

# MCMC algorithms for structured multivariate normal models

William Browne\*, Suna Akkol†<sup>1</sup> and Harvey Goldstein\*

\*University of Bristol, †University of Yuzuncu Yil, Turkey

## Summary

In this paper we look at the use of MCMC methods for the general class of structured multivariate Normal (SMVN) models. These models consider a response vector as following a multivariate normal distribution with a limited number of parameters representing the mean vector and variance matrix. We in particular look at how multilevel models can be represented in this framework and show that a simple random walk Metropolis algorithm for such models can be very quick and outperform MCMC algorithms for the equivalent model in the standard ‘random effects’ framework on some simple examples. We show however that this computational advantage only holds for very simple models. We discuss model comparison which is more natural in this framework. We finish with discussion of other models that fit naturally in the framework and link to other work on these models.

**Keywords:** Multilevel Modelling, Markov chain Monte Carlo (MCMC), structured multivariate normal models, Metropolis Hastings

## 1 Introduction

Statistical models that take account of the structure of the data that is collected are increasingly widely used. Often this structure involves some clustering in the dataset where certain observations are collected from the same cluster and it is believed that such observations should be more similar than observations collected from different clusters. The term ‘multilevel model’ is typically used to describe such models that deal with clustered data.

There are (at least) two ways of describing a model for data with such clustering and we illustrate this here for a general normal response linear model with response vector  $y$ , predictor variables  $X$  and data collected on  $J$  clusters with  $n_j$  observations in the  $j$ th cluster.

---

<sup>1</sup>We are grateful to (a) the ESRC for financial support for the first author, and (b) TUBITAK-BIDEB for financial support for the second author’s visit to the UK.

Firstly random effect (latent variable) models describe this structure by adding random (unobserved) effects,  $u_j$  to a standard linear model so that we have the following:

$$\begin{aligned} y_{ij} &= X_{ij}\beta + u_j + e_{ij} \\ u_j &\sim N(0, \sigma_u^2), e_{ij} \sim N(0, \sigma_e^2), i = 1, \dots, n_j, j = 1, \dots, J \end{aligned} \quad (1)$$

Here  $i$  indexes the observations in cluster  $j$  and so  $y_{ij}$  is the  $i$ th response on the  $j$  cluster,  $\beta$  is the fixed effect vector associated with the predictors  $X$  and  $u_j$  are the random effects, one for each cluster.

This model formulation is known as a random intercept or variance components model as the  $u_j$  create differing intercepts for each cluster and is the standard one (with the addition of priors for  $\beta$ ,  $\sigma_u^2$  and  $\sigma_e^2$ ) used with MCMC estimation. Assuming conjugate priors are used then this model can be fitted using a four step Gibbs sampling algorithm (see Browne (2003) for details of an implementation in MLwiN). Note that additional random effects that account for cluster specific effects of predictor coefficients (random slopes) can also be included in a multilevel model.

The other formulation is to consider the whole response vector  $y$  following a multivariate Normal distribution as follows:

$$y \sim MVN(X\beta, V)$$

The clustering is now accounted for by the form given to  $V$ . We will describe such a model formulation as a structured multivariate normal (SMVN) model. In fact a simple linear model can also be described in this framework by letting  $V = \sigma_e^2 I$  and in the random intercepts case given by (1) we can write

$$\begin{aligned} y_j &\sim MVN(X_j\beta, V_j) \\ V_j &= \sigma_e^2 I_{n_j} + \sigma_u^2 J_{n_j} \end{aligned} \quad (2)$$

where  $y_j$  is the response vector for the  $j$ th cluster,  $X_j$  is the matrix of  $p$  predictor variables for the  $j$ th cluster,  $I_{n_j}$  is the  $n_j \times n_j$  identity matrix and  $J_{n_j}$  is an  $n_j \times n_j$  matrix of 1s. This formulation is the basis of the IGLS algorithm (Goldstein, 1986) that is used in the software package MLwiN (Rasbash et al. 2000). It is a straightforward exercise to verify that (1) does indeed imply (2).

In this paper we consider using MCMC methods for this second formulation of the model and highlight some advantages and disadvantages. In particular we shall focus on the speed advantages from the use of this formulation. As computer processing speeds and memory continue, apparently, to obey Moore's law, namely a power doubling approximately every 2 years, it is sometimes argued that searching for more efficient computing algorithms should have a low priority since the performance of the hardware can be expected to achieve the same or superior effect within a reasonable time scale. While this may be true for certain applications, our own experience in the

social and medical sciences is that, as processing capacity improves, so does the ambition of the data analyst. One might even argue that Moore's law should also be applied to the size of problems that data analysts are prepared to contemplate. This is particularly relevant to the use of very large administrative datasets from, for example health and education. These datasets are now available for analysis and will often include several million individual subjects with repeated observations and many variables, with various levels of clustering, so that processing time is crucial to the feasibility of completing an analysis within a realistic time scale. We would suggest, therefore, that the search for more efficient software algorithms is a worthwhile activity, and the practical justification for the topic of the present paper.

In the next section we describe their use in the simple family of random intercept models. In section 3 we look at choosing between models and in particular testing whether clustering is important. In section 4 we look at random slopes models and see that adding extra random terms often removes the speed advantage of the framework. In section 5 we look at other models that naturally fit into this framework before finishing with some discussion.

