Implementations of the Monte Carlo EM Algorithm

Richard A. LEVINE and George CASELLA

The Monte Carlo EM (MCEM) algorithm is a modification of the EM algorithm where the expectation in the E-step is computed numerically through Monte Carlo simulations. The most flexible and generally applicable approach to obtaining a Monte Carlo sample in each iteration of an MCEM algorithm is through Markov chain Monte Carlo (MCMC) routines such as the Gibbs and Metropolis–Hastings samplers. Although MCMC estimation presents a tractable solution to problems where the E-step is not available in closed form, two issues arise when implementing this MCEM routine: (1) how do we minimize the computational cost in obtaining an MCMC sample? and (2) how do we choose the Monte Carlo sample size? We address the first question through an application of importance sampling whereby samples drawn during previous EM iterations are recycled rather than running an MCMC sampler each MCEM iteration. The second question is addressed through an application of regenerative simulation. We obtain approximate independent and identical samples by subsampling the generated MCMC sample during different renewal periods. Standard central limit theorems may thus be used to gauge Monte Carlo error. In particular, we apply an automated rule for increasing the Monte Carlo sample size when the Monte Carlo error overwhelms the EM estimate at any given iteration. We illustrate our MCEM algorithm through analyses of two datasets fit by generalized linear mixed models. As a part of these applications, we demonstrate the improvement in computational cost and efficiency of our routine over alternative MCEM strategies.

Key Words: Generalized linear mixed models; Gibbs sampler; Importance sampling; Markov chain Monte Carlo; Metropolis–Hastings algorithm; Regenerative simulation; Renewal theory.

1. INTRODUCTION

The EM algorithm provides a tool for obtaining maximum likelihood estimates under models that yield analytically formidable likelihood equations. The EM algorithm is an iterative routine requiring two primary calculations each iteration: Computation of a particular conditional expectation of the log-likelihood (E-step) and maximization of this expectation over the relevant parameters (M-step). The Monte Carlo EM (MCEM), introduced by Wei

Richard A. Levine is Assistant Professor, Department of Statistics, One Shields Avenue, University of California, Davis, CA 95616 (E-mail: ralevine@ucdavis.edu). George Casella is Professor and Chair, Department of Statistics, University of Florida, Griffin/Floyd Hall P.O. Box 118545, Gainesville, FL 32611-8545 (E-mail: casella@stat.ufl.edu).

Journal of Computational and Graphical Statistics, Volume 10, Number 3, Pages 422–439

and Tanner (1990), is a modification of the EM algorithm where the expectation in the E-step is computed numerically through Monte Carlo simulations. Although the Monte Carlo estimate presents a tractable solution to problems where the E-step is not available in closed form, we must account for the additional Monte Carlo (MC) error inherent in the approach and try to minimize the increased computational cost in obtaining the MC sample.

Booth and Hobert (1999) presented a method for gauging Monte Carlo error in the MCEM algorithm through an automated routine to increase the number of MC samples as the algorithm converges. Their routine relies on MC estimation in the E-step via either independent samples from the distribution of interest or importance weighted random samples from a candidate distribution "close" to that distribution. Independent samples allow for computationally inexpensive and straightforward assessment of Monte Carlo error through the central limit theorem. However, such sampling routines are often not available due either to the complexity of the target distribution or lack of a reasonable importance density (under which a large number of samples are needed to attain a good estimator).

Alternatively, McCulloch (1994, 1997) suggested obtaining a sample via Markov chain Monte Carlo techniques, in particular the Gibbs sampler and Metropolis–Hastings algorithm. Though the random variates are dependent in such a scenario, the E-step estimator is still unbiased and approaches the true value as the sample size increases. Furthermore, MCMC techniques are applicable to a wider range of distributions than approaches based on independent samples.

The greater flexibility introduced by MCMC sampling, however, is countered by greater computational cost and difficulties in assessing Monte Carlo error. Application of, say, a Metropolis–Hastings algorithm each iteration of the EM algorithm to evaluate the E-step is significantly more expensive than comparable independent samplers, if such are available (Booth and Hobert 1999). In situations where MCMC is the only alternative, the computational cost could be quite restrictive in that estimation of the E-step to a satisfactory number of significant digits requires an inordinate amount of time (McCulloch 1997). Furthermore, validating central limit theorems under Markov chain sampling could be a mathematically and computationally complex task.

In this article we overcome the computational burdens of MCMC sampling in the MCEM algorithm by applying results from importance sampling and renewal theory. First, we show how to recycle variates generated from previous iterations of the EM algorithm through importance weighting. Hence, the MCEM algorithm generates only one sample for evaluation of subsequent E-steps, rather than obtaining an MCMC sample for each iteration of the EM algorithm. Second, we construct a central limit theorem based on renewals of the Markov chain. Markov chains often exhibit *regeneration times* during which the chain essentially restarts. Excursions between these renewal times are hence independent and identically distributed (see Mykland, Tierney, and Yu 1995). Here, we implement a method of Robert, Rydén, and Titterington (1999) to identify samples in each of these excursions. Consequently, we are able to collect an iid sample, being a subsample of our original MCMC sample. We may then apply a central limit theorem based on these iid observations as in Booth and Hobert (1999) to assess MC error.

The article unfolds as follows. In Section 2 we formalize the MCEM routine and present the importance sampling modification. We also discuss the application of regenerative simulation for assessing MC error. We conclude the section with an outline of our MCEM

algorithm. In Section 3 we apply our MCEM to fitting two models: the logit-normal model of McCulloch (1997) fit to simulated data and the probit-normal model of McCulloch (1994) fit to the salamander data of McCullagh and Nelder (1989). As part of these applications, we compare our model with the performance of other MCEM routines using MCMC for evaluating the E-step.

2. MONTE CARLO EM

Let $\mathbf{y} = (y_1, \ldots, y_n)'$ denote observed data with distribution $f(\mathbf{y} \mid \boldsymbol{\Psi})$ characterized by the s-vector of parameters $\boldsymbol{\Psi}$. Here $'$ denotes vector transpose. The EM algorithm is an iterative routine for computing the maximum likelihood estimate of $\boldsymbol{\Psi}$, denoted $\hat{\boldsymbol{\Psi}}$. The driving force behind the EM algorithm is that the MLE is simpler to compute on the data $\mathbf{y}$ augmented by a set of latent variables $\mathbf{u} = (u_1, \ldots . u_q)'$. For example, we may think of the latent variables as unknown random effects in a generalized linear model.

The EM algorithm thus works on the augmented log-likelihood $\ln f(\mathbf{y}, \mathbf{u} \mid \boldsymbol{\Psi})$ to obtain the MLE of $\boldsymbol{\Psi}$ over the distribution $f(\mathbf{y} \mid \boldsymbol{\Psi})$ where it is assumed that $f(\mathbf{y} \mid \boldsymbol{\Psi}) = \int f(\mathbf{y}, \mathbf{u} \mid \boldsymbol{\Psi})\, d\mathbf{u}$. More specifically, the EM algorithm iterates between a calculation of the expected complete-data likelihood

$$Q\left(\boldsymbol{\Psi} \mid \hat{\boldsymbol{\Psi}}^{(r)}\right) = E_{\hat{\boldsymbol{\Psi}}^{(r)}}\left\{\ln f(\mathbf{y}, \mathbf{u} \mid \boldsymbol{\Psi}) \mid \mathbf{y}\right\}, \tag{2.1}$$

and the maximization of $Q\left(\boldsymbol{\Psi} \mid \hat{\boldsymbol{\Psi}}^{(r)}\right)$ over $\boldsymbol{\Psi}$, where the maximum value of $\boldsymbol{\Psi}$ is denoted by $\hat{\boldsymbol{\Psi}}^{(r+1)}$ and $\hat{\boldsymbol{\Psi}}^{(r)}$ denotes the maximum of $\boldsymbol{\Psi}$ at the rth iteration. Wu (1983) showed that, under regularity conditions, the sequence of values $\{\hat{\boldsymbol{\Psi}}^{(r)}\}$ converges to the MLE $\hat{\boldsymbol{\Psi}}$.

In situations where the E-step is analytically troublesome, we may estimate the quantity (2.1) from Monte Carlo simulations. Note the expectation in (2.1) is over the latent variable $\mathbf{u}$. In particular,

$$E_{\hat{\boldsymbol{\Psi}}^{(r)}}\left\{\ln f(\mathbf{y}, \mathbf{u} \mid \boldsymbol{\Psi}) \mid \mathbf{y}\right\} = \int \ln f(\mathbf{y}, \mathbf{u} \mid \boldsymbol{\Psi})\, g\left(\mathbf{u} \mid \mathbf{y}, \hat{\boldsymbol{\Psi}}^{(r)}\right) d\mathbf{u},$$

where $g(\mathbf{u} \mid \mathbf{y}, \boldsymbol{\Psi})$ is the conditional distribution of the latent variables given the observed data and $\boldsymbol{\Psi}$. If we obtain a sample $\mathbf{u}_1^{(r)}, \ldots, \mathbf{u}_m^{(r)}$ from the the distribution $g(\mathbf{u} \mid \mathbf{y}, \hat{\boldsymbol{\Psi}}^{(r)})$, this expectation may be estimated by the Monte Carlo sum

$$Q_m\left(\boldsymbol{\Psi} \mid \hat{\boldsymbol{\Psi}}^{(r)}\right) = \frac{1}{m}\sum_{t=1}^{m} \ln f\left(\mathbf{y}, \mathbf{u}_t^{(r)} \mid \boldsymbol{\Psi}\right), \tag{2.2}$$

where the subscript m denotes the dependence of this estimator on the MC sample size. By the law of large numbers, the estimator in (2.2) converges to the theoretical expectation in (2.1). The EM algorithm can thus be modified into an *MCEM* whereby the E-step is replaced by the estimated quantity from (2.2). The M-step maximizes then the sum (2.2) over $\boldsymbol{\Psi}$. See Chan and Ledolter (1995) for more details.

This article focuses solely on the problem of an intractable E-step. Although the M-step may also require sophisticated numerical routines, our applications in Section 3 yield straightforward maximizations in the M-step. Implementation of the MCEM algorithm

then presents two important issues for us: how do we obtain a random sample from the distribution $g(\mathbf{u} \mid \mathbf{y}, \Psi)$, and how do we choose m? We discuss these issues in the following two subsections. The third subsection summarizes our MCEM algorithm.

2.1 IMPORTANCE SAMPLING

We consider the situation where the sample of latent variables $\mathbf{u}_1, \ldots, \mathbf{u}_m$ in the E-step is obtained from a Markov chain Monte Carlo routine such as the Gibbs sampler or Metropolis–Hastings algorithm with stationary distribution $g(\mathbf{u} \mid \mathbf{y}, \Psi)$. As mentioned in Section 1, drawing an MCMC sample each iteration of the EM algorithm could be prohibitively costly particularly for large m.

The computational expense of the MCMC based MCEM algorithm can be substantially improved through an application of importance sampling. We initialize the algorithm with a sample $\mathbf{u}_1, \ldots, \mathbf{u}_m$ from the distribution $g(\mathbf{u} \mid \mathbf{y}, \Psi^{(0)})$, where $\Psi^{(0)}$ is the initial value of the parameter Ψ at the start of the EM algorithm. At each iteration r, rather than obtaining a new sample from $g(\mathbf{u} \mid \mathbf{y}, \hat{\Psi}^{(r)})$ with the most recent iterate $\hat{\Psi}^{(r)}$, we can importance weight the original sample through the updated distribution $g(\mathbf{u} \mid \mathbf{y}, \hat{\Psi}^{(r)})$. That is,

$$Q_m\left(\Psi \mid \hat{\Psi}^{(r)}\right) = \sum_{t=1}^{m} w_t \ln f(\mathbf{y}, \mathbf{u}_t \mid \Psi) \Big/ \sum_{t=1}^{m} w_t, \tag{2.3}$$

where the original sample is corrected for the new information we have at iteration r through the weights

$$w_t = \frac{g\left(\mathbf{u}_t \mid \mathbf{y}, \hat{\Psi}^{(r)}\right)}{g\left(\mathbf{u}_t \mid \mathbf{y}, \Psi^{(0)}\right)}.$$

[See Robert and Casella (1999, chap. 3) for a discussion of importance sampling.]

The cost in obtaining the weights w_t is less than obtaining a new sample. The reason for this expense saving is that the weights are not dependent on the unknown likelihood $L(\Psi \mid \mathbf{y})$. Note that

$$w_t = \frac{L\left(\hat{\Psi}^{(r)} \mid \mathbf{u}_t, \mathbf{y}\right) / L\left(\hat{\Psi}^{(r)} \mid \mathbf{y}\right)}{L\left(\Psi^{(0)} \mid \mathbf{u}_t, \mathbf{y}\right) / L\left(\Psi^{(0)} \mid \mathbf{y}\right)}.$$

Hence the likelihood $L(\Psi \mid \mathbf{y})$ cancels in the formulation (2.3) so that

$$Q_m\left(\Psi \mid \hat{\Psi}^{(r)}\right) = \frac{\sum_{t=1}^{m} w_t' \ln f(\mathbf{y}, \mathbf{u}_t \mid \Psi)}{\sum_{t=1}^{m} w_t'},$$

where

$$w_t' = \frac{L\left(\hat{\Psi}^{(r)} \mid \mathbf{u}_t, \mathbf{y}\right)}{L\left(\Psi^{(0)} \mid \mathbf{u}_t, \mathbf{y}\right)}.$$

The importance sampling estimator may, alternatively, be calculated as

$$Q_m\left(\Psi \mid \hat{\Psi}^{(r)}\right) = \frac{1}{m}\sum_{t=1}^{m} w_t \ln f(\mathbf{y}, \mathbf{u}_t \mid \Psi), \tag{2.4}$$

where $\sum_{t=1}^{m} w_t$ is replaced by m in the denominator. This choice will not affect the EM algorithm since the unknown normalizing constant

$$\frac{L(\Psi^{(0)} \mid \mathbf{y})}{L(\hat{\Psi}^{(r)} \mid \mathbf{y})}$$

depends only on the known values $\Psi^{(0)}$ and $\hat{\Psi}^{(r)}$ and not the unknown value of Ψ. This constant does not then come into play in the maximization step. However, we choose the estimator (2.3) in order to avoid calculation of this normalizing constant in the routine described in the next subsection for choosing m. The estimator (2.3) may be further rationalized by the fact that $E[w] = 1$, where the expectation is taken with respect to $g(\mathbf{u} \mid \mathbf{y}, \Psi^{(0)})$. Furthermore, the estimator (2.3) often has smaller mean squared error than (2.4); see Liu (1996) and Casella and Robert (1998).

Importance sampling estimators are not without drawbacks. Most notably, if the importance density $g(\mathbf{u}_t \mid \mathbf{y}, \hat{\Psi}^{(r)})$ is not close enough to $g(\mathbf{u}_t \mid \mathbf{y}, \Psi^{(0)})$, the weights w_t will vary widely giving many samples little weight and allowing for a few variates to be overinfluential. Consequently, the estimator (2.3) will be imprecise (Robert and Casella 1999, sec. 3.3.2; see also Geyer 1991). In our setting, if the initial values $\Psi^{(0)}$ are poor, the importance sampling estimator will take a long time to converge. We alleviate this problem by initiating a burn-in whereby for the first few iterations, a new sample is obtained from $g(\mathbf{u}_t \mid \mathbf{y}, \hat{\Psi}^{(r)})$ rather than importance weighting. McCulloch (1997) showed that such an algorithm reaches the neighborhood of the MLE quickly. Following the burn-in, the target density should be close enough to the distribution $g(\mathbf{u}_t \mid \mathbf{y}, \hat{\Psi})$ based on the MLE $\hat{\Psi}$ to ensure well-behaved weights; that is, weights with small variance.

