A Monte-Carlo algorithm for maximum likelihood estimation of variance components

S Xu, Wr Atchley

Genet Sel Evol (1996) 28, 329-343

Original article

A Monte-Carlo algorithm for maximum likelihood estimation of variance components

S Xu [1], WR Atchley [2]

[1] *Department of Botany and Plant Sciences, University of California, Riverside, CA 92521;*
[2] *Department of Genetics, North Carolina State University, Raleigh, NC 27695, USA*

(Received 30 January 1995; accepted 24 May 1996)

Summary – A new algorithm for finding maximum likelihood (ML) solutions to variance components is introduced. This algorithm first treats random effects as fixed, then expresses the pseudo-fixed effects as linear transformations of a set of standard normal deviates which eventually are integrated out numerically through Monte-Carlo simulation. An iterative algorithm is employed to estimate the standard deviation (rather than the variance) of the random effects. This method is conceptually simple and easy to program because repeated updating and inverting the variance–covariance matrix of data is not required. It is potentially useful for handling large data sets and data that are not normally distributed.

maximum likelihood / restricted maximum likelihood / variance component / Monte-Carlo / mixed model

Résumé – Un algorithme de Monte-Carlo pour estimer des composantes de variance par le maximum de vraisemblance. *Un nouvel algorithme pour résoudre le maximum de vraisemblance de composantes de variance est présenté. Cet algorithme traite d'abord les effets aléatoires comme des effets fixes, puis exprime ces pseudo-effets fixes sous la forme de transformations linéaires d'un ensemble de variables normales centrées réduites. Celles-ci sont ensuite éliminées par intégration à l'aide d'un processus numérique de Monte-Carlo. Un algorithme itératif est employé pour estimer l'écart type (et non la variance) des effets aléatoires. Cette méthode est simple conceptuellement et facile à programmer parce que des inversions de la matrice de variance-covariance des données répétées à chaque itération ne sont plus nécessaires. La méthode peut être utile pour traiter de grands ensembles de données et des données qui ne sont pas distribuées normalement.*

maximum de vraisemblance / maximum de vraisemblance restreinte / composante de variance / Monte-Carlo / modèle mixte

INTRODUCTION

Estimates of variances and covariances have been used extensively in animal breeding (Henderson, 1986). Recently, much attention has been paid to natural populations (Guo and Thompson, 1991). A thorough knowledge of genetic variances and covariances is useful in determining genetic variability of a population, interpreting the genetic mechanism of quantitative traits and estimating heritabilities and genetic correlation of quantitative traits. Those genetic parameters are necessary in planning breeding programs, constructing selection indices, estimating breeding values of candidate breeders and predicting selection responses (Henderson, 1986).

Various methods for estimation of variance components have been developed. A general review can be found in Henderson (1984) and Searle (1989). Among these, the maximum likelihood (ML) of Hartley and Rao (1967) and restricted maximum likelihood (REML) of Patterson and Thompson (1971) are the most popular methods. With the ML-related methods, large computer resources and CPU times are required due to repeatedly inverting the variance–covariance matrix of the data. The derivative-free algorithm for REML (DF-REML) has been suggested by Graser et al (1987) where matrix inversion is not required, rather, Gaussian elimination is used (Smith and Graser, 1986). With the advent of efficient computer programs for ML and REML estimation of variance components such as the DF REML program of Meyer (1988), large data set ($N > 100\,000$) can be handled by utilizing the sparse matrix technique (see Misztal, 1994; Kriese et al, 1994 for up-to-date REML programs).

Recently, Guo and Thompson (1991) developed a Monte-Carlo expectation–maximization (EM) method for variance component estimation which uses jointly the EM algorithm (Dempster et al, 1977) and the Gibbs sampler (Geman and Geman, 1984). Their method has avoided repeated inversion of the variance–covariance matrix.

An alternative algorithm for solving ML and REML solutions, similar to Guo and Thompson (1991), is reported in this paper. The new method first treats random effects as fixed and then integrates out these pseudo-fixed effects via Monte-Carlo simulation. In the case where data are normally distributed, there is an explicit form of the multiple integral, but the explicit form involves inverse of the variance–covariance matrix. Instead of using the explicit form of the integral, the integration is carried out via Monte-Carlo simulation. With this algorithm, the standard deviation, rather than the variance, of the random effects is estimated using an iterative algorithm similar to the EM algorithm (Dempster et al, 1977). This method does not require updating the variance–covariance matrix, thus may need less computer memory than other methods. As a result, it may potentially handle large dimensional data sets. For data that are not normally distributed, an explicit form of the multiple integral is not available, thus the Monte-Carlo method may be the only appropriate way to solve the problem. In this paper application of the Monte-Carlo method to normal data for ML (or REML) estimation is reported. The result serves as a necessary step approaching the application the the new method to non-normal data.

THEORY AND METHODS

The mixed model

We will use the simplest mixed model with one class of random effects to demonstrate the new algorithm. The model is shown below:

$$\mathbf{y} = \mathbf{Xb} + \mathbf{Zu} + \mathbf{e} \qquad [1]$$

where

$\mathbf{y}$ $n \times 1$ vector of observations or data,
$\mathbf{b}$ $p \times 1$ vector of fixed effects,
$\mathbf{u}$ $q \times 1$ vector of random effects with $\mathrm{N}(\mathbf{0}, \mathbf{I}\sigma_u^2)$,
$\mathbf{e}$ $n \times 1$ vector of residuals with $\mathrm{N}(\mathbf{0}, \mathbf{I}\sigma_e^2)$,
$\mathbf{X}$ $n \times p$ known incidence matrix for the fixed effects with a rank r,
$\mathbf{Z}$ $n \times q$ known incidence matrix for the random effects, usually with full column rank.

It is assumed that $\mathrm{E}(\mathbf{y}|\boldsymbol{\theta}) = \mathbf{Xb}$ and $\mathrm{Var}(\mathbf{y}|\boldsymbol{\theta}) = \mathbf{V} = \mathbf{ZZ}^{\mathrm{T}}\sigma_u^2 + \mathbf{I}\sigma_e^2$, where $\boldsymbol{\theta} = [\mathbf{b}\sigma_u^2\sigma_e^2]^{\mathrm{T}}$ denotes the unknown parameters and the superscript T stands for matrix transposition. In the genral framework of animal breeding, $\mathbf{u}$ may represent sire effects. If every progeny within each sire has a different dam from each other, the variance among sires, σ_u^2, will account for a quarter of the additive genetic variance and σ_e^2 will contain three-quarters of the additive genetic variance plus the variance solely due to environmental effects (deviation of phenotype from genotype).

