Small Dataset

It might be helpful to actually display a network for a small dataset. Accordingly, let us consider the following logistic regression model with two covariates and a constant term,

$$\log\left(\frac{\pi_j}{1-\pi_j}\right) = \lambda_1 + \lambda_2 a_{2j} + \theta w_j, \tag{10}$$

in which there are six covariate groups, $j = 1, 2, \ldots 6$. The data for these six covariate groups are tabulated in Table 1.

Suppose that we wish to make inferences about θ. We thus will need the distribution of its sufficient statistic, $r = \sum_j w_j y_j$, conditional on the observed values of the sufficient statistics for λ_1 and λ_2. The observed sufficient statistic for λ_1 is $\sum_j y_j = 5$. The observed sufficient statistic for λ_2 is $\sum_j a_{2j} y_j = 17$. These observed values, along with the ranges of the y_j's determine the reference set Γ. Accordingly, Γ consists of all possible vectors $\mathbf{y} \equiv (y_1, y_2, \ldots, y_6)$ satisfying the following constraints:

$$y_1 + y_2 + y_3 + y_4 + y_5 + y_6 = 5, \tag{11}$$

$$0y_1 + 2y_2 + 3y_3 + 5y_4 + 7y_5 + 17y_6 = 17, \tag{12}$$

$$0 \le y_1 \le 4, \qquad 0 \le y_2, y_3, y_4, y_5, y_6 \le 2, \tag{13}$$

and y_j is integer for all j. The network representation for Γ is shown in Figure 1. An algorithm for constructing this network is described in Section 3.2.

This network has seven stages, labelled $j = 0, 1, \ldots 6$. Each stage contains a collection of nodes, with each node identified uniquely by a triple of the form (j, s_{1j}, s_{2j}), where j refers to the stage, $s_{1j} = \sum_{l=1}^{j} y_l$ is the sum of the contributions of the first j groups of data to the sufficient statistic for λ_1, and $s_{2j} = \sum_{l=1}^{j} a_{2l} y_l$ is the sum of the contributions of the first j groups of data to the sufficient statistic for λ_2. There are as many nodes at stage j as there are possible values of the pair (s_{1j}, s_{2j}). There is a single initial node (0, 0, 0) implying that at stage 0, the start of the network, nothing has been contributed to the values of these two sufficient statistics. There is a single terminal node (6, 5, 17) implying that by stage 6, the end of the network, the total contribution to the sufficient statistic for λ_1 must equal 5 and the total contribution to the sufficient statistic for λ_2 must equal 17, in accordance with the set of constraints Γ. Each node at stage $j, j \le 5$, is connected to one or more nodes at stage $j + 1$ by arcs. The arc connecting node (j, s_{1j}, s_{2j}) to node $(j + 1, s_{1,j+1}, s_{2,j+1})$ corresponds to a value of $y_{j+1} = (s_{1,j+1} - s_{1j})$. A path through the network is a sequence of connected arcs of the form

$$(0, 0, 0) \to (1, s_{11}, s_{21}) \to (2, s_{12}, s_{22}) \cdots \to (6, 5, 17)$$

joining the initial node to the terminal node, and corresponds to a unique vector $\mathbf{y} \in \Gamma$ whose components are

$$y_1 = s_{11}, \qquad y_2 = (s_{12} - s_{11}), \ldots y_6 = (5 - s_{15}).$$

Table 1. Data for Network Example

Group ID (j)	*Group response (y_j)*	*Group size (n_j)*	*Variable 1 (a_{2j})*	*Variable 2 (w_j)*
1	4	4	0	1
2	0	2	2	2
3	0	2	3	3
4	0	2	5	4
5	0	2	7	5
6	1	2	17	6

For example, the network path indicated by the heavy line in Figure 1 corresponds to $\mathbf{y} = (4, 0, 0, 0, 0, 1)$, which is the value of $\mathbf{y} \in \Gamma$ actually observed. The foregoing network has been so constructed that there is one and only one path through it for each $\mathbf{y} \in \Gamma$.

3.2 Network Construction

Constructing the network displayed in Figure 1 is a nontrivial task. This section outlines a general algorithm for constructing a network representation for the reference set Γ defined by (5). Many details of implementation, dealing with nonstatistical issues like the underlying data structures and the role of hashing (Knuth 1969), have been omitted. The network has $g + 1$ stages, labelled $0, 1, \ldots, g$. It contains a single initial node (0, **0**) at stage 0 and a single terminal node $(g, \mathbf{s}^*)$ at stage g. At each intermediate stage, j, the network contains a set of nodes of the form $(j, \mathbf{s}_j)$, where the $(c \times 1)$ vector,

$$\mathbf{s}_j = \sum_{l=1}^{j} \mathbf{a}_l y_l, \tag{14}$$

is one possible partial sum contributed by the first j groups of data to the final value of the sufficient statistic, $\mathbf{s}^*$. Each node at stage j is connected to one or more nodes at stage $j + 1$ by arcs. Let $\mathcal{R}(j, \mathbf{s}_j)$ denote the set of arcs emanating from node $(j, \mathbf{s}_j)$. At the extremity of each such arc is a successor node of the form $(j + 1, \mathbf{s}_{j+1})$. The act of connecting any node $(j, \mathbf{s}_j)$ to a successor node $(j + 1, \mathbf{s}_{j+1})$ by an arc is equivalent to assigning a specific value to the binomial response variable y_{j+1}. The value so assigned is $y_{j+1} = (s_{1,j+1} - s_{1j})$, where s_{1l} is the first component of $\mathbf{s}_l$. Thus the arc-set $\mathcal{R}(j, \mathbf{s}_j)$ also denotes the set of allowable values for the binomial response variable y_{j+1} that are consistent with the linear integer constraint $\mathbf{y} \in \Gamma$, given that the accumulated value of the sufficient statistic through stage j is $\mathbf{s}_j$. The network is fully specified by listing all of the nodes, $(j, \mathbf{s}_j)$, together with their respective arc-sets, $\mathcal{R}(j, \mathbf{s}_j)$, for $j = 0, 1, \ldots, g - 1$. The actual members of $\mathcal{R}(j, \mathbf{s}_j)$ can be identified only on completion of the algorithm for network construction described next.

