STA601: Lab 7

Akbota Anuarbek

<u>Full conditionals:</u>

$\pi(\beta_{0i}|Y, X, parameters) \propto \prod_{j=1}^{10} L(Y_{ij}|x_j, \beta_{0i}, \beta_{1i}, \tau)\pi(\beta_{0i}) \propto$

$\propto \prod_{j=1}^{10} \exp\left(\frac{1}{2\tau}(Y_{ij} - \beta_{0i} - \beta_{1i}x_j)^2\right) \exp\left(\frac{1}{2\tau_0}(\beta_{0i} - \mu_0)^2\right) \propto$

$\propto \exp\left[\frac{1}{2\tau}(-2\beta_{0i}\sum_{j=1}^{10} Y_{ij} + 10\beta_{0i}^2 + 2\beta_{0i}\beta_{1i}\sum_{j=1}^{10} x_j) + \frac{1}{2\tau_0}(\beta_{0i}^2 - 2\mu_0\beta_{0i})\right] \propto$

$\propto \exp\left[(\frac{10}{2\tau} + \frac{1}{2\tau_0})\beta_{0i}^2 - 2\beta_{0i}(\frac{\sum_{j=1}^{10} Y_{ij}}{2\tau} - \frac{\beta_{1i}\sum_{j=1}^{10} x_j}{2\tau} + \frac{\mu_0}{2\tau_0})\right]$

$\propto N(M_0, V_0)$, where

$M_0 = V_0\big(\tau^{-1}(\sum_{j=1}^{10} Y_{ij} - \beta_{1i}\sum_{j=1}^{10} x_j) + \mu_0\tau_0^{-1}\big),$

$V_0 = (10\tau^{-1} + \tau_0^{-1})^{-1}$

$\pi(\beta_{1i}|Y, X, parameters) \propto \prod_{j=1}^{10} L(Y_{ij}|x_j, \beta_{0i}, \beta_{1i}, \tau)\pi(\beta_{1i}) \propto$

$\propto \prod_{j=1}^{10} \exp\left(\frac{1}{2\tau}(Y_{ij} - \beta_{0i} - \beta_{1i}x_j)^2\right) \exp\left(\frac{1}{2\tau_1}(\beta_{1i} - \mu_1)^2\right) \propto$

$\propto \exp\left[\frac{1}{2\tau}(-2\beta_{1i}\sum_{j=1}^{10} Y_{ij}x_j + 2\beta_{0i}\beta_{1i}\sum_{j=1}^{10} x_j + \beta_{1i}^2\sum_{j=1}^{10} x_j^2) + \frac{1}{2\tau_1}(\beta_{1i}^2 - 2\beta_{1i}\mu_1)\right] \propto$

$\propto \exp\left[(\frac{\sum_{j=1}^{10} x_j^2}{2\tau} + \frac{1}{2\tau_1})\beta_{1i}^2 - 2\beta_{1i}(\frac{\sum_{j=1}^{10} Y_{ij}x_j}{2\tau} - \frac{\beta_{0i}\sum j=1^{10} x_j}{2\tau} + \frac{\mu_1}{2\tau_1})\right] \propto$

$\propto N(M_1, V_1)$, where

$M_1 = V_1(\tau^{-1}(\sum_{j=1}^{10} Y_{ij}x_j - \beta_{0i}\sum_{j=1}^{10} x_j) + \mu_1\tau_1^{-1})$

$V_1 = (\tau^{-1}\sum_{j=1}^{10} x_j^2 + \tau_1^{-1})^{-1}$

The part of the code we needed to complete is given below *(you can also find the complete R code attached):*

```
### Fill in the blanks

 sigmai0= (10*(tau[k])^(-1)+(tau0[k])^(-1))^(-1/2)
 for(i in 1:N){
   mui0[i] = (sum(Y[i,])-beta1[k-1,i]*sum(X))/tau[k]+mu0[k]/tau0[k]
 }
 beta0[k,] = rnorm(N, mui0*sigmai0^2, sigmai0)

 sigmai1 = (SUMX.sq/tau[k]+ 1/tau1[k])^(-1/2)
 # SUMX.sq = sum(X^2) is defined above (outside of loop)
 for(i in 1:N){
   mui1[i] = (sum(Y[i,]*X)-beta0[k,i]*sum(X))/tau[k] + mu1[k]/tau1[k]
 }
 beta1[k,] = rnorm(N, mui1*sigmai1^2, sigmai1)
```

Burn-in first 1000 observations and calculate probability that new patient will have slope > 0.5 using remaining 4000 observations. We can approximate the probability by:

