Chapter 1

Discussion and Outlook - Some extenstions

In the following chapter some extenstions of the LMG formula in the Bayesian framework beyond the simple linear regression model are shown. The focus is on repeated measurements models. These models extend the simple linear regression be allowing intra subject correlation between repeated measures.

The dependence within subjects can be modeled by including random effects (mixed model) or by assuming correlated errors within a subject (marginal model). Using a random intercept model or a compound symmetry matrix for the error will result in the same model for the fixed predictors. A mixed model can be extended by including a random slope per subject, allowing for less restrictive longitudinal shapes. The marginal approach can get more freedom by different specified covariance matrices of the error terms. An unstructured covariance matrix, where no restriction are imposed, allows for the most freedom. However, depending on the number of repeated measuresments and the sample size the covariance matrix can get too large to make reasonable inference about it.

The extenstion of the LMG formula in the Bayesian framework presented in chapter is restricted to models where the conditional variance formula can easily be applied to get the explained variance of the submodel from the regression parameters of the full model. The focus is on the fixed predictors and not on the random effects. Using the conditional variance formula to get the explained variance of the fixed predictors of the submodels should be applicable in the marginal models, where only the fixed effects are modelled anyway. In the mixed model framework the conditional variance formula is applicable to models including only random intercepts and the focus lies in the explained variance of the fixed predictors. For random-slope models there are atleast some difficulties involved, if it is possible at all the get the expalined variance of the submodel. This chapter shows a random intercept model and a repeated measurement model with an unstructured covariance matrix.

The first example concerns a simple random intercept model with time varying predictors.

1.1 random intercept model

Different R^2 metrics exist for linear mixed models. The variance of a random intercept model with regression parameter $\boldsymbol{\beta}$ can be written as

$$\text{Var}(y) = \sigma_f^2 + \sigma_\alpha^2 + \sigma_\epsilon^2, \tag{1.1}$$

where $\sigma_f^2 = \text{Var}(\mathbf{X}\boldsymbol{\beta}) = \boldsymbol{\beta}^\top \boldsymbol{\Sigma}_{\mathbf{XX}} \boldsymbol{\beta}$, σ_α^2 is the random intercept and σ_ϵ^2 is the error term. An R^2 that is guaranteed to be positive can be defined as

$$R^2_{\text{LMM}} = \frac{\sigma_f^2}{\sigma_f^2 + \sigma_\alpha^2 + \sigma_\epsilon^2}, \tag{1.2}$$

Referenz Naka, Snyder.... It is theoretical possible that the R^2_{LMM} decreases when adding predictors. This may be problematic for the LMG metric, because of violation of the non-negative property. This should rarely be the case with real data. The R^2 can not decrease when adding predictor by using the conditional variance formula on the full model to calculate the R^2 of the submodels. In the Bayesian framework we would the sample from the posterior distributions of the parameters.

The total variance of the full can be calculated as in equation 6 or by using the samples of the random intercept for each subject directly. The same total variance is then used for one sample of the posterior. (In a repeated measure study we often have within and between subject predictors. If we use the total variance of the full model the random intercept is fitted including all predictors. If a between subject predictor is excluded (e.g. Sex) and we would fit a new random intercept model, the random intercept parameter will in addition explain the variance that was explained by the excluded predictor. In other words it means that the model with the exluded between subject predictor will explain almost as much as the model where the predictor is included when each time a new model and therefore a new random intercept term is fitted in each model.

When using the conditional variance formula for the R^2 of the submodels, it only takes into account the explained variance of the fixed predictors.)

In repeated measurement studies the focus is often in within subject changes. The between subject variance estimated with the random intercept term may not be so important. The more important question may be how much the fixed predictors explain compared to the within subject error, which is

$$R^2_{\text{repeated}} = \frac{\sigma_f^2}{\sigma_f^2 + \sigma_\epsilon^2}, \tag{1.3}$$

The square root of this term is known under the name correlation within subjects by ref(bland Altman 1995). Although in the paper the subject term is fitted as a factor. If we are interested in the within subject effects we can use the model including only the between subject predictor as the null model.

The following example shows a simple random intercept model with time-varying predictors. The main question is which within subject predictors are the most important ones. The between subject variance is of lower importance.

