Is it feasible to use MCMC to estimate a GLMM with a complex correlation structure in an R-environment?

Submitted on: 6 September 2005.

Submitted By: Md. Moudud Alam

Supervisor: Kenneth Carling

This paper has been prepared for the partial fulfilment (a D-level essay consists of 10 Swedish credits) for the Advanced Courses in Statistics at Dalarna University.

Abstract

This paper is motivated by the desire to estimate a Credit-risk model which allows a complex dependency structure between companies. Previously, simple versions of such models have been estimated in SAS environment with the use of Penalized Quasi Likelihood approach (using %glimmix macro). I discuss the advantage of using Markov chain Monte Carlo to small data set but most of the data used to establish Credit-risk models are large and in those situations the MCMC implementation is too time-consuming.

Contents

1. Introduction

A large number of sophisticated credit risk models have been developed during the last decade by several Banking institutions with a desire to quantify the risk value and to develope a well-equipped risk management system [1]. In spite of the all efforts none of the models has got absolute supremacy due to their sensitivity to parameter estimates [2]. Quoting the reference of Jacobson, Linde´ and Roszbach [3], Carling et.al. [4] claims that empirical evidence [3] with data from Swedish banks agrees with the above theoretical statement formulated by Gordy [2].

The limitations of the existing credit risk models as stated above leads to rethinking about those model structures. In this connection Giesecke [5] suggests incorporation of correlation in the loans between the companies. Carling et.al. [4] develops a model which introduces interdependence. Their [4] empirical results using real data from two Swedish Banks support correlation within companies.

However, while interpreting the results of Carling et.al. [4] it is found very important to specify a form of correlation between the farms within and between different sectors (clusters). In this connection one may consider the input-output matrix to be a source correlation structure of the defaults of the loans between and within sectors.

The input-output matrix[1] the is produced and maintained by Statistiska Cetralbyrån (Statistics Sweden) as part of Integrated National Accounts, publication NR 10 SM 0301. It consists of 14 tables, Supply table (1-7) and Output table (8-14). In particular we are interested in the supply tables that document division of total output of each individual industry into market output for the use of their own and other industries. Through these tables (supply tables) we get the information how a particular industry's total output in a particular time is going to different other industries for their use. Therefore, when we see a particular industry A (say) sells an amount x of its total output to another industry B, then it gives us an idea that if industry B is in trouble then it implies the revenue of amount x of company A is under risk. So, this risk might have a proportional effect on the propensity to default of the companies in industry (A).

[1] Detail input-output matrix can be obtained online from http:// www.scb.se /statistik/ NR/NR0102/2003A01/ InputOutputTables1995o2000.xls

Therefore, this paper aims to incorporate the Swedish inter industry input-output matrix (with proposed modification) as a proxy to the inter industry correlation matrix of defaults and then wish to find an estimate of the model parameters within the scope of existing computing facilities and to examine the methods applicability to real data in terms of computing intensity.

Therefore the objectives of this paper can be stated as

1) To formulate a model for credit risk that incorporates within and between cluster defaults correlation using the inter industry input-output matrix as guiding correlation structure.
2) Find a suitable technique for estimation of the model parameters
3) Study the feasibility of the proposed estimation technique in terms of computational speed.

2. Credit Risk Model with Interdependence

The credit risk model described in Carling et.al. [4] is a binary regression model with random effect and first order autocorrelation. The mathematical form of the model can be presented as

$$y_i^\tau = x_i(\tau)\beta + \varepsilon_i^\tau + \varepsilon_k^\tau + \delta_k \varepsilon_k^{\tau-1} \qquad (1)$$

Where, y_i^τ is the propensity of the *i*:th loan in the portfolio at calendar year τ to default in a predetermined time-unit ahead of τ, $x_i(\tau)$ are the covariates associated to the *i*:th loan at time τ,β is a vector of regression coefficients, ε_i^τ is the loan specific shock, ε_k^τ is the cluster specific shocks, δ_k is dependency parameter and $\varepsilon_k^{\tau-1}$ is the cluster specific shock at time $\tau - 1$.

The special assumptions of model (1) are

i) Between cluster defaults are uncorrelated i.e. $f(\varepsilon_k^\tau, \varepsilon_{k+j}^\tau) = f(\varepsilon_k^\tau) f(\varepsilon_{k+j}^\tau)$ for all $j \neq 0$

ii) The dependence between any two loans in the same cluster can be expressed in terms of correlation measure ρ_k which should be positive.

Therefore from the above model it is evident that the dependence of defaults between clusters is considered to be zero. But in reality they might not be so. For example, the car industry buys lather and textile items for making the seats of the cars from leather and textile industries. So, there may be some leather and textile firms who sell their product mainly to car industry. Consequently, market depreciation in car industry may also affect the business of the related textile and lather firms. Therefore, we find a reasonable logic behind between farm interdependency of loan default. On this ground this research aims at including the inter cluster dependency to model (1) and then using the input/output matrix as a structure of inter industry dependency. It also aims to check if the adoption of the proposed correlation structure increases the strength of the model. Now, considering a between cluster dependency the model becomes

$$y_i^\tau = x_i(\tau)\beta + \varepsilon_i^\tau + \varepsilon_k^\tau + \delta_k \varepsilon_k^{\tau-1} + \sum_{\nu_k} \delta_{i\nu_k} \varepsilon_{\nu_k}^\tau \qquad (2)$$

Where, υ_k denotes the other industries but k whose loan defaults are correlated with *k*:th cluster.

3. Estimation of the Parameters

Since we have no access to y_i^τ other than the information whether a loan is default or not, we define an indicator variable D_i such that:

$$D_i = \begin{cases} 1 & \textit{if the loan is default} \\ 0 & \textit{otherwise} \end{cases}$$

Then the model (2) becomes a model with binary response variable D_i. So, the possible way of dealing with the model is trough a logistic or probit model technique with random effect. The model then falls into the class of generalized linear mixed model. In matrix notation (under the framework of generalized linear model [12]) model (2) can be expressed as

$$\eta = \mathrm{X}\beta + \mathrm{ZU} \qquad (3)$$
$$\mathrm{E}(\mathrm{Y} \mid \mathrm{X}, \mathrm{U}) = \mathrm{g}(\mu)$$

Where, Y is a vector of response (0 and 1 in this case), X is a design matrix related to covariates and fixed intercept term, U is a vector of random effects, Z is also a design matrix

related to random effects, η represents the linear predictor and g(.) is a link function having the following options

i) Logistic (Canonical for this case)

$$g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$$

ii) Probit

$$g(\mu) = \Phi^{-1}(\mu)$$

Where $\Phi(.)$ is a standard normal CDF

iii) Complementary Log-log

$$g(\mu) = \log(-\log(1-\mu))$$

Under the formal assumptions of general linear model the BLUE of β and BLUP of U would be reasonably straight forward when we know the variance of U (say G and assume it is the input output matrix).

However, under the framework of generalized linear model the model estimates and their interpretation is comparatively difficult. Some very popular ways of estimating the parameters are through the Maximum Likelihood[2] method or Penalized Quasi Likelihood (PQL)[3] method. Since none of the two packages MASS (function glmPQL()) or glmmML (function glmmML()) in R allow us to use such complex correlation structure as it appears in this study we can not carry out our estimation directly with R. One alternative way could iterate a linear mixed model of the same form to get a generalized linear mixed model as it is done in PQL technique. But the PQL approach has a major flaw, it is biased. The full ML estimate itself is not unbiased but the unbiasedness would not be a serious concern if the amount of expected bias is very small. However, the PQL bias does not tend to zero even though the sample size tends to infinity which implies they are not consistent [11].

The R-package, nlme [9] provides flexible estimate for linear mixed models. There is another package Schall's[7] reglm for R and S+ (especially for S+) but it is not chosen since it is not

[2] Boström's [6] glmmML package for R uses this technique. However, it is useful for estimating the parameters of the marginal models only.
[3] Ripley's MASS package that comes with R base library contains glmPQL() function that uses PQL method for estimating the fixed and random effect coefficients. glmPQL is useful for conditional models.

efficient for R environment. The NLINMIX[4] macro [10] in SAS may be another alternative but it is too slow and it allows us only to use some specified correlation structures [10] for the random intercept term. Now the only option left is, either to write a new programme for R to estimate the coefficients or to compromise with the model structure to get it suitable to be handled with existing packages. Finally this paper aims at writing a programme in R to estimate the model.

Now, we assume the correlation in the random effect appears since each of the industry sales some portion of their product to the other industries. We assume that, at each time an independent random effect (γ_i) hits each industry. Now, a portion of this random effect of any particular industry also hits all other industries that sell their product to that industry. Again, how much an industry sales to other industry is documented in the Swedish inter industry Input-output matrix (as stated in section 1). So, the actual random effect that hits any industry can be expressed as a linear combination of the independent random effect where the input output matrix determines the structure of linear dependency. However, time lagged dependency among the random effects has been neglected. It can be regarded as a compromise with the initial model but it is reasonable since no such significant effect of time was found in the previous study conducted by Carling et.al.[4].

So, the random effect part of the model (3) can be written as, $U=A\Gamma$. Where A is an input-output Matrix and Γ is a vector of independent random effects $(\gamma_1 , \gamma_2 , \ldots \gamma_k)'$. For example of an input-output matrix let us assume we have only three industries, **A**, **B** and **C**. Then their input-output matrix can be shown as Table 1.

Table 1. A hypothetical input-output matrix

Industry	A	B	C
A	1.2	0.2	0.3
B	0.2	1.7	0.2
C	0.3	0.2	0.9

The rows of table 1 shows how much (may be in million dollars) a particular industry sales to other companies of other industries. For example value 1.2 in the first row and first column

[4] SAS GLIMMIX macro could be another alternative but NLINMIX provides better flexibility than GLIMMIX.

indicated industry **A** sells 1.2 million within itself on the other hand value 0.2 in the second column in the same row indicates the industry **A** sells 0.2 million to industry **B**.

Now, let us consider the independent industry specific random effect is normally distributed with mean 0 and variance covariance matrix $\sigma^2 I$. Therefore from the properties of Multivariate Normal distribution [13] we can conclude that U follows a Multivariate Normal Distribution with mean vector equals 0 and the variance-covariance matrix of U can be written as

$$Var(U) = \sigma^2 AA' = G \text{ (say)} \qquad (4)$$

Further more, with reference to the findings of Carling et.al.[4] we need not worry about time wise auto correlation among the random effects.

Still, we can not use any existing R package to estimate the proposed model. Therefore to find an ML estimate of the model parameter we have to proceed through analytical calculation of the Likelihood function. Now, the Likelihood function for this model (3) can be presented as:

$$L = \int\int...\int \left\{\prod f_{Y_i}(y_i \mid U, G)\right\} f_U(U \mid G) dU \qquad (5)$$

Further simplification of (5) is not so easy since it involves a non-tractable multi-dimensional integral. To find ML estimate Marcov Chain Monte Carlo (MCMC) technique may be an attractive option since the multidimensional integral can be computed via MCMC simulation. But the main drawback of the MCMC methods is that they are very slow [14]. However, as we have to estimate only one extra parameter in addition to the regression coefficients, MCMC may be reasonable in this case. Therefore, this paper aims to use Marcov Chain Monte Carlo Newton-Rapson (MCMC-NR)[11] technique.

3.1 MCMC for Model Estimation

Under this technique we draw random sample form f(U|Y) using MCMC Metropolis algorithm with candidate distribution f(u), as suggested by McCulloch and Searle [11], and acceptance probability

$$\alpha(u^*,u)=\min\left\{1,\frac{\prod_{i=1}^{n} f_{Yi|U}(y_i \mid u^*,\beta,G)}{\prod_{i=1}^{n} f_{Yi|U}(y_i \mid u\ ,\beta,G)}\right\} \qquad (6)$$

Since, given U the distribution of Y is very straight forward (Bernoulli distribution in this case) so the simulation is still very reasonable. Then we estimate the coefficients as

$$\beta^{(m+1)}=\beta^{m}+(X'E[W \mid y]X)^{-1}X'E[W\Delta(Y-\mu) \mid y] \qquad (7)$$

Where, $W=\left\{{}_d\left[v(\mu)g_\mu^2(\mu)\right]^2\right\}$, $\Delta=\left\{{}_d\, g_\mu(\mu)\right\}$, and g(.) , and v(.) represents the link and variance functions[12] respectively and the subscript in g(.) denotes differentiation with respect to that parameter.

The expectations in the right hand side of (7) are calculated through MCMC simulation. At the same time σ_u is also estimated through the MCMC expectation (find the computational details R codes in the appendix A) i.e. to maximize $E\left[\frac{\delta \ln f(U \mid G)}{\delta G} \mid y\right]$ through MCMC expectation which implies to solve the equation

$$\frac{1}{N}\sum_{i=1}^{N}\frac{\delta}{\delta\sigma}\ln f(U^{(i)} \mid G)=0$$

$$\Rightarrow \frac{1}{N}\sum_{i=1}^{N}\frac{\delta}{\delta\sigma}\ln\left(\frac{1}{(2\pi)^{\frac{k}{2}} \mid \sigma^2 AA' \mid^{\frac{1}{2}}}Exp\left[-\frac{1}{2}U'^{(i)}(\sigma^2 AA')^{-1}U^{(i)}\right]\right)=0$$

$$\Rightarrow \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\delta}{\delta\sigma}(-\frac{k}{2}\log(2\pi))-\frac{\delta}{\delta\sigma}(k\log\sigma)-\frac{\delta}{\delta\sigma}(-\frac{1}{2}\log \mid AA' \mid)-\frac{\delta}{\delta\sigma}(\frac{1}{2\sigma^2}U'^{(i)}(AA')^{-1}U)\right)=0$$

$$\Rightarrow \frac{1}{N}\sum_{i=1}^{N}\left(-\frac{k}{\sigma}+\frac{1}{\sigma^3}U'^{(i)}(AA')^{-1}U\right)=0$$

$$\Rightarrow \frac{1}{N\sigma^3}\sum_{i=1}^{N}U'^{(i)}(AA')^{-1}U=\frac{k}{\sigma}$$

$$\therefore \hat{\sigma}^2=\frac{1}{kN}\sum_{i=1}^{N}U'^{(i)}(AA')^{-1}U^{(i)} \qquad (8)$$

Where, *N* is the size of MCMC sample and *k* is the dimensionality of the square matrix *G*.