A path through the network is a sequence of connected arcs of the form

$$(0, \mathbf{0}) \to (1, \mathbf{s}_1) \to (2, \mathbf{s}_2) \cdots \to (g, \mathbf{s}_g)$$

joining the initial node to the terminal node, and corresponds to a unique vector $\mathbf{y} \in \Gamma$ whose components are

$$y_1 = (s_{11} - 0), \qquad y_2 = (s_{12} - s_{11}), \ldots y_g = (s_{1g} - s_{1,g-1}).$$

The goal is to construct the network in such a way that the paths through it are in one-to-one correspondence with the members of the reference set Γ. The network is constructed recursively in two phases. In phase 1 we begin at stage 0 with a single initial node (0, **0**), and proceed to either extend or terminate nodes stage by stage until stage g is reached, in accordance with the following four-step procedure:

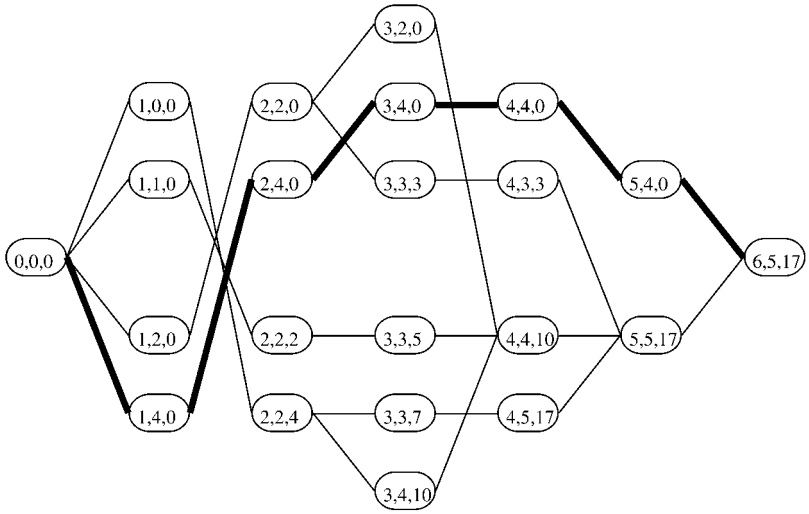

Figure 1. Network Representation for Γ

Step 1. Pick up a node (j, s_j) at stage j.

Step 2. Determine whether this node can be terminated, according to the termination rule specified here. If so, terminate and return to step 1. If not, proceed to step 3.

Termination rule. Node $(j, \mathbf{s}_j)$ may be terminated if it can be established that there exists no continuation $(y_{j+1}, y_{j+2} \dots y_g)$ having the property that

$$\mathbf{s}_j + \sum_{l=j+1}^{g} \mathbf{a}_l y_l = \mathbf{s}^*. \tag{15}$$

We use the infeasibility criteria developed by Hirji, Mehta, and Patel (1987) to establish whether (15) is impossible. Briefly, these criteria rapidly compute a lower bound on the smallest possible value that can be assumed by $\mathbf{s}_j + \sum_{l=j+1}^{n} \mathbf{a}_l y_l$, and an upper bound on the largest possible value that this sum can assume, by choice of $(y_{j+1}, y_{j+2}, \dots y_n)$. Unless $\mathbf{s}^*$ lies inside the interval defined by these two bounds, (15) is impossible.

Early termination of infeasible nodes is desirable because it cuts down on the storage requirements during this phase of network construction. The tighter the infeasibility checks, the earlier one can terminate infeasible nodes (see also Hirji 1992).

Step 3. Generate $n_{j+1} + 1$ arcs from node $(j, \mathbf{s}_j)$, corresponding to the possible realizations $0, 1, \dots n_{j+1}$ for y_{j+1}. At the extremity of each arc, attach a successor node $(j + 1, \mathbf{s}_j + \mathbf{a}_{j+1} y_{j+1})$.

Step 4. If all the nodes at stage j have been processed, set $j \leftarrow (j - 1)$ and return to step 1. Otherwise, return to step 1.

On reaching stage $(g, \mathbf{s}^*)$, we will have built a network that contains all of the paths in Γ plus some additional paths that have been prematurely terminated at infeasible nodes prior to their reaching the final node $(g, \mathbf{s}^*)$. We next proceed with phase 2 of the algorithm in which all these prematurely terminated paths are weeded out, leaving us with the final network that contains only the paths satisfying $\mathbf{y} \in \Gamma$. This is accomplished by the following three-step procedure. To begin, set $j = g - 1$.

Step 1. Drop from the network all terminated stage j nodes; that is, nodes having no arcs moving forward to stage $(j + 1)$. When a node is dropped, eliminate all of the arcs connected to that node from nodes at stage $(j - 1)$. This effectively will terminate several nodes at stage $(j - 1)$.

Step 2. There now remain at stage j only the nodes connected by arcs to successor nodes at stage $(j + 1)$. This enables us to identify the arc-sets $\mathcal{R}(j, \mathbf{s}_j)$ attached to all of the nodes at stage j.

Step 3. Set $j \leftarrow (j - 1)$ and return to step 1.

By the time stage 0 is reached, we will have constructed and stored the entire network representation for Γ. This representation consists of all the nodes $(j, \mathbf{s}_j)$, together with their respective arc-sets $\mathcal{R}(j, \mathbf{s}_j)$, for $j = 0, 1, \dots, g - 1$.

3.3 Network Sampling

A path through the network is a sequence of directed arcs from node $(0, \mathbf{0})$ to node $(g, \mathbf{s}^*)$. Each such path is in one-to-one correspondence with a binomial sequence $\mathbf{y} \in \Gamma$. The objective is to sample paths $\mathbf{y} \in \Gamma$ with probability

$$\frac{\prod_{j=1}^{g} \binom{n_j}{y_j} \exp(\theta w_j y_j)}{\sum_{\mathbf{y} \in \Gamma} \prod_{j=1}^{g} \binom{n_j}{y_j} \exp(\theta w_j y_j)}. \tag{16}$$

Before sampling network paths, we need to gather some preliminary information by means of a rapid backward induction pass through the network. First define the subnetwork $\Lambda(j, \mathbf{s}_j)$ to be all the paths of the form $(j, \mathbf{s}_j) \rightarrow (j + 1, \mathbf{s}_{j+1}) \rightarrow \cdots \rightarrow (g, \mathbf{s}^*)$, from node $(j, \mathbf{s}_j)$ to the terminal node $(g, \mathbf{s}^*)$. Thus $\Lambda(0, \mathbf{0}) = \Gamma$. At any node $(j, \mathbf{s}_j)$ with $j < g$, define

$$\mu_0(j, \mathbf{s}_j) = \sum_{\Lambda(j, \mathbf{s}_j)} \prod_{l=j+1}^{g} \binom{n_l}{y_l} e^{\theta w_l y_l}. \tag{17}$$

Notice that $\mu_0(0, \mathbf{0})$ is the denominator of (16). We can compute $\mu_0(j, \mathbf{s}_j)$ at every node in the network by setting $\mu_0(g, \mathbf{s}^*) = 1$ and arguing recursively as follows. Suppose that we have already evaluated $\mu_0(j+1, \mathbf{s}_{j+1})$ for all the nodes at stage $j+1$. Then at each stage-j node of the form $(j, \mathbf{s}_j)$, we can compute

$$\mu_0(j, \mathbf{s}_j) = \sum_{y_{j+1} \in \mathcal{R}(j, \mathbf{s}_j)} \binom{n_{j+1}}{y_{j+1}} \times \exp(\theta w_{j+1} y_{j+1}) \mu_0(j+1, \mathbf{s}_{j+1}). \quad (18)$$

We proceed in this manner until we reach node (0, **0**).

To sample a path through the network, start at stage 0 with the initial node (0, **0**). Follow a two-step procedure to move from a node at stage j to a successor node at stage $j+1$, until the end node, $(g, \mathbf{s}^*)$, is reached at stage g:

Step 1. At node $(j, \mathbf{s}_j)$, select arc $y_{j+1} \in R(j, \mathbf{s}_j)$ with probability

$$\binom{n_{j+1}}{y_{j+1}} \exp(\theta w_{j+1} y_{j+1}) \frac{\mu_0(j+1, \mathbf{s}_j + \mathbf{a}_{j+1} y_{j+1})}{\mu_0(j, \mathbf{s}_j)}.$$

Step 2. Set $j \leftarrow (j+1), \mathbf{s}_j \leftarrow (\mathbf{s}_j + \mathbf{a}_{j+1} y_{j+1})$, and return to step 1.

On reaching stage g, we will have sampled a path $\mathbf{y} \in \Gamma$ with probability

$$\binom{n_1}{y_1} \exp(\theta w_1 y_1) \frac{\mu_0(1, \mathbf{a}_1 y_1)}{\mu_0(0, \mathbf{0})} \times \binom{n_2}{y_2} \exp(\theta w_2 y_2) \frac{\mu_0(2, \mathbf{a}_1 y_1 + \mathbf{a}_2 y_2)}{\mu_0(1, \mathbf{a}_1 y_1)} \times \cdots \times \binom{n_g}{y_g} \exp(\theta w_g y_g) \times \frac{\mu_0(g, \mathbf{s}^*)}{\mu_0(g-1, \mathbf{a}_1 y_1 + \mathbf{a}_2 y_2 + \cdots + \mathbf{a}_{g-1} y_{g-1})}.$$

Because $\mu_0(0, \mathbf{0})$ is the denominator of (16), $\mu_0(g, \mathbf{s}^*) = 1$, and all other $\mu_0(j, \mathbf{s}_j)$ terms cancel, the foregoing sampling probability reduces to (16).

The advantage of this sampling scheme is that there is no rejection. Each $\mathbf{y}$ is sampled independently and is automatically a member of Γ. The disadvantage is that memory is needed to construct and store the network. The actual memory requirements for any given dataset and model are, however, hard to quantify a priori. They depend on the specifics of the problem such as the sample size, the number of covariate groups, the number of covariates in the model, the proportion of responses, and the spacing between successive covariate values.

4. ANALYSIS OF DATASETS

This section presents a Monte Carlo analysis of three datasets. Inference was based on evaluating (6), (8), and (9). Both exact and Monte Carlo inference was performed for the first dataset, whereas only Monte Carlo inference was computationally feasible for the other two datasets. The corresponding asymptotic inference was performed for all three datasets in the standard way, by maximizing the unconditional likelihood function. For the second dataset we also obtained single and double saddlepoint p values for the individual regression coefficients. Simultaneous multiple hypothesis tests on more than one regression coefficients were based on the likelihood ratio statistic. The asymptotic p values were computed by treating this statistic as a random variable with a chi-squared distribution, whereas the Monte Carlo estimates of the exact p values were obtained by sampling from the exact conditional distribution of this test statistic. All Monte Carlo results were based on directly sampling 100,000 times from the appropriate conditional distribution. All models were fitted with a constant term, but for brevity, the estimates for this term are excluded from the results that follow. All computations were performed on a Pentium 200 PC. Computation times for network building plus Monte Carlo sampling ranged between 1 and 20 minutes. For consistency, all results are displayed to four decimal digits.

4.1 Evaluation of Drug Withdrawal Symptoms

This dataset was supplied by Spadille Biostatistik, Denmark. It is small enough to be amenable to exact inference using LogXact. Our purpose in analysing it here is to demonstrate that the Monte Carlo solution is very accurate indeed, providing answers that are practically the same as the corresponding exact ones. The asymptotic results, on the other hand, are not as accurate for this dataset.

The data displayed in Table 2 consists of children in a pediatric intensive care unit treated with with varying doses and for varying duration with the drug Dormicum. The response variable (RSP) is 1 if withdrawal symptoms were exhibited and 0 otherwise. The covariates are drug dose (DOSE) in mg/kg and number of days treated (DAYS).

Table 3 displays the asymptotic, exact, and Monte Carlo p values (two-sided) and 95% confidence intervals for the parameters of the model

$$\text{RSP} = \text{DOSE} + \text{DAYS}.$$

Notice how well the Monte Carlo p values and confidence intervals estimate the corresponding exact quantities. In contrast, the asymptotic p value for DOSE overestimates the corresponding exact p value by a factor of five. The asymptotic 95% lower confidence bound for dose is likewise off by a factor of two.