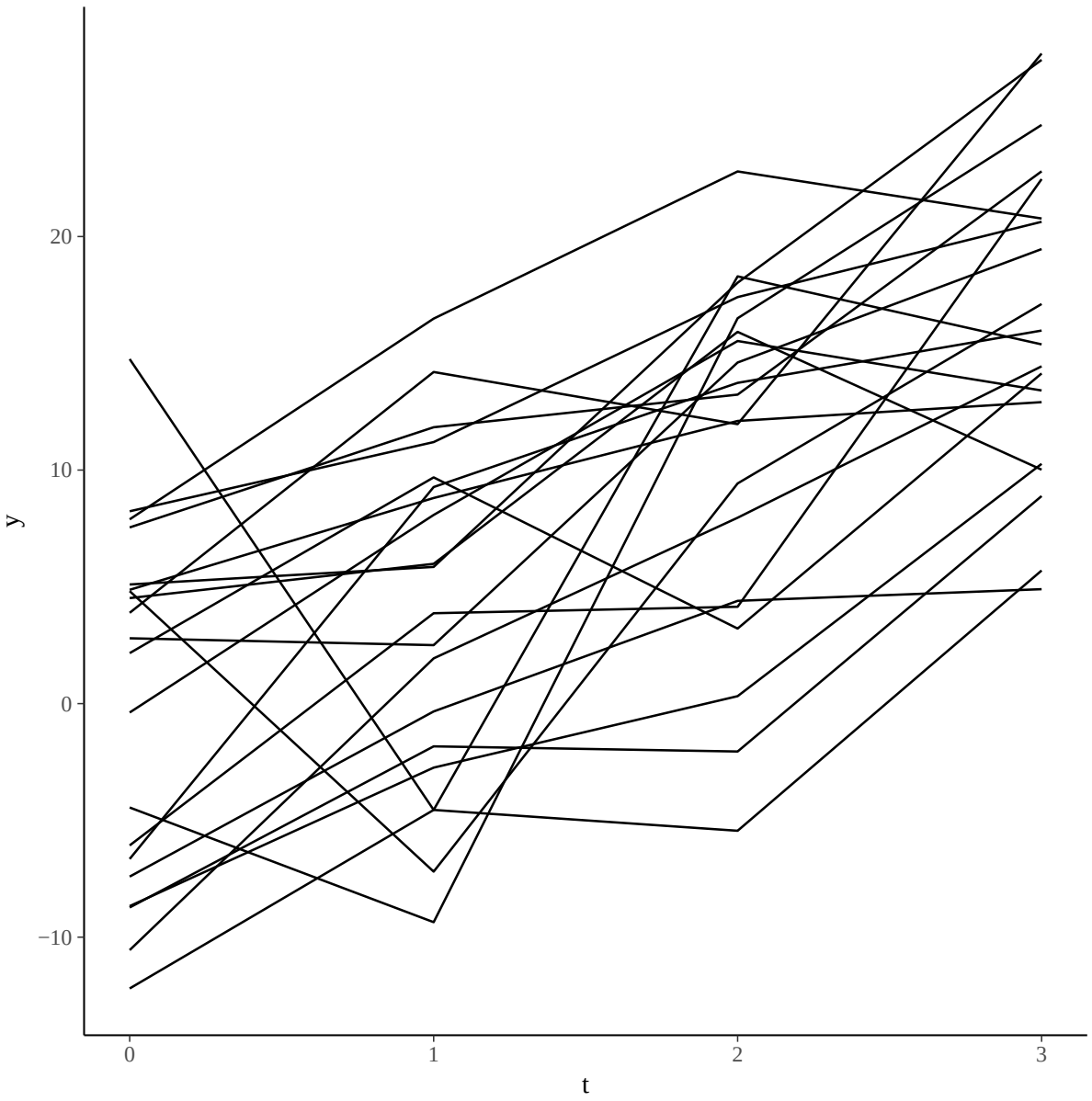
The data are simulated from the following regression setting with $m = 4$ timepoints,

$$Y_{i,j} \sim \mathcal{N}(\beta_0 + x_{1_{i,j}}\beta_1 + x_{2_{i,j}}\beta_2 + x_{3_{i,j}}\beta_3 + x_{4_{i,j}}\beta_4 + \alpha_i, \sigma^2), \qquad i = 1, \ldots, n \qquad j = 1, \ldots, m \tag{1.4}$$

where $\beta_1 = 1$, $\beta_2 = 1$, $\beta_3 = 2$, $\beta_4 = 2$, $\sigma^2 = 1$, $\alpha_i \sim \mathcal{N}(0, \sigma^2_\alpha)$, $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$.

The following R code is used to simulate the data:

```
sub<- 1:20
subi <- rnorm(20, 0, 4)
subi<-rep(subi, 4)
t <- c(0, 1, 2,3)
t <- c(rep(0, 20), rep(1,20), rep(2, 20), rep(3,20))

mu <- rep(0,4)
sig <- matrix(0.4, 4, 4)
diag(sig) <- 1
sig[3,4] <- 0.9
sig[4,3] <- 0.9
sig[1,2] <- 0.3
sig[2,1] <- 0.3


rawvars <- mvrnorm(n=80, mu=mu, Sigma=sig)

x1 <- t+rawvars[,1]
x2 <- t+rawvars[,2]
x3 <- t+rawvars[,3]
x4 <- t+rawvars[,4]

b1 <- b2  <-1
b3 <- b4 <- 2

y<- x1*b1 +x2*b2 +x3*b3+  x4*b4 + subi+ rnorm(80, 0, 0.1)

df <- data.frame(y=y, x1 = x1, x2=x2, x3 = x3, x4 = x4, sub = rep(sub,4))

p <- ggplot(data = df, aes(x = t, y = y, group = sub))
p + geom_line()
```



The R^2 of the submodels were first calculated according to the formula in equation ...,. The random intercept effect is not of interest. We see that most of the within subject variance is explained by the predictors. The credible intervals are very narrow. For the information about the between subject varinace we can look at the random intercept variance directly. In the second part the random intercept is included in the total variance calculation and the calculation of the R^2 values. We see that there is a large between subject variance in this dataset. The LMG values including the between subject variance are much lower. The credible intervals are also much wider, because the uncertainty about the between subject variance is also included. In my opinion we can get more useful infromation from separating the between and within variance components when it is easy possible. Note that we assumed non stochastic predictors otherwise the credible intervals would be larger. In general it seems more reasonable to assume stochastic time-varying predictors. The variance could then be estimated by non-parameteric bootstrap, resampling whole subjects (all repeated measurements of a subject).

```
fit <- stan_glmer(y ~ x1+x2+x3+x4 + (1|sub) ,
                                                    data = df,
                                                    chains = 4, cores = 4)

post.sample <- as.matrix(fit)
post.sample.r <- post.sample[,c(2:5,(ncol(post.sample)-1):ncol(post.sample))]

df.rtwos <- rtwos.ri.r(df[,2:5], post.sample.r)


LMG.Vals<-matrix(0, 4, dim(df.rtwos)[2])

for(i in 1:dim(df.rtwos)[2]){

        gofn<-df.rtwos[,i]

        obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

        LMG.Vals[,i]=obj.Gelman$IJ[,1]
}

# posterior LMG distribution of each variable
quantile(LMG.Vals[1,], c(0.025, 0.5, 0.975))

##      2.5%       50%     97.5%
## 0.1759721 0.1778704 0.1797420

quantile(LMG.Vals[2,], c(0.025, 0.5, 0.975))

##      2.5%       50%     97.5%
## 0.1805498 0.1824204 0.1841952

quantile(LMG.Vals[3,], c(0.025, 0.5, 0.975))

##      2.5%       50%     97.5%
## 0.3213042 0.3228832 0.3244004

quantile(LMG.Vals[4,], c(0.025, 0.5, 0.975))

##      2.5%       50%     97.5%
## 0.3149818 0.3165701 0.3180643

# explained compared to total variance

df.rtwos <- rtwos.ri.a(df[,2:5], post.sample)
```

```
LMG.Vals<-matrix(0, 4, dim(df.rtwos)[2])

for(i in 1:dim(df.rtwos)[2]){

        gofn<-df.rtwos[,i]

        obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

        LMG.Vals[,i]=obj.Gelman$IJ[,1]
}

# posterior LMG distribution of each variable
quantile(LMG.Vals[1,], c(0.025, 0.5, 0.975))

##        2.5%         50%       97.5%
## 0.009191661 0.019669347 0.047444048

quantile(LMG.Vals[2,], c(0.025, 0.5, 0.975))

##       2.5%        50%      97.5%
## 0.01137392 0.03076685 0.06097521

quantile(LMG.Vals[3,], c(0.025, 0.5, 0.975))

##       2.5%        50%      97.5%
## 0.02238598 0.05498246 0.10306689

quantile(LMG.Vals[4,], c(0.025, 0.5, 0.975))

##       2.5%        50%      97.5%
## 0.02545344 0.06020207 0.11190935
```

1.2 marginal model

The next example concerns a repeated measurement model with an unstructured covariance error structure. The data are generated from the following model:

$$Y_i \sim \mathcal{N}(\mathbf{X}_i\boldsymbol{\beta}, \boldsymbol{\Sigma}), \qquad i = 1, \ldots, n \tag{1.5}$$

where $\boldsymbol{\Sigma}$ represents an unstructured error covariance matrix.

In the variance calculation we need to take into account that we do not have just one σ^2 parameter, but a covariance matrix $\boldsymbol{\Sigma}$. The diagonal elements of $\boldsymbol{\Sigma}$ represent the variance of each timepoint. The sum of the diagonal elements of $\boldsymbol{\Sigma}$ represents the variance for a whole

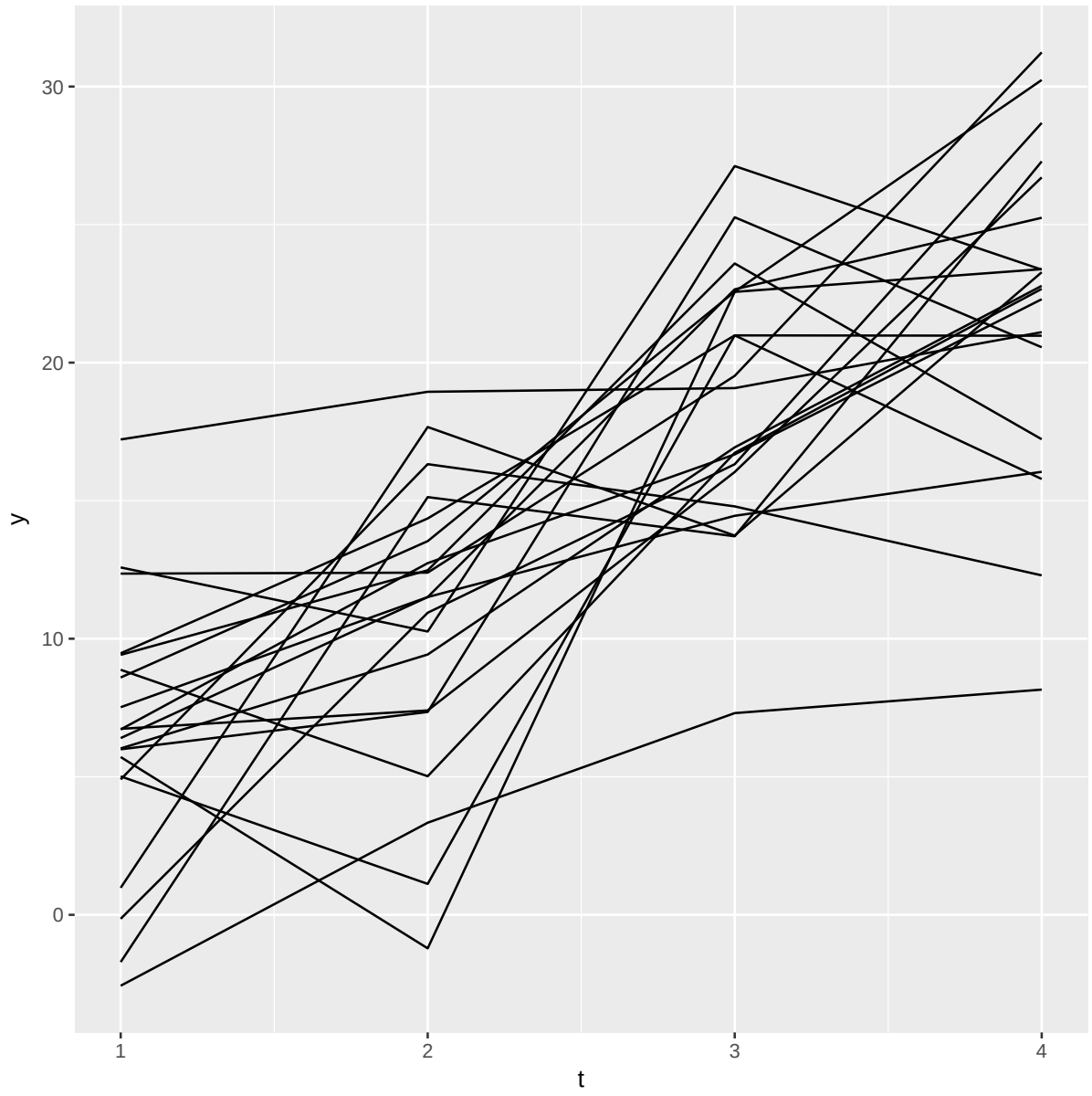
subject. When there are no missing timepoints in each subject, we can take the mean of diag($\mathbf{\Sigma}$) to make the formula compatiable with the $\boldsymbol{\beta}^\top \mathbf{\Sigma_{XX}} \boldsymbol{\beta}$ of equation .... resulting in the total variance term

$$\text{Var}(\mathbf{Y}) = \boldsymbol{\beta}^\top \mathbf{\Sigma_{XX}} \boldsymbol{\beta} + \text{mean}(\text{diag}(\mathbf{\Sigma})), \tag{1.6}$$