## 2 Random intercept models

Multilevel models are used to account for underlying structure (clustering) in the data which are used in an analysis. The simplest form of multilevel model is the random intercept model (1) where a single random term is used for each cluster to account for within cluster similarity. There are many motivations for fitting a multilevel model. Generally we assume that the main focus in the analysis is the relationship between the response variable  $y$  and the predictor variables  $X$ . Then the structure of the data might be perceived as a nuisance that has to be accounted for in some way. This is the motivation behind techniques such as generalised estimating equations (GEE, Diggle, 1994) and the use of sandwich estimators (White, 1980) that adjust fixed effect standard errors to account for the clustering.

Alternatively the structure of the data itself might be important and we might like to make inferences about the structure through the magnitude of the cluster level variance  $\sigma_u^2$  or the individual random effects  $u_j$ . For example the between cluster variance (along with the level 1 variance) can be used to construct the variance partitioning coefficient (VPC) which estimates the importance of the clustering in the data for a particular observation and for a random intercept model is

$$VPC = \frac{\sigma_u^2}{\sigma_u^2 + \sigma_e^2}$$

Note for the random intercept model the VPC does not depend on the value of the predictors and is identical to the intra-class correlation (ICC). It can be debated whether inferences about individual random effects should be

made. If the particular units that the random effects represent are themselves important then their exchangeability might be questioned and they could instead be fitted as fixed effects. If however exchangeability is not an issue and we have few observations on a particular cluster then a random effects model will shrink the corresponding cluster effect towards zero and prevent overconfidence in the effects associated with small clusters. Furthermore, as we shall see when we look at random coefficient models, fitting a fixed effect for every (potentially random) coefficient for every cluster becomes unwieldy when there is a large number of clusters and the random effects model which assumes a distribution for these coefficients is an efficient modelling procedure.

The estimation of random effects to assess goodness of fit of the model assumptions, for example normality in (1), and to look for outliers is also appropriate. A disadvantage of the alternative SMVN formulation is that it doesn't contain the random effects  $u_j$  and so no goodness of fit testing of this type can be done. It is however possible (see Goldstein 2003 Appendix 2.2) to construct residuals conditional on the parameter estimates obtained from a model fitted using SMVN. The other disadvantage of the SMVN formulation is that the Bayesian random effects formulation (with conjugate priors) results in conditional conjugacy for all parameters and hence the use of a simple Gibbs sampling algorithm, whilst this is not true for the SMVN formulation.

The two formulations will result in equivalent estimates assuming the same priors are used for  $\beta$ ,  $\sigma_u^2$  and  $\sigma_e^2$ . It should be noted that in the SMVN formulation the parameter  $\sigma_u^2$  is really a covariance in the model and not a variance and therefore it can feasibly take negative values which can be interpreted as greater variability within clusters than in the population in general. A good example of this is litter effects in animal populations where competition for resources may generate more variability of weight or size in a litter than in a random sample from the population generally. In the SMVN formulation therefore, a negative value for this parameter presents no problems, and we can choose a prior that allows such negative values and this could be useful for model selection as we discuss later.

We will here consider the SMVN model in a Bayesian framework and devise a random walk Metropolis algorithm for fitting the models. It should be noted that the fixed effects  $\beta$  will have a Gaussian full conditional distribution (Chib and Carlin, 1999) and so could alternatively be updated using Gibbs sampling but here we consider an algorithm with random-walk Metropolis steps for all parameters.

## 2.1 Constructing a random walk Metropolis algorithm

For Bayesian SMVN models we have 3 sets of unknown parameters, the fixed effects  $\beta$ , the level 2 random variance terms  $\Omega_u$ , which for the random intercept model consist of a single variance  $\sigma_u^2$ , and the level 1 variance  $\sigma_e^2$ .

All these parameters require prior distributions and so for generality we will assume generic priors  $p(\beta)$ ,  $p(\sigma_u^2)$  and  $p(\sigma_e^2)$ . These then need to be combined with the likelihood to form the posterior distribution. The likelihood is as follows:

$$L(y \mid \beta, \sigma_u^2, \sigma_e^2) = \prod_{j=1}^J (2\pi)^{-n_j/2} |V_j| \exp \left[ -\frac{1}{2} (y_j - X_j \beta)^T V_j^{-1} (y_j - X_j \beta) \right]$$

where  $V_j = \sigma_e^2 I_{n_j} + \sigma_u^2 J_{n_j}$ .

As illustrated in McCulloch and Searle (2001) for special cases this likelihood can be simplified, for example, for the variance components model (a random intercept model with only a constant ( $\beta_0$ ) in the fixed effects) we have

$$\log(L(y \mid \beta, \sigma_u^2, \sigma_e^2)) = -\frac{1}{2} N \log 2\pi - \frac{1}{2} \sum_j \log(\sigma_e^2 + n_j \sigma_u^2) - \frac{1}{2} (N - J) \log(\sigma_e^2) - \frac{1}{2\sigma_e^2} \sum_{i,j} (y_{ij} - \beta_0)^2 + \frac{\sigma_u^2}{2\sigma_e^2} \sum_j \frac{n_j (\bar{y}_j - \beta_0)^2}{\sigma_e^2 + n_j \sigma_u^2}$$

where  $N = \sum_j n_j$  and  $\bar{y}_j = \sum_i y_{ij}/n_j$ . For a general random intercept model this becomes

$$\log(L(y \mid \beta, \sigma_u^2, \sigma_e^2)) = -\frac{1}{2} N \log 2\pi - \frac{1}{2} \sum_j \log(\sigma_e^2 + n_j \sigma_u^2) - \frac{1}{2} (N - J) \log(\sigma_e^2) - \frac{1}{2\sigma_e^2} \sum_{i,j} (y_{ij} - X_{ij} \beta)^2 + \frac{\sigma_u^2}{2\sigma_e^2} \sum_j \frac{(\sum_i (y_{ij} - X_{ij} \beta))^2}{\sigma_e^2 + n_j \sigma_u^2}.$$