4.2 Fraudulent Automobile Insurance Claims

We are grateful to Richard Derrig of the Automobile Insurers Bureau of Massachusetts and the Insurance Fraud Bureau of Massachusetts for providing these data. The dataset consists of 127 claims arising from automobile accidents in 1989. Each claim was classified as either fraudulent (RESP = 1) or legitimate (RESP = 0) by consensus among four independent claims adjusters who examined each case file thoroughly. An exploratory analysis by Weisberg and Derrig (1993) identified 10 binary indicators, each of which

Table 2. Withdrawal Symptoms From Treatment With Dormicum

ID	RSP	DOSE	DAYS
1	0	24.4	4
2	0	1.1	3
3	0	4.6	2
4	0	.5	1
5	0	1.3	2
6	1	51.5	3
7	0	1.1	1
8	0	14.7	4
9	0	.3	1
10	1	39.9	8
11	1	27.2	4
12	1	83.8	8
13	0	4.5	3
14	0	15.8	3
15	1	4.8	3
16	0	5	3
17	1	4.3	5
18	1	32.4	4
19	0	1.1	3
20	0	1.3	2
21	0	.3	1
22	1	.7	1
23	1	99.2	13
24	0	4.3	3
25	1	33.2	8
26	1	29.3	4
27	0	8.9	4
28	0	33.3	11
29	0	10.7	3
30	1	.1	1
31	0	18.5	4
32	0	10	6
33	0	2.3	2
34	0	.9	1
35	0	3.1	2
36	0	7.3	3
37	0	24.4	5

denotes the presence or absence of a potential fraud characteristic in the claim situation. These 10 binary covariates fall into three broad groups relating to Accident (AC1, AC9, and AC16), Claimant (CL7 and CL11), and Injury (IJ2, IJ3, IJ4, IJ6, and IJ12). Presence of the potential fraud characteristic is indicated by a 1, while absence is indicated by a 0. The data are tabulated in Table 4.

We fit the logistic regression model

$$\text{RESP} = \text{AC1} + \text{AC9} + \text{AC16} + \text{CL7} + \text{CL11} + \text{IJ2} + \text{IJ3} + \text{IJ4} + \text{IJ6} + \text{IJ12}$$

to these data in an attempt to identify the statistically significant potential fraud characteristics. Although this is a fairly small dataset, the presence of 10 covariates in the regression causes the exact inference to be computationally infeasible. But we were able to obtain Monte Carlo estimates for the exact p values, accompanied by very small corresponding standard errors. These results are displayed in Table 5. For comparison purposes, Table 5 also displays three types of asymptotic p values: maximum likelihood, single saddlepoint, and double saddlepoint. We discuss the saddlepoint results in Section 5.

Notice that there are substantial differences between the Monte Carlo and the asymptotic maximum likelihood (MLE)–based p values. These MLE results are rather optimistic, suggesting statistically significant effects where none exist. For example, the MLE-based p value for the coefficient AC9 is .0122, whereas the corresponding exact p value, obtained up to Monte Carlo accuracy, is .0769.

One reason for including this dataset in this section is to demonstrate that the Monte Carlo method is especially useful for performing simultaneous tests on multiple regression coefficients. In particular, we wish to perform three separate hypothesis tests: that all of the coefficients belonging to the Accident group (AC1, AC9, and AC16) are simultaneously 0, that all of the coefficients belonging to the Claimant group (CL7 and CL11) are simultaneously 0, and that all of the coefficients belonging to the Injury group (IJ2, IJ3, IJ4, IJ6, and IJ12) are simultaneously 0. We discussed the theory and computational methods underlying these multiple hypothesis tests in Section 1. Table 6 gives p values based on the likelihood ratio test.

The exact versions of these tests are computationally infeasible, but the Monte Carlo versions can provide accurate estimates with extremely small standard errors. As in the univariate case, there are substantial differences between the Monte Carlo p values and MLE-based asymptotic p values that, in the case of the Claimant group at least, might lead to different inferences.

4.3 Calibration of Crash Dummies in Automobile Safety Tests

Hardle and Stoker (1989) analyzed data from 58 simulated side-impact collisions to calibrate "crash dummies" in automobile safety tests. The response variable is binary, taking the value 1 if the collision is judged to have resulted in a fatality. The covariates are the age of the subject, the velocity of the automobile, and the maximal acceleration induced on the subject's abdomen at the time of impact. We have included this dataset here because Strawderman, Casella, and Wells (SCW; 1996) pointed out that LogXact ran out of memory in attempting the exact inference for the foregoing three-covariate logistic regression model. These investigators then obtained estimates of the regression coefficients by using saddlepoint methods to approximate the unconditional joint density function of the MLE and eliminating nuisance parameters through a Bayesian integration.

Table 3. Regression Analysis of Withdrawal Symptoms Data

Regression coefficient	Two-sided P value			95% confidence interval for regression coefficients		
	Asymptotic	Exact	Monte Carlo (SE)	Asymptotic	Exact	Monte Carlo
DOSE	.0286	.0057	.0057 (.0002)	(.0122, .2207)	(.0249, .2195)	(.0250, .2202)
DAYS	.3437	.4507	.4488 (.0026)	(−.8466, .2951)	(−.9292, .2977)	(−.9351, .2891)