The burn-in idea is further rationalized by the choice of m. Tanner (1993, sec. 4.5) suggests increasing m as the EM algorithm progresses. Hence, the first few iterations are implemented with a relatively small MC sample size, perhaps as small as $m = 10$ as we will see in Section 3. Generating a sample of size 10 each iteration is a computationally inexpensive task. In later iterations, when m is of the order of a thousand to tens of thousands, the cost of generating a sample is too much to perform each iteration. Importance sampling thus becomes particularly crucial for feasible running of the MCEM under MCMC sampling at these later stages.

The requisite burn-in time is problem specific, dependent on how close we need to be to the MLE in order to obtain stable importance weights in subsequent M-steps. In the application of Section 3.2, 16 burn-in steps are performed. This burn-in follows our suggested implementation of running the MCEM algorithm for a burn-in period of one minute. In our applications and experience, the EM algorithm converges to within a small enough neighborhood of the local maximum relatively quickly, ensuring stable importance weights beyond such a short burn-in period. In practice, of course, trial runs of the EM algorithm with importance sampling, gauging the variability in the importance weights as the algorithm converges, will provide the user with an idea for the appropriate burn-in time.

2.2 CENTRAL LIMIT THEOREM

The choice of Monte Carlo sample size m is equally important to the sampling method chosen for the MCEM algorithm. We do not want to start with a large value of m when the iterates $\hat{\Psi}^{(r)}$ are far from the MLE $\hat{\Psi}$. The tradeoff between improving accuracy and the computational cost in obtaining more samples favors starting the algorithm with small MC samples. However, as the EM algorithm progresses, we may wish to increase m as the approximation $\hat{\Psi}^{(r)}$ approaches the true MLE (Tanner 1993, sec. 4.5).

Up until recently, the choice of m was purely ad hoc; that is, one would arbitrarily increase m at predetermined iterations of the EM algorithm. Booth and Hobert (1999) presented an automated routine where the MC sample size is chosen through considerations of the Monte Carlo error inherent in the Monte Carlo sum (2.2). In particular, since the random variates $\mathbf{u}_1, \ldots, \mathbf{u}_m$ are generated independently, the central limit theorem (CLT) ensures that, conditional on the iterate $\hat{\Psi}^{(r)}$, the current iterate $\hat{\Psi}^{(r+1)}$ is approximately normally distributed. Thus, we may estimate Monte Carlo error via the normal distribution. In particular, if the past value $\hat{\Psi}^{(r)}$ lies in an approximate confidence interval about $\hat{\Psi}^{(r+1)}$, then the Monte Carlo error is said to swamp the EM step and the number of simulations, m, is increased (see Booth and Hobert 1999 for more details).

This idea may be generalized to dependent samples $\mathbf{u}_1, \ldots, \mathbf{u}_m$ generated by MCMC routines. The difficulty, of course, is obtaining an appropriate CLT and then estimating the corresponding Monte Carlo variance for the requisite confidence interval. The MCMC literature presents a number of CLTs depending on the induced Markov chain (see, e.g., Kipnis and Varadhan 1986; Tierney 1994; Robert 1995a; and Robert and Casella 1999 to name a few). Based on this work, application of a CLT appears feasible in the MCMC setting upon checking the appropriate regularity conditions. However, computation of the variance of the asymptotic distribution is particularly difficult under these CLTs. The dependency between MC samples forces a variance estimation equivalent to estimating the spectral density function at frequency zero (see Geyer 1992 and Tierney 1994).

As an alternative approach, renewal theory and regenerative simulation presents methods for extracting independent subsets of the Markov chain under which the classical CLT may be applied (see Robert 1995a and Mykland et al. 1995). We apply the ideas of Robert et al. (1999) along these lines.

Recall we have a sample $\mathbf{u}_1, \ldots, \mathbf{u}_m$ taken from the distribution $g(\mathbf{u} \mid \mathbf{y}, \Psi)$ under stationarity of the Markov chain. We will assess Monte Carlo error in our MCEM algorithm through a confidence interval about some function $h(\mathbf{u})$ as defined later. The confidence interval will require three pieces: a CLT to ensure normal critical values, and estimates of both $E_g[h(\mathbf{u})]$ and $\text{var}_g[h(\mathbf{u})]$. We can estimate the expectation with

$$\hat{\mu}_m = \frac{\sum_{t=1}^{m} w_t \, h(\mathbf{u}_t)}{\sum_{t=1}^{m} w_t}, \tag{2.5}$$

and the variance by

$$\hat{v}_m = \left\{ \sum_{t=1}^{m} w_t \, [h(\mathbf{u}_t)]^2 \Big/ \sum_{t=1}^{m} w_t \right\} - \hat{\mu}_m^2, \tag{2.6}$$

where the weights w_t are as defined in Section 2.1. Implementation of a central limit theorem, however, is complicated by the correlation between sample points from the Markov chain.

We use a subsampling scheme from Robert et al. (1999) to overcome Markov chain correlation issues. Specifically, choose the sequence $X_1, \ldots X_N$ such that $X_k - 1 \sim$ Poisson(ν_k), where $\nu_k = \nu k^d$ for some $\nu \geq 1$ and $d > 0$ as in Robert et al. (1999). The sums $t_k = x_1 + \cdots + x_k$ are used as the subsampling points; that is, we consider $\mathbf{u}_{t_1}, \ldots, \mathbf{u}_{t_N}$ where $N = \sup\{k : t_k \leq m\}$ is the number of subsamples taken from the m random variates. Note that ν and d specify the distance required, on average, to ensure subsampling from different renewal periods when constructing the subchain. In our experience, $\nu = 1$ and $d = .5$ provides for appropriate Poisson mean ν_k for the subsampling scheme.

Let

$$S_m = \frac{1}{\sqrt{N\hat{v}_m}} \sum_{k=1}^{N} \left[h(\mathbf{u}_{t_k}) - \hat{\mu}_m\right] \tag{2.7}$$

for $j = 1, \ldots, s$. Recall that a Markov chain $\mathbf{u}_1, \ldots \mathbf{u}_m$ is *strongly mixing* or α-*mixing* if

$$\alpha(t) = \sup_{A,B} |P(\mathbf{u}_t \in A, \mathbf{u}_0 \in B) - P(\mathbf{u}_t \in A)P(\mathbf{u}_0 \in B)|$$

approaches zero as t goes to infinity and $\mathbf{u}_0 \sim g(\mathbf{u} \mid \mathbf{y}, \boldsymbol{\Psi})$. If $\alpha(t) \leq C\gamma^t$ for some constant $C \geq 0$ and $\gamma \in (0, 1)$, then the mixing coefficients are *geometrically decaying*. We then have the following central limit theorem.

Theorem 1. *If the Markov chain* $\{\mathbf{u}_k\}$ *is ergodic and strongly mixing with geometrically decaying mixing coefficients, and*

$$E_g\, |h(\mathbf{u})|^{2+\delta} < \infty$$

for some $\delta > 0$, *then the normalized sum* S_m *converges weakly to the standard normal distribution (Robert et al. 1999).*

We choose s functions $h_j(\mathbf{u}) = \frac{\partial}{\partial\psi(j)} \ln f(\mathbf{u}, \mathbf{y} \mid \boldsymbol{\Psi})$, $j = 1, \ldots, s$, to study MC error in estimation of each of the s components of $\boldsymbol{\Psi}$. Let $\mathbf{h}(\mathbf{u}) = (h_1(\mathbf{u}), \ldots, h_s(\mathbf{u}))'$. Under this choice, the expectation is

$$E_g[\mathbf{h}(\mathbf{u})] = Q^{(1)}(\boldsymbol{\Psi} \mid \boldsymbol{\Psi}') = \frac{\partial Q(\boldsymbol{\Psi} \mid \boldsymbol{\Psi}')}{\partial \boldsymbol{\Psi}}.$$

Theorem 1 will be used to choose the Monte Carlo sample size m each iteration. For direct application of the theorem, we assume the components of the s-vector $\mathbf{h}(\mathbf{u})$ are independent. In particular, we will construct a $(1 - \alpha)$ confidence region around $Q^{(1)}(\boldsymbol{\Psi} \mid \hat{\boldsymbol{\Psi}}^{(r)})$, namely, the set of intervals for each $j = 1, \ldots, s$,

$$\hat{\mu}_{m;j} \pm z_{1-\alpha/2} \cdot \hat{v}_{m;j}. \tag{2.8}$$

Here $z_{1-\alpha}$ denotes the $(1 - \alpha)$ critical value of the standard normal distribution and $\hat{\mu}_{m;j}$ and $\hat{v}_{m;j}$ are the estimators of the expectation and variance of $h_j(\mathbf{u})$ from (2.5) and (2.6), respectively. Note these estimators for the mean and variance are computed by substituting

the most recent MLE update $\hat{\boldsymbol{\Psi}}^{(r+1)}$. In our experience, a value of $\alpha = .25$ provides sufficient width to our confidence intervals for gauging Monte Carlo error. This value was also chosen by Booth and Hobert (1999) for their Monte Carlo error estimation routine.

We will increase m following iteration $(r+1)$ if for any $j = 1, \ldots, s$ the value from the previous iteration,

$$Q^{(1)}_{m;j}\left(\hat{\boldsymbol{\Psi}}^{(r)} \mid \boldsymbol{\Psi}^{(r-1)}\right) = \sum_{k=1}^{N} w_{t_k} \frac{\partial}{\partial \psi(j)} \ln f(\mathbf{u}_{t_k}, \mathbf{y} \mid \boldsymbol{\Psi}) \Bigg/ \sum_{t=1}^{m} w_{t_k} \Bigg|_{\boldsymbol{\Psi}=\hat{\boldsymbol{\Psi}}^{(r)}},$$

based on the subsample, lies in the confidence interval (2.8). The rationale is that the limiting distribution dispersion measures Monte Carlo error in consecutive steps of the EM algorithm. If the confidence interval (2.8), based on the $(r+1)$st iteration estimate, covers the value from the rth EM iteration, then we can not distinguish the EM estimates from consecutive iterations above the Monte Carlo error. That is to say that the Monte Carlo error swamps the current MLE estimate $\hat{\boldsymbol{\Psi}}^{(r+1)}$. More Monte Carlo samples are thus required to attain reasonable precision at the next EM iteration. We choose to increase m to $m + m/c$ where c is a positive constant, with $c = 3$ as suggested by Booth and Hobert (1999).

A number of remarks are in order.

1. Most Markov chains induced by MCMC routines such as the Metropolis–Hastings algorithm and the Gibbs sampler applied in practice are ergodic (see, e.g., Robert and Casella 1999, chaps. 6 and 7).
2. The α-mixing criterion of Theorem 1 may seem foreboding. However, every positive recurrent aperiodic Markov chain is α-mixing (Robert 1994). Additionally, relationships exist between α-mixing and other types of mixing as well as minorization conditions (see, e.g., Tierney 1994). This theory lends well to showing Markov chains induced by independent and dependent Metropolis–Hastings samplers and Gibbs samplers are strongly mixing.
3. The presumed independence between components of $\mathbf{h}(\mathbf{u})$ provides for a conservative evaluation of Monte Carlo error; that is, we are constructing rectangular confidence regions (2.8) instead of confidence ellipsoids over these components. The independence assumption may be lifted through an application of the Cramér–Wold device (Robert et al. 1999); namely, s-dimensional random vectors $\mathbf{X}_n$, converge in distribution to an s-dimensional standard normal random vector $\mathbf{Z}$ if and only if $\mathbf{a}'\mathbf{X}_n$ converges in distribution to $\mathbf{a}'\mathbf{Z}$ for all s-dimensional vectors $\mathbf{a}$. In particular, we may reduce the multivariate case to limit theorems in one dimension. Here, for simplicity, we restrict ourselves to the rectangular confidence region in (2.8).
4. Note that $Q^{(1)}_m(\boldsymbol{\Psi}^{*(r+1)} \mid \hat{\boldsymbol{\Psi}}^{(r)}) = \mathbf{0}$ for the true MLE $\boldsymbol{\Psi}^{*(r+1)}$ at iteration $(r+1)$. Therefore the mean value $Q^{(1)}_m(\hat{\boldsymbol{\Psi}}^{(r+1)} \mid \hat{\boldsymbol{\Psi}}^{(r)}) \approx \mathbf{0}$. We may thus reduce computations slightly by constructing the confidence interval (2.8) to be symmetric around zero.
5. The parameters α and c determine the width of the confidence interval for gauging Monte Carlo error and the amount by which the Monte Carlo sample size m should be increased respectively. Optimal choices for these parameters, as well as the parameters ν and d in the Poisson subsampling scheme, will be investigated in future work.

2.3 ALGORITHM

1. Initialize m, $\boldsymbol{\Psi}^{(0)}$.
2. Generate $\mathbf{u}_1, \ldots \mathbf{u}_m \sim g(\mathbf{u} \mid \mathbf{y}, \boldsymbol{\Psi}^{(0)})$ via an MCMC algorithm.

   At iteration $r+1$:
3. Compute the importance weights

$$w_t = \frac{L(\hat{\boldsymbol{\Psi}}^{(r)} \mid \mathbf{u}_t, \mathbf{y})}{L(\boldsymbol{\Psi}^{(0)} \mid \mathbf{u}_t, \mathbf{y})}.$$

4. *E-step*: Estimate $Q(\boldsymbol{\Psi} \mid \hat{\boldsymbol{\Psi}}^{(r)})$ by

$$Q_m(\boldsymbol{\Psi} \mid \hat{\boldsymbol{\Psi}}^{(r)}) = \sum_{t=1}^{m} w_t \ln f(\mathbf{u}_t, \mathbf{y} \mid \boldsymbol{\Psi}) \Big/ \sum_{t=1}^{m} w_t.$$

5. *M-step*: Maximize $Q_m(\boldsymbol{\Psi} \mid \hat{\boldsymbol{\Psi}}^{(r)})$ to obtain $\hat{\boldsymbol{\Psi}}^{(r+1)}$.
6. *MC error estimation*:

   (a) Compute for each $j = 1, \ldots, s$

$$\hat{\mu}_{m;j} = \sum_{t=1}^{m} w_t \frac{\partial}{\partial \psi(j)} \ln f(\mathbf{u}_t, \mathbf{y} \mid \boldsymbol{\Psi}) \Big/ \sum_{t=1}^{m} w_t \Big|_{\boldsymbol{\Psi}=\hat{\boldsymbol{\Psi}}^{(r+1)}}.$$

   (b) Compute for each $j = 1, \ldots, s$

$$\hat{v}_{m;j} = \sum_{t=1}^{m} w_t \left[\frac{\partial}{\partial \psi(j)} \ln f(\mathbf{u}_t, \mathbf{y} \mid \boldsymbol{\Psi})\right]^2 \Big/ \sum_{t=1}^{m} w_t - \hat{\mu}_{m;j}^2 \Big|_{\boldsymbol{\Psi}=\hat{\boldsymbol{\Psi}}^{(r+1)}}.$$

   (c) Obtain for each $j = 1, \ldots, s$ a $(1-\alpha)$ confidence interval about $Q_j^{(1)}(\boldsymbol{\Psi} \mid \hat{\boldsymbol{\Psi}}^{(r)})$

$$\hat{\mu}_{m;j} \pm z_{1-\alpha/2} \cdot \hat{v}_{m;j},$$

   where $z_{1-\alpha/2}$ is the $(1-\alpha/2)$ cutoff of the standard normal distribution.
7. Obtain subsampling instants $t_k = x_1 + \cdots + x_k$ where $x_k - 1 \sim \text{Poisson}(\nu_k)$, $k = 1, \ldots, N$ and $N = \sup\{n : t_n \leq m\}$.
8. If $Q_m^{(1)}(\hat{\boldsymbol{\Psi}}^{(r)} \mid \boldsymbol{\Psi}^{(r-1)})$ lies in the confidence region from step 6, then

   (a) Set $m_o = m$.
   (b) Set $m = m_o + \lfloor m_o/c \rfloor$ for some $c > 0$.
   (c) Obtain $\mathbf{u}_{m_o+1}, \ldots, \mathbf{u}_m \sim g(\mathbf{u} \mid \mathbf{y}, \boldsymbol{\Psi}^{(0)})$ via an MCMC algorithm.
9. Compute for each $j = 1, \ldots, s$

$$Q_{m;j}^{(1)}\left(\hat{\boldsymbol{\Psi}}^{(r+1)} \mid \hat{\boldsymbol{\Psi}}^{(r)}\right) = \sum_{k=1}^{N} w_{t_k} \frac{\partial}{\partial \psi(j)} \ln f(\mathbf{u}_{t_k}, \mathbf{y} \mid \boldsymbol{\Psi}) \Big/ \sum_{t=1}^{m} w_{t_k} \Big|_{\boldsymbol{\Psi}=\hat{\boldsymbol{\Psi}}^{(r+1)}},$$

10. Repeat Steps 3 through 9 until convergence.

As mentioned in Section 2.1, the importance weights are sensitive to the "distance" between the target and candidate distribution, namely, at iteration r, $g(\mathbf{u} \mid \mathbf{y}, \hat{\Psi}^{(r)})$ and $g(\mathbf{u} \mid \mathbf{y}, \Psi^{(0)})$. We can include the following burn-in in Step 1 to alleviate this problem.

1. Initialize m, $\Psi^{(0)}$, and run burn-in.

   (a) Set importance weights $w_t = 1$ for all $t = 1, \ldots, m$.
   At iteration b
   (b) Generate $\mathbf{u}_1, \ldots \mathbf{u}_m \sim g(\mathbf{u} \mid \mathbf{y}, \Psi^{(b)})$ via an MCMC algorithm.
   (c) Run E and M steps above with $r = b$.
   (d) Repeat Steps 1b and 1c for B burn-in iterations.
   (e) Reinitialize $\Psi^{(0)} = \Psi^{(B)}$.

In the applications of Section 3, we run the burn-in for one minute. Note that m is not changed during the burn-in process. This burn-in is essentially the McCulloch (1997) MCEM where random variates are generated each iteration.

The choice of a particular MCEM algorithm implementation from those discussed here is problem specific. In this article, we present three possible MCEM implementations in terms of importance sampling: (1) importance reweighting in the E-step; (2) burn-in period without reweighting followed by importance reweighting; and (3) no importance reweighting. Implementation (1) provides the most computationally cost-efficient approach since we need perform only a single MCMC run without specifying a burn-in period. If the importance weights are highly variable, however, implementations (2) or (3) are appropriate depending on whether the weights stabilize quickly or not. In particular, if the M-step is expensive involving say hundreds of parameters in a nonlinear complete data density, then we may prefer implementation (3) since the weights are likely to be quite variable. Furthermore, in determining burn-in time, we are left to diagnostic measures for determining how close $\Psi^{(0)}$ is to the local maximum. If expensive diagnostic measures, such as computation of gradients of the likelihood, are required for checking for convergence of the EM estimates, implementation (3) may again be preferred. These expensive measures will typically become an issue when the EM algorithm converges very slowly. In our experience, as mentioned above, the EM estimates converge to within a reasonable neighborhood of the local maximum. We thus recommend the burn-in implementation (2) requiring only one MCMC run while ensuring stable importance weights.

3. APPLICATIONS

We apply the algorithm in Section 2.3 to fit a simple logit-normal model from McCulloch (1997) and the salamander data of McCullagh and Nelder (1989).

3.1 LOGIT-NORMAL MODEL

Suppose for $i = 1, 2, \ldots, q$ and $j = 1, 2, \ldots, n$

$$Y_{ij} \mid \mathbf{u} \sim \text{Bernoulli}(\pi_{ij}) \tag{3.1}$$

Table 1. Simulated data from Booth and Hobert (1999) for the logit-normal model.

i/j	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
1	1	0	0	0	0	1	1	0	1	1	1	1	1	1	1
2	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1
6	0	0	0	1	0	1	1	1	0	1	1	1	1	1	1
7	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	1	0	0	1	1	0	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

conditionally independent. Furthermore,

$$u_i \sim N(0,\ \sigma^2) \tag{3.2}$$

independent and identically distributed over i. The logit link function relates $\mathbf{Y}$ to $\mathbf{u}$ in that

$$\ln\left(\frac{\pi_{ij}}{1-\pi_{ij}}\right) = \beta x_{ij} + u_i. \tag{3.3}$$

Thus, the latent variables here are the random effects $\mathbf{u}$. McCulloch (1997) simulated data from this single random effects model with $q = 10$, $n = 15$, $\beta = 5$, $\sigma^2 = 0.5$, and $x_{ij} = i/15$. We use the data generated by Booth and Hobert (1999) and presented in Table 1. The exact MLE computed from numerical integration for this data is $\hat{\beta} = 6.132$ and $\hat{\sigma}^2 = 1.766$.

Figures 1 and 2 display output from our algorithm with and without importance sampling. The algorithms are run on a 533 MHz DEC alpha with 128 MB RAM under initial values $\Psi^{(0)} = (\beta^{(0)}, \sigma^{(0)}) = (2, 1)$ and $m = 100$. Each iteration of the sample $\mathbf{u}_1^{(r)}, \ldots \mathbf{u}_m^{(r)}$ is generated from a Metropolis–Hastings algorithm with $N(0,\ \sigma^2)$ candidate distribution. McCulloch (1997) presented the relevant likelihoods and acceptance probabilities.

Both MCEM approaches converge to a reasonable neighborhood around the true MLEs, but continue to show variation. As stated by McCulloch (1997), the number of replications required to obtain the MLE within three or four decimal places of accuracy would be very large. As expected, the importance sampling algorithm is faster performing 108 iterations including 16 burn-in in 60 minutes as opposed to 58 iterations run by MCEM without importance sampling. Generation of an MCMC sample each iteration of the algorithm is quite expensive.

The Markov chain induced by the Metropolis–Hastings algorithm with normal candidate distribution is strongly mixing with geometrically decaying mixing rates (see the Appendix). We may thus monitor the Monte Carlo sample size through the central limit theorem of Theorem 1. The sample size m increases from 100 to 4197. Figure 3 displays the iterations at which the sample size is increased and by how much for the MCEM algorithm with and without importance sampling, but each using the MC error estimation routine for changing the Monte Carlos sample size m. Also included in Figure 3 is McCulloch's (1997) predetermined choice for increasing m. As mentioned previously, the algorithms are run for 60 minutes, thus the difference in the number of iterations performed for each of the

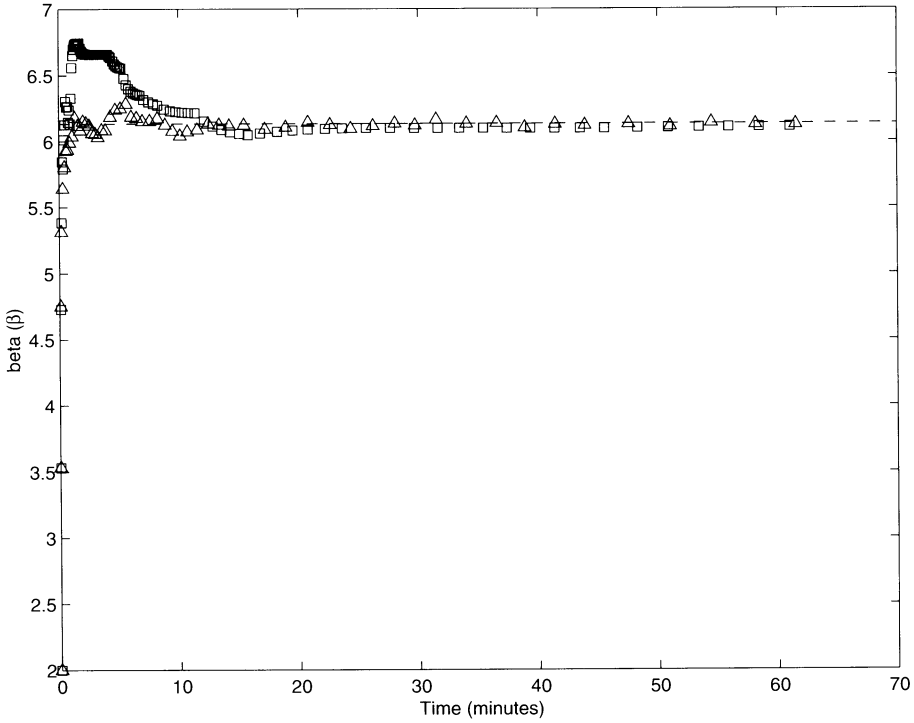

Figure 1. MLE of β at each iteration of MCEM algorithms with importance sampling (□) and without importance sampling (△). The iteration is measured in minutes. Dotted line specifies the true MLE $\hat{\beta} = 6.132$.

three routines. For comparison purposes, we excluded the burn-in from the iteration count as m is not increased during the burn-in process. Note that the predetermined choices of m proposed by McCulloch (1997) for running MCEM without importance sampling follows the sample size for MCEM under importance for the first 40 iterations. Following iteration 40, the predetermined m's take a large jump to 5,000, overshooting the sample size for MCEM without importance sampling. Thus, at early stages of the algorithm the McCulloch (1997) MCEM does not increase the MC sample size quickly enough. At later stages, the McCulloch (1997) MCEM runs at an unnecessarily high computational cost by generating more MCMC variates than are needed.

Our stopping rule at this point is based purely on time; namely, stop after 60 minutes. We can apply the stopping rules suggested by Booth and Hobert (1999) used to diagnose convergence. A temporal stopping rule is used to allow for easy cost comparisons between the various MCEM algorithms.

3.2 SALAMANDER DATA

As a final illustration of the MCEM algorithm developed in this article, we consider the salamander data of McCullagh and Nelder (1989, sec. 14.5). The data consists of mating success between two species of salamanders—rough-butt (R) and whiteside (W). Twenty males and twenty females, ten of each of the two species were mated resulting in four

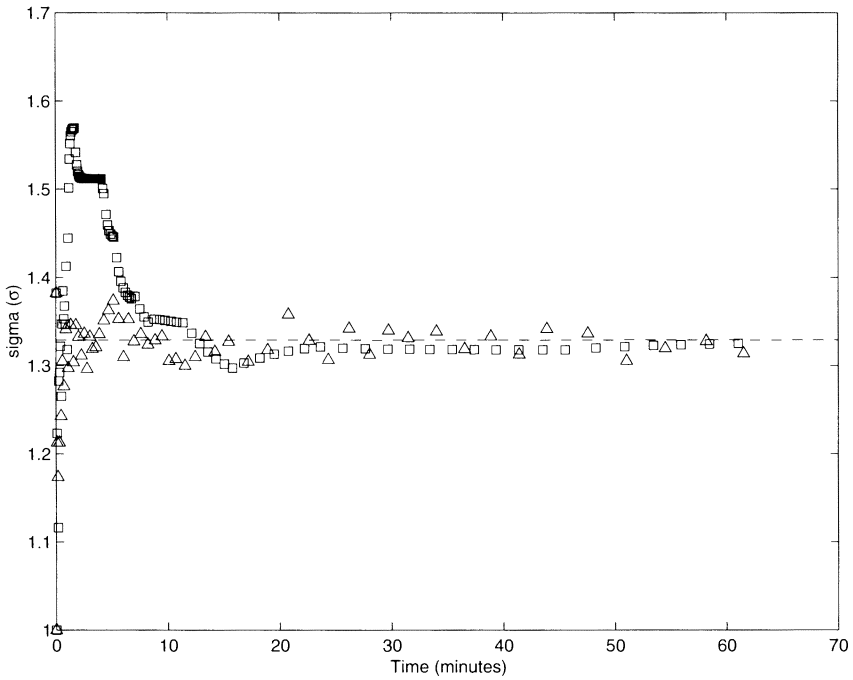

Figure 2. MLE of σ at each iteration of MCEM algorithms with importance sampling (□) and without importance sampling (△). The iteration is measured in minutes. Dotted line specifies the true MLE $\hat{\sigma} = 1.329$.

types of crosses. The design we follow is from Table 14.3 of McCullagh and Nelder (1989) consisting of $n = 120$ matings from the first of three experiments. The observed outcome is binary denoted by W_i taking a value one if the ith mating is successful and zero otherwise.

We apply the MCEM algorithm to perform a hierarchical probit regression. The algorithm uses the Gibbs sampler to estimate the expectation in the E-step. The model fitting is thus analogous to the work of McCulloch (1994), Chan and Kuk (1997), and van Dyk (in press).

The probit-normal model is a threshold model, assuming the underlying response is an unobserved continuous random n-vector $\mathbf{Y}$ which governs mating success. The observed W_i states whether Y_i exceeds the threshold of zero or not. The effect of each salamander on the outcome is assumed random while the effect of the species cross is assumed fix. In this way, the mixed model is written

$$\begin{aligned} \mathbf{Y} &= \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}_f\mathbf{U}_f + \mathbf{Z}_m\mathbf{U}_m + \boldsymbol{\epsilon} \\ \mathbf{U}_f &\sim N_{20}(\mathbf{0}, \sigma_f^2\mathbf{I}) \\ \mathbf{U}_m &\sim N_{20}(\mathbf{0}, \sigma_m^2\mathbf{I}) \\ \boldsymbol{\epsilon} &\sim N_n(\mathbf{0}, \mathbf{I}), \end{aligned} \tag{3.4}$$

where $\mathbf{U}_f$ and $\mathbf{U}_m$ are normal random q-vectors denoting the effect of the female and male salamanders, respectively; $\boldsymbol{\beta} = (\beta_{\mathrm{RR}}, \beta_{\mathrm{RW}}, \beta_{\mathrm{WR}}, \beta_{\mathrm{WW}})^T$ denotes the fixed effects of each of the four species crossings; $\mathbf{X}$ is the design matrix for the type of cross; $\mathbf{Z}_f$ and $\mathbf{Z}_m$

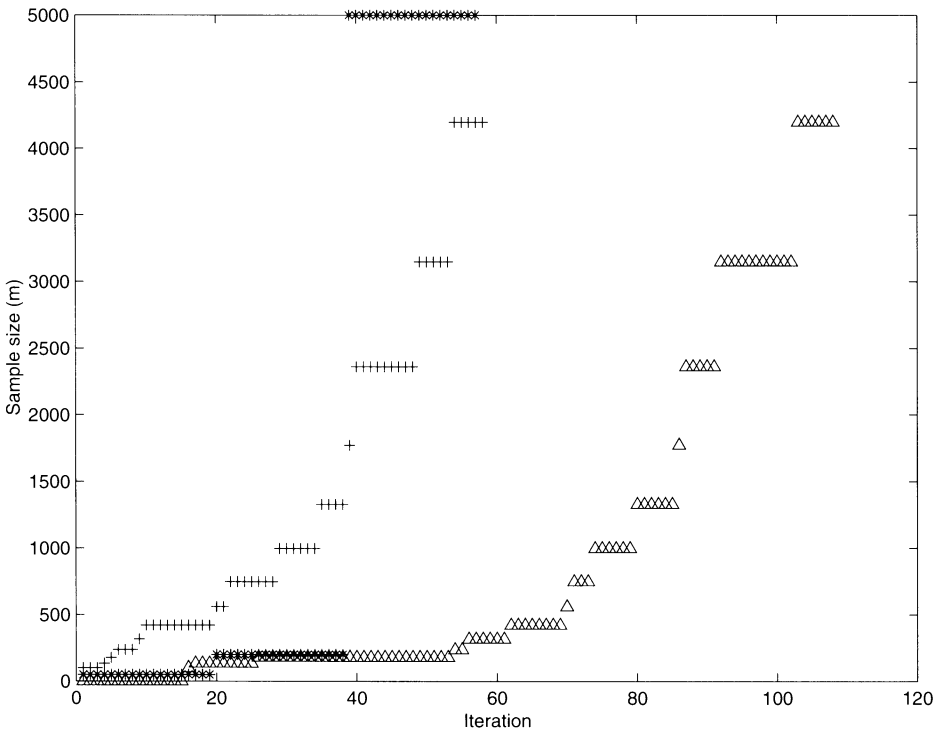

Figure 3. Increases in Monte Carlo sample size m for three MCEM algorithms: MCEM with importance sampling ($\triangle$), without importance sampling ($+$), and the predetermined values ($*$) from McCulloch (1997).

denotes the design matrix for the female and male salamanders, respectively; and $\mathbf{I}$ denotes an identity matrix of the appropriate size, $\mathbf{0}$ denotes a vector of zeros of the appropriate dimension, and T denotes vector transpose. The random effects are assumed independent of each other and mutually independent of the normal random errors ϵ. Let the parameter vector be denoted by $\Psi = (\beta^T, \sigma_f^2, \sigma_m^2)^T$.

Under model (3.4), the complete data is $(\mathbf{Y}, \mathbf{U}_f, \mathbf{U}_m)$, as the latent variable $\mathbf{Y}$ is unobserved in addition to the random effects. However, through results about the multivariate normal density, we can show, at iteration r of the EM algorithm, calculation of $Q(\Psi \mid \hat{\Psi}^{(r)})$ is dependent solely on $\mathbf{Y}$ (see McCulloch 1994 for details). Thus, the MC estimation in the E-step requires a sample from the latent variables $\mathbf{Y}$.

Recall that the observations are binary responses $\mathbf{W} = (W_1, \ldots, W_n)^T$. Thus the distribution from which we must sample is the truncated multivariate normal conditional distribution of $\mathbf{Y}$ given $\mathbf{W}$ truncated at zero. Note that the full conditional distributions of $Y_i \mid Y_1, \ldots, Y_{i-1}, Y_{i+1}, \ldots, Y_n$ for $i = 1, \ldots, n$ is univariate truncated normal distributions, truncated at zero. As the univariate truncated normal is easy to sample, a Gibbs sampler may be implemented to obtain a sample $\mathbf{Y}^{(1)}, \ldots, \mathbf{Y}^{(m)}$ from the truncated multivariate normal distribution of interest (Robert 1995b). McCulloch (1994) presented the Gibbs routine for sampling from the truncated multivariate normal distribution in detail. See Albert and Chib (1993) for a detailed exposition for the latent variable representation

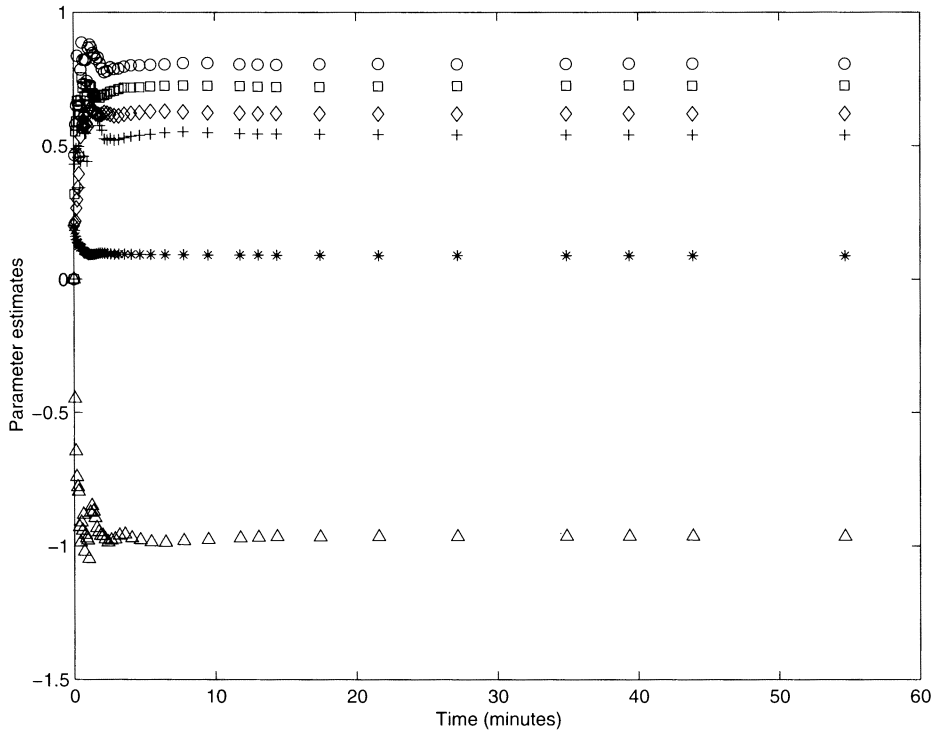

Figure 4. MLE of β_{RR} (O), β_{RW} (+), β_{WR} (△), β_{WW} (□), σ_f^2 (◇), and σ_m^2 (*) at each iteration of MCEM algorithm with importance sampling.

of this binary probit model.

At iteration i of the Gibbs sampler, Y_i is generated from the truncated univariate normal distribution given the most recent value of the other $n-1$ components. Thus, at each iteration of the Gibbs chain, we must compute n conditional expectations as well as generate a sample from the truncated normal distribution. This task is costly, particularly for later iterations of the MCEM algorithm in which m may be of the order of tens of thousands. Hence we implement the MCEM algorithm with importance sampling as discussed in Section 2.3. The algorithm is initiated at $\mathbf{\Psi} = (0, 0, 0, 0, .2, .2)$ and $m = 10$.

Figure 4 displays convergence of the ML estimates for the parameter $\mathbf{\Psi}$. The algorithm was run for 60 minutes during which it performed 46 EM iterations including 16 burn-in. The final estimates are $\hat{\beta}_{\mathrm{RR}} = .81$, $\hat{\beta}_{\mathrm{RW}} = .54$, $\hat{\beta}_{\mathrm{WR}} = -.96$, $\hat{\beta}_{\mathrm{WW}} = .73$, $\hat{\sigma}_f^2 = .62$, $\hat{\sigma}_m^2 = .088$, similar to the values found by McCulloch (1994).

Robert (1995b) stated the Gibbs sampler used to generate the truncated multivariate normal sample induces a geometrically ergodic Markov chain. Consequently, the chain is strongly mixing with geometrically decaying mixing rates (Chan and Geyer 1994) putting Theorem 1 into use. The MC samples size m increases from 10 to 12,857. The large increase in MC sample size suggests the recommendation of Chan and Kuk (1997) to generate a Gibbs sample of 1,000 iterates each EM iteration is not appropriate. This fixed sampling routine oversamples at early iterations of the MCEM algorithm and does not draw enough samples in later iterations to obtain precise EM estimates.