The likelihood function of the mixed model is proportional to

$$f(\mathbf{y}|\boldsymbol{\theta}) = |\mathbf{V}|^{-1/2}\,\mathrm{Exp}\left\{-\frac{1}{2}(\mathbf{y} - \mathbf{Xb})^{\mathrm{T}}\mathbf{V}^{-1}(\mathbf{y} - \mathbf{Xb})\right\} \qquad [2]$$

Note that this likelihood function involves inverting the variance–covariance matrix of the data ($\mathbf{V}^{-1}$).

Conditional on the random effects ($\mathbf{u}$), equation [1] is a fixed-effect model so that $\mathrm{E}(\mathbf{y}|\boldsymbol{\theta}\mathbf{u}) = \mathbf{Xb} + \mathbf{Zu}$ and $\mathrm{Var}(\mathbf{y}|\boldsymbol{\theta}\mathbf{u}) = \mathbf{I}\sigma_e^2$. Furthermore, $\mathbf{u}$ can be obtained by linear transformation of a set of standard normal deviates, ie, $\mathbf{u} = \mathbf{s}\sigma_u$, where $\mathbf{s} \sim N_{q\times 1}(\mathbf{0}, \mathbf{I})$. Thus, conditional on $\mathbf{s}$, the fixed model is reformulated as

$$\mathbf{y} = \mathbf{Xb} + \mathbf{Zs}\sigma_u + \mathbf{e} \qquad [3]$$

With such a fixed model, the conditional likelihood function is proportional to

$$f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}) = \frac{1}{(\sigma_e^2)^{n/2}}\mathrm{Exp}\left\{-\frac{1}{2\sigma_e^2}(\mathbf{y} - \mathbf{Xb} - \mathbf{Zs}\sigma_u)^{\mathrm{T}}(\mathbf{y} - \mathbf{Xb} - \mathbf{Zs}\sigma_u)\right\} \qquad [4]$$

Clearly, matrix inversion is not involved here. However, $\mathbf{s}$ is a vector of random variables which must be integrated out to obtain unconditional estimates of $\boldsymbol{\theta} = [\mathbf{b}, \sigma_u \text{ and } \sigma_e^2]$. Note that σ_u^2 in $\boldsymbol{\theta}$ is now replaced by σ_u.

The marginal likelihood function

This likelihood function has the form

$$f(\mathbf{y}|\boldsymbol{\theta}) = \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}) \frac{1}{(2\pi)^{q/2}} \operatorname{Exp}\left\{-\frac{1}{2}\mathbf{s}^{\mathrm{T}}\mathbf{s}\right\} d\mathbf{s} \qquad [5]$$

whose explicit form [5] is equivalent to [2]. The reason to reformulate [2] as [5] is that equation [5] can be approximated by Monte-Carlo simulation, as suggested by Fahrmeir and Tutz (1994). Because the distribution of $\mathbf{s}$ is completely known (standard normal), we can generate M sets of random normal deviates and denote the ith set by $\mathbf{s}_i$. Then the marginal likelihood function can be approximated by

$$g(\mathbf{y}|\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i) \qquad [6]$$

where $f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)$ is given in [4] with $\mathbf{s}$ substituted by $\mathbf{s}_i$, and it is an exponential function not the logarithm. Hereafter, M is called the length of the Monte-Carlo simulation. We can see that as $M \to \infty, g(\mathbf{y}|\boldsymbol{\theta}) \to f(\mathbf{y}|\boldsymbol{\theta})$. Therefore, maximum likelihood estimators can be obtained by maximizing $g(\mathbf{y}|\boldsymbol{\theta})$ instead of $f(\mathbf{y}|\boldsymbol{\theta})$ provided that M is large. More importantly, when $g(\mathbf{y}|\boldsymbol{\theta})$ is used, the maximum likelihood solution of the parameters can be easily solved via an iteratively reweighted least squares scheme.

The iterative algorithm

When $g(\mathbf{y}|\boldsymbol{\theta})$ is used, the vector of parameters becomes $\boldsymbol{\theta} = [\mathbf{b}^{\mathrm{T}}\sigma_{\mathrm{u}}\sigma_{\mathrm{e}}^2]^{\mathrm{T}}$, namely σ_{u} is estimated. As usual, the maximum likelihood solution of $\boldsymbol{\theta}$ is found by maximizing $L = \log[g(\mathbf{y}|\boldsymbol{\theta})]$ instead of $g(\mathbf{y}|\boldsymbol{\theta})$.

Let us first define the following partial derivatives:

$$\frac{\partial f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\partial \mathbf{b}} = -\frac{1}{2\sigma_{\mathrm{e}}^2} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)(\mathbf{X}^{\mathrm{T}}\mathbf{y} - \mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{b} - \mathbf{X}^{\mathrm{T}}\mathbf{Z}\mathbf{s}_i\sigma_{\mathrm{u}})$$

$$\frac{\partial f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\partial \sigma_{\mathrm{u}}} = -\frac{1}{2\sigma_{\mathrm{e}}^2} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)(\mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{y} - \mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{X}\mathbf{b} - \mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{Z}\mathbf{s}_i\sigma_{\mathrm{u}})$$

$$\frac{\partial f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\partial \sigma_{\mathrm{e}}^2} = -\frac{1}{2\sigma_{\mathrm{e}}^2} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)\left[n - \frac{1}{\sigma_{\mathrm{e}}^2}(\mathbf{y} - \mathbf{X}\mathbf{b} - \mathbf{Z}\mathbf{s}_i\sigma_{\mathrm{u}})^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\mathbf{b} - \mathbf{Z}\mathbf{s}_i\sigma_{\mathrm{u}})\right]$$