Table 4. Data on Fraudulent Insurance Claims

RESP	Covariate group									
rate	AC1	AC9	AC16	CL7	CL11	IJ2	IJ3	IJ4	IJ6	IJ12
0/22 (0%)	0	0	0	0	0	0	0	0	0	0
0/1 (0%)	0	0	0	0	0	0	0	0	0	1
0/4 (0%)	0	0	0	0	0	0	0	0	1	0
0/2 (0%)	0	0	0	0	0	0	0	1	0	0
0/3 (0%)	0	0	0	0	0	0	1	0	0	0
0/10 (0%)	0	0	0	0	0	1	0	0	0	0
0/2 (0%)	0	0	0	0	0	1	0	0	1	0
0/4 (0%)	0	0	0	0	0	1	1	0	0	0
1/1 (100%)	0	0	0	0	0	1	1	0	0	1
1/4 (25%)	0	0	0	0	0	1	1	0	1	0
1/1 (100%)	0	0	0	0	1	0	0	0	0	1
0/1 (0%)	0	0	0	0	1	1	1	0	1	0
0/8 (0%)	0	0	0	1	0	0	0	0	0	0
0/1 (0%)	0	0	0	1	0	0	0	1	1	0
0/3 (0%)	0	0	0	1	0	1	0	0	0	0
0/1 (0%)	0	0	0	1	0	1	0	0	1	0
1/1 (100%)	0	0	0	1	0	1	1	1	0	0
0/1 (0%)	0	0	0	1	1	0	0	0	0	0
1/1 (100%)	0	0	0	1	1	1	0	0	0	0
1/1 (100%)	0	0	0	1	1	1	1	1	1	0
0/1 (0%)	0	0	1	0	0	0	0	0	0	0
1/1 (100%)	0	0	1	0	0	1	0	0	1	0
1/1 (100%)	0	0	1	0	1	1	0	0	0	0
0/1 (0%)	0	1	0	0	1	0	1	0	0	0
0/10 (0%)	1	0	0	0	0	0	0	0	0	0
0/2 (0%)	1	0	0	0	0	0	0	0	1	0
0/1 (0%)	1	0	0	0	0	0	0	0	1	1
0/8 (0%)	1	0	0	0	0	1	0	0	0	0
1/7 (14%)	1	0	0	0	0	1	0	0	1	0
0/1 (0%)	1	0	0	0	1	0	0	0	0	0
1/6 (17%)	1	0	0	0	1	1	0	0	0	0
0/1 (0%)	1	0	0	0	1	1	0	0	0	1
0/3 (0%)	1	0	0	0	1	1	0	0	1	0
0/3 (0%)	1	0	0	1	0	0	0	0	0	0
0/1 (0%)	1	0	0	1	0	0	0	0	1	0
0/1 (0%)	1	0	0	1	0	0	0	1	0	0
0/2 (0%)	1	0	0	1	0	1	0	0	1	0
0/1 (0%)	1	0	0	1	1	0	0	0	1	0
1/1 (100%)	1	0	0	1	1	1	0	0	0	0
1/1 (100%)	1	0	0	1	1	1	0	0	0	1
0/1 (0%)	1	1	0	0	0	0	0	0	0	0
1/1 (100%)	1	1	0	0	0	1	0	0	1	0

It is instructive to compare the asymptotic, Monte Carlo, and SCW results displayed in Table 7.

Notice that for this dataset, the usual unconditional MLEs are closer than the corresponding SCW estimates to the exact estimates, as computed by the Monte Carlo method. This is possibly due to the SCW's choice of a flat prior for the underlying parameters. A prior specifically designed to elicit marginal highest posterior density regions with frequentist coverage properties would likely yield answers closer to those displayed in the "Asymptotic" and "Monte Carlo" columns of Table 7.

5. OTHER SMALL-SAMPLE APPROACHES

The exact permutational approach to inference for the parameters of the logistic regression model might be regarded as the gold standard, for it guarantees unconditional protection from type 1 error at the desired level of significance and unconditional coverage of the parameters of interest at the desired confidence level. When the exact approach is computationally infeasible, direct Monte Carlo sampling from the appropriate permutation distribution is the most suitable alternative. By increasing the size of the Monte Carlo sample, one is guaranteed to produce estimates that are arbitrarily close to the corresponding exact results. Here we consider two other methods of inference that many regard as preferable to the usual unconditional maximum likelihood approach in small samples: saddlepoint inference and Markov chain Monte Carlo (MCMC) inference.

5.1 Saddlepoint Inference

The results in this section apply only to the model (1) for the special case where $\boldsymbol{\theta} = \theta$, a scalar parameter. For conditional inference on θ, we must evaluate the tail areas, (8) and (9), of the distribution of R. Let $\kappa(t) = \log E(e^{Rt}|\theta)$ be the cumulant-generating function of R. A single saddlepoint approximation to (9), developed by Lugannani and Rice (1980), is given by

$$G_\theta(r^*) \simeq 1 - \Phi(z^*) + \phi(z^*)\left(\frac{1}{c^*} - \frac{1}{z^*}\right), \qquad (19)$$

with $z^* = \text{sign}(\hat{T})[2\{\hat{T}u_0 - \kappa(\hat{T})\}]^{1/2}$ and $c^* = \{1 - \exp(-\hat{T})\}\kappa''(\hat{T})^{1/2}$, where $\hat{T}$ is the unique solution to $\kappa'(\hat{T}) = r^*$. To use a continuity correction, replace r^* by $r^* - 1/2$ throughout and redefine $c^* = 2\sinh(\hat{T}/2)\kappa''(\hat{T})^{1/2}$.

It is generally held that the foregoing single saddlepoint approximation is impractical for logistic regression problems because $\kappa(t)$ is computationally intractable (see, e.g., Bedrick and Hill 1992). A byproduct of storing the reference set Γ as a network in the memory of the computer, however, is the ability to evaluate (19) by backward induction. To evaluate (19), we need the cumulant-generating function of R and its first two moments. Using arguments similar to those made in an earlier work (Mehta, Patel, and Senchaudhuri 1998), we can show that these cumulants can be expressed as functions of moments of R of the form

$$\mu_q(0, \mathbf{0}) = \sum_{\mathbf{y}\in\Gamma} \prod_{j=1}^{g} \binom{n_j}{y_j} r^q e^{(\theta+t)r} \qquad (20)$$

Table 5. One-Sided (Right-Tailed) p Values for Fraudulent Insurance Claims Data

Regression	Monte Carlo		Saddlepoint		Asymptotic
coefficient	P value	(SE)	Single	Double[a]	MLE
AC1	.5701	(.0031)	.8386	.6368	.3586
AC9	.0769	(.0009)	.0779	.1290	.0122
AC16	.0490	(.0007)	.0507	.0732	.0190
CL7	.1593	(.0012)	.1610	.1759	.0591
C11	.0535	(.0007)	.0535	.0884	.0254
IJ2	.0075	(.0003)	.0080	NA[b]	.0061
IJ3	.4946	(.0016)	.6127	.5830	.3030
IJ4	.3955	(.0016)	.3969	.5280	.1060
IJ6	.3631	(.0015)	.3660	.4051	.2556
IJ12	.0133	(.0004)	.0137	.0197	.0056

[a] Computations performed by Davison's S-PLUS macros based on Davison (1988).
[b] Convergence not attained.