The following R-code is used to generate the data:

```
sub<- 1:20
subi <- rnorm(20, 0, 1)
subi<-rep(subi, 4)

mu <- rep(0,4)
sig <- matrix(0.4, 4, 4)
diag(sig) <- 1
sig[3,4] <- 0.9
sig[4,3] <- 0.9
sig[1,2] <- 0.3
sig[2,1] <- 0.3

rawvars <- mvrnorm(n=80, mu=mu, Sigma=sig)
cov(rawvars)

##           [,1]      [,2]      [,3]      [,4]
## [1,] 0.8185433 0.1544539 0.4858672 0.4857820
## [2,] 0.1544539 0.9208422 0.2848089 0.3715627
## [3,] 0.4858672 0.2848089 1.0513613 1.0061088
## [4,] 0.4857820 0.3715627 1.0061088 1.1152070

t <- c(rep(1, 20),rep(2,20), rep(3, 20), rep(4, 20))
x1 <- t+rawvars[,1]
x2 <- t+rawvars[,2]
x3 <- t+rawvars[,3]
x4 <- t+rawvars[,4]

Sig<- matrix(3, 4,4)
diag(Sig) <- 10
u <- rep(0, 4)
Sig[1,1] <- 5
Sig[2,2] <- 7
Sig[3,4] <- 8
Sig[4,3] <- 8

Sig[1,2] <- 4
```

```
Sig[2,1] <-4

Sig

##      [,1] [,2] [,3] [,4]
## [1,]    5    4    3    3
## [2,]    4    7    3    3
## [3,]    3    3   10    8
## [4,]    3    3    8   10

error <- mvrnorm(20, u, Sig)

y<- x1*b1 +x2*b2 + x3*b3 +x4*b4 +c(error)


t <- c(rep(1, 20), rep(2, 20), rep(3, 20), rep(4, 20))
df <- data.frame(y=y, x1 = x1, x2=x2 , x3 = x3, x4 = x4, sub = rep(sub,4), t =t)



p <- ggplot(data = df, aes(x = t, y = y, group = sub))
p + geom_line()
```




```
# Bayesian framework

Y <- matrix(df[,'y'], 20, 4, byrow=F)
x1 <- matrix(df[,'x1'], 20, 4, byrow=F)
x2 <- matrix(df[,'x2'], 20, 4, byrow=F)
x3 <- matrix(df[,'x3'], 20, 4, byrow=F)
x4 <- matrix(df[,'x4'], 20, 4, byrow=F)

N = 20 #subjects
M = 4 # repeated measures
```

```
#-----------------------------------------

modelString <- "model{
```

```
# Likelihood
for(i in 1:N){
Y[i,1:M] ~ dmnorm(mu[i,1:M],Omega[1:M,1:M])
for(j in 1:M){
mu[i,j] <- beta0 + beta1*x1[i,j]+ beta2*x2[i,j]+ beta3*x3[i,j] + beta4*x4[i,j]
}}

# Priors

Omega[1:M, 1:M] ~dwish(zRmat[1:M,1:M] , zRscal)
Sigma[1:M, 1:M] <- inverse(Omega)

beta0      ~ dnorm(0,0.001)
beta1      ~ dnorm(0,0.001)
beta2      ~ dnorm(0,0.001)
beta3      ~ dnorm(0,0.001)
beta4      ~ dnorm(0,0.001)

}"


writeLines( modelString , con="Jags-MultivariateNormal-model.txt" )

model <- jags.model(textConnection(modelString),
                                                                    data = list

                                                                    n.chains=3)









samp <- coda.samples(model,
                                                                     variable.n
                                                                     n.iter=200


summary(samp)
```

```
##
## Iterations = 1:20000
## Thinning interval = 1
## Number of chains = 3
## Sample size per chain = 20000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean     SD Naive SE Time-series SE
## Sigma[1,1]  5.978 2.3508 0.009597       0.013099
## Sigma[2,1]  4.576 2.3490 0.009590       0.012624
## Sigma[3,1]  2.766 2.6219 0.010704       0.011703
## Sigma[4,1]  3.512 2.4949 0.010185       0.011590
## Sigma[1,2]  4.576 2.3490 0.009590       0.012624
