

Using SMCMC for normal response multilevel models

William Browne

Universities of Bristol and Nottingham, UK

pmzwjb@nottingham.ac.uk

<http://www.maths.nottingham.ac.uk/personal/pmzwjb/>

Sixth International Amsterdam conference on Multilevel Analysis

Amsterdam

April 2007

Joint work with Mousa Golalizadeh

Outline

1. Introduction to MVN formulations
2. Structured MCMC
3. Variance Components model
4. Random slopes model
5. General 2 level model
6. Extensions to more levels and non-normal responses

Two level Multilevel models

We can write out a general 2 level model as follows:

$$y_{ij} = \mathbf{X}_{ij}^T \boldsymbol{\beta} + \mathbf{Z}_{ij}^T \mathbf{u}_j + e_{ij}$$

$$\mathbf{u}_j \sim MVN(0, \Omega_u), e_{ij} \sim N(0, \sigma_e^2)$$

where \mathbf{X}_{ij}^T and \mathbf{Z}_{ij}^T are row vectors whilst $\boldsymbol{\beta}$ and \mathbf{u}_j are column vectors.

Here i indexes the level 1 units nested within j the level 2 units. $\boldsymbol{\beta}$ are fixed effects common to all observations whilst \mathbf{u}_j are random effects specific to particular level 2 units. We assume J level 2 units and N level 1 units.

How is THIS (or a similar) model fitted using various approaches?

The IGLS approach - Integrating out the random effects

Here we write the model as a single vector response multivariate Normal model as

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

where V is an $N \times N$ matrix with $V_{ii'} = f(\Omega_u, \sigma_e^2)$.

For example in a variance components model:

$V_{ii} = \sigma_u^2 + \sigma_e^2$, $V_{ii'} = \sigma_u^2$ if i and i' are distinct but from the same level 2 unit else $V_{ii'} = 0$.

This model is equivalent to the standard multilevel model if additional constraints are added. Without these constraints estimates of higher level variance matrices need not be positive definite.

The HGLM approach - additional pseudo data

Here we again write the model as a single vector response multivariate Normal model but the y vector is augmented by rearranged terms for the level 2 effects to give

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

where we effectively rearrange $\mathbf{u}_j \sim MVN(0, \Omega_u)$ as $0 \sim MVN(-\mathbf{u}_j, \Omega_u)$ and then add these zeroes to create y^* .

Note that β^* is a vector containing β and all elements of \mathbf{u}_j .

This model is just a rearrangement of the multilevel model and so has the constraints built in.

MCMC approaches

The standard MCMC (Gibbs Sampling) approach is to include priors for the fixed effects and variance parameters and iterate around 4 steps of generating draws from conditional posterior distributions.

We can also think about MCMC algorithms that are analagous to the two ‘classical’ approaches described above.

It is possible (and fairly easy) to write a Metropolis Hastings algorithm to emulate the IGLS approach and we are investigating this and it’s extension to multilevel time series models.

Here we are going to talk about structured MCMC (Sargent, Hodges and Carlin 2000) and its use on multilevel models which has links with the HGLM approach.

Structured MCMC

The term structured MCMC or SMCMC was coined in a paper by Sargent, Hodges and Carlin (2000) and describes a ‘general method for Bayesian computing in richly-parameterized models, based on a blocked hybrid of the Gibbs sampling and Metropolis Hastings algorithms’.

The basic premise for normal response models is to be able to write the model in multivariate normal form:

$$Y = X\boldsymbol{\theta} + E, \text{cov}(E) = \Lambda$$

Then

$$\boldsymbol{\theta}|Y, X, \Lambda \sim MVN \left((X^T \Lambda^{-1} X)^{-1} X^T \Lambda^{-1} Y, (X^T \Lambda^{-1} X)^{-1} \right)$$

The method updates all fixed and random effects in one step and hence improves mixing of the MCMC algorithm but relies on ‘nice’ forms for the large matrices that need inverting.