We then have

$$\frac{\partial L}{\partial \mathbf{b}} = -\frac{1}{\sum_{j=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)} \sum_{i=1}^{M} \frac{\partial f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\partial \mathbf{b}}$$

$$= -\frac{1}{2\sigma_e^2} \sum_{i=1}^{M} \frac{f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\sum_{j=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)} \left[\mathbf{X}^{\mathrm{T}}\mathbf{y} - \mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{b} - \mathbf{X}^{\mathrm{T}}\mathbf{Z}\mathbf{s}_i\sigma_u\right]$$

$$\frac{\partial L}{\partial \sigma_u} = -\frac{1}{\sum_{j=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)} \sum_{i=1}^{M} \frac{\partial f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\partial \sigma_u}$$

$$= -\frac{1}{2\sigma_e^2} \sum_{i=1}^{M} \frac{f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\sum_{j=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)} \left[\mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{y} - \mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{X}\mathbf{b} - \mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{Z}\mathbf{s}_i\sigma_u\right]$$

and

$$\frac{\partial L}{\partial \sigma_e^2} = -\frac{1}{\sum_{j=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)} \sum_{i=1}^{M} \frac{\partial f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\partial \sigma_e^2}$$

$$= -\frac{1}{2\sigma_e^2} \sum_{i=1}^{M} \frac{f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\sum_{j=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)} \left[n - \frac{1}{\sigma_e^2}(\mathbf{y} - \mathbf{X}\mathbf{b} - \mathbf{Z}\mathbf{s}\sigma_u)^{\mathrm{T}}(\mathbf{y} - \mathbf{X}\mathbf{b} - \mathbf{Z}\mathbf{s}\sigma_u)\right]$$

The maximum likelihood estimators (MLEs) are obtained by setting

$$\frac{\partial L}{\partial \mathbf{b}} = \mathbf{0} \quad \text{and} \quad \frac{\partial L}{\partial \sigma_u} = \frac{\partial L}{\partial \sigma_e^2} = 0$$

Dropping the constant $-1/2\sigma_e^2$ in the above equations and defining

$$p_i = \frac{f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_i)}{\sum_{j=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)}$$

we have the following ML equations:

$$\begin{bmatrix} \widehat{\sigma}_u \\ \widehat{\mathbf{b}} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{M} p_i \mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{Z}\mathbf{s}_i & \sum_{i=1}^{M} p_i \mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{X} \\ \sum_{i=1}^{M} p_i \mathbf{X}^{\mathrm{T}}\mathbf{Z}\mathbf{s}_i & \sum_{i=1}^{M} p_i \mathbf{X}^{\mathrm{T}}\mathbf{X} \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{M} p_i \mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{y} \\ \sum_{i=1}^{M} p_i \mathbf{X}^{\mathrm{T}}\mathbf{y} \end{bmatrix} \qquad [7]$$

and

$$\widehat{\sigma}_e^2 = \left[n \sum_{i=1}^{M} p_i \right]^{-1} \sum_{i=1}^{M} p_i (\mathbf{y} - \mathbf{X}\widehat{\mathbf{b}} - \mathbf{Z}\mathbf{s}_i \widehat{\sigma}_u)^T (\mathbf{y} - \mathbf{X}\widehat{\mathbf{b}} - \mathbf{Z}\mathbf{s}_i \widehat{\sigma}_u) \quad [8]$$

Unfortunately, the weight p_i is a function of the unknown parameters, therefore, an iterative scheme must be employed. We can see that the iterative scheme is essentially an iteratively reweighted least squares approach. The iteration takes the following steps:

Step 1: set up initials for $\mathbf{b}$, σ_u and σ_e^2;
Step 2: evaluate p_i;
Step 3: solve for $\mathbf{b}$, σ_u and σ_e^2 using [7] and [8];
Step 4: update $\mathbf{b}$, σ_u and σ_e^2 which completes one cycle of iteration;
Step 5: repeat Steps 2–4 until convergence.

The MLE of σ_u^2 is simply the square of the MLE of σ_u due to the invariance property of the ML method (DeGroot, 1986). The iterative algorithm does not require inversion of matrix $\mathbf{V}$; rather, it only requires storing the following quantities:

$$\{\mathbf{s}_i^T \mathbf{Z}^T \mathbf{Z}\mathbf{s}_i\}_{M\times 1}, \{\mathbf{s}_i^T \mathbf{Z}^T \mathbf{y}\}_{M\times 1} \text{ and } \{\mathbf{s}_i^T \mathbf{Z}^T \mathbf{X}\}_{M\times k}$$

These quantities do not involve the unknown parameters, and thus do not need to be updated; rather, they can be calculated after the random normal deviates are generated and before the iteration is invoked. In addition, if the starting value of σ_u is positive, the solution to σ_u remains positive at each round of iteration. If σ_u starts at a negative value, it remains negative subsequently, but its square is still a valid ML estimate of σ_u^2.

Note that p_i is the posterior probability density whose denominator

$$\sum_{j=1}^{M} f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)$$

is constant across i, and thus can be dropped without altering the solutions. For computational conveneince, p_i is redefined as $p_i = f(\mathbf{y}|\boldsymbol{\theta}\mathbf{s}_j)$ hereafter. Furthermore, with a large data set, p_i will be very close to zero and may cause the method to fail due to numerical problems. This can be circumvented by multiplying p_i by the exponential of a very large positive number whose magnitude is comparable to

$$\frac{1}{2\sigma_e^2}(\mathbf{y} - \mathbf{X}\mathbf{b} - \mathbf{Z}\mathbf{s}\sigma_u)^T (\mathbf{y} - \mathbf{X}\mathbf{b} - \mathbf{Z}\mathbf{s}\sigma_u)$$

A candidate of such a number may be

$$c = \frac{1}{2\,\sigma_{e(0)}^2}(\mathbf{y} - \mathbf{X}\mathbf{b}_{(0)} - \mathbf{Z}\mathbf{s}_{(0)}\sigma_{u(0)})^T (\mathbf{y} - \mathbf{X}\mathbf{b}_{(0)} - \mathbf{Z}\mathbf{s}_{(0)}\sigma_{u(0)})$$

where $\sigma_{e(0)}^2$, $\mathbf{b}_{(0)}$ and $\sigma_{u(0)}$ are chosen such that they are close to the true parametric values, and $\mathbf{s}_{(0)}$ is an arbitrary Monte-Carlo realization of vector $\mathbf{s}$. The modified p_i has the following form:

$$p_i = \text{Exp}(c) f(\mathbf{y}|\boldsymbol{\theta}\,\mathbf{s})$$
$$= \frac{1}{(\sigma_e^2)^{n/2}} \text{Exp}\left\{-\frac{1}{2\sigma_e^2}(\mathbf{y} - \mathbf{Xb} - \mathbf{Zs}\sigma_u)^T(\mathbf{y} - \mathbf{Xb} - \mathbf{Zs}\sigma_u) + c\right\} \quad [9]$$

It should be warned that the value of c stays the same throughout the iterations and should not be updated.

The REML

The Monte-Carlo algorithm also works for the REML estimation of variance components. As in the ML analysis described earlier, we first treat the random effects as pseudo-fixed effects, then express the model by a linear transformation of the original data, ie,

$$\mathbf{K}^T\mathbf{y} = \mathbf{K}^T\mathbf{Zs}\sigma_u + \mathbf{K}^T\mathbf{e} \quad [10]$$

where $\mathbf{s}$ is a $q \times 1$ vector of standard random normal deviates generated via Monte-Carlo simulation, $\mathbf{K}$ is a known matrix with n rows and $n - r$ columns. Matrix $\mathbf{K}$ is chosen such that $\mathbf{K}^T\mathbf{X} = \mathbf{0}$ (Patterson and Thompson, 1971). One such a choice is given by Harville (1977) as any $n - r$ independent columns of matrix $\mathbf{I} - \mathbf{X}(\mathbf{X}^T\mathbf{X})\mathbf{X}^T$. After the linear transformation, the model does not depend on the fixed effects, $\mathbf{b}$. Thus, the vector of unknown parameters becomes $\boldsymbol{\theta} = [\sigma_u\sigma_e^2]^T$. Conditional on $\mathbf{s}$, the expectation and variance of $\mathbf{K}^T\mathbf{y}$ are $\text{E}(\mathbf{K}^T\mathbf{y}|\boldsymbol{\theta}\,\mathbf{s}) = \mathbf{K}^T\mathbf{Zs}\sigma_u$ and $\text{Var}(\mathbf{K}^T\mathbf{y}|\boldsymbol{\theta}\,\mathbf{s}) = \mathbf{K}^T\mathbf{K}\sigma_e^2$, respectively, Thus, the conditional likelihood function is proportional to

$$f(\mathbf{K}^T\mathbf{y}|\boldsymbol{\theta}\,\mathbf{s}) = \frac{1}{(\sigma_e^2)^{(n-r)/2}} \text{Exp}\left\{-\frac{1}{2\sigma_e^2}(\mathbf{y} - \mathbf{Zs}\sigma_u)^T\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}(\mathbf{y} - \mathbf{Zs}\sigma_u)\right\} \quad [11]$$

The marginal likelihood function is similarly approximated via Monte-Carlo simulation and has the following form

$$g(\mathbf{K}^T\mathbf{y}|\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M} f(\mathbf{K}^T\mathbf{y}|\boldsymbol{\theta}\,\mathbf{s}_i) \quad [12]$$

where $f(\mathbf{K}^T\mathbf{y}|\boldsymbol{\theta}\,\mathbf{s}_i)$ is given by [11] with $\mathbf{s}$ replaced by $\mathbf{s}_i$. Setting partial derivatives of $L = \log[g(\mathbf{K}^T\mathbf{y}|\boldsymbol{\theta})]$ with respect to $\boldsymbol{\theta}$ equal to zero, we have

$$\widehat{\sigma}_u = \frac{\sum_{i=1}^{M} p_i \mathbf{s}_i^T\mathbf{Z}^T\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T\mathbf{y}}{\sum_{i=1}^{M} p_i \mathbf{s}_i^T\mathbf{Z}^T\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T\mathbf{Zs}_i} \quad [13]$$

and

$$\widehat{\sigma}_e^2 = \left[(n-r)\sum_{i=1}^{M} p_i\right]^{-1} \sum_{i=1}^{M} p_i(\mathbf{y} - \mathbf{Zs}_i\widehat{\sigma}_u)^T\mathbf{Z}^T\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T(\mathbf{y} - \mathbf{Zs}_i\widehat{\sigma}_u) \quad [14]$$

where

$$p_i = \frac{f(\mathbf{K}^{\mathrm{T}}\mathbf{y}|\boldsymbol{\theta}\,\mathbf{s}_i)}{\sum_{j=1}^{M} f(\mathbf{K}^{\mathrm{T}}\mathbf{y}|\boldsymbol{\theta}\,\mathbf{s}_j)}$$

is again the posterior probability density. Iteration is required because the weight p_i is a function of the unknown parameters. Note that the quantities to be stored are $\{\mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{K}(\mathbf{K}^{\mathrm{T}}\mathbf{K})^{-1}\mathbf{K}^{\mathrm{T}}\mathbf{y}\}_{M\times 1}$ and $\{\mathbf{s}_i^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{K}(\mathbf{K}^{\mathrm{T}}\mathbf{K})^{-1}\mathbf{K}^{\mathrm{T}}\mathbf{Z}\mathbf{s}_i\}_{M\times 1}$ which do not involve the unknown parameters, hence updating is not necessary.

The animal model

Under an individual animal model, the genetic value of each animal is included. It has the form

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{u} + \mathbf{e} \qquad [15]$$

where $\mathbf{u}$ is an $n \times 1$ vector of additive genetic values for all animals. There is no $\mathbf{Z}$ matrix here and $\mathbf{u}$ are correlated through the additive relationship matrix (matrix $\mathbf{A}$), namely, $\mathrm{Var}(\mathbf{u}) = \mathbf{A}\sigma_{\mathrm{u}}^2$ and σ_{u}^2 is the additive genetic variance. Let $\mathbf{Z} = \mathbf{A}^{1/2}$, the lower Choleskey decomposition of $\mathbf{A}$. Now the fixed model version of [15] becomes

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{Z}\mathbf{s}\sigma_{\mathrm{u}} + \mathbf{e}$$

This equation is exactly the same as [3] except that $\mathbf{Z}$ is an $n \times n$ lower triangular matrix and $\mathbf{s}$ is an $n \times 1$ vector of standard normal deviates generated via Monte-Carlo simulation. Therefore, the Monte-Carlo algorithm works equally well with an animal model. Note that $\mathrm{Var}(\mathbf{Z}\mathbf{s}\sigma_{\mathrm{u}}) = \mathbf{Z}\mathrm{Var}(\mathbf{s})\mathbf{Z}^{\mathrm{T}}\sigma_{\mathrm{u}}^2 = \mathbf{Z}\mathbf{Z}^{\mathrm{T}}\sigma_{\mathrm{u}}^2 = \mathbf{A}\,\sigma_{\mathrm{u}}^2$ because $\mathbf{Z}\mathbf{Z}^{\mathrm{T}} = \mathbf{A}$ and $\mathrm{Var}(\mathbf{s}) = \mathbf{I}$. The likelihood function for an animal model can be formulated either as [6] for ML estimation or as [12] for REML estimation.

With an animal model, the typical size of a data set could be very large, making factorization of $\mathbf{A}$ into $\mathbf{Z}$ very expensive. In addition, matrix $\mathbf{A}$ is dense and storing it may not be feasible. Here, we introduce a new algorithm that can avoid these difficulties.