## Sigma[2,2]  8.095 3.2934 0.013445       0.016489
## Sigma[3,2]  1.312 3.1313 0.012783       0.013701
## Sigma[4,2]  3.566 3.2325 0.013197       0.014454
## Sigma[1,3]  2.766 2.6219 0.010704       0.011703
## Sigma[2,3]  1.312 3.1313 0.012783       0.013701
## Sigma[3,3] 12.290 5.6598 0.023106       0.025608
## Sigma[4,3]  7.791 5.3037 0.021652       0.023547
## Sigma[1,4]  3.512 2.4949 0.010185       0.011590
## Sigma[2,4]  3.566 3.2325 0.013197       0.014454
## Sigma[3,4]  7.791 5.3037 0.021652       0.023547
## Sigma[4,4]  8.247 5.9496 0.024289       0.026969
## beta1       1.223 0.2524 0.001031       0.002759
## beta2       1.031 0.2018 0.000824       0.001686
## beta3       1.283 0.5261 0.002148       0.013322
## beta4       2.411 0.5703 0.002328       0.014934
##
## 2. Quantiles for each variable:
##
##               2.5%     25%    50%    75%  97.5%
## Sigma[1,1]  2.9681  4.3945  5.516  7.030 11.608
## Sigma[2,1]  1.5064  3.0655  4.159  5.594 10.003
## Sigma[3,1] -0.8815  1.3080  2.474  3.891  8.056
## Sigma[4,1]  0.5801  2.1839  3.183  4.451  8.282
## Sigma[1,2]  1.5064  3.0655  4.159  5.594 10.003
## Sigma[2,2]  4.1045  5.9916  7.479  9.474 15.605
## Sigma[3,2] -3.2649 -0.1773  1.170  2.630  6.553
## Sigma[4,2]  0.1671  2.0992  3.240  4.647  8.726
```

```
## Sigma[1,3] -0.8815  1.3080  2.474  3.891  8.056
## Sigma[2,3] -3.2649 -0.1773  1.170  2.630  6.553
## Sigma[3,3]  6.3027  9.1570 11.403 14.372 23.292
## Sigma[4,3]  3.3046  5.4649  7.110  9.314 16.010
## Sigma[1,4]  0.5801  2.1839  3.183  4.451  8.282
## Sigma[2,4]  0.1671  2.0992  3.240  4.647  8.726
## Sigma[3,4]  3.3046  5.4649  7.110  9.314 16.010
## Sigma[4,4]  4.0997  6.0544  7.586  9.621 16.025
## beta1       0.7338  1.0517  1.221  1.393  1.724
## beta2       0.6451  0.8955  1.026  1.163  1.441
## beta3       0.2668  0.9337  1.278  1.628  2.326
## beta4       1.2668  2.0383  2.418  2.794  3.502
```

```
#LMG calculations

samp <- coda.samples(model,
                                                                    variable.n
                                                                    n.iter=200


post.sample <- samp[[1]][,c(5:8, 1:4)]

df.rtwos <- rtwos.marg(df[,2:5], post.sample, 4) # 4 repeated measures

LMG.Vals<-matrix(0, 4, dim(df.rtwos)[2])

for(i in 1:dim(df.rtwos)[2]){

        gofn<-df.rtwos[,i]

        obj.Gelman<-partition(gofn, pcan = 4, var.names = names(df[,2:5]))

        LMG.Vals[,i]=obj.Gelman$IJ[,1]
}

# posterior LMG distribution of each variable
quantile(LMG.Vals[1,], c(0.025, 0.5, 0.975))

##      2.5%       50%     97.5%
## 0.1077745 0.1390563 0.1671079

quantile(LMG.Vals[2,], c(0.025, 0.5, 0.975))

##       2.5%        50%      97.5%
## 0.08137171 0.10369238 0.13062228
```

```
quantile(LMG.Vals[3,], c(0.025, 0.5, 0.975))

##      2.5%       50%     97.5%
## 0.1450403 0.1852359 0.2153531

quantile(LMG.Vals[4,], c(0.025, 0.5, 0.975))

##      2.5%       50%     97.5%
## 0.1522790 0.1930420 0.2260583
```