Variance components example

In this talk we will consider the tutorial example from the MLwiN manual that consists of data on 4059 (N) students in 65 (J) schools. The response of interest is (normalised) exam score at 16.

Let y_{ij} be the mark for pupil i in school j then perhaps the simplest model we could fit is

$$y_{ij} = \beta_0 + u_j + e_{ij}, \quad u_j \sim N(0, \sigma_u^2), \quad e_{ij} \sim N(0, \sigma_e^2)$$

Here we have an estimated average mark, β_0 , school residuals u_j with variance σ_u^2 and pupil level residuals, e_{ij} with variance σ_e^2 .

For a Bayesian model we will use ‘diffuse’ priors as follows:

$$p(\beta_0) \propto 1, \quad p(1/\sigma_u^2) \sim \Gamma(\epsilon, \epsilon), \quad p(1/\sigma_e^2) \sim \Gamma(\epsilon, \epsilon)$$

Standard Gibbs sampling algorithm for VC model

In this example with conjugate priors we have full conditionals that can be easily evaluated and hence used in a Gibbs sampling algorithm as follows:

Step 1: $\beta_0 \sim N(\hat{\beta}_0, \hat{D}_0)$ where $\hat{D}_0 = \frac{\sigma_e^2}{N}$ and $\hat{\beta}_0 = \frac{\sum_{i,j}(y_{ij} - u_j)}{N}$.

Step 2: $u_j \sim N(\hat{u}_j, \hat{D}_j)$ where $\hat{D}_j = (\frac{n_j}{\sigma_e^2} + \frac{1}{\sigma_u^2})^{-1}$ and

$$\hat{u}_j = \frac{\hat{D}_j}{\sigma_e^2} \sum_{i=1}^{n_j} (y_{ij} - \beta_0)$$

Step 3: $1/\sigma_u^2 \sim \Gamma(c_u, d_u)$ gives $c_u = J/2 + \epsilon$ and $d_u = \sum_{j=1}^J \frac{u_j^2}{2} + \epsilon$.

Step 4: $1/\sigma_e^2 \sim \Gamma(c_e, d_e)$ gives $c_e = N/2 + \epsilon$ and $d_e = \sum_{i,j} \frac{e_{ij}^2}{2} + \epsilon$.

where $e_{ij} = y_{ij} - \beta_0 - u_j$.

SMCMC algorithm for VC Model

Firstly steps 3 and 4 will be as in the standard algorithm. The main difference is the combination of the steps for β_0 and u_j into one 66 dimensional normal update.

We start by reparameterising the model into a centred formulation by letting $u_j^* = \beta_0 + u_j$ and so we have:

$$y_{ij} = u_j^* + e_{ij}, u_j^* \sim N(\beta, \sigma_u^2), e_{ij} \sim N(0, \sigma_e^2)$$

Then reconstructing the random effects statement as $0 \sim N(\beta - u_j^*, \sigma_u^2)$ we can construct our model in the form

$$Y = X\boldsymbol{\theta} + E, \text{cov}(E) = \Lambda$$

Here we have

$$Y = \begin{pmatrix} y \\ 0_J \end{pmatrix} = \left[\begin{array}{ccc|c} 1_{n_1} & 0 & 0 & 0_N \\ 0 & \ddots & 0 & \\ 0 & 0 & 1_{n_J} & \\ \hline & -I_J & & 1_J \end{array} \right] \begin{pmatrix} u_1^* \\ \vdots \\ u_J^* \\ \beta_0 \end{pmatrix} + \begin{pmatrix} e \\ \delta \end{pmatrix}$$

with

$$\Lambda = \begin{pmatrix} \sigma_e^2 I_N & 0 \\ 0 & \sigma_u^2 I_J \end{pmatrix}$$