Let $\mathbf{u} = \mathbf{a}\sigma_{\mathrm{u}}$, then $\mathbf{y}$ is equation [15] can be remodeled as

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{a}\sigma_{\mathrm{u}} + \mathbf{e} \qquad [16]$$

where $\mathbf{a}$ is an $n \times 1$ vector with $N(\mathbf{0}, \mathbf{A})$ distribution. The vector $\mathbf{a}$ can be simulated as follows: for founder j, a_j is sampled from an $N(0,1)$ distribution; for non-founder j, $a_j = (a_{\mathrm{m}} + a_{\mathrm{f}})/2 + \gamma_j$, where m and f are the parents of j and γ_j is sampled from an $N(0, \sqrt{1/2})$ distribution. We have now avoided the use of matrix factorization by directly generating vector $\mathbf{a}$ (instead of $\mathbf{s}$) and subsequently replacing $\mathbf{Z}\mathbf{s}$ by $\mathbf{a}$.

AN ILLUSTRATION

To demonstrate the Monte-Carlo algorithm, data originally used by Cunningham and Henderson (1968) were reanalyzed. There were 18 observations classified into two treatments (fixed) and three blocks (random). No correlation between blocks was assumed. The data was described by model [1]. The estimates of σ_e^2 and σ_u^2 were given by Patterson and Thompson (1971), which are ML: $\widehat{\sigma}_e^2 = 2.3518$ and $\widehat{\sigma}_u^2 = 2.5051$; REML: $\widehat{\sigma}_e^2 = 2.5185$ and $\widehat{\sigma}_u^2 = 3.9585$. The length of the Monte-Carlo algorithm varied from 10 to 31 623, which have a $\log_{10}$ scale ranging from 1.0 to 4.5 incremented by 0.5. The experiment was replicated 20 times.

Results of the ML estimates are plotted against the length in $\log_{10}$ scale as shown in figures 1 and 2. When $\log_{10}(M) = 3.5$, the MLE of σ_u^2 stabilized at the true maximum likelihood estimate (fig 1), while the estimate of σ_e^2 took only $\log_{10}(M) = 3.0$ to stabilize (fig 2). From these two figures we also observed that when $\log_{10}(M) < 3.0$ the Monte-Carlo estimation tends to be biased (downward). The standard deviation among the 20 replicates are plotted against the length as shown in figure 3. Similar results were also observed for the REML estimates (see figs 4–6). Downward baseness may be a general property of the Monte-Carlo method for small M because biases in both ML and REML estimates are downward. Further investigation is necessary to quantify the expected bias for small M.

In conclusion, the Monte-Carlo algorithm does converge to the true ML or REML estimates for both σ_u^2 and σ_e^2. Estimate of σ_e^2 converges more quickly than that of σ_u^2. For this particular data set, a length of $M = 1\,000$–$5\,000$ seemed to be sufficient. $M \approx 5\,000$ may also serve as a guideline for other data sets (see *Discussion*).

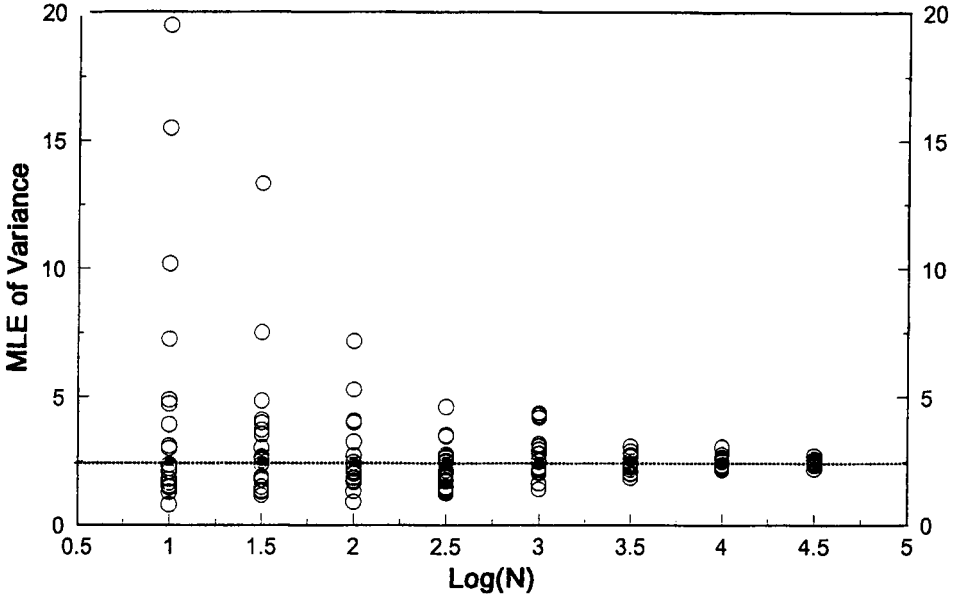

Fig 1. Maximum likelihood estimates of σ_u^2 plotted against the length of simulation in $\log_{10}$ scale. The dashed line represents the true MLE of $\sigma_u^2 = 2.5051$.

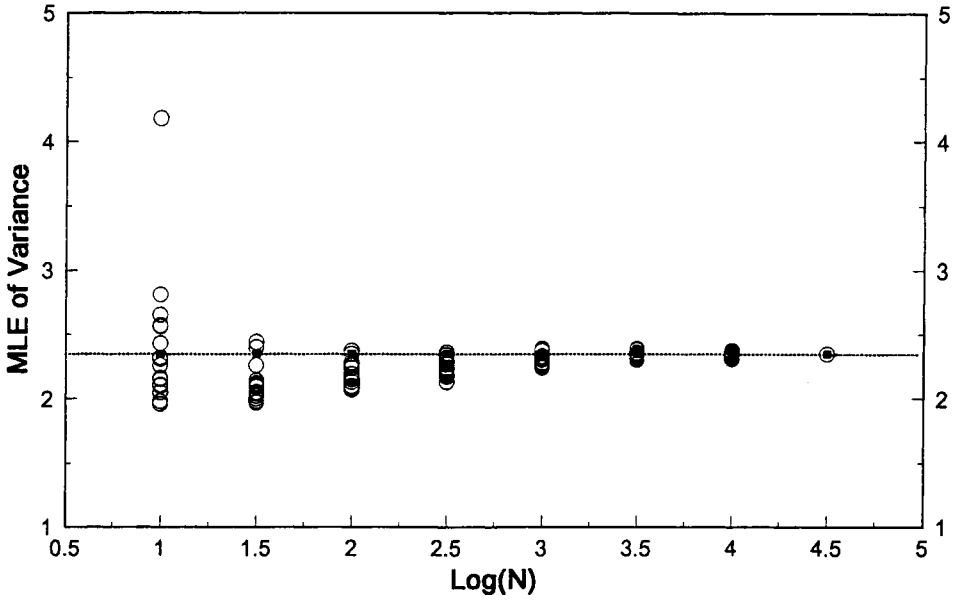

Fig 2. Maximum likelihood estimates of σ_e^2 plotted against the length of simulation in $\log_{10}$ scale. The dashed line represents the true MLE of $\sigma_e^2 = 2.3518$.

