STA 601: Homework 10

Akbota Anuarbek

$y_{ij} = x'_{ij}\beta + z'_{ij}\alpha_i + e_{ij}, e_{ij} \sim N(0, \sigma^2)$

Use Pregnancy Blood Pressure data.

- m = #of observations in total
- n = #of people

Part (a) Inverse Wishart prior for Ω

$\beta \sim \mathcal{N}_2(0, I)$
$\sigma^{-2} \sim Gamma(1, 1)$
$\alpha_i \sim \mathcal{N}(0, \Omega)$
$\Omega \sim \mathcal{W}^{-1}(I, 3)$

Derive full conditionals:

Let $y^*_{ij} = y_{ij} - z'_{ij}\alpha_i$, then $y^*_{ij} \sim \mathcal{N}(x'_{ij}\beta, \sigma^2)$

- $\underline{\pi(\beta|y^*, X, \sigma^2) \sim \mathcal{N}(M_1, V_1)}$, where $V_1 = (X^T X/\sigma^2 + I_2)^{-1}$ and $M_1 = VX^T y^*/\sigma^2$

- $\underline{\pi(\sigma^{-2}|\beta, y^*, X) \sim Gamma(1 + \frac{m}{2}, 1 + \frac{1}{2}(y - X\beta)^T(y - X\beta))}$

Let $\tilde{y}_{ij} = y_{ij} - x'_{ij}\beta$, then $\tilde{y}_{ij} \sim \mathcal{N}(z_{ij}\alpha_i, \sigma^2)$

- $\underline{\pi(\alpha_i|\sigma^2, \Omega, z_i, \tilde{y}_i) \sim \mathcal{N}(M_2, V_2)}$, where $V_2 = (z_i^T z_i/\sigma^2 + \Omega^{-1})^{-1}$ and $M_2 = V_2 z_i^T \tilde{y}_i/\sigma^2$
($\tilde{y}_i$ and z_i are observation for person i)

- $\underline{\pi(\Omega|\alpha_{1:n}, Z, \tilde{y}) \sim \mathcal{W}^{-1}(\alpha\alpha^T + I_2, n + 3)}$, where $\alpha = [\alpha_1, ..., \alpha_n]$

Run Gibbs sampling in R. Code is given in the end.

Part (b)

$\beta \sim \mathcal{N}_2(0, I_2)$
$\sigma^{-2} \sim Gamma(1, 1)$
$\alpha_i = \Lambda \alpha_i^*$
$\alpha_i^* \sim \mathcal{N}_2(0, I_2)$
$\Lambda \sim \mathcal{MN}_{2x2}(M, U, V)$, where $M = \mathbf{0}_{2x2}$, $U = V = I_2$
Λ *could be any matrix, not necessarily PSD.*

Derive full conditionals:
Full conditionals for β and σ^{-2} are exactly the same as in part (a). Λ will be updated using Metropolis Hasting, we don't derive its full conditional.

Let $y_{ij}^* = y_{ij} - z_{ij}'\alpha_i$, then $y_{ij}^* \sim \mathcal{N}(x_{ij}'\beta, \sigma^2)$

- $\underline{\pi(\beta|y^*, X, \sigma^2) \sim \mathcal{N}(M_1, V_1)}$, where $V_1 = (X^T X/\sigma^2 + I_2)^{-1}$ and $M_1 = V X^T y^*/\sigma^2$

- $\underline{\pi(\sigma^{-2}|\beta, y^*, X) \sim Gamma(1 + \frac{m}{2}, 1 + \frac{1}{2}(y - X\beta)^T(y - X\beta))}$

Let $\tilde{y}_{ij} = y_{ij} - x_{ij}'\beta$, then $\tilde{y}_{ij} \sim \mathcal{N}(z_{ij}\alpha_i, \sigma^2)$

- $\underline{\pi(\alpha_i^*|\sigma^2, z_i, \tilde{y}_i, \Lambda) \sim \mathcal{N}(M_2, V_2)}$, where $V_2 = (\Lambda^T z_i^T z_i \Lambda/\sigma^2 + I)^{-1}$ and $M_2 = V_2 \Lambda^T z_i^T \tilde{y}_i/\sigma^2$