MVN Step

Recall we now have:

$$\boldsymbol{\theta} | Y, X, \Lambda \sim \text{MVN} \left((X^T \Lambda^{-1} X)^{-1} X^T \Lambda^{-1} Y, (X^T \Lambda^{-1} X)^{-1} \right)$$

where

$$X^T \Lambda^{-1} X = \left[\begin{array}{ccc|c} 1/\sigma_u^2 + n_1/\sigma_e^2 & 0 & 0 & -1/\sigma_u^2 \\ 0 & \ddots & 0 & \vdots \\ 0 & 0 & 1/\sigma_u^2 + n_J/\sigma_e^2 & -1/\sigma_u^2 \\ \hline -1/\sigma_u^2 & \dots & -1/\sigma_u^2 & J/\sigma_u^2 \end{array} \right]$$

Note for now we will simply invert this 66×66 matrix by standard matrix inversion before looking at speeding up this operation as this matrix is a partition matrix.

Results of the 2 methods

Both methods were run for 50,000 iterations following a burnin of 500 iterations. The table below gives point estimates and effective sample sizes (ESS) for the fixed effects and variances along with some ESS for the residuals (Gibbs on left, SMCMC on right):

| Parameter | Estimate (SE) | ESS | Estimate (SE) | ESS |
|--------------|----------------|-------|----------------|-------|
| β_0 | -0.012 (0.056) | 1.9k | -0.014 (0.055) | 52.0k |
| σ_u^2 | 0.177 (0.036) | 32.4k | 0.177 (0.036) | 34.3k |
| σ_e^2 | 0.848 (0.019) | 49.7k | 0.848 (0.019) | 47.9k |
| u_1 | 0.479 (0.117) | 11.9k | 0.482 (0.116) | 48.2k |
| u_{14} | 0.026 (0.084) | 4.6k | 0.029 (0.084) | 50.1k |
| u_{48} | -0.117 (0.353) | 48.9k | -0.118 (0.351) | 50.0k |
| u_{65} | -0.280 (0.113) | 10.8k | -0.278 (0.114) | 49.6k |

Results and Matrix speedups

The previous results took 28s for Gibbs and 1m 54s for SMCMC.

We can see that both β_0 and the u have low ESS for the standard algorithm but are effectively independently sampled in SMCMC.

Note that we can speed up SMCMC by using the following matrix inversion result when inverting $X^T \Lambda^{-1} X$:

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} -A^{-1}B \\ I \end{bmatrix} \begin{bmatrix} C - B^T A^{-1}B \end{bmatrix}^{-1} \begin{bmatrix} -B^T A^{-1} & I \end{bmatrix}$$

A here is diagonal and so easy to invert and using this results in SMCMC taking 1m 3s only.

Hierarchical centering

SMCMC improves things due to removing correlation by updating parameters together in a multivariate step. An alternative approach is to use hierarchical centering and then a standard univariate updating algorithm:

$$y_{ij} = u_j^* + e_{ij}, u_j^* \sim N(\beta, \sigma_u^2), e_{ij} \sim N(0, \sigma_e^2)$$

Implementing this in WinBUGS results in 50k iterations taking 47s and the ESS values for β_0 is 40.9k and for the residuals we have nearly as good ESS's (45.8k - 50k) as for SMCMC so here centering seems also to be a good approach and possibly preferable to SMCMC on speed grounds.

Random slopes regression model

Here we introduce one predictor, a reading test at age 11 (x) and allow its coefficient to vary across schools.

$$y_{ij} = \beta_0 + \beta_1 x_{ij} + u_{0j} + u_{1j} x_{ij} + e_{ij}$$

$$\mathbf{u}_j \sim MVN(0, \Omega_u), e_{ij} \sim N(0, \sigma_e^2)$$

where $\mathbf{u}_j = (u_{0j}, u_{1j})^T$ and Ω_U is the variance/covariance matrix at the school level which consists of the following terms:

$$var(u_{0j}) = \sigma_{u0}^2, var(u_{1j}) = \sigma_{u1}^2 \text{ and } cov(u_{0j}, u_{1j}) = \sigma_{u01}.$$

For a Bayesian model we will use ‘diffuse’ priors as follows:

$$p(\beta_0) \propto 1, p(\beta_1) \propto 1, p(\Omega_u) \sim IWishart(\nu_p, S_p), p(1/\sigma_e^2) \sim \Gamma(\epsilon, \epsilon)$$

Standard Gibbs sampling algorithm for RSR model

In this example with conjugate priors we have full conditionals that can be easily evaluated and hence used in a Gibbs sampling algorithm as follows:

Step 1: $\boldsymbol{\beta} \sim MVN(\hat{\boldsymbol{\beta}}, \hat{\Sigma}_{\boldsymbol{\beta}})$ gives $\hat{\Sigma}_{\boldsymbol{\beta}} = \sigma_e^2 (X^T X)^{-1}$ and $\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T d$ where $d_{ij} = y_{ij} - \mathbf{X}_{ij}^T \mathbf{u}_j$.

Step 2: $\mathbf{u}_j \sim MVN(\hat{\mathbf{u}}_j, \hat{\Sigma}_j)$ where $\hat{\Sigma}_j = \left[\sum_{i=1}^{n_j} \frac{\mathbf{x}_{ij}^T \mathbf{x}_{ij}}{\sigma_e^2} + \Omega_u^{-1} \right]^{-1}$

and $\hat{\mathbf{u}}_j = \frac{\hat{\Sigma}_j}{\sigma_e^2} \sum_{i=1}^{n_j} \mathbf{X}_{ij}^T (y_{ij} - \mathbf{X}_{ij}^T \boldsymbol{\beta})$.

Step 3: $\Omega_u^{-1} \sim Wishart_2(\nu_u, S_u)$ gives $\nu_u = J + \nu_p$ and $S_u = (\sum_{j=1}^J \mathbf{u}_j \mathbf{u}_j^T + S_p^{-1})^{-1}$.

Step 4: $1/\sigma_e^2 \sim \Gamma(c_e, d_e)$ gives $c_e = N/2 + a_e$ and $d_e = \sum_{i,j} \frac{e_{ij}^2}{2} + b_e$ where $e_{ij} = y_{ij} - \mathbf{X}_{ij}^T \boldsymbol{\beta} - \mathbf{X}_{ij}^T \mathbf{u}_j$.

SMCMC algorithm for RSR model

Again steps 3 and 4 are as for the standard algorithm and we will have one step to replace steps 1 and 2 with a 132 dimensional normal update.

We start by reparameterising the model into a centred formulation by letting $u_{0j}^* = \beta_0 + u_{0j}$ and $u_{1j}^* = \beta_1 + u_{1j}$ so we have:

$$y_{ij} = u_{0j}^* + x_{ij}u_{1j}^* + e_{ij}, \mathbf{u}_j^* \sim MVN(\boldsymbol{\beta}, \Omega_u), e_{ij} \sim N(0, \sigma_e^2)$$

Then reconstructing the random effects statements as previously we can construct our model in the form

$$Y = X\boldsymbol{\theta} + E, cov(E) = \Lambda$$

Here we have

$$Y = \begin{pmatrix} y \\ 0_{2J} \end{pmatrix} = \left[\begin{array}{ccccc|cc} 1_{n_1} & x_{.1} & 0 & 0 & 0 & & \\ 0 & 0 & \ddots & 0 & 0 & 0_N & 0_N \\ 0 & 0 & 0 & 1_{n_J} & x_{.J} & & \\ \hline & -I_2 & 0 & 0 & 0 & I_2 & \\ & & \ddots & 0 & 0 & \vdots & \\ 0 & 0 & & & & I_2 & \\ 0 & 0 & 0 & & -I_2 & & \end{array} \right] \begin{pmatrix} u_{01}^* \\ u_{11}^* \\ \vdots \\ u_{0J}^* \\ u_{1J}^* \\ \beta_0 \\ \beta_1 \end{pmatrix} + \begin{pmatrix} e \\ \delta \end{pmatrix}$$

with

$$\Lambda = \begin{pmatrix} \sigma_e^2 I_N & 0 \\ 0 & \text{diag}_J(\Omega_u) \end{pmatrix}$$

MVN Step

Recall we now have:

$$\boldsymbol{\theta} | Y, X, \Lambda \sim \text{MVN} \left((X^T \Lambda^{-1} X)^{-1} X^T \Lambda^{-1} Y, (X^T \Lambda^{-1} X)^{-1} \right)$$

where

$$X^T \Lambda^{-1} X = \left[\begin{array}{ccc|c} A_1 & 0 & 0 & -\Omega_u^{-1} \\ 0 & \ddots & 0 & \vdots \\ 0 & 0 & A_J & -\Omega_u^{-1} \\ \hline -\Omega_u^{-1} & \dots & -\Omega_u^{-1} & J\Omega_u^{-1} \end{array} \right]$$

where $A_i = \Omega_u^{-1} + \frac{X_i^T X_i}{\sigma_e^2}$. Note that this makes the top left partition block diagonal and hence the matrix speed up result can be used.

Results

We will compare 3 methods again - the standard Gibbs sampling algorithm, the SMC MC algorithm and hierarchical centering. All 3 methods were run for 50,000 iterations following a 500 burnin.

The standard algorithm takes 1m 18s in MLwiN, the hierarchical centering algorithm takes 2m 5s in WinBUGS whilst the SMC MC algorithm takes 8m 19s before speeding up and 2m 44s afterwards.

All three methods give similar point estimates for 50,000 iterations and overleaf we compare the ESS for the methods.

ESS comparison

The table below gives effective sample sizes (ESS) for the 3 methods.

| Parameter | standard | SMCMC | HC |
|-----------------|----------|-------|-------|
| β_0 | 2.3k | 49.5k | 23.1k |
| β_1 | 8.3k | 48.1k | 13.9k |
| σ_{u0}^2 | 28.0k | 30.9k | 30.2k |
| σ_{u1}^2 | 10.3k | 10.5k | 9.7k |
| σ_{u01} | 17.2k | 17.8k | 17.2k |
| σ_e^2 | 47.6k | 45.9k | 45.8k |
| $u_{1,0}$ | 15.9k | 49.5k | 41.6k |
| $u_{1,1}$ | 32.4k | 43.3k | 38.9k |
| $u_{14,0}$ | 6.7k | 48.5k | 34.7k |
| $u_{14,1}$ | 29.8k | 45.2k | 38.0k |
| $u_{48,0}$ | 49.8k | 48.2k | 50.6k |
| $u_{48,1}$ | 48.4k | 50.3k | 50.4k |
| $u_{63,0}$ | 29.4k | 45.9k | 42.4k |
| $u_{63,1}$ | 30.4k | 42.6k | 32.2k |

General 2 level model

Let us consider a general 2 level model with J level 2 units and in total N level 1 units. Let there be p fixed effects of which r are also random and hence $f = p - r$ are fixed only.

We will use a centred formulation and so

$$y_{ij} = X_{ij}\mathbf{u}_j^* + X_{Fij}\boldsymbol{\beta}_f + e_{ij}$$
$$\mathbf{u}_j^* \sim MVN(\boldsymbol{\beta}_r, \Omega_u), e_{ij} \sim N(0, \sigma_e^2)$$

Note this class does not include predictors that are random only or complex level 1 variation but these can also be included with little added work.