Given the above we can choose starting values for all the parameters in the model and then construct a three step Metropolis Hastings algorithm.

Step 1 - Update the fixed effects  $\beta$  using univariate random walk Metropolis at iteration  $t + 1$  as follows, for  $j = 1, \dots, p$  and with  $\beta_{(-j)}$  signifying the  $\beta$  vector without component  $j$ :

$$\begin{aligned} \beta_j(t+1) &= \beta_j(*) \text{ with probability } \min \left[ 1, \frac{p(\beta_j(*) \mid y, \sigma_u^2, \sigma_e^2, \beta_{(-j)})}{p(\beta_j(t) \mid y, \sigma_u^2, \sigma_e^2, \beta_{(-j)})} \right] \\ &= \beta_j(t) \text{ otherwise,} \end{aligned}$$

where  $\beta_j(*) \sim N(\beta_j(t), s_{\beta_j}^2)$  and  $p(\beta_j \mid y, \sigma_u^2, \sigma_e^2, \beta_{(-j)}) \propto L(y \mid \beta, \sigma_u^2, \sigma_e^2) p(\beta_j)$ .

Step 2 - Update  $\sigma_u^2$  using univariate random walk Metropolis at iteration  $t + 1$  as follows :

$$\begin{aligned} \sigma_u^2(t+1) &= \sigma_u^2(*) \text{ with probability } \min \left[ 1, \frac{p(\sigma_u^2(*) \mid y, \beta, \sigma_e^2)}{p(\sigma_u^2(t) \mid y, \beta, \sigma_e^2)} \right] \\ &= \sigma_u^2(t) \text{ otherwise,} \end{aligned}$$

where  $\sigma_u^2(*) \sim N(\sigma_u^2(t), s_{p_u}^2)$  and  $p(\sigma_u^2 \mid y, \beta, \sigma_e^2) \propto L(y \mid \beta, \sigma_u^2, \sigma_e^2) p(\sigma_u^2)$ .

Here as described previously in Browne (2006) we use a normal proposal distribution and assume that inadmissible values for  $\sigma_u^2$  will have zero likelihood or a prior of zero. The alternative is to use a truncated Normal proposal (and calculate the resulting Hastings ratio for each iteration) where the truncation point is zero for priors that force  $\sigma_u^2$  to be positive and  $-\sigma_e^2/\max_j(n_j)$  otherwise.

Step 3 - Update  $\sigma_e^2$  using univariate random walk Metropolis at iteration  $t+1$  as follows :

$$\begin{aligned}\sigma_e^2(t+1) &= \sigma_e^2(*) \text{ with probability } \min \left[ 1, \frac{p(\sigma_e^2(*)|y,\beta,\sigma_u^2)}{p(\sigma_e^2(t)|y,\beta,\sigma_u^2)} \right] \\ &= \sigma_e^2(t) \text{ otherwise,}\end{aligned}$$

where  $\sigma_e^2(*) \sim N(\sigma_e^2(t), s_{pe}^2)$  and  $p(\sigma_e^2 | y, \beta, \sigma_u^2) \propto L(y | \beta, \sigma_u^2, \sigma_e^2)p(\sigma_e^2)$ .

Again we have the alternative here of a truncated normal proposal distribution with truncation point 0. To choose the values of the proposal standard deviations,  $s_{\beta_j}$ ,  $s_{pu}$  and  $s_{pe}$  we use an adapting procedure due to Browne and Draper (2000) that consists of an additional adaptation period prior to burning in, in which the proposal distributions are tuned to give acceptance rates of roughly 50%.

## 2.2 Efficient Likelihood Evaluation

It is easier to work with the log-likelihood than the likelihood and for the general variance components model we have

$$\begin{aligned}\log(L(y | \beta, \sigma_u^2, \sigma_e^2)) &= -\frac{1}{2}N \log 2\pi - \frac{1}{2} \sum_j \log(\sigma_e^2 + n_j \sigma_u^2) - \frac{1}{2}(N - J) \log(\sigma_e^2) \\ &\quad - \frac{1}{2\sigma_e^2} \sum_{i,j} (y_{ij} - X_{ij}\beta)^2 + \frac{\sigma_u^2}{2\sigma_e^2} \sum_j \frac{(\sum_i (y_{ij} - X_{ij}\beta))^2}{\sigma_e^2 + n_j \sigma_u^2}.\end{aligned}$$

Here the majority of the computation lies in the last two terms with their summations of squared terms over the length of the dataset. We can speed up evaluation by considering summary statistics of the data that are constant through iterations.