MH step for Λ :

- draw a candidate $\Lambda^* \sim \mathcal{MN}_{2x2}(\Lambda^{(t-1)}, I_{2x2}, I_{2x2})$

- set $\Lambda^t = \Lambda^*$ with probability $\min\left\{1, \frac{\pi(\Lambda^*)\prod_{i=1}^n N(\tilde{y}_i;\mu=z_i\Lambda^*\alpha_i^*,\Sigma=\sigma^2 I)}{\pi(\Lambda^{(t-1)})\prod_{i=1}^n N(\tilde{y}_i;\mu=z_i\Lambda^{(t-1)}\alpha_i^*,\Sigma=\sigma^2 I)}\right\}$
- otherwise $\Lambda^t = \Lambda^{(t-1)}$

Posterior Inferences:

Run Gibbs and MH using R. We also make sure that our samplers have converges buy cehing their traceplots or acf function. Then after burning-in first 1000 observation following posterior inferences were obtained.

Table 1: Method a

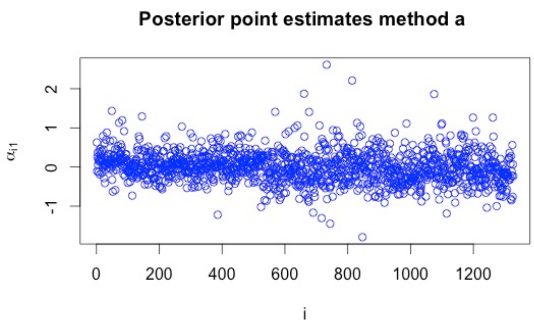
	β_1	β_2	σ^2
posterior point estimates	**0.0849**	**-0.0023**	**0.67945**
95% CIs	(0.0205 0.1523)	(-0.0053 0.0006)	(0.6612 0.7072)

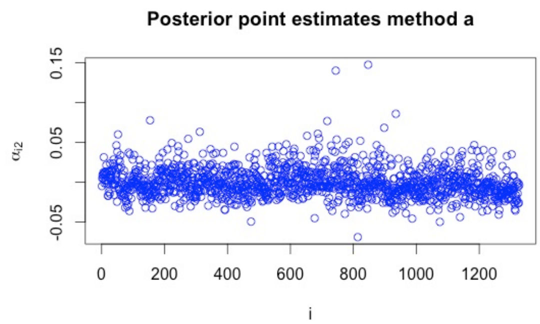
$$\text{Method a: } \hat{\Omega} = \begin{pmatrix} 0.1609 & -0.0025 \\ -0.0025 & 0.0035 \end{pmatrix}$$

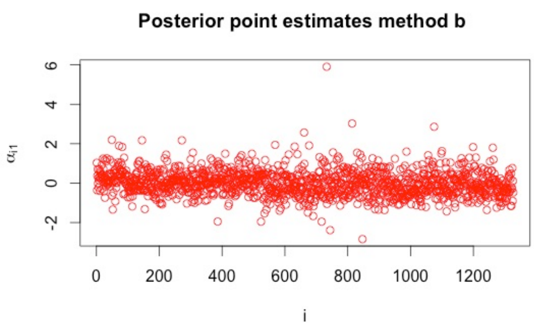
Table 2: Method b

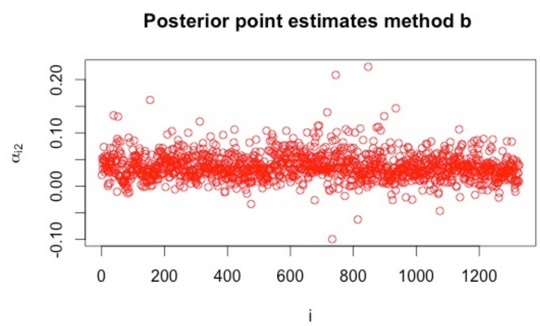
	β_1	β_2	σ^2
posterior point estimates	**0.0871**	**-0.0051**	**0.7092**
95% CIs	(-0.0117 0.1444)	(-0.0077 0.0021)	(0.6572 0.7565)

Method b: $\hat{\Omega} = \begin{pmatrix} 0.4163 & -0.0101 \\ -0.0101 & 0.0006 \end{pmatrix}$









Overall, two methods give similar results. We can see that there are both within subject variance σ^2 and between subject variance $z'_{ij}\Omega z_{ij}$.

R code part a:

```
library(mvtnorm)
library(optimbase)

setwd('/Users/akbota/Documents/STA601/HWs/hw10')
data<- read.table("BP.csv", header=TRUE)

m<-length(data$id) #number of observations in total
```

```
n<-length(unique(data$id)) #number of people

#modify data id's
data$newid[1]=1
d=1
for(j in 2:m){
  if(data$id[j]!=data$id[j-1]){
    d=d+1
  }
  data$newid[j]=d
}
data$id<-NULL
names(data)[4]="id"

#normalize y's
y<-data$BP_syst/data$BP_dia
y<-(y-mean(y))/sd(y)

id=data$id

x=x=matrix(data=c(rep(1,m), data$age), ncol=2, nrow=m)
z=x

#Gibbs

N<-1000
a<-1
b<-1

beta<-matrix(rep(NA), nrow=N, ncol=2)
beta[1,]=rep(0,2)

alpha<-matrix(rep(NA), nrow=n, ncol=2*N)
alpha[,1:2]=0

sig.sq<-c(1)

Omega<-matrix(rep(NA),nrow=2*N, ncol=2)
Omega[1:2,]<-riwish(v=3, S=diag(2))

for(k in 2:N){

  #y*
  y.star<-c()
```

```
  j=1
  for(i in id){
    y.star[j]=y[j]-z[j,]%*%alpha[i,(2*(k-1)-1):(2*(k-1))]
    j=j+1
  }

  #update beta
  V1=solve(t(x)%*%x/sig.sq[k-1]+diag(2))
  M1=V1%*%t(x)%*%y.star/sig.sq[k-1]
  beta[k,]=rmvnorm(1, mean=M1, sigma=V1)

  #update sigma^2
 f=rgamma(1,shape=a+m/2, rate=b+0.5*t(y.star-x%*%beta[k-1,])%*%(y.star-x%*%beta[k-1,]))
  sig.sq[k]=1/f

  #y.tilde
  y.t=y-x%*%beta[k-1,]


  #update alpha's
  i2=0
  i1=1
  for(t in 1:n){
    l=length(which(id==t))
    i2=i2+l
    z_i=z[i1:i2,]
   V2=solve(transpose(z_i)%*%z_i/sig.sq[k-1]+solve(Omega[(2*(k-1)-1):(2*(k-1)),]))
    M2=V2%*%transpose(z_i)%*%y.t[i1:i2]/sig.sq[k-1]
    alpha[t,(2*k-1):(2*k)]=rmvnorm(1,mean=M2, sigma=V2)
    i1=i1+l
  }

  #update Omega
  A=t(alpha[,(2*(k-1)-1):(2*(k-1))])%*%alpha[,(2*(k-1)-1):(2*(k-1))]
  Omega[(2*k-1):(2*k),]=riwish(v=3+n, S=A+diag(2))


  print(k)
}
```