Table 6. Multiple Hypothesis Tests on Subsets of Automobile Insurance Covariates

Hypothesis test	LR statistic	Asymptotic p value	Monte Carlo p value (SE)
AC1 = AC9 = AC16 = 0	12.89	.0049	.0073 (.0003)
CL7 = CL11 = 0	6.98	.0304	.0660 (.0008)
IJ2 = IJ3 = IJ4 = IJ6 = IJ12 = 0	24.86	.0001	.0011 (.0001)

for $q = 0, 1, 2$. Therefore, it suffices to evaluate these moments. We have already shown in Section 3.3 how, through backward induction on the network, the recursive formula (18) for $\mu_0(j, \mathbf{s}_j)$ can be derived and used to compute $\mu_0(0, \mathbf{0})$. Similar recursive formulas can be obtained for $\mu_1(j, \mathbf{s}_j)$ and $\mu_2(j, \mathbf{s}_j)$ also. Indeed, we can show that

$$\begin{aligned}\mu_1(j, \mathbf{s}_j) &= \sum_{y_{j+1} \in \mathcal{R}(j, \mathbf{s}_j)} \binom{n_{j+1}}{y_{j+1}} \exp\{(t+\theta)(w_{j+1}y_{j+1})\} \\ &\quad \times [(w_{j+1}y_{j+1})\mu_0(j+1, \mathbf{s}_{j+1}) + \mu_1(j+1, \mathbf{s}_{j+1})] \quad (21)\end{aligned}$$

and

$$\begin{aligned}\mu_2(j, \mathbf{s}_j) &= \sum_{y_{j+1} \in \mathcal{R}(j, \mathbf{s}_j)} \binom{n_{j+1}}{y_{j+1}} \exp\{(t+\theta)(w_{j+1}y_{j+1})\} \\ &\quad \times \{(w_{j+1}y_{j+1})^2 \mu_0(j+1, \mathbf{s}_{j+1}) + 2w_{j+1}y_{j+1} \\ &\quad \times \mu_1(j+1, \mathbf{s}_{j+1}) + \mu_2(j+1, \mathbf{s}_{j+1})\}, \quad (22)\end{aligned}$$

where $\mathbf{s}_{j+1} = \mathbf{s}_j + \mathbf{a}_{j+1}y_{j+1}$. These recursions were derived earlier (Mehta et al. 1998) for the special case of stratified $2 \times c$ contingency tables, but the same method can be applied to the more general problem considered here. Backward induction on the network enables us to obtain $\mu_1(0, \mathbf{0})$ and $\mu_2(0, \mathbf{0})$, from which (19) can be readily evaluated.

Single saddlepoint computations have usually been considered intractable (see, e.g., Bedrick and Hill 1992). A practical alternative, developed by Skovgaard (1987) and implemented for generalized linear models by Davison (1988), is the double saddlepoint version of the Lugannani–Rice formula. In the double saddlepoint method,

$$G_\theta(r^*) \simeq 1 - \Phi(z^{**}) + \phi(z^{**}) \left(\frac{1}{c^{**}} - \frac{1}{z^{**}} \right), \quad (23)$$

where, under the null hypothesis that $\theta = 0$, $z^{**} = \text{sign}(\hat{\theta})(D - D_w)^{1/2}$, $\hat{\theta}$ and D_w are the MLE for θ and the deviance statistic based on the full model (1), D is the deviance statistic for the restricted model in which $\theta = 0$, $c^{**} = \{1 - \exp(-\hat{\theta})\}\{\det(\mathbf{X}_w'\hat{\mathbf{V}}_w\mathbf{X}_w)/\det(\mathbf{X}'\hat{\mathbf{V}}\mathbf{X}\}^{1/2}$, $\mathbf{X}_w \equiv \{(\mathbf{a}_1', w_1)', (\mathbf{a}_2', w_2)', \ldots (\mathbf{a}_g', w_g)'\}$ is the $((c+1) \times g)$ design matrix for the full model, $\mathbf{X} \equiv \{\mathbf{a}_1', \mathbf{a}_2', \ldots \mathbf{a}_g'\}$ is the corresponding $(c \times g)$ design matrix for the restricted model, $\hat{\mathbf{V}}_w$ is the estimated covariance matrix of $\mathbf{Y}$ for the full model, and $\hat{\mathbf{V}}$ is the corresponding estimated covariance matrix of $\mathbf{Y}$ for the restricted model. A continuity correction involves replacing r^* with $r^* - 1/2$ and setting $c^{**} = 2 \sinh(\hat{\theta}/2) \det(\mathbf{X}_w'\hat{\mathbf{V}}_w\mathbf{X}_w)/\det(\mathbf{X}'\hat{\mathbf{V}}\mathbf{X}\}^{1/2}$. There is no computational difficulty in evaluating (23) and Davison provides S-PLUS macros on his web site (http://dmawww.epfl.ch/∼brazzale/cond.html) for doing so.

Table 5 displays one-sided right-tailed p values, computed by both the single and double saddlepoint methods for the Fraudulent Automobile Insurance Claims data. The single saddlepoint results are fairly close to the Monte Carlo results and appear to be satisfactory. They were obtained without any continuity correction. The double saddlepoint results are not very accurate, but the continuity-corrected versions appear to perform better than the versions without a continuity correction. On the other hand, even without a continuity correction, the double saddlepoint results are more accurate than the results based on asymptotic normality of the MLE. The accuracy of the single saddlepoint estimates are consistent with the findings of Strawderman and Wells (1998), and the superiority of the single to the double saddlepoint estimates support the findings of Bedrick and Hill (1992).