If we write  $y$  for the vector of responses  $y_{ij}$  and  $X$  for the matrix of predictor variables then we can write the penultimate term,  $\sum_{i,j} (y_{ij} - X_{ij}\beta)^2 = y^T y - 2y^T X \beta + \beta^T X^T X \beta$ . Now storing the sums of squares and cross product objects  $y^T y$ ,  $y^T X$  and  $X^T X$  results in far fewer numerical operations. By a similar argument the same objects can be constructed for each cluster and the operations for the inner  $i$  summation in the final term can be reduced. These speed ups rely on the fact that the number of observations in the dataset  $N$  (and the size of each cluster  $n_j$ ) is far greater than the number of fixed effects  $p$  which is true in many cases.

### 2.3 Comparison of results for an example from education

We here consider a dataset of student exam scores (Rasbash et al. 2004) taken by students at age 16. We begin by fitting a variance components model and we have 4059 students in 65 schools. We compare the results and the MCMC chains from the above algorithm and the standard Gibbs sampling algorithm in MLwiN. For comparison of the chains we use the effective sample size (ESS) statistic described in Kass et al. (1998) which for each chain gives the size of an equivalent independent sample from the posterior distribution. For this example we use improper uniform priors for  $\beta$ ,  $\sigma_u^2$  and  $\sigma_e^2$  although similar conclusions can be obtained with other ‘diffuse’ priors and we run for 100k iterations after a burnin of 5,000. The results can be seen in table 1.

Table 1: Estimates and effective sample sizes for the variance components model

Parameter	MLwiN		SMVN formulation	
	Estimate (S.D.)	ESS	Estimate (S.D.)	ESS
$\beta_0$	-0.012 (0.056)	4k	-0.013 (0.056)	24k
$\sigma_u^2$	0.184 (0.038)	65k	0.185 (0.038)	21k
$\sigma_e^2$	0.849 (0.019)	97k	0.849 (0.019)	25k
Time	50s	-	2s	-

We can see that although the SMVN formulation has worse ESS for the two variance parameters it is far quicker and wins on the metric ESS/s. It should be noted that we are using the standard MCMC algorithm in MLwiN which is not using hierarchical centering (Gelfand et al. 1995) which will increase the ESS for  $\beta_0$ . This is possible in the WinBUGS software package and the resulting ESS is 82k although the algorithm in WinBUGS took 250s.

## 3 Model comparison

The education dataset has a predictor variable,  $LRT$  which is a reading test that the students took at age 11. We can include this parameter in our model and fit a random intercept model using either of the formulations. The results are given in table 2

Here as with the variance components model we see generally smaller ESS for the SMVN formulation but a far faster algorithm means that the SMVN formulation gives better ESS/s values.

If we wish to choose between the two models there are many possibilities. Perhaps the simplest comparison tool is to look at the chain for the additional parameter  $\beta_1$  as the variance components model is simply the random intercept model with  $\beta_1$  constrained to zero. Using either formulation this

Table 2: Estimates and effective sample sizes for the random intercept model

Parameter	MLwiN		SMVN formulation	
	Estimate (S.D.)	ESS	Estimate (S.D.)	ESS
$\beta_0$ - intercept	0.004 (0.042)	5k	0.002 (0.042)	25k
$\beta_1$ - LRT	0.563 (0.013)	83k	0.563 (0.012)	25k
$\sigma_u^2$	0.101 (0.021)	59k	0.101 (0.022)	20k
$\sigma_e^2$	0.566 (0.013)	97k	0.566 (0.013)	24k
Time	57s	-	3s	-

chain never crosses the zero value and so there is no support for the simpler model.

A more complicated example is testing for the existence of a set of random effects or equivalently their variance,  $\sigma_u^2$ . The model without the random effects effectively has  $\sigma_u^2$  constrained to equal zero, however in the random effects formulation  $\sigma_u^2$  is truly a variance and so requires a prior that has all its mass on positive values and hence it's posterior chain will never cross the zero value. An advantage of the SMVN approach is that a prior with mass on negative values can be used, as in this framework  $\sigma_u^2$  is really representing covariance within the larger variance matrix  $V$  (and perhaps should be relabelled  $\sigma_{uu}$ ).

To illustrate this point we simulated 4059 fully independent standard normal responses and then match these responses to the data structure of the tutorial dataset. We then fitted a variance components model using each framework and got the following results:

Table 3: Estimates and effective sample sizes for a simulated variance components model with zero level 2 variance

Parameter	MLwiN		SMVN formulation	
	Estimate (S.D.)	ESS	Estimate (S.D.)	ESS
$\beta_0$	-0.001 (0.017)	72k	-0.003 (0.016)	23k
$\sigma_u^2$	0.002 (0.002)	2.3k	0.000 (0.003)	18k
$\sigma_e^2$	1.022 (0.023)	99k	1.024 (0.023)	24k
Time	50s	-	2s	-

Here we have similar estimates from the two approaches however the MLwiN estimate is based upon a chain that never goes negative whilst the SMVN formulation chain can be seen in figure 1.

We see here that the posterior has considerable mass below zero and so we can use a 95% credible interval to show that zero cannot be rejected as a potential estimate. Note that chapter 5 of Box and Tiao (1973) give a nice discussion of the problem of negative variance in random effect models which can result from a sampling theory approach.