The amount by which the MC sample should be increased in our algorithm, of course, is flexible. Our algorithm tells the user when to increase the MC sample size, but the actual value taken by m each EM iteration is user determined. As van Dyk (in press) states, trial and error is necessary for choosing appropriate MC sample sizes. Nonetheless, a linear increase of m seems to be the preferred choice in the literature (see, e.g., McCulloch, 1994 and van Dyk in press). Though our choice of increasing m by 4/3 appears to allow a reasonable tradeoff of computational cost with EM precision, users may choose to increase or decrease this value as deemed appropriate for the application of interest.

APPENDIX

The Metropolis–Hastings algorithms used in Section 3.1 is an independence chain in that the candidate distribution does not depend on the most recent sample point (see Robert and Casella 1999, chap. 6, for more details). We will use the following theorem concerning convergence of independent Metropolis–Hastings algorithms, a combination of a result from Chan and Geyer (1994) and Mengersen and Tweedie (1996, theorem 2.1).

Theorem 2. *The Markov chain induced by the independent Metropolis–Hastings algorithm is a strongly mixing chain with geometrically decaying mixing coefficients if there exists some $M > 0$ such that for all $\mathbf{u} \in supp(\pi)$, $q(\mathbf{u}) > 0$ and*

$$\frac{\pi(\mathbf{u})}{q(\mathbf{u})} \leq M,$$

where π is the stationary distribution of the chain and q is the candidate distribution.

Mengersen and Tweedie (1996) actually showed under the conditions of Theorem 2, the chain is *uniformly ergodic*, a much stronger conclusion than α-mixing. Chan and Geyer (1994) show uniformly ergodic chains are strongly mixing with geometrically decaying mixing rate. We will apply this theorem to the logit-normal example.

Recall the candidate distribution $q(\mathbf{u})$ in the logit-normal example of Section 3.1 is multivariate normal $N(\mathbf{0}, \sigma^2\mathbf{I})$ where $\mathbf{I}$ is an $s \times s$ identity matrix with $s = 10$. Hence generations from the candidate distribution are independent of variates generated earlier by the Metropolis–Hastings sampler.

The invariant distribution $\pi(\mathbf{u})$ of the Markov chain induced by the Metropolis–Hastings algorithm is the conditional density

$$\begin{aligned} f(\mathbf{u} \mid \mathbf{Y}, \beta, \sigma^2) &= \frac{f(\mathbf{Y} \mid \mathbf{u}, \beta, \sigma^2) \cdot f(\mathbf{u} \mid \beta, \sigma^2)}{f(\mathbf{Y} \mid \beta, \sigma^2)} \\ &\propto \prod_{j=1}^{s} \prod_{i=1}^{n} \frac{\exp\{y_{ij}(\beta x_{ij} + u_j)}{1 + \exp\{y_{ij}(\beta x_{ij} + u_j)\}} \cdot \frac{\exp\{-u_j^2/(2\sigma^2)\}}{\sqrt{2\pi\sigma^2}}, \end{aligned}$$

where the variables are all as defined in Section 3.1.

Theorem 2 is applicable as

$$\begin{aligned} \frac{f(\mathbf{u} \mid \mathbf{Y}, \beta, \sigma^2)}{q(\mathbf{u})} &\propto \prod_{j=1}^{s} \prod_{i=1}^{n} \frac{\exp\{y_{ij}(\beta x_{ij} + u_j)}{1 + \exp\{y_{ij}(\beta x_{ij} + u_j)\}} \cdot \frac{\exp\{-u_j^2/(2\sigma^2)\}/\sqrt{2\pi\sigma^2}}{\exp\{\sum_{j=1}^{s} u_j^2/(2\sigma^2)\}} \\ &\leq \left(\frac{1}{2\pi\sigma^2}\right)^n, \end{aligned}$$

where the proportionality constant in the first equation is

$$\frac{(2\pi\sigma^2)^{-q/2}}{f(\mathbf{Y} \mid \beta, \sigma^2)}.$$

ACKNOWLEDGMENTS

This research was done while the first author was visiting the Department of Biometics and the Department of Statistical Science, Cornell University, supported by National Science Foundation Grant DMS-9625440.

[Received August 1999. Revised April 2000.]

REFERENCES

Albert, J. H., and Chib, S. (1993), "Bayesian Analysis of Binary and Polychotomous Response Data," *Journal of the American Statistical Association* 88, 669–679.

Booth, J. G., and Hobert, J. P. (1999), "Maximizing Generalized Linear Mixed Model Likelihoods with an Automated Monte Carlo EM Algorithm," *Journal of the Royal Statistical Society*, Ser. B, 61, 265–285

Casella, G., and Robert, C. P. (1998), "Post-Processing Accept–Reject Samples: Recycling and Rescaling," *Journal of Computational and Graphical Statistics*, 7, 139–157.

Chan, J. S. K., and Kuk, A. Y. C. (1997), "Maximum Likelihood Estimation for Probit-linear Mixed Models with Correlated Random Effects," *Biometrics* 53, 86–97.

Chan, K. S., and Geyer, C. (1994), Discussion of "Markov Chains for Exploring Posterior Distributions," by L. Tierney, *The Annals of Statistics*, 22, 1747–1758.

Chan, K. S., and Ledolter, J. (1995), "Monte Carlo EM Estimation for Time Series Models Involving Counts," *Journal of the American Statistical Association* 90, 242–252.

Geyer, C. J. (1991), "Reweighting Monte Carlo Mixtures," Technical Report 568, University of Minnesota, School of Statistics.

——— (1992), "Practical Markov Chain Monte Carlo," *Statistical Science*, 7, 473–483.

Kipnis, C., and Varadhan, S. R. S. (1986), "Central Limit Theorem for Additive Functionals of Reversible Markov Processes and Applications to Simple Exclusions," *Communications in Mathematical Physics* 104, 1–19.

Liu, J. S. (1996), "Metropolized Independent Sampling with Comparisons to Rejection Sampling and Importance Sampling," *Statistics and Computing* 6, 113–119.

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

McCulloch, C. E. (1994). Maximum Likelihood Variance Components Estimation for Binary Data. *Journal of the American Statistical Association* 89, 330-335.

——— (1997), "Maximum Likelihood Algorithms for Generalized Linear Mixed Models," *Journal of the American Statistical Association*, 92, 162–170.

Mengersen, K. L., and Tweedie, R. L. (1996), "Rates of Convergence of the Hastings and Metropolis Algorithms," *The Annals of Statistics*, 24, 101–121.

Mykland, P., Tierney, L., and Yu, B. (1995), "Regeneration in Markov Chain Samplers," *Journal of the American Statistical Association*, 90, 233–241.

Robert, C. P. (1994), Discussion of "Markov Chains for Exploring Posterior Distributions," by L. Tierney, *The Annals of Statistics*, 22, 1742–1747.

——— (1995a), "Convergence Control Methods for Markov Chain Monte Carlo Algorithms," *Statistical Science*, 10, 231–253.

——— (1995b), "Simulation of Truncated Normal Variables," *Statistics and Computing*, 5, 121–125.

Robert, C. P., and Casella, G. (1999), *Monte Carlo Statistical Methods*, New York: Springer-Verlag.

Robert, C. P., Rydén, T., and Titterington, D. M. (1999), "Convergence Controls for MCMC Algorithms, With Applications to Hidden Markov Chains," *Journal of Statistical Computation and Simulation*, 64, 327–355.

Tanner, M. A. (1993), *Tools for Statistical Inference* (2nd ed.), New York: Springer-Verlag.

Tierney, L. (1994), "Markov Chains for Exploring Posterior Distributions," *The Annals of Statistics*, 22, 1701–1762.

Wei, G. C. G., and Tanner, M. A. (1990), "A Monte Carlo Implementation of the EM Algorithm and the Poor Man's Data Augmentation Algorithms," *Journal of the American Statistical Association*, 85, 699–704.

Wu, C. F. J. (1983), "On the Convergence Properties of the EM Algorithm," *The Annals of Statistics*, 11, 95–103.

van Dyk, D. (in press), "Nesting EM Algorithms for Computational Efficiency," *Statistica Sinica*.