Now the total MCMC-NR method can be summarised as

i) Set m (number of iteration)=0, $\beta = \beta^{(0)}$, $\sigma = \sigma^{(0)}$ which implies $G^{(0)} = (\sigma^{(0)})^2(AA')$

ii) Draw MCMC sample form $f(U|Y)$ using $f(U)$ as candidate distribution. For doing this we have two options. Either we can draw all elements in $U^{(i)}$ at the same time or we can draw them element wise. The first option is quite straight forward however for the second option we start from any arbitrary point $(u_1^0, u_2^0, ..., u_k^0)'$ then we draw the next sample when the all other sample units are the same except for the first one i.e. draw $(u_1^1, u_2^0, ..., u_k^0 \mid y)'$. To do this we have to specify the full conditional distribution of $f(u_1 | u_2, u_3, \ldots, u_k)$. Since U follows Multivariate Normal (as per assumption) distribution the specification of the full conditional distribution of U is given by

$$f(U_1 = u_1 | U_2 = (u_2, u_3, \ldots, u_k)', G) \sim N(G_{12} G_{22}^{-1} U_2, G_{11} - G_{12} G_{22}^{-1} G_{21}) \qquad (9)$$

So, at each time draw a random sample from the univariate normal distribution above (9) when all the other values of u^i's are from the previous draw except for the current one. Then accept or reject a new sample (and stay in the old state) according to (6). Therefore if we accept all the new samples until i:th sample (for simplicity lets assume $i<k$) is drawn then the i:th sample consists actually of

$$U^{(i)} = (u_1^{(1)}, u_2^{(2)}, \ldots, u_i^{(i)}, u_{i+1}^{(0)}, \ldots, u_k^{(0)}).$$

iii) Calculate new estimates of $\beta^{(i)}$ and $\sigma^{(i)}$ according to (7) and (9) where the expectations are evaluated over Monte Carlo samples i.e.

$$E(W \mid Y) = \frac{1}{N}\sum_{i=1}^{N} W^{(i)} \text{ and } E(W\Delta(Y-\mu) \mid Y) = \frac{1}{N}\sum_{i=1}^{N} W^{(i)}\Delta^{(i)}(Y-\mu^{(i)})$$

iv) Stop if convergence is achieved otherwise set $m=m+1$ and go to step (ii).

3.2 Some Practical Points for MCMC

There are some practical points on which there is no exact suggestion in any standard text book like McCulloch and Searle [11] or in any other references. For example, what should be the exact size of the Monte Carlo sample, what should be the stopping criteria, what should be the starting value for the parameter at the initial iteration and so on? In stead we have to make our own judgement on these points or to pick some reasonable suggestion from some other references.

For the MCMC sample sizes, Booth and Hobert [14] reports McCulloch [15] has used a linear function of iteration to determine the sample size in particular N=50 for 1-19 iteration, N=200 for 20-39 iteration and N=5000 for iteration greater than 40. Though Booth and Hobert (*op.cit.*) used EM algorithm, they have suggested a different way of determining the MCMC sample size. It should be noted that the size of MCMC sample has a remarkable effect on the timing to convergence. For example, Kuk [16] has used a fixed number of MCMC sample and in one of his experiments the algorithm required 300 iterations to converge.

Since, we are using McCulloch [15]'s algorithm, we have used a similar way of determining MCMC sample size however with some more samples in each iteration. For this study N has been chosen to be 500 for first 19 iteration, 2000 for iteration 20-39 and 10000 for each iteration after 39 (with some exceptions which will be described in the respective discussion).

Moreover, this paper has followed a suggestion of Searle et.al. [17] to the stopping criteria. They suggest taking two arbitrary constants (as small as possible), δ1 and δ2 and then express the convergence criteria as

$$\max_i \left(\frac{|\psi_i^{(r+1)} - \psi_i^{(r)}|}{|\psi_i^{(r)}| + \delta_1} \right) < \delta_2 \qquad (10)$$

Where ψ_i represents the parameters to be estimated including the variance of the random effect.

Searle et.al [17] suggest[5] using δ1=0.001 and δ2=0.0001 but others [14] comment δ_2 =0.0001 is too ambitious in MCMC estimation. Rather δ_2 between 0.002 and 0.005 may be a reasonable choice. Since we are not going to evaluate the unbiasedness or consistency of the estimates, δ_2=0.005 has been used for the experiments shown in this paper. It is worth noting that any lower value of δ_2 increases the execution time significantly and the programme sometimes fails to find converge within 100 iterations. Other options for convergence criteria are also available [17] however none of them (including the one I have used in this paper) are free from criticism.

Besides, to get rid of 0 and 1 in estimated Bernoulli proportions an error at the 9th decimal point has been considered for experiments with Probit and Complementary Log-log links. It is worth noting that the standard statistical packages like glmm [6] has also used some approximations for $|\hat{\eta}| > 6.5$. The same is true for Probit model. Although the normal probability curve covers the whole real line however, after some $k\sigma$ distance ($|k|>3$) from the mean, there does not exist any numerically computable probabilities.

Moreover, for simplicity a simulated and simple data set has been used to demonstrate the accuracy of the algorithm for simple binary models with a random intercept. With subject to available computer facility the algorithm can be applied for the actual data set.

4. Some Examples from Hypothetical Data

To study the performance of the MCMC algorithm in our present settings we have implemented the algorithm for several sets of hypothetical data with different number of clusters and different number of observations per cluster. Execution times have been recorded and trace of convergence has been plotted.

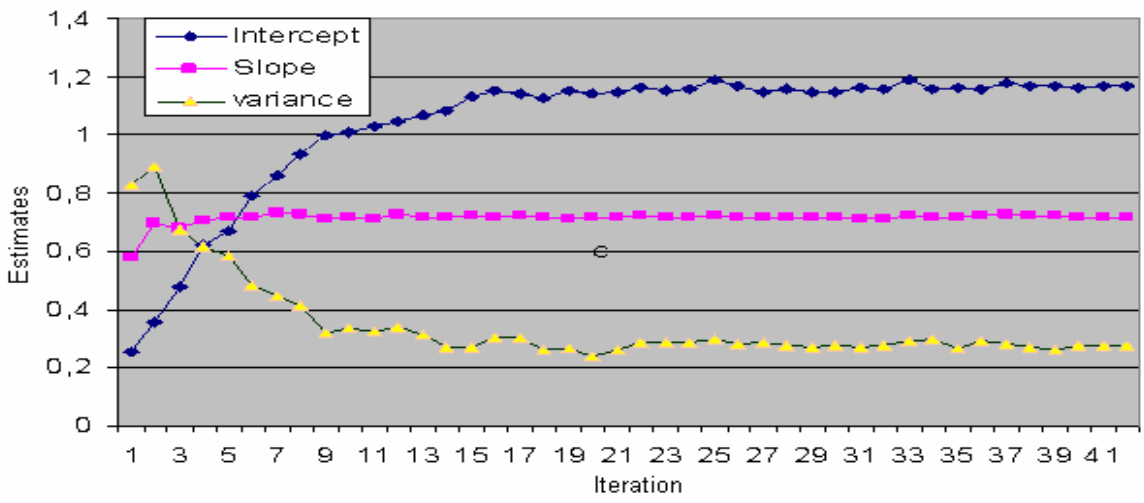
Regarding the execution time it should be mentioned that, it depends greatly on the choice of starting point and the number of observation per cluster. Where as, the effect of increasing the number of clusters is less pronounced. On the other hand, as Booth and Hobert [14] noted that

[5] It should be noted that Searle et.al [17] suggests the criteria for simple linear mixed models when the parameter estimates does not require MCMC sample. However, Booth and Hobert [14] has used the same in the context of generalized mixed models.

employment of a suitable automation might decrease the total execution time. For example, when we see the estimates are showing a sharp trend (after a certain number of iteration) then we can push them some more to the same direction. For example please see figure B.1 in the appendix where the estimates were pushed down after 40 iterations and then a convergence were obtained after the next 40 iterations.

On the other hand, as stated in section 3.2, a good choice of the size of MCMC sample plays an important role. Large sample size at the beginning is just waste of time whereas with small sample size the estimates show huge fluctuation (see figure-B.2). What is the optimal balance may be a topic of another research. Figure B.2 also reveals that a small sample size might show a huge fluctuation in the estimates over iterations even if they are very close to the true parameter estimates. In that particular experiment the sample size was reduced after 60 iteration then it was found that the estimates were showing a huge fluctuation but neither did they behave so just immediately before we reduce the MCMC sample nor after some iteration when the sample size was again increased and finally it converged very nearly to the point that was obtained at the 60th iteration.

Figure 1. Trace of Convergence for a Logistic Model with Random Intercept and One Covariate (number of cluster =3 and sample size per cluster = 33)



Selection of a good starting point for variance parameter (σ) matters a lot on convergence however, the slope parameter goes very quickly to the neighbourhood of the actual estimate (see Figure 1). From the above figure we see that the slope parameter reached the neighbourhood of the actual estimate within just five iterations while the others took about 20

iterations. In this regard, a start from the estimate of a restricted model may be a good choice especially for the variance parameter. However, starting from the restricted model does not reduce the convergence time so remarkably as it is revealed in figure 1 that the estimates reach to the neighbourhood of convergence within 20 iterations when the MCMC sample size is very small and therefore time required for this 20 iteration is negligible (see Table 2) with respect to the total time required.

Table 2. Time required (in second) and number of iteration for convergence for logit model with a fixed sample size (approximately 100) and one covariate

Size of MCMC Sample	No. of Cluster							
	3	4	5	6	7	8	9	10
	Time required per iteration (in seconds)							
500	3	3	3	3	3	3	3	3
2000	9.5	9.5	9.5	11	11	10.5	10.5	11
10000	45	45	47.5	50	53	53	53	58
Iter. req.	60	55	53	75	86	81	94	136*

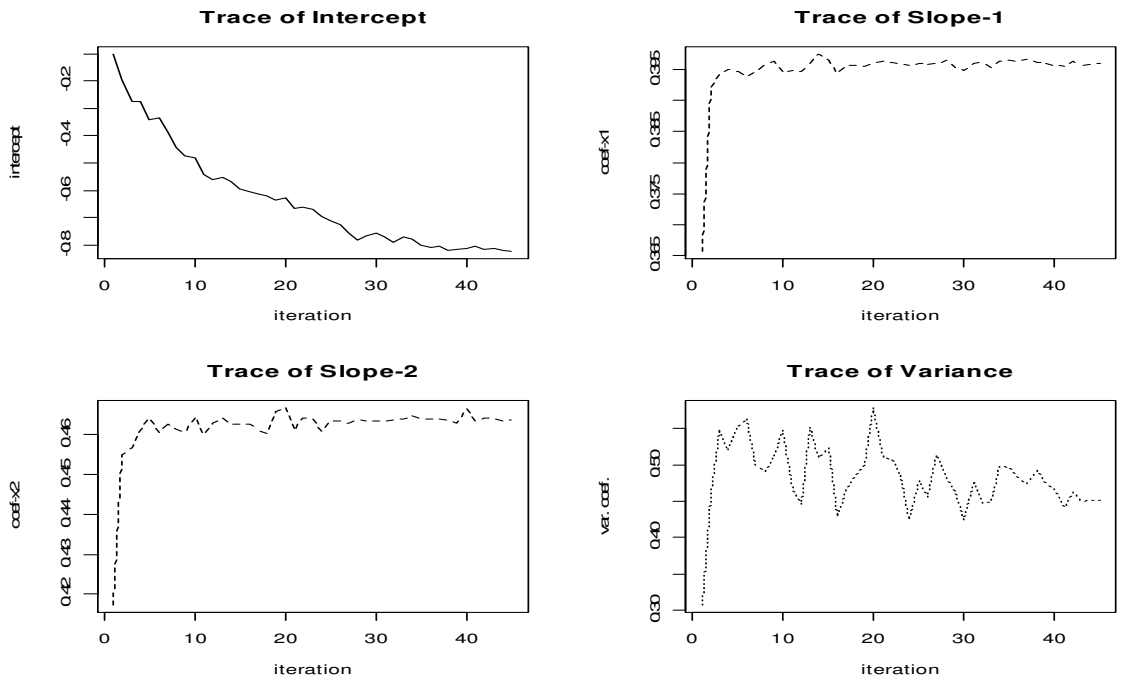
*Note: For n=10 the sample size was further increased after 100 iteration to 30000 and then a convergence was found (for this special case total time required 3hr. 24 minutes and 13 seconds).

The strange behaviour of the simulation is that the estimates continuously fluctuate even if it is very near to the true estimate (Figure 1, Figure 2, Figure A.1 and Figure A.2). However, a very bad starting point increases the convergence time a lot especially when the estimates do not reach at the neighbourhood of the true estimate within 20 iteration (see Figure 2). This is because after 20 iterations we increase the MCMC sample size (as stated in section 3.2) and therefore each iteration after 20 is very expensive (see Table 2) in terms of computational time.

Finally let us have a look how computationally feasible they are. Well, for a fixed sample size (around 100 for example) increasing the number of clusters does not matter a lot. Though the time required for a single iteration with a fixed number of MCMC sample does not even increase so dramatically when the number of cluster exceeds 5 however, they require a huge number of extra iteration before the estimate converges (see Table 2).

From the practical experience it has been found that when the number of cluster exceeds 6, pretty often the iterations do not converge with 10000 MCMC sample . Especially for 10 clusters 30000 MCMC sample were required to get the iteration converged and hence a huge computation time.

Figure 2. Trace of Convergence for Complementary Log-log Model with Two Covariates and a Random Intercept. No. of Cluster is 4 with 25 observations per Cluster



Now let us have a look at how the sample size affects the computation time. From a single experiment with 300 sample (and three clusters) for a logistic model it was found that for 500 MCMC sample it takes 19 seconds for each iteration, for 2000 sample it takes 85 seconds and for 10000 sample it takes 390 seconds for the same. The iteration converged after 84 iterations and it took totally 2 hours, 27 minutes and 23 seconds for convergence. So, it is very evident that increment in the sample size is highly expensive from the consideration of the computational time. Further experiment in this line therefore has not been conducted. For a details comparison for logistic model (up to 300 samples) please see appendix Table B.1.

The last concern regarding the computation time may be the use of different link function and increasing the number of covariates. As it is found in figure 1 and figure 2, the covariate specific parameters goes very quickly to the neighbourhood of convergence and they do not

fluctuate so much from iteration to iteration. Therefore, an additional number of covariate does not increase the convergence time a lot but the choice of link function does matter.

Table 3. Summary of computation time for 3 covariates, seven clusters (15 observations per cluster) and different link functions

Link	Logit	Probit	Complementary log-log
MCMC Sample Size	Time (in seconds)		
500	3.5	7	6.5
2000	12	32	28
10000	55	65	65
20000	108	129	116
Iteration required for Convergence	55 (32)	53 (29)	77 (66)

Note: values in parentheses show total time required in minutes.

Table 3 reveals that performance in logit link is faster followed by complementary log-log and probit. From the above table we see that with an MCMC-sample of size 500 the programme needs 3.5 seconds to perform each iteration with a logistic link whereas it takes almost double (7 and 6.5 seconds) time for the probit and complementary log-log model. Though the table shows probit model convergence with the lowest number of iteration however, it should be noted that this is just an example from one experiment and therefore it does not guarantee the faster convergence of probit model. On the contrary, the time required for each iteration has been observed over several occasions and that figure can be relied on. Moreover, with different starting point and intuitively pushing the estimates forth and back we can obtain a convergence with less number of iteration but in every case the time requirement for each iteration will not alter (provided that we conduct experiment with computers of same hard were configuration).

The fact behind faster performance in logit link is that it is the canonical link for binary regression and therefore it simplifies the calculation a lot. Moreover the link function transforms the value the most smoothly than the other two links. It is because π (binomial probability) does not become numerically indistinguishable from 0 or 1 (for which an approximation would be required in numerical calculation of likelihood function to get rid of

0 or infinity) unless η exceeds 665.1416. On the other hand the likelihood function does become numerically zero or infinity as $|\eta|$ exceeds 6.5 for complementary log-log and 5.1 for probit model.

Table 4. Comparison between the estimates of restricted and unrestricted model for a logistic model with 2 covariates, 3 clusters and 33 observations per cluster.

Parameters	Estimates		True value
	Unrestricted Model	Restricted Model	
Intercept	0.319	0.898	0.300
Slope-1	0.676	0.665	0.600
Slope-2	0.325	0.323	0.400

Finally, let us have a look on the effect of estimates when the random effects are not independent but we wrongly assume they are independent. Table 4 shows the respective difference in the parameter estimates between a restricted and an unrestricted model when the data was generated by the unrestricted model. From the above table we see that the slope estimates does not change a lot however the estimate of the intercept term differs a lot. Actually the restricted model highly overestimates the intercept term. Besides the most important information that we lose due to the use of restricted model is the correlation of between the random effects. For instance, for the above experiment, the restricted model gives an identical variance estimate 0.761 while it fails to provide any information on the heteroscadastisity and the correlation between the random effects. On the contrary the unrestricted model provides the estimate of the variance covariance matrix of the random effects as follows

$$\hat{G} = \begin{bmatrix} 1.645 & 0.670 & 0.708 \\ 0.670 & 3.311 & 0.608 \\ 0.702 & 0.608 & 0.985 \end{bmatrix} \text{ when the true value was } G = \begin{bmatrix} 1.931 & 0.787 & 0.824 \\ 0.787 & 3.653 & 0.713 \\ 0.824 & 0.713 & 1.156 \end{bmatrix}$$

So, it is very clear that if the random effects are correlated then it is not wise to estimate the model under the assumption of independent random effects. Under the violation of

dependency we might obtain unrealistic estimate of the model parameters as well al we loose information on the correlation structure between the random effects.

5. Concluding Discussion

Model estimation under the settings of generalized linear mixed models more or less depends on numerical computation mainly due to high dimensional integration involved in the specification of the likelihood function. To get rid of the analytically intractable integration problem either we follow PQL approach (for example [8]) which provides approximate inference or we adopt a numerical integration technique as Gauss-Hermit quadrant (for example [6]) or we bypass the integration via MCMC expectation.

The approximation based on PQL does not provide us with consistent estimates, Gauss-Hermit quadrant performs poorly when the likelihood is not properly centred and smoothed [11] and therefore in this paper I have tried to adopt an MCMC approach. Throughout the exercise it has been found that, although the MCMC approach has theoretically very good properties, they are time consuming even in the case the number of free parameter in the variance component is small. So, they are a good choice for small data-sets however, they are infeasible for large data-sets.

It should be mentioned that all the exercises of this paper have been conducted with R language so the speed and accuracy of the computation are subject to programming codes used for this paper. Using a low level language such as C or FORTRAN might reduce the execution time.

Considering the scope of this essay no further effort has been made to incorporate FORTRAN or C codes to estimate the target model. Considering the time requirement we do not recommend to carry out any experiment with large data with this technique completely in R environment.

However, from the experiments with simulated data it has been found that the technique works reasonably well with small data sets. The estimates it provides are pretty often closer to the true parameter value than that we obtain from PQL based estimates (as we have seen in

Table 4). Though much study has not been done in this line however, from small scale experiments (for example [14] and [15]) with independent random effects it has been found that MCMC estimates are more closed to true parameter value as well as ML estimate provided by Bayesian approach in comparison to PQL estimates. From those experiments, it can be assumed that they should perform similarly when the random effects are not independent, although the computational speed may be a burden. In this connection I would like to quote from [14], "… still a way to go before exact Monte Carlo method for fitting generalized linear mixed models are competitive with approximate analytical method in terms of computational speed". Nevertheless we can use MCMC method when the sample size is small and enjoy a greater flexibility in model formulation.

References:

[1] Basel Committee on Banking Supervision (1999), Credit Risk Modelling Current Practice and Applications, Basel.

[2] Gordy Michael B., (2000), A comparative anatomy of credit risk models, *Journal of Banking and Finance*, 24 (1-2) , pp. 119-149.

[3] Jacobson, Tor, Jesper Lindé and Kasper Roszbach, (2003), Internal ratings systems, implied credit risk and the consistency of banks' risk classification policies, Sveriges Riksbank Working Papers No. 155.

[4] Carling, Kenneth, Rönnegård, Lars and Rosczbach, Kasper (2004), Is firm interdependence within industries important for portfolio credit risk?, Sverige Riskbank Working Paper Series No. 168

[5] Giesecke, Kay, and Stefan Weber, (2004), Cyclical correlations, credit contagion, and portfolio losses, *Journal of Banking & Finance 28 (2004) 3009–3036* (also available at- http://www.orie.cornell.edu/~giesecke/paper_sk_voter5.pdf, last referred- 07/06/2005)

[6] Broström Göran, Generalized linear models with random intercepts, Umeå University, http://www.stat.umu.se/forskning/reports/glmmML.pdf, (last referred-07/06/2005)

[7] Schall, R. (1991), Estimation in generalized linear models with random effects, *Biometrica*, 78, 719-727.

[8] Wolfinger, R. (1995), GLIMMIX A SAS Macro for Generalized Linear Mixed Models, Cary, NC SAS Institute Inc.

[9] Pinheiro J.C. and Bates, D. M. (1999), *Linear Mixed Models in S and S Plus*, Springer-Verlag, New York.

[10] Littell C., Milliken G. A., Stroup W.W., and Wolfinger R.D. (1996), *The SAS System for Mixed Models*, SAS Institution Inc., Cary N.Y.

[11] McCulloch C. E. and Searle S. R. (2001), *Generalized Linear and Mixed Models*, John Wiley & Sons, Inc. New York.

[12] McCullagh P. and Nelder J.A. (1989), *Generalized Linear Models*, Chapman and Hall, New York.

[13] Anderson T.W. (2003), *An Introduction to Multivariate Statistical Analysis*, John Wiley and Sons (first edition 1958, third edition 2003), New York.

[14] Booth, James G. and Hobert, James P. (1999), Maximizing generalized linear mixed model likelihoods with an automated Monte Carlo EM algorithm, *Journal of Royal Statistical Society (B),* Vol.-61 (part-1), pp. 265-285.

[15] McCulloch, C. E. (1997),Maximum likelihood algorithms for generalized linear mixed models, *Journal of the American Statistical Association*, vol. 92, pp. 162-170.

[16] Kuk, A.Y.C. (1995) Asymptotic unbiased estimates in generalized linear models with random effects, *Journal of Royal Statistical Society (B)*, vol. 57, pp. 395-407.

[17] Searle, S.R., Casella, G. and McCulloch, C.E. (1992), *Variance Component*, New York: Wiley.

Appendix A: R Codes

Codes A.1 Logistic Model

```
#This Programme has been developed to find MCMC NR estimate of GLMM
#with a random intercept and correlation within the levels of a random effect

### Read Input Output Matrix ########
a<-matrix(1.57 0.64 0.67 0.64 2.97 0.58 0.67 0.58 0.94),nrow=3,byrow=T)
a<-a%*%t(a)    #### Define Covariance Structure
ca<-chol(a)    ### Choleski decomposition of a
ai<-solve(a)   ## Inverse of a
n<-100
k<-nrow(a)
cl<-LETTERS[1:k]
data<-data.frame(cl=rep(cl,rep(n/k,k)),x=rnorm(n,0,1))
err<-runif(n,min=0,max=1)
err<-log(err/(1-err))
######## Define Model Matrices#############
X<-cbind(rep(1,n),data$x)
Z<-matrix(c(rep(c(rep(1,n/k),rep(0,n)),(k-1)),rep(1,n/k)),nrow=n,ncol=k)
y<-(X%*%c(.7,.6)+Z%*%t(rnorm(k)%*%(1.23*ca))
+err)>0
data<-data.frame(y,data)
library(MASS)
glmm<-summary(glmmPQL(y~x,random=~1|cl,family=binomial,data=data))
b<-c(.01,.01)
su<-1
#####################################################
#######Define MCMC Algorithm for Drawing Sample#########
#####################################################
#### Define Alpha Function######
################################
alp<-function(u,u1){
p1<-exp(X%*%b+Z%*%u1)/(1+exp(X%*%b+Z%*%u1))
p2<-exp(X%*%b+Z%*%u)/(1+exp(X%*%b+Z%*%u))
l<-data.frame(y=data$y,p=(p1/p2),q=((1-p1)/(1-p2)))
l1<-subset(l,y==1)
l2<-subset(l,y==0)
tst<-prod(l1$p)*prod(l2$q)
return(min(1,tst))}
#### Main MCMC ################
ru<-function(nsam){
U<-matrix(rep(NA,(nsam+100)*k),nrow=100+nsam,byrow=T)
U[1,]<-mvrnorm(1,mu=rep(0,k),Sigma=(su*a))
### define the conditional mean and variances#####
ct<-1
for(i in seq(2,nsam+100,1)){
U[i,]<-U[i-1,]
if(ct>k){ct<-1}
uu<-U[i,]
tu<-uu[ct]
uu[ct]<-uu[1]
uu<-uu[2:k]
mv<-cmv(ct,uu)
U[i,ct]<-rnorm(1, mean=mv[1], sd=sqrt(mv[2]))
if(runif(1,0,1)>alp(U[i-1,],U[i,]))U[i,]<-U[i-1,]
ct<-ct+1
}
return(U[101:nrow(U),])
}

### Conditional mean and variance calculation
cmv<-function(m,cu){
cmn<-rep(0,2)
v<-(su)*a
if(m!=1){
vr<-v[m,]
v[m,]<-v[1,]
v[1,]<-vr
vc<-v[,m]
v[,m]<-v[,1]
v[,1]<-vc}
```

```
cmn[1]<-as.numeric(v[1,2:k]%*%solve(v[2:k,2:k])%*%cu)
cmn[2]<-as.numeric(v[1,1]-v[1,2:k]%*%solve(v[2:k,2:k])%*%v[2:k,1])
return(cmn)
}


## Calculate MCMC NR Solutions####
#################################
con=T                       ## Convergence criteria, T if not converged
iter<-0
while(con & iter<100){      ### Check the convergence within 100 iteration
if(iter<20){nsam<-500}
else if (iter>=20 & iter<40) {nsam<-2000}
else {nsam<-20000}
U<-ru(nsam)                                  ### Draw MCMC sample form f(u)
W<-matrix(rep(0,nrow(X)^2),nrow=nrow(X),byrow=T)  ### Initialize diagonal Weight matrix, W
s<-(rep(0,nrow(X)))
su1<-0
                            ### Initialize sigma_u
mu<-rep(0,nrow(X))
eta<-rep(0,nrow(X))
for(j in 1:nrow(U)){
eta<-X%*%b+Z%*%U[j,]         ### Calculate linear predictor
mu<-exp(eta)/(1+exp(eta))
W1<-matrix(rep(0,nrow(X)^2),nrow=nrow(X),byrow=T)
diag(W1)<-mu*(1-mu)
W<-W+W1                     ### calculate weight W
s<-s+(data$y-mu)            ### for Logit link dl=1/W
su1<-su1+(1/k)*(t(U[j,])%*%ai%*%U[j,])
}
c<-1/nrow(U)
W<-W*c
s<-s*c
b1<-b+solve(t(X)%*%W%*%X)%*%(t(X)%*%s) ### calculate MCMC-NR solution
su1<-su1*c                               ### Calculate variance coefficient, sigma_u of V(U)
if (max(abs(b1-b)/(abs(b)+.001))<0.005 & (abs(su-su1)/(abs(su)+.001))<.005)con=F  ### check
                                                                                  #convergence
b<-b1
su<-as.numeric(su1)
iter<-iter+1
}
###############################
##### Output Section ##########
###############################
if(con) print("Iteration Did not converge")else
{print("Fixed effects estimates")
print(b)
print("Estimate of the random effect variance coefficient:")
print(su)
}
```

Codes A.2 Probit Model

```
#This Programme has been developed to find MCMC NR estimate of GLMM
#with a random intercept and correlation within the levels of a random effect#

#############################################################################
### Generate a Hypothetical Dataset                                     #####
#############################################################################
a<-matrix(c(.8,.4,.52,.3,.6,.63,.42,.6,.5),nrow=3,byrow=T)
a<-a%*%t(a)    #### Define Covariance Structure
ca<-chol(a)                 ### Choleski decomposition of a
ai<-solve(a)   ## Inverse of a
n<-99
k<-nrow(a)
cl<-LETTERS[1:k]
data<-data.frame(cl=rep(cl,rep(n/k,k)),x=rnorm(n,0,1))
######## Define Model Matrices#############
X<-cbind(rep(1,n),data$x)
Z<-matrix(c(rep(c(rep(1,n/k),rep(0,n)),(k-1)),rep(1,n/k)),nrow=n,ncol=k)
y<-(X%*%c(-.4,.3)+Z%*%t(rnorm(k)%*%(sqrt(1.23)*ca))+rnorm(n))>0
data<-data.frame(data,y)
```

```
library(MASS)
### Use MASS to Calculate parameters with independence and use them as initial value
lme1<-summary(glmmPQL(y~x, random=~1|cl, family=binomial(link=probit), data=data))
print(lme1)
#b<-lme1$coefficients$fixed
#names(b)<-NULL
b<-c(.1,.1)
su<-1 ### Remark: I did not find any way to extract the variance or random effect
#############################################################
#### MCMC Algorith for Drawing Random Numbers from U|Y #####
#############################################################
###################################
## Difine Alpha (U,U*) Function #
###################################
alp<-function(u,u1){### U stands for old and U1 stands for candidate
eta1<-X%*%b+Z%*%u1
eta<-X%*%b+Z%*%u
p1<-pnorm(eta1)
p1<-(p1+1e-12)/(1+2*1e-12)
p2<-pnorm(eta)
p2<-(p2+1e-12)/(1+2*1e-12)
l<-data.frame(y=data$y,p=(p1/p2),q=((1-p1)/(1-p2)))
l1<-subset(l,y==1)
l2<-subset(l,y==0)
tst<-prod(l1$p)*prod(l2$q)
return(min(1,tst))
}

########################################
###### Main Algorithm for MCMC ###########
########################################
ru<-function(nsam=1000,sigma){
U<-matrix(numeric((nsam+100)*k),nrow=nsam+100,byrow=T) # initialize the matrix for MCMC sample
U[1,]<-mvrnorm(1,mu=rep(0,k),Sigma=sigma) ### Draw first candidate from f(u)
ct<-1
for(i in seq(2,(nsam+100),1)){
U[i,]<-U[i-1,]
if(ct>k){ct<-1}
uu<-U[i,]
uu[ct]<-uu[1]
uu<-uu[2:k]
mv<-cmv(ct,uu,sigma)
U[i,ct]<-rnorm(1,mean=mv[1], sd=sqrt(mv[2]))
if(runif(1,0,1)>alp(U[i-1,],U[i,]))U[i,]<-U[i-1,]
ct<-ct+1
}
return(U[101:nrow(U),])}
#### Define conditional mean and variance of f(u1|u2) ############
cmv<-function(m,cu,sigma){
cmn<-rep(0,2)
v<-sigma
if(m!=1){
vr<-v[m,]
v[m,]<-v[1,]
v[1,]<-vr
vc<-v[,m]
v[,m]<-v[,1]
v[,1]<-vc}
cmn[1]<-as.numeric(v[1,2:k]%*%solve(v[2:k,2:k])%*%cu)
cmn[2]<-as.numeric(v[1,1]-v[1,2:k]%*%solve(v[2:k,2:k])%*%v[2:k,1])
return(cmn)
}

############################
### End MCMC ###############
############################

###################################
## Calculate MCMC NR Solutions####
###################################
###initialize beta and sigma #####
con=T                       ## Convergence criteria, T if not converged
iter<-0
while(con & iter<100){      ### Check the convergence within 100 iteration
if(iter<20){nsam<-1000}
else if (iter>=20 & iter<40) {nsam<-5000}
else {nsam<-10000}
```

```
U<-ru(nsam,sigma=a*su)                    ### Draw MCMC sample form f(u)
s<-(rep(0,nrow(X)))
W<-matrix(numeric(nrow(X)*nrow(X)),nrow=nrow(X),byrow=T)  ### Initialize diagonal Weight
##matrix, W
mu<-rep(0,nrow(X))
eta<-rep(0,nrow(X))
su1<-0
                                          ### Initialize sigma_u
for(j in 1:nrow(U)){
eta<-X%*%b+Z%*%U[j,]                       ### Calculate linear predictor
mu<-pnorm(eta)                             ### Define inverse link function evaluated at eta
#mu<-(mu+1e-10)/(1+2*1e-10)
dl<-rep(0,nrow(X))
dl<-1/dnorm(eta) ### calculate diagonal del i.e. dg(mu)/dmu
W1<-matrix(rep(0,nrow(X)^2),nrow=nrow(X),byrow=T)
diag(W1)<-1/(mu*(1-mu)*dl^2)               ### calculate weight, W
W<-W+W1
s<-s+(diag(W1)*dl*(data$y-mu))
su1<-su1+(1/k)*t(U[j,])%*%solve(a)%*%U[j,]
}
c<-1/nrow(U)
W<-W*c
s<-s*c
b1<-b+solve(t(X)%*%W%*%X)%*%(t(X)%*%s)      ### calculate MCMC-NR solution of beta for this
##iteration
su1<-c*su1                  ### Calculate variance coefficient, sigma_u of V(U)
if (max(abs(b1-b)/(abs(b)+.001))<0.005 & (abs(su-su1)/(abs(su)+.001))<.005)con=F
### check convergence
b<-b1
su<-as.numeric(su1)
iter<-iter+1
print(paste(iter,b[1],b[2],su))
}
###############################
##### Output Section ##########
###############################
if(con) print("Iteration Did not converge")else
{print("Fixed effects estimates")
print(b)
print("Estimate of the random effect variance coefficient:")
print(su)
}
```

Codes A.3 Complementary Log-log Model

```
#### GLMM With Complementary log log Link######
### Read Input Output Matrix ########
a<-matrix(c(.8,.4,.2,.3,.6,.3,.2,.6,.5,.4,.6,.23,.42,.48,.23,.45),nrow=4,byrow=T)
a<-a%*%t(a)    #### Define Covariance Structure
ca<-chol(a)    ### Choleski decomposition of a
ai<-solve(a)     ## Inverse of a
n<-105                    ### Total Sample size N
k<-nrow(a)                ###Number of cluster
cl<-LETTERS[1:k]
data<-data.frame(cl=rep(cl,rep(n/k,k)),x=rnorm(n,0,1),x2=rnorm(n,1.5),x3=rnorm(n,0,1.2))
err<-runif(n,min=0,max=1)
err<-log(-log(1-err))
######## Define Model Matrices############# %*%t(rnorm(4)%*%(sqrt(1.23)*ca))
X<-cbind(rep(1,n),data$x,data$x2,data$x3)
Z<-matrix(c(rep(c(rep(1,n/k),rep(0,n)),(k-1)),rep(1,n/k)),nrow=n,ncol=k)
r<-rnorm(k)%*%(sqrt(1.23)*ca)
y<-(X%*%c(-.7,.6,-.2,.3)+Z%*%t(r)
+err)
y<-(1-exp(-exp(y)))>.5
data<-data.frame(data,y)
library(MASS)
glmm<-summary(glmmPQL(y~x+x2+x3,random=~1|cl,family=binomial(link=cloglog),data=data))
b<-c(.01,.01,.01,.01)
su<-.1
#################################################
#######Define MCMC Algorithm for Draing Sample#########
#################################################
```

```
#### Define Alpha Function######
################################
alp<-function(u,u1){
eta1<-X%*%b+Z%*%u1
eta<-X%*%b+Z%*%u
p1<-1-exp(-exp(eta1))
p1<-(p1+1e-20)/(1+2*1e-20)
p2<-1-exp(-exp(eta))
p2<-(p2+1e-20)/(1+2*1e-20)
l<-data.frame(y=data$y,p=(p1/p2),q=((1-p1)/(1-p2)))
l1<-subset(l,y==1)
l2<-subset(l,y==0)
tst<-prod(l1$p)*prod(l2$q)
return(min(1,tst))}
#### Main MCMC ################
ru<-function(nsam){
U<-matrix(rep(NA,(nsam+100)*nrow(a)),nrow=100+nsam,byrow=T)
U[1,]<-mvrnorm(1,mu=rep(0,k),Sigma=(su*a))
### define the conditional mean and variances#####
ct<-1
for(i in seq(2,nsam+100,1)){
U[i,]<-U[i-1,]
if(ct>k){ct<-1}
uu<-U[i,]
uu[ct]<-uu[1]
uu<-uu[2:k]
mv<-cmv(ct,uu)
U[i,ct]<-mv[1]+sqrt(mv[2])*rnorm(1)
if(runif(1,0,1)>alp(U[i-1,],U[i,]))U[i,]<-U[i-1,]
ct<-ct+1
}
return(U[101:nrow(U),])
}

#### Define conditional mean and variance of f(u1|u2) ############
cmv<-function(m,cu){
cmn<-rep(0,2)
v<-su*a
if(m!=1){
vr<-v[m,]
v[m,]<-v[1,]
v[1,]<-vr
vc<-v[,m]
v[,m]<-v[,1]
v[,1]<-vc}
cmn[1]<-as.numeric(v[1,2:k]%*%solve(v[2:k,2:k])%*%cu)
cmn[2]<-as.numeric(v[1,1]-v[1,2:k]%*%solve(v[2:k,2:k])%*%v[2:k,1])
return(cmn)
}

## Calculate MCMC NR Solutions####
###################################

con=T                       ## Convergence criteria, T if not converged
iter<-0
while(con & iter<100){                    ### Check the convergence within 20 iteration
if(iter<20){nsam<-1000}
else if (iter>=20 & iter<40) {nsam<-5000}
else if (iter>=40 & iter<60) {nsam<-10000}
else{nsam<-20000}
U<-ru(nsam)                                          ### Draw 10000 MCMC sample form f(u)
W<-matrix(rep(0,nrow(X)^2),nrow=nrow(X),byrow=T)  ### Initialize diagonal Weight matrix, W
#s1<-matrix(rep(0,(ncol(X))^2),nrow=ncol(X),byrow=T)     ### Define first part of the MCMC-NR
##solution
s<-(rep(0,nrow(X)))                          ### Define second part of the MCMC-NR solution
su1<-0                                       ### Initialize sigma_u
mu<-rep(0,nrow(X))
eta<-rep(0,nrow(X))
for(j in 1:nrow(U)){
eta<-X%*%b+Z%*%U[j,]                      ### Calculate linear predictor
mu<-1-exp(-exp(eta))
mu<-(mu+1e-10)/(1+2*1e-10)
W1<-matrix(rep(0,nrow(X)^2),nrow=nrow(X),byrow=T)
dl<-rep(0,nrow(X))
dl<--1/((1-mu)*log(1-mu))
diag(W1)<-1/((mu*(1-mu))*dl^2)             ### calculate weight, W
W<-W+W1
```

```
s<-s+(diag(W1)*dl*(data$y-mu))
su1<-su1+(1/nrow(a))*(t(U[j,])%*%ai%*%U[j,])
}
c<-1/nrow(U)
W<-W*c
s<-s*c
b1<-b+solve(t(X)%*%W%*%X)%*%(t(X)%*%s) ### calculate new MCMC-NR solution of beta
su1<-su1*c                  ### Calculate variance coefficient, sigma_u of V(U)
if (max(abs(b1-b)/(abs(b)+.001))<0.005 & (abs(su-su1)/(abs(su)+.001))<.005)con=F
### check convergence
b<-b1
su<-as.numeric(su1)
iter<-iter+1
print(paste(iter,b[1],b[2],b[3],b[4],su))
}
##############################
##### Output Section ##########
##############################
if(con) print("Iteration Did Not Converge")else
{print("Fixed effects estimates")
print(b)
print("Estimate of the random effect variance coefficient:")
print(su)
}
```