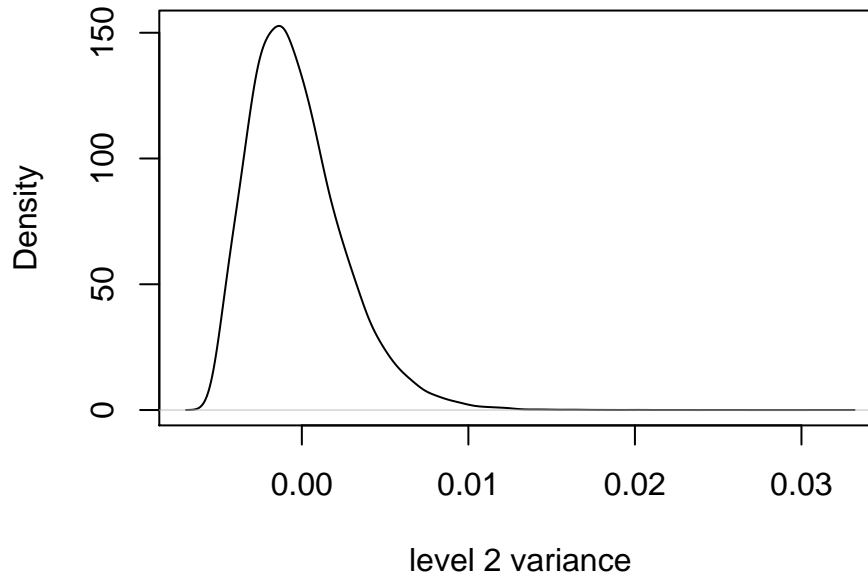


Figure 1: Kernel density plot of posterior of  $\sigma_u^2$  under the SMVN formulation

### 3.1 Deviance Information Criterion

The Deviance information criterion (DIC) was proposed by Spiegelhalter et al. (2002) as a method for comparing statistical models fitted using MCMC. Although the DIC has its critics, including several of the discussants of the original paper read at the RSS, its ease of calculation and the lack of implementation of alternatives has led to considerable use in applied articles. The DIC like other information criteria is a combination of the fit and complexity of a model. The fit in this case is the mean deviance and the complexity is twice the ‘effective’ number of parameters.

One problem with the DIC is its lack of invariance to the ‘focus’ of the model, so for example a reparameterisation of a model and calculation of the DIC using this new parameterisation may result in different numerical estimates. One nice feature of the use of the DIC with the SMVN formulation described in this paper is that due to the lack of random effects, the ‘effective’ number of parameters is (usually) close to the actual number of parameters

in the model.

Table 4 gives DIC values for the two models considered earlier along with the random slopes model in the next section based on the SMVN method.

Table 4: DIC Estimates for three models using the SMVN formulation

Model	$D(\bar{\theta})$	$D(\bar{\theta})$	$p_D$	DIC
Model 1: Variance Components	11013.8	11010.9	2.9	11016.7
Model 2: Random Intercepts	9361.4	9357.5	3.9	9365.3
Model 3: Random Slopes	9323.5	9321.2	2.3	9325.7

Here we see that the models give successively smaller DIC values suggesting better models. The introduction of the LRT predictor (moving from models 1 to 2) and the introduction of random slopes (moving from models 2 to 3) both result in lower DIC values. The effective number of parameters  $p_D$  is approximately equal to the actual number of parameters in the first two model but is lower in the random slopes model. This gives some evidence that the DIC in this framework might be easier to understand and actually will be equivalent to the AIC.

## 4 Random slopes models

A natural extension of the variance components and random intercepts model is the random slopes model where we allow predictor variables to have different effects for each higher level unit. In the education example we can consider allowing the effect of the predictor  $LRT$  to vary across schools. In the random effects framework we would write this as

$$\begin{aligned} y_{ij} &= \beta_0 + LRT_{ij}\beta_1 + u_{0j} + LRT_{ij}u_{1j} + e_{ij} \\ u_j &\sim MVN(0, \Omega_u), \quad e_{ij} \sim N(0, \sigma_e^2), \quad i = 1, \dots, n_j, \quad j = 1, \dots, J \end{aligned} \quad (3)$$

where  $u_j = (u_{0j}, u_{1j})^T$ .

In the SMVN framework we can write a random slopes model as

$$\begin{aligned} y_j &\sim MVN(X_j\beta, V_j) \\ V_j &= \sigma_e^2 I_{n_j} + X_j\Omega_u X_j^T \end{aligned} \quad (4)$$

where  $X_j = (X_{1j}, \dots, X_{n_jj})^T$  and  $X_{ij} = (1, LRT_{ij})$  in our example. This is similar to (2) but with the within group variance matrices  $V_j$  depending on the predictor variable  $LRT$ . We then have the general MVN likelihood function for each block and the likelihood is as follows:

$$L(y \mid \beta, \Omega_u, \sigma_e^2) = \prod_{j=1}^J (2\pi)^{-n_j/2} |V_j| \exp \left[ -\frac{1}{2} (y_j - X_j\beta)^T V_j^{-1} (y_j - X_j\beta) \right].$$

We adapt the random walk Metropolis algorithm described in section 2.1 to random slopes models. The steps (1) and (3) for the fixed effects and the level 1 variance are unchanged and so we adapt step (2) as follows:

Step 2 - Update  $\Omega_u[i, j], i = 0, 1, j \leq i$  using univariate random walk Metropolis at iteration  $t + 1$  as follows :

$$\begin{aligned}\Omega_u[i, j](t + 1) &= \Omega_u[i, j](*) \text{ with probability } \min \left[ 1, \frac{p(\Omega_u[i, j](*) | y, \beta, \sigma_e^2)}{p(\Omega_u[i, j](t) | y, \beta, \sigma_e^2)} \right] \\ &= \Omega_u[i, j](t) \text{ otherwise,}\end{aligned}$$

where  $\Omega_u[i, j](*) \sim N(\Omega_u[i, j](t), s_{pu[i, j]}^2)$  and  $p(\Omega_u[i, j] | y, \beta, \sigma_e^2) \propto L(y | \beta, \Omega_u, \sigma_e^2)p(\Omega_u)$ .

Note that  $s_{pu[i, j]}$  are the proposal standard deviations that will be adapted as before (Browne and Draper (2000)). Here as described previously in Browne (2006) we use a normal proposal distribution for each element of  $\Omega_u$  and assume that inadmissible sets of values for  $\Omega_u$  will have likelihood or prior zero.

For the variance components model we were able to calculate a negative lower bound ( $-\sigma_e^2 / \max_j(n_j)$ ) for the parameter  $\sigma_u^2$  that resulted in positive definite matrices  $V_j$  and this allowed us to test if  $\sigma_u^2$  was significantly different from 0. For the random slopes model there is not an equivalent single lower bound since we now have several parameters in the matrix and hence the bound for each parameter will depend on the values of the rest of the matrix. Moreover, even if we did carry out such a check, this would not guarantee that any of the  $V_j$  matrices remained positive definite for all possible values of the predictor  $X$  and would raise interpretational difficulties. Instead, therefore, we shall treat the matrix  $\Omega_u$  as a true variance matrix (as in the random effects framework) by using a prior that only has support for positive definite matrices so that we can easily test that a proposed variance matrix is positive definite by using a Cholesky decomposition. Assuming that  $\Omega_u$  is strictly a variance matrix rather than a set of parameters that describe correlations in the variance matrix  $V$  does result in the lack of a nice model comparison test as used in section 3 for variance components models and so in further work we will investigate if we can calculate conditional bounds for each of the parameters in  $\Omega_u$  when assuming only that  $V$  is positive definite but not necessarily  $\Omega_u$  and also study interpretational issues. The alternative approach to using normal proposals is to use truncated Normal proposals (and calculate the resulting Hastings ratio for each iteration) where the truncation points are such that  $\Omega_u$  remains positive definite. In our  $2 \times 2$  example we have  $\Omega_u[0, 0] > 0, \Omega_u[1, 1] > 0$  and  $-\sqrt{\Omega_u[0, 0]\Omega_u[1, 1]} < \Omega_u[0, 1] < \sqrt{\Omega_u[0, 0]\Omega_u[1, 1]}$  but such constraints increase in number as the size of the variance matrix increases and the truncated normal method becomes impractical (see Browne (2006)).

#### 4.1 Efficient Likelihood Evaluation

The likelihood function  $L(y \mid \beta, \Omega_u, \sigma_e^2)$  has been simplified by considering the MVN vector as a set of independent MVN blocks corresponding to level 2 units whose likelihood can be evaluated separately. This however still leaves blocks of size of up to 198 in the education example. For these blocks we need to invert the variance matrices  $V_j$  every time we propose new values for any of the variance parameters in the model. Inverting these blocks directly is computationally prohibitive and the algorithm can potentially be incredibly slow. We can, however, use matrix algebra results to speed things up.

We can use the following matrix identity given in Goldstein(1986):

$$(A + BCB^T)^{-1} = A^{-1} - A^{-1}BC(I + B^T A^{-1}BC)^{-1}B^T A^{-1}.$$

In our example we then have

$$V_j^{-1} = \frac{1}{\sigma_e^2} \left[ I_{n_j} - \frac{1}{\sigma_e^2} X_j \Omega_u \left( I_2 + \frac{X_j^T X_j \Omega_u}{\sigma_e^2} \right)^{-1} X_j^T \right].$$

This will remove the need to invert matrices of upto size  $198 \times 198$  and instead we will have matrices of size  $2 \times 2$  to invert albeit with some additional matrix multiplications. We will now look at using this algorithm on a random slopes model for the education dataset.

Table 5 gives results for fitting a random slopes regression model to the tutorial dataset with a burnin of 5,000 iterations and a run of 100,000 iterations.

Table 5: Estimates and effective sample sizes for the RSR model

Parameter	MLwiN		SMVN formulation	
	Estimate (S.D.)	ESS	Estimate (S.D.)	ESS
$\beta_0$ - intercept	-0.011 (0.042)	4.3k	-0.012 (0.043)	19k
$\beta_1$ - LRT	0.556 (0.021)	15k	0.556 (0.021)	18k
$\Omega_u[0, 0]$	0.103 (0.022)	58k	0.103 (0.022)	11k
$\Omega_u[0, 1]$	0.020 (0.008)	36k	0.020 (0.008)	9k
$\Omega_u[1, 1]$	0.018 (0.006)	22k	0.018 (0.006)	11k
$\sigma_e^2$	0.554 (0.013)	93k	0.554 (0.013)	24k
Time	96s	-	3hrs	-

Here we see good agreement between the two formulations and not much to choose in terms of ESS however even with our efficient likelihood evaluations the fact that the SMVN formulation takes over 100 times longer means it is clearly for this example not a practical competitor. Before we dismiss the SMVN formulation as being totally unsuitable for random slopes model

we will consider another example and application of random effect models, namely repeated measures models.