R code part b:

```
library(mvtnorm)
library(optimbase)

setwd('/Users/akbota/Documents/STA601/HWs/hw10')
data<- read.table("BP.csv", header=TRUE)

m<-length(data$id) #number of observations in total
n<-length(unique(data$id)) #number of people

#modify data id's
data$newid[1]=1
d=1
for(j in 2:m){
  if(data$id[j]!=data$id[j-1]){
    d=d+1
  }
  data$newid[j]=d
}
data$id<-NULL
names(data)[4]="id"

#normalize y's
y<-data$BP_syst/data$BP_dia
y<-(y-mean(y))/sd(y)

id=data$id

x=x=matrix(data=c(rep(1,m), data$age), ncol=2, nrow=m)
z=x

#Gibbs

N<-2000
a<-1
b<-1

beta<-matrix(rep(NA), nrow=N, ncol=2)
beta[1,]=rep(0.1,2)

alpha.star<-matrix(rep(NA), nrow=n, ncol=2*N)
alpha.star[,1:2]=0.1

sig.sq<-c(1)
```

```
Lambda<-matrix(rep(NA),nrow=2*N, ncol=2)
#Lambda[1:2,]<-matrix(data=c(0,0,0,0), nrow=2, ncol=2) #any matrix may be PSD too!
lll<-matrix(nrow=2,ncol=2)
for(i in 1:2){
  for(j in 1:2){
    lll[i,j]<-rnorm(1,mean=0, sd=1)
  }
}
Lambda[1:2,]=lll

alpha<-matrix(rep(NA), nrow=n, ncol=2*N)
alpha[,1:2]=alpha.star[,1:2]%*%transpose(Lambda[1:2,])

for(k in 2:N){

  #y*
  y.star<-c()
  j=1
  for(i in id){
    y.star[j]=y[j]-z[j,]%*%alpha[i,(2*(k-1)-1):(2*(k-1))]
    j=j+1
  }

  #update beta
  V1=solve(t(x)%*%x/sig.sq[k-1]+diag(2))
  M1=V1%*%t(x)%*%y.star/sig.sq[k-1]
  beta[k,]=rmvnorm(1, mean=M1, sigma=V1)

  #update sigma^2
 f=rgamma(1,shape=a+m/2, rate=b+0.5*t(y.star-x%*%beta[k-1,])%*%(y.star-x%*%beta[k-1,]))
  sig.sq[k]=1/f

  #y.tilde
  y.t=y-x%*%beta[k-1,]



  #update alpha*'s
  i2=0
  i1=1
  for(t in 1:n){
    l=length(which(id==t))
    i2=i2+l
    z_i=z[i1:i2,]
```

```
 V2=solve(transpose(Lambda[(2*(k-1)-1):(2*(k-1)),])%*%transpose(z_i)%*%z_i%*%Lambda[(2*(k-1)-1):(2
 M2=V2%*%transpose(Lambda[(2*(k-1)-1):(2*(k-1)),])%*%transpose(z_i)%*%y.t[i1:i2]/sig.sq[k]
  alpha.star[t,(2*k-1):(2*k)]=rmvnorm(1,mean=M2, sigma=V2)
  i1=i1+l
}


#MH
L<-matrix(nrow=2,ncol=2)

for(i in 1:2){
  for(j in 1:2){
    L[i,j]<-rnorm(1,mean=Lambda[2*(k-1)-(2-i),j], sd=7)
  }
}


prod1<-0
prod2<-0
i2=0
i1=1
for(t in 1:n){

  l=length(which(id==t))
  i2=i2+l
  z_i=z[i1:i2,]

 prod1<-prod1+dmvnorm(x=y.t[i1:i2], mean=z_i%*%L%*%alpha.star[t,(2*(k-1)-1):(2*(k-1))], sigma=sig.s
 prod2<-prod2+dmvnorm(x=y.t[i1:i2], mean=z_i%*%L%*%alpha.star[t,(2*(k-1)-1):(2*(k-1))], sigma=sig.s

  i1=i1+l
}

p=exp(prod1-prod2)*(exp(-0.5*matrix.trace(t(L)%*%L)))/(exp(-0.5*matrix.trace(t(Lambda[(2*(k-1)-1):

prty=min(1,p)

q=runif(1,min=0,max=1)

if(q<p){
  Lambda[(2*k-1):(2*k),]=L
}else{
  Lambda[(2*k-1):(2*k),]=Lambda[(2*(k-1)-1):(2*(k-1)),]
}
```

```
  alpha[,(2*k-1):(2*k)]=alpha.star[,(2*k-1):(2*k)]%*%transpose(Lambda[(2*k-1):(2*k),])

   print(k)


}
```