5.2 Indirect Monte Carlo Sampling via Markov Chains

We have seen that direct Monte Carlo sampling of vectors $\mathbf{Y} \in \Gamma$, with probabilities as given by (16), can be very demanding on memory, because the network representation of Γ must be stored. The MCMC approach attempts to circumvent this difficulty by sampling $\mathbf{Y}$ vectors from a Markov chain whose transition matrix is such that, in the steady state, the probability of sampling vector $\mathbf{Y}$ is (16). Forster, McDonald, and Smith (1996) used this approach by developing a Gibbs sampler to generate the Monte Carlo samples. The method requires a starting vector $\mathbf{y}_0 \in \Gamma$. A readily available starting vector is the one actually observed. The Gibbs sampler then generates a sequence of vectors, $\mathbf{Y}_1, \mathbf{Y}_2, \ldots$, with the components of $\mathbf{Y}_i \equiv (Y_{i1}, Y_{i2}, \ldots, Y_{ig})$ generated from a conditional distribution that depends only on the components of $\mathbf{Y}_{i-1}$. Specifically, Forster et al. suggested conditional distributions of the form

$$\begin{aligned}&f(Y_{i+1,j}, Y_{i+1,g-p+1}, Y_{i+1,g-p+2}, \ldots Y_{i+1,g} | Y_{i+1,1} = y_{i+1,1}, \\ &\quad Y_{i+1,2} = y_{i+1,2}, \ldots Y_{i+1,j-1} = y_{i+1,j-1}, Y_{i+1,j+1} = y_{i,j+1}, \\ &\quad Y_{i+1,j+2} = y_{i,j+2}, \ldots Y_{i+1,g-p} = y_{i,g-p}),\end{aligned}$$

for $j = 1, 2, \ldots, g - p$, and $i = 1, 2, \ldots$. We found that this method is effective for problems with equally spaced covariate values and n_j moderately large for each covariate group. For datasets where these conditions do not hold, however, the method is unreliable. For illustration, consider the dataset given in Table 1. The Gibbs sampler Monte

Table 7. Three Methods of Inference for the Crash Dummies Data

Regression coefficient	95% confidence intervals by three methods		
	Asymptotic	SCW	Monte Carlo
Age	(.0862, .2556)	(.112, .295)	(.0864, .2461)
Velocity	(−.0736, .3662)	(−.033, .461)	(−.0721, .3540)
Acceleration	(−.0122, .0446)	(−.011, .051)	(−.0111, .0456)

Carlo estimate for the exact p value of the conditional test of the hypothesis that $\theta = 0$ is 1.0 even after the sampler has been run for 500,000 iterations. The correct p value (easily computed by direct Monte Carlo sampling and confirmed by LogXact) is .0426, suggesting significance at the 5% level. The Gibbs sampler fails because the underlying Markov chain generating the samples is not irreducible. (For a detailed discussion of this type of problem, see Gilks, Richardson, and Spiegelhalter 1996, pp. 52–54.) For this problem, the conditional reference set Γ contains six members: $\Gamma = \{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \mathbf{y}_4, \mathbf{y}_5\mathbf{y}_6\}$, where $\mathbf{y}_1 = (4,0,0,0,0,1), \mathbf{y}_2 = (2,0,0,2,1,0), \mathbf{y}_3 = (0,2,2,0,1,0), \mathbf{y}_4 = (1,1,1,1,1,0), \mathbf{y}_5 = (2,0,1,0,2,0)$, $\mathbf{y}_6 = (0,2,1,2,0,0)$. We can show that the transition matrix for the underlying Markov chain is the (6×6) identity matrix. That is, no matter what $\mathbf{y}_0$ you use for the starting vector, you will never sample a different one by the Gibbs sampler approach. Clearly the Markov chain is not irreducible. We have communicated this difficulty to Forster and colleagues, and they agree that the Gibbs sampler can be unreliable as it stands. They are currently developing MCMC methods that overcome this shortcoming by using Metropolis–Hastings approaches (personal communication, 1997). A MCMC method for conditional inference in logistic regression that does not have the shortcomings of the Gibbs sampler described here is the method devised by Diaconis and Sturmfels (1998), which uses tools from computational algebraic geometry to define the transition matrix of the Markov chain. We have yet to investigate this method's computational feasibility for problems of the size considered in Section 4.

6. CONCLUDING REMARKS

In this article we have developed a numerical algorithm for direct Monte Carlo sampling from the conditional likelihood of the logistic regression model, given the sufficient statistics for its nuisance parameters. The approach developed here is applicable to more general discrete regression problems as well. For the entire class of generalized linear models with canonical link functions (see McCullagh and Nelder 1989), the sufficient statistic is of the form (3). Thus, in principle, the network representation of the reference set can be extended to Poisson regression and to certain types of polytomous regression models. The method cannot, however, be applied to generalized linear models with noncanonical links (e.g., the probit or tobit for binary regression). In contrast, the saddlepoint and MCMC approaches have no such limitation and are more broadly applicable.

The availability of memory is important to the success of the network sampling. If the entire network cannot fit into the memory of the computer, then it is possible to reduce the memory requirements by relaxing some of the linear integer constraints. But in such a case we will introduce some rejection into the sampling scheme. We will discuss this modification of the network algorithm in a future article.

The option to perform Monte Carlo inference has long been available in the StatXact (1998, version 4) software package, but so far has eluded us for logistic and other discrete regression problem. This is because StatXact deals with data mainly in the form of contingency tables. Therefore, the reference set Γ is simply the collection of all contingency tables with fixed margins, and the network representation of such a reference set can be constructed easily. We have seen that the reference set for logistic regression problems, being the solution to a system of linear integer constraints, is much more difficult to represent as a network. In fact, notwithstanding the efficiency of our algorithm for network construction, there will always be problems for which the network representation of Γ is computationally infeasible. At the heart of the problem lies the fact that finding all the solutions to a system of linear integer constraints is NP-hard. Nevertheless, with improved computer technology enabling us to store intermediate redundant nodes to disk, and the fact that the linear integer constraints restrict the final number of network nodes, we should be able to solve substantially larger problems than was possible with the exact algorithms alone. The examples in Sections 4.2 and 4.3, for instance, were not amenable to exact inference using the LogXact (1996) package, but were easily analyzed by direct Monte Carlo sampling.

As a byproduct of the network construction, we showed how to obtain single saddlepoint estimates for regression parameters. From a computational perspective, the main advantage of this single saddlepoint algorithm over direct Monte Carlo sampling is to replace the repeated forward sampling of network paths with a single backward induction pass through the network. This can be a substantial advantage when the network is too large to be fully stored in memory and parts of it are called up from disk storage on an as-needed basis. The results in Section 4.2 show that single saddlepoint inference performs satisfactorily.