## 4.2 A balanced repeated measures model

The education dataset has some large clusters of observations and is unbalanced both in terms of cluster size and predictor values. We will here consider a smaller example on the weights of laboratory rats studied in Gelfand et al. (1990). This dataset consists of the weights of 30 rats that were measured weekly for 5 weeks. We will consider the following random slopes model:

$$\begin{aligned} y_{ij} &= \beta_0 + \text{age}_i \beta_1 + u_{0j} + \text{age}_i u_{1j} + e_{ij} \\ u_j &\sim MVN(0, \Omega_u), \quad e_{ij} \sim N(0, \sigma_e^2), \quad i = 1, \dots, n_j, \quad j = 1, \dots, J \end{aligned} \quad (5)$$

where  $u_j = (u_{0j}, u_{1j})^T$ ,  $i$  indexes time points and  $j$  indexes rats. In the SMVN framework we can write the model as

$$\begin{aligned} y_j &\sim MVN(X_j \beta, V_j) \\ V_j &= \sigma_e^2 I_{n_j} + X_j \Omega_u X_j^T \end{aligned} \quad (6)$$

but as all rats are the same age at each measurement then we have  $X_j = X'_j \forall j, j'$  and hence we can replace  $V_j$  with  $V_1 \forall j$ . This will reduce the computation as rather than evaluating 30 separate  $V_j^{-1}$  we can evaluate one and use it for each rat. In table 6 we give estimates from the random effects and SMVN formulations of the model.

Table 6: Estimates and effective sample sizes for the rats model

Parameter	MLwiN		SMVN formulation	
	Estimate (S.D.)	ESS	Estimate (S.D.)	ESS
$\beta_0$ - intercept	106.65 (2.584)	6.5k	106.59 (2.598)	18k
$\beta_1$ - age	6.188 (0.119)	3.6k	6.186 (0.119)	18k
$\Omega_u[0, 0]$	154.73 (61.74)	36k	156.43 (63.46)	7.6k
$\Omega_u[0, 1]$	-1.440 (2.099)	33k	-1.474 (2.114)	6.9k
$\Omega_u[1, 1]$	0.344 (0.131)	41k	0.344 (0.129)	7.9k
$\sigma_e^2$	37.91 (5.856)	97k	37.83 (5.857)	20k
Time	20s	-	7s	-

Here we see again that both formulations give similar estimates and have reasonably similar overall performance in terms of ESS, with the random effects framework giving better ESS for random parameters but worse for fixed effects. The SMVN formulation is competitive in terms of computation time and is actually quicker than the standard formulation. This is in part due to the speed up due to the balanced data and without the speed up the method took 86 seconds. This does however only represent a factor of

4 rather than 100 between the two methods for the rats dataset and this is because the largest block is of size 5 (as opposed to 198) and the number of multiplications involved in the evaluation of the likelihood is  $O(n_j^2)$ . This leads us to conclude that the SMVN formulation may not be so bad for random slopes models as long as the block sizes aren't too big but also that the speed up due to the balanced structure might not be so impressive for longer repeated measures series.

## 5 Other structured multivariate normal models

The number of models that can be written in the general multivariate normal

$$y \sim MVN(X\beta, V)$$

is vast. We have here concentrated on two level models that are generally considered in a random effects framework when using MCMC. One feature of these models is their block structure and the fact that we can write the model as:

$$y_j \sim MVN(X_j\beta, V_j)$$

for some blocks  $j = 1, \dots, J$ . This means that the computationally intensive part of the likelihood evaluation (inverting the matrix  $V$ ) is replaced by the computationally quicker inversion of the variance matrices for the individual blocks  $V_j$ . This suggests that other models that might be suitable for MCMC fitting via the structural MVN formulation would have different structural forms for the matrices  $V_j$  while maintaining between block independence.

Examples of such models that have been fitted (in a maximum likelihood setting) by extending the IGLS algorithm are multilevel time series models (Goldstein et al. 1994). Here the correlation between observations within a cluster depends on the time gap between the two observations. Browne (2006) describes an MCMC algorithm for the rats repeated measure data that assumes an autoregressive (AR(1)) structure for the time gaps. As the time gaps are all equal this results in the following correlation matrix with  $\rho$  estimated at 0.94 :

$$D = \begin{pmatrix} 1 & \rho & \rho^2 & \rho^3 & \rho^4 \\ \rho & 1 & \rho & \rho^2 & \rho^3 \\ \rho^2 & \rho & 1 & \rho & \rho^2 \\ \rho^3 & \rho^2 & \rho & 1 & \rho \\ \rho^4 & \rho^3 & \rho^2 & \rho & 1 \end{pmatrix}.$$