Here we have

$$Y = \begin{pmatrix} y \\ 0_{rJ} \end{pmatrix} = \left[\begin{array}{cccc|c|c} X_1 & 0 & 0 & 0 & 0 & X_{F1} \\ 0 & X_2 & 0 & 0 & 0 & X_{F2} \\ & & \ddots & & \vdots & \vdots \\ 0 & 0 & 0 & X_J & 0 & X_{FJ} \\ \hline -I & 0 & 0 & 0 & I & 0 \\ 0 & -I & 0 & 0 & I & 0 \\ & & \ddots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \vdots & \vdots \\ 0 & 0 & 0 & -I & I & 0 \end{array} \right] \begin{pmatrix} \mathbf{u}_1^* \\ \mathbf{u}_2^* \\ \vdots \\ \mathbf{u}_J^* \\ \beta_r \\ \beta_f \end{pmatrix} + \begin{pmatrix} e \\ \delta \end{pmatrix}$$

with

$$\Lambda = \begin{pmatrix} \sigma_e^2 I_N & 0 \\ 0 & \text{diag}_J(\Omega_u) \end{pmatrix}$$

MVN Step

Recall we now have:

$$\boldsymbol{\theta}|Y, X, \Lambda \sim MVN \left((X^T \Lambda^{-1} X)^{-1} X^T \Lambda^{-1} Y, (X^T \Lambda^{-1} X)^{-1} \right)$$

where

$$X^T \Lambda^{-1} X = \left[\begin{array}{ccc|c|c} A_1 & 0 & 0 & -\Omega_u^{-1} & \frac{X_1^T X_F}{\sigma_e^2} \\ 0 & \ddots & 0 & \vdots & \vdots \\ 0 & 0 & A_J & -\Omega_u^{-1} & \frac{X_J^T X_F}{\sigma_e^2} \\ \hline -\Omega_u^{-1} & \dots & -\Omega_u^{-1} & J\Omega_u^{-1} & 0 \\ \hline \frac{X_F^T X_1}{\sigma_e^2} & \dots & \frac{X_F^T X_J}{\sigma_e^2} & 0 & \frac{X_F^T X_F}{\sigma_e^2} \end{array} \right]$$

where $A_i = \Omega_u^{-1} + \frac{X_i^T X_i}{\sigma_e^2}$. Once again the top left partition is block diagonal and the matrix speed up result can be used.

Comments and Conclusions

In the two example models fit here there appears to be some merit to the SMCMC methodology. Clearly the ESS results are a great improvement from the standard Gibbs algorithm in both examples however less impressive is the fact that we get a good improvement simply by reparameterising the model via hierarchical centering.

What is less impressive is what happens when J the number of level 2 units increases when things begin to slow down and there are even sometimes numerical problems with calculating the MVN posterior distribution.

It should also be noted that the models here run very quickly in all methods so speed ups are less important.

Extension 1: 3 level models

The methodology can be extended to 3 (or more) level models without much difficulty and also to cross-classified models.

Obviously the dimension of the θ vector is now of length the number of fixed effects and the number of residuals at all higher levels (p say).

Finding $(X^T \Lambda^{-1} X)^{-1}$ will involve inverting a $p \times p$ matrix but for nested models ordering the parameters in θ carefully can result in submatrices that have the partition form of the 2 level model and the matrix speed up trick can be used recursively. Note we haven't yet programmed this.

For cross-classified models if we place the residuals for the classification with most units first we will have a block-diagonal submatrix which may speed things up slightly.

Extension 2: Binary responses

As partially described in Sargent, Hodges and Carlin (2000) for non-normal responses one can linearise the problem and construct an approximate MVN model. This model can then be used to generate proposals as part of the Metropolis Hastings sampling algorithm.

We are currently investigating using the quasi-likelihood estimates from MLwiN for binary response models to construct such an MH algorithm. Preliminary work using R hasn't as yet had much success in particular in getting a good acceptance rate for the MVN step. However this method may yet prove effective.

Reference: Sargent D.J., Hodges J.S., and Carlin B.P. (2000).
Structured Markov chain Monte Carlo. *Journal Of Computational And Graphical Statistics*. 9 (2): 217-234