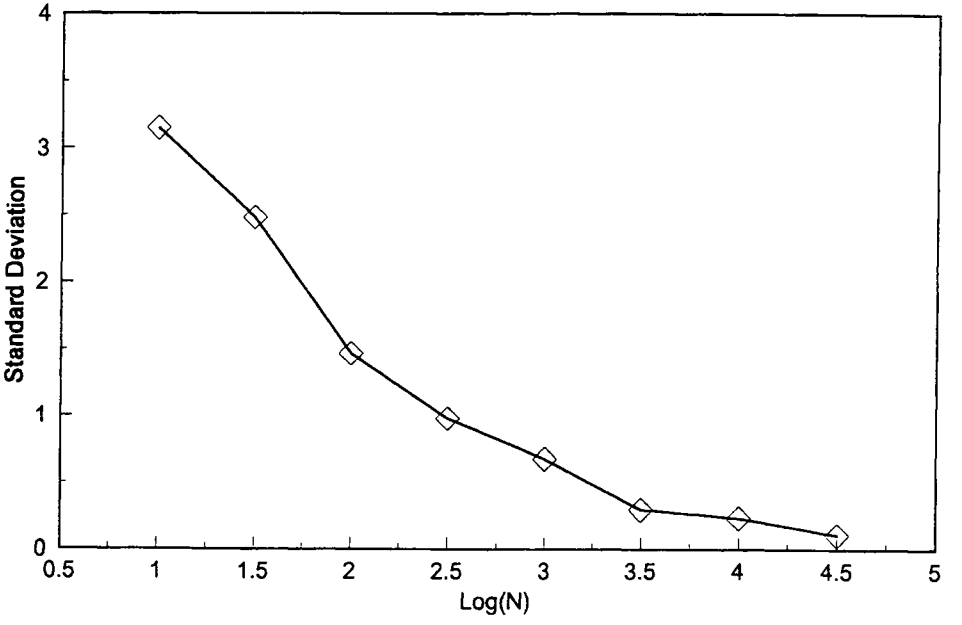

Fig 3. Standard deviation of the ML estimates of σ_u^2 among 20 replicates plotted against the length of simulation in $\log_{10}$ scale.

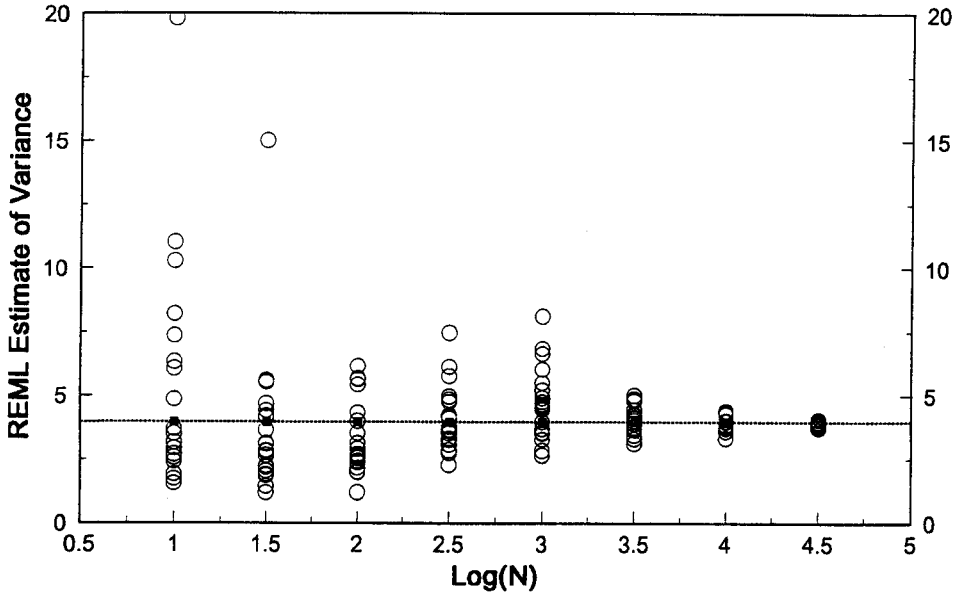

Fig 4. Restricted maximum likelihood estimates of σ_u^2 plotted against the length of simulation in $\log_{10}$ scale. The dashed line represents the true REML estimate $\sigma_u^2 = 3.9585$.

DISCUSSION

The major advantages of the Monte-Carlo algorithm presented in this paper rely on its conceptual simplicity and easy programming. There are two strategies to implement the Monte-Carlo algorithm. One is to generate the random numbers before invoking the iteration. In ML analysis, this requires storing the following quantities: $\{s_i^T\mathbf{Z}^T\mathbf{Z}\mathbf{s}_i\}_{M\times 1}$, $\{\mathbf{s}_i^T\mathbf{Z}^T\mathbf{y}\}_{M\times 1}$ and $\{\mathbf{s}_i^T\mathbf{Z}^T\mathbf{X}\}_{M\times k}$. The number of storage units required is $(k+2)\times M$. In REML analysis, the quantities to be stored are $\{\mathbf{s}_i^T\mathbf{Z}^T\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T\mathbf{y}\}_{M\times 1}$ and $\{s_i^T\mathbf{Z}^T\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T\mathbf{Z}\mathbf{s}_i\}_{M\times 1}$, where the number of storage units is $2M$. Although REML needs less computer memory than ML, additional CPU time is required for evaluating $\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T$. Once these quantities are generated, however, ML takes more CPU time than REML for iteration because ML needs repeated evaluation of equation [7] which is not trivial for a large number of fixed effects. The second strategy is to generate the random numbers as needed using a reseed with the same number strategy in the pseudo-random number generator. This would have a substantial CPU load but does not require storing those quantities needed by the first strategy. In either case, generating the quantities such as $\mathbf{s}_i^T\mathbf{Z}^T\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T\mathbf{Z}\mathbf{s}_i$ must resort to special techniques for large data sets (eg, Perez-Enciso et al, 1994).