Browne (2006) then uses separate mean and variance parameters for each timepoint resulting in a model with 5 parameters to explain mean behaviour

and 6 parameters to explain the variance matrices  $V_j$ . The random slopes model described in section 4 uses only 2 parameters to explain mean and 4 parameters to explain the variance matrices  $V_j$  which results in a correlation between two observations at times  $s$  and  $t$ ,  $\rho_{s,t}$  as

$$\rho_{s,t} = \frac{\Omega_u[0,0] + (s+t)\Omega_u[0,1] + st\Omega_u[0,1]}{\sqrt{(\Omega_u[0,0] + 2s\Omega_u[0,1] + s^2\Omega_u[0,1] + \sigma_e^2)(\Omega_u[0,0] + 2t\Omega_u[0,1] + t^2\Omega_u[0,1] + \sigma_e^2)}}$$

which will change both with the time gap between the measures and also the time of the measures. If we use the estimates we have for the random slopes model we get

$$D = \begin{pmatrix} 1 & 0.779 & 0.720 & 0.653 & 0.592 \\ 0.779 & 1 & 0.829 & 0.800 & 0.765 \\ 0.720 & 0.829 & 1 & 0.875 & 0.863 \\ 0.653 & 0.800 & 0.875 & 1 & 0.910 \\ 0.592 & 0.765 & 0.863 & 0.910 & 1 \end{pmatrix}.$$

These correlations are smaller than those in the AR(1) model in Browne (2006) but this is in part due to the AR(1) model using more parameters to estimate the mean and variance structure than the random slopes model.

It is possible to fit many other structures to the within block variance matrices. The balanced repeated measures situation with the same structure for the within block matrix for each block clearly will have the greatest computational speed advantages but the same models can be fitted to non-balanced structures. When the model does not have within block variance matrices of the form  $(A + BCB^T)$ , then we cannot use the result in section 4.1 which will result in slower execution.

Between block correlations as might be present if there are spatial relationships between blocks of responses, for example distances between schools are probably best not considered in the SMVN formulation and instead a latent variable random effects framework is much better. We will consider between block correlations in future work.

## 6 Discussion

In this paper we have considered how structured MVN models might be implemented using MCMC estimation. We have in the main considered how multilevel models that are commonly fitted using a random effects formulation when using MCMC might be fitted in the structured MVN formulation. We have shown the advantage this presents in terms of model comparison and computational speed for simple examples. We have also shown how for more complex models with further random effects the computational speed

advantage disappears and the structured MVN framework can be impractical. We have briefly looked at other structured MVN models that fit the general framework.

In practical implementations it should be possible to interrogate the data structure in order to estimate the relative efficiencies of the two different algorithms we have studied. In further work we will look at rapid methods for carrying out such calculations so that a software package can choose the most efficient procedure for any given specified model.

## References

- Box, G.E.P. and Tiao, G.C. (1973). *Bayesian Inference in Statistical Analysis*. Reading, Mass., Addison-Wesley.
- Browne, W.J. (2003). *MCMC Estimation in MLwiN*. London: Institute of Education, University of London
- Browne, W.J. (2006). MCMC Algorithms for constrained variance matrices. *Computational Statistics and Data Analysis* 50: 1655-1677.
- Browne, W.J. and Draper D. (2000). Implementation and performance issues in Bayesian and likelihood fitting of multilevel models. *Computational Statistics* 15: 391-420.
- Chib, S and Carlin, B.P. (1999). On MCMC sampling in hierarchical longitudinal models. *Statistics and Computing*, 9, 17-26.
- Diggle, P.J., Liang, K-Y. and Zeger, S.L. (1994) *Analysis of Longitudinal Data*. Oxford, Clarendon Press.
- Gelfand, A.E., Hills, S.E., Racine-Poon, A. and Smith, A.F.M (1990). An illustration of Bayesian inference in normal data models using Gibbs Sampling. *Journal of the American Statistical Association* 85: 972-985
- Gelfand, A.E., Sahu, S.K. and Carlin, B.P. (1995). Efficient parameterizations for normal linear mixed models. *Biometrika* 82, 479-488.
- Goldstein, H. (1986). Multilevel mixed linear model analysis using iterative generalised least squares. *Biometrika* 73, 43-56.
- Goldstein, H. (2003). *Multilevel Statistical Models*. (Third edition) London: Edward Arnold.
- Goldstein, H., Healy, M.J.R. and Rasbash, J. (1994). Multilevel Time Series Models with applications to repeated measures data. *Statistics in Medicine* 13: 1643-1655.
- Kass, R.E., Carlin, B.P., Gelman, A. and Neal, R. (1998). Markov chain Monte Carlo in practice: a roundtable discussion. *American Statistician*, 52, 93-100.
- McCulloch, C.E. and Searle, S.R. (2001). *Generalized, Linear and Mixed Models*. New York: John Wiley and Sons.
- Rasbash, J., Browne, W.J., Healy, M, Cameron, B and Charlton, C. (2000). The MLwiN software package version 1.10. London: Institute of Education, University of London.
- Rasbash, J., Steele, F., Browne, W.J. and Prosser, B. (2004). *A User's Guide to MLwiN*. London: Institute of Education, University of London.

- Spiegelhalter, D.J., Best, N.G., Carlin, B.P. and van der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society, Series B* 64: 583-640.
- White H (1980). A heteroscedasticity-consistent covariance matrix estimator and a direct test for heteroscedasticity. *Econometrica*, 48, 817-830.