Thanks to the pioneering research of Lugannani and Rice (1980), Daniels (1987), Skovgard (1987), Davison (1988), Reid (1988), Bedrick and Hill (1992), Pierce and Peters (1992), Diaconis and Sturmfels (1998), Kolassa and Tanner (1994), and Forster et al. (1996), we also have the double saddlepoint and MCMC options available for problems where the network cannot be stored at all. But additional research, possibly of an empirical nature, is needed before these methods can be implemented in software. The double saddlepoint method performed poorly in Section 4.2. Those results highlight the great difficulty encountered when attempting to rely on mathematically derived error bounds for the tail areas of conditional densities. Theoretically, both the single and double saddlepoint methods have error terms of the same order, $O(n^{-3/2})$—but yet the single saddlepoint results are appreciably superior to the double saddlepoint results for this example. The term $O(n^{-3/2})$ really means that the error is $Kn^{-3/2}$ for some unknown constant K that is typically ignored. But we can see from this example that K is different for the single and double saddlepoint settings. Additionally, we suspect that K depends on the number of covariates in the regression model, the spacing between successive values of a covariate, and possibly

other unknown factors. The big advantage of the Monte Carlo method is that we can provide explicit error bounds regardless of the number of covariates in the model or the structure of the data, and can make these error bounds arbitrarily small by increasing the Monte Carlo sample size. For example, by taking 100,000 Monte Carlo samples, we were able to establish that the exact p-value estimate for AC9 in Table 4 is .0769 with a standard error of .0009. Thus, for all practical purposes, we have obtained the exact p value. The single saddlepoint answer (.0846) is a good approximation, whereas the double saddlepoint answers (.0291 with no continuity correction, .1290 with continuity correction) are not. But we would not know this unless we had the Monte Carlo answer available for comparison. Therefore, it would be useful to undertake a systematic empirical investigation to establish conditions under which the double saddlepoint estimates would be accurate. A similar investigation is needed for the MCMC approach, to establish conditions under which the probability of irreducibility of the Markov chain is high, thereby minimizing the chance of problems like that uncovered in Section 5.2. With such investigations in place, we envisage software packages of the future providing a smooth transition from exact inference to network-based Monte Carlo inference, to saddlepoint inference, and to MCMC inference as the size of the problem increases progressively. Software implementing the methods described here is included in the new LogXact-4 (2000) package.

[Received May 1998. Revised June 1999.]

REFERENCES

Baglivo, J., Pagano, M., and Spino, C. (1996), "Permutation Distributions via Generating Functions With Applications to Sensitivity Analysis of Discrete Data," *Journal of the American Statistical Association*, 91, 1037–1046.

Bedrick, E. I., and Hill, J. R. (1992), "An Empirical Assessment of Saddlepoint Approximations for Testing a Logistic Regression Parameter," *Biometrics*, 48, 529–544.

Daniels, H. E. (1987), "Tail Probability Approximations," *International Statistical Review*, 55, 37–48.

Davison, A. C. (1988), "Approximate Conditional Inference in Generalized Linear Models," *Journal of the Royal Statistical Society*, Ser. B, 50, 445–461.

Diaconis, P., and Sturmfels, B. (1998), "Algebraic Algorithms for Sampling from Conditional Distributions," *Annals of Statistics*, 26, 363–397.

Forster, J. J., McDonald, J. W., and Smith, P. W. F. (1996), "Monte Carlo Exact Conditional Tests for Log-Linear and Logistic Models," *Journal of the Royal Statistical Society*, Ser. B, 58(2), 445–453.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996), *Markov Chain Monte Carlo in Practice*, Cambridge, U.K.: Chapman and Hall.

Hardle and Stoker (1989), "Investigating Smooth Multiple Regression by the Method of Average Derivatives," *Journal of the American Statistical Association*, 84, 986–995.

Hirji, K. F., Mehta, C. R., and Patel, N. R. (1987), "Computing Distributions for Exact Logistic Regression," *Journal of the American Statistical Association*, 82, 1110–1117.

——— (1988), "Exact Inference for Matched Case-Control Studies," *Biometrics*, 44, 803–814.

Hirji, K. F. (1992), "Computing Exact Distributions for Polytomous Response Data," *Journal of the American Statistical Association*, 87, 487–492.

Knuth, D. E. (1969), *The Art of Computer Programming: Volume 2, Seminumerical Algorithms*, Addison-Wesley, Reading, MA.

Kolassa, J. E., and Tanner, M. A. (1994), "Approximate Conditional Inference in Exponential Families Via the Gibbs Sampler," *Journal of the American Statistical Association*, 89, 697–702.

Lugannani and Rice (1980), "Saddlepoint Approximation for the Distribution of the Sum of Independent Random Variables," *Advanced Applied Probabilities*, 12, 475–490.

LogXact for Windows (1996), *A Software Package for Exact Logistic Regression*, Cambridge, MA: Cytel Software Corporation.

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

Mehta, C. R., Patel, N. R., and Senchaudhuri, P. (1998), "Comment: On Single Saddlepoint Computations for Stratified Contingency Tables," *Journal of the American Statistical Association*, 93, 1313–1316.

Pierce, D. A., and Peters, D. (1992), "Practical Use of Higher Order Asymptotics for Multiparameter Exponential Families," *Journal of the Royal Statistical Society*, Ser. B, 54, 701–725.

Reid, N. (1988), "Saddlepoint Methods and Statistical Inference," *Statistical Science*, 3, 213–238.

Skovgaard, I. (1987), "Saddlepoint Expansions for Conditional Distributions," *Journal of Applied Probability*, 24, 875–887.

StatXact-4 for Windows (1998), *A Software Package for Exact Nonparametric Inference*, Cytel Software Corporation: Cambridge, MA.

Strawderman, R. L., Casella, G., and Wells, M. T. (1996), "Practical Small Sample Asymptotics for Regression Problems," *Journal of the American Statistical Association*, 91, 643–655.

Strawderman, R. L., and Wells, M. T. (1998), "Approximately Exact Inference for the Common Odds Ratio in Several 2×2 Tables," *Journal of the American Statistical Association*, 93, 1294–1307.

Tritchler, D. (1984), "An Algorithm for Exact Logistic Regression," *Journal of the American Statistical Association*, 79, 709–711.