When applied to large data sets, the Monte-Calro algorithm is powerful for ML analysis, but inefficient for REML analysis because it involves matrix $\mathbf{K}(\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T$ which looks formidable. The $\mathbf{K}$ matrix may be avoided by absortion of fixed effects using Guassian elimination, but it is still expensive for more than

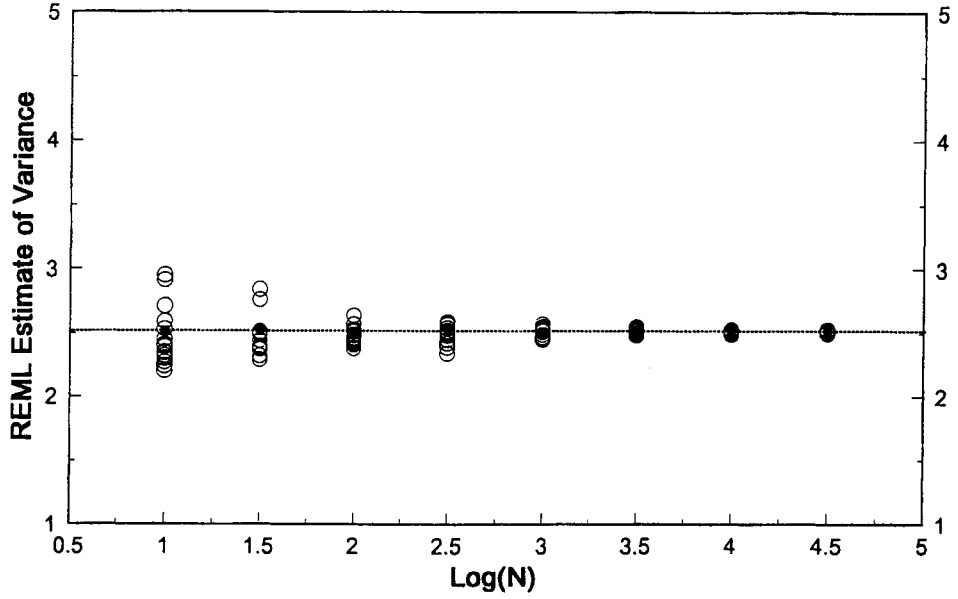

Fig 5. Restricted maximum likelihood estimates of σ_e^2 plotted against the length of simulation in $\log_{10}$ scale. The dashed line represents the true REML estimate of $\sigma_e^2 = 2.5185$.

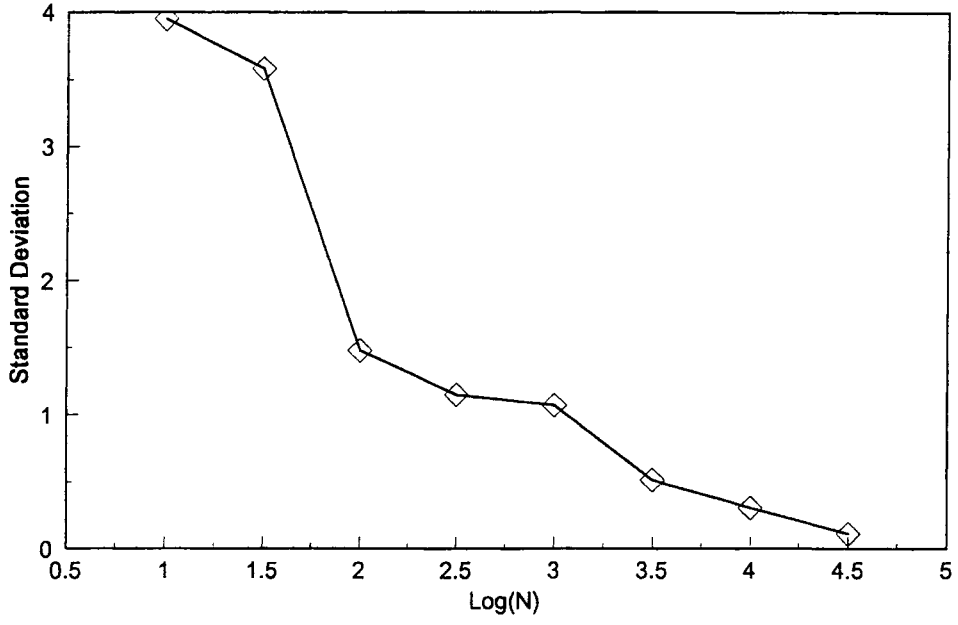

Fig 6. Standard deviation of the REML estimates of σ_u^2 among 20 replicates plotted against the length of simulation in $\log_{10}$ scale.

one fixed effect. Further investigation is necessary to implement this algorithm to REML analysis.

Using the first strategy of Monte-Carlo simulation, matrix factorization and inversion (the latter is required in REML) are done only once, which makes the first strategy more desirable than the second. For a reasonable number of levels of fixed effects (eg, $k = 10$), the number of storage units required is solely determined by M. For example, if $M = 5\,000$, the number of storage units will be $(10+2) \times 5\,000$ for ML and $2 \times 5\,000$ for REML. Such a number of storage units may be easily handled by standard PCs. The question becomes whether $M = 5\,000$ is sufficient or not. The example previously described shows that $M = 5\,000$ is sufficient. This number may also serve as a general guideline because it does not depend on the size of the data set. Fahrmeir and Tutz (1994) state that the computing load of Monte-Carlo multiple integration increases linearly with the dimension of the integral. Here, the computing load is not M but $M \times q$ (the total number of random normal deviates generated), where q is the dimension of $\mathbf{u}$. We can see that $M \times q$ is already proportional to q. Thus, we will not expect M to increase with q.

We have demonstrated the Monte-Carlo algorithm under the situation of one class of random effects. Extension to more classes of random effects is straightforward. Suppose that the mixed model has the form

$$\mathbf{y} = \mathbf{Xb} + \mathbf{Zu} + \mathbf{Wv} + \mathbf{e}$$

where $\mathbf{W}$ is a known matrix and $\mathbf{v}$ is another vector of random effects with $\mathbf{v} \sim N(\mathbf{0}, \mathbf{I}\sigma_{\mathrm{v}}^2)$. Here we assume $\mathrm{Cov}(\mathbf{u}, \mathbf{v}^{\mathrm{T}}) = \mathbf{0}$. Conditional on two independent sets of random normal deviates, $\mathbf{s}$ and $\mathbf{z}$, the fixed model equivalence of the above model is

$$\mathbf{y} = \mathbf{Xb} + \mathbf{Zs}\sigma_{\mathrm{u}} + \mathbf{Wz}\sigma_{\mathrm{v}} + \mathbf{e}$$

To obtain the marginal likelihood function, one simply generates M sets of $\mathbf{s}_i$ and $\mathbf{z}_i$ to approximate the multiple integral. A similar algorithm can be employed to simultaneously estimate $\mathbf{b}$, $\sigma_{\mathrm{u}}, \sigma_{\mathrm{v}}$ and σ_{e}^2.

The Monte-Carlo algorithm presented in this paper should be considered as an alternative method for estimation of variance components. Similar to the Monte-Carlo method of Guo and Thompson (1991), the method proposed here is ineffective for estimating the likelihood under mixed inheritance. In addition, large pedigrees (say 1 000 members) would be needed to demonstrate its effectiveness under polygenic inheritance. Furthermore, with a normal distribution, the new method perhaps does not offer too much advantage over the existing algorithms that use the sparse matrix technique. Therefore, by no means should the Monte-Carlo method replace those conventional algorithms. For data that are not normally distributed, however, the Monte-Carlo algorithms is perhaps the only convenient way to solve the problem. Many economically important traits of animal species are not normally distributed, such as binary disease resistant traits or ordinal categorical traits. When analyzing such traits, one usually assumes that there is a continuous latent variable (liability) controlling the phenotype. The link between the liability and the discrete phenotype may be described by Wright's (1934) physiological threshold model. To estimate the genetic variance of the liability and the thresholds, one can easily establish a fixed model equivalence of the likelihood function similar to

equation [4]. The random effects are then integrated out to obtain an integrated likelihood function. Certainly, there is no explicit form for the multiple integral and Monte-Carlo method may be the only appropriate way for such a large dimensional multiple integration. Results from this research (normal data) serve as a necessary first step approching to application of the new method to non-normal data.

Finally, it is necessary to clarify a potential confusion between this method and the Gibbs samplers of Guo and Thompson (1991) and Wang et al (1993). Each method involves maximizing integrated likelihoods and the integration is numerically approximated by drawing pseudo-random deviates from presumably known conditional distributions. However, the Monte-Carlo EM method of Guo and Thompson (1991) generates the joint posterior distribution of the *random effects* using the Gibbs sampling. This method generates the same estimates of variance components as the usual ML method. The Gibbs sampler of Wang et al (1993) generates the marginal posterior distribution of each *variance component.* This method is a Bayesian approach in that it requires prior distributions of the unknown parameters and then integrates out all other parameters except the one of interest. Each parameter is then estimated by maximizing its own marginal likelihood function. The Gibbs sampler of Wang et al (1993) does not produce new estimators but the Bayesian estimates of the variance components. The Monte-Carlo algorithm presented in this paper first treats the random effects as fixed and then generates the marginal distribution of the data via Monte-Carlo sampling. Similar to the derivative-free algorithm for REML (Graser et al, 1987), our method is only an alternative algorithm to obtain the ML and REML estimates. It does not generate new estimates of variance components, provided M is sufficiently large.

ACKNOWLEDGMENTS

We thank two anonymous reviewers for their suggestions. This work was supported by NRI Competitive Grants Programs/USDA 95-37205-2313 to SX and the National Science Foundation BSR-9107108 and the National Institutes of Health GM 45344 to WRA.

REFERENCES

Cunningham EP, Henderson CR (1968) An iterative procedure for estimating fixed effects and variance components in mixed model situations. *Biometrics* 24, 13-25

DeGroot MH (1986) *Probability and Statistics.* Addison-Wesley, MA, USA

Dempster AP, Laird NM, Ribin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. *J R Stat Soc* B 39, 1-38

Fahrmeir L, Tutz G (1994) *Multivariate Statistical Modelling Based on Generalized Linear Models.* Springer-Verlag, New York, USA

Geman S, Geman D (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal Machine Intell* 6, 721-741

Graser HU, Smith SP, Tier B (1987) A derivative-free approach for estimating variance components in animal models by restricted maximum likelihood. *J Anim Sci* 64, 1362-1370

Guo SW, Thompson EA (1991) Monte-Carlo estimation of variance component models for large complex pedigrees. *IMA J Math Appl Med Biol* 8, 171-189

Hartley HO, Rao JNK (1967) Maximum-likelihood estimation for the mixed analysis of variance model. *Biometrika* 54, 93-108

Harville DA (1977) Maximum likelihood approaches to variance component estimation and to related problems. *J Am Stat Assoc* 72, 320-339

Henderson CR (1984) ANOVA, MIVQUE, REML, and ML algorithms for estimation of variances and covariances. In: *Statistics: An Appraisal: Proceedings 50th Anniversary Conference* (David HA, David HT, eds), The Iowa State University Press, Ames, IA, 257-280

Henderson CR (1986) Recent developments in variance and covariance estimation. *J Anim Sci* 63, 208-216

Kriese LA, Boldman KG, Van Vleck LD, Kachman SD (1994) MTDFREML: A flexible set of programs to estimate (co)variances for messy multiple trait animal models using derivative-free REML and sparse matrix techniques. In: *Proc 5th World Congress on Genetics Applied to Livestock Production, 7–12 August 1994*, University of Guelph, Guelph, ON, Canada, 22, 43-46

Meyer K (1988) DFREML – a set of programs to estimate variance components under an individual animal model. *J Dairy Sci* 71, 33-34

Misztal I (1994) Comparison of software packages in animal breeding. In: *Proc the 5th World Congress on Genetics Applied to Livestock Production, 7–12 August 1994*, University of Guelph, Guelph, ON, Canada, 22, 3-10

Patterson HD, Thompson R (1971) Recovery of inter-block information when block sizes are unequal. *Biometrika* 58, 545-554

Perez-Enciso M, Misztal I, Elzo MA (1994) FSPAK: An interface for public domain sparse matrix subroutines. In: *Proc 5th World Congress on Genetics Applied to Livestock Production, 7–12 August 1994*, University of Guelph, Guelph, ON, Canada, 22, 87-88

Searle SR (1989) Variance components – some history and a summary account of estimation methods. *J Anim Breed Genet* 106, 1-29

Smith SP, Graser HU (1986) Estimating variance components in a class of mixed models by restricted maximum likelihood. *J Dairy Sci* 69, 1156-1165

Wang CS, Rutledge JJ, Gianola D (1993) Marginal inferences about variance components in a mixed linear model using Gibbs sampling. *Genet Sel Evol* 25, 41-62

Wright S (1934) The results of crosses between inbred strains of guinea pigs differing in number of digits. *Genetics* 19, 537-551