Appendix B: Some Additional Facts and Figures

Figure: B.1 Trace of Convergence for Logistic Model with One Covariate (Number of cluster =4, No. of observation per cluster =25)

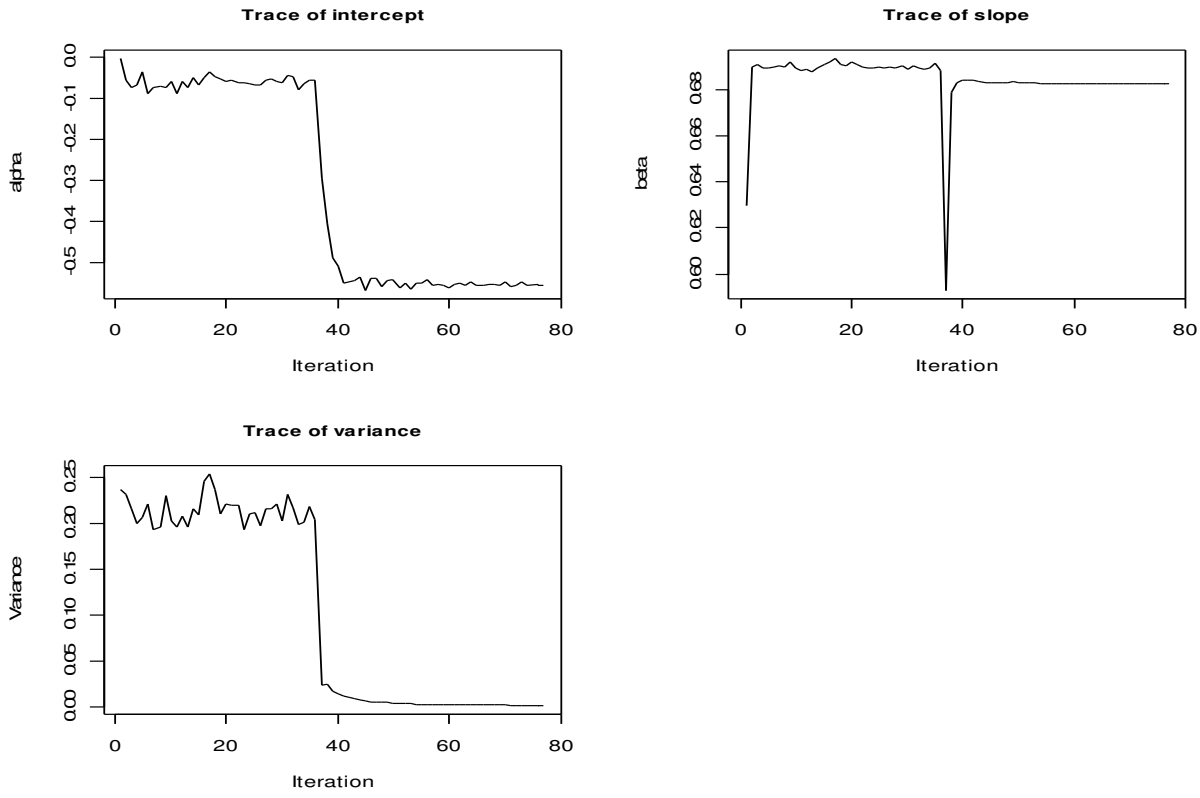
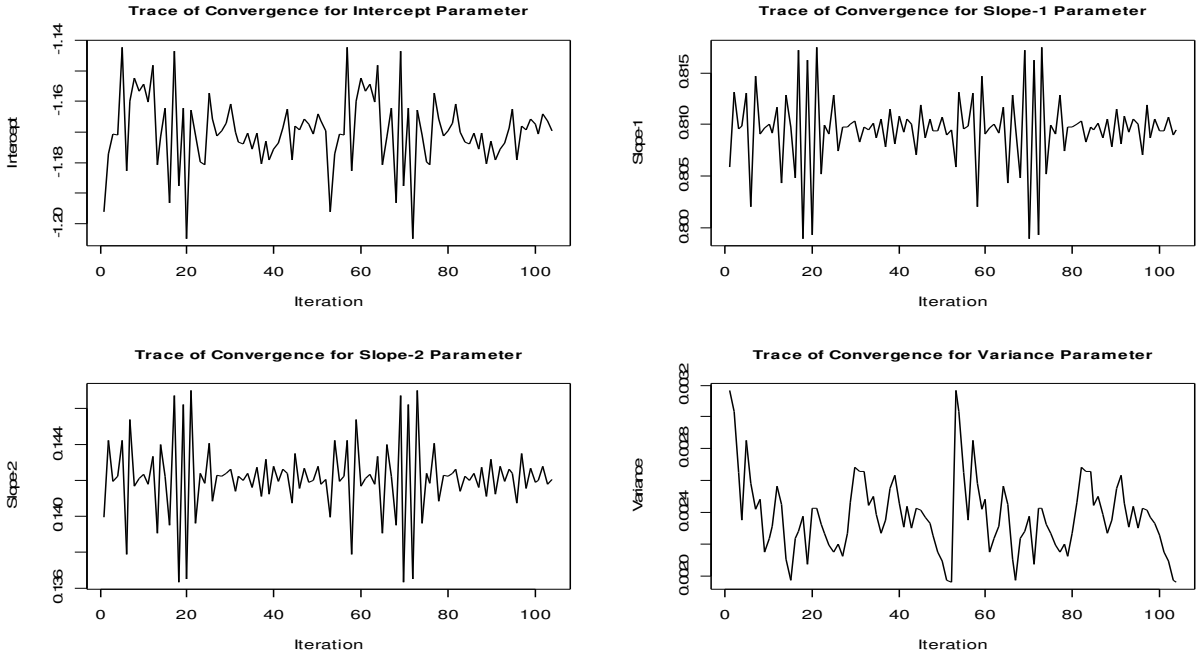


Table-B.1 Performance of Logistic model with increasing sample size

Number of observations	100	200	300
MCMC Sample	Time (in seconds)		
500	3	9	19
2000	9.5	33	85
10000	45	157	390
Iteration required for Convergence	55	78	84

This experiment has been conducted with one covariate and four clusters

Figure B.2: Trace of convergence of the parameters in logit model with 4 clusters and 25 observations per cluster.



Note: For this experiment, initial maximum iteration was set as 60 then after the maximum iteration limit had been reached the programme was restarted from the same position. Consequently, for small sample size (for MCMC) a big fluctuation is produced just after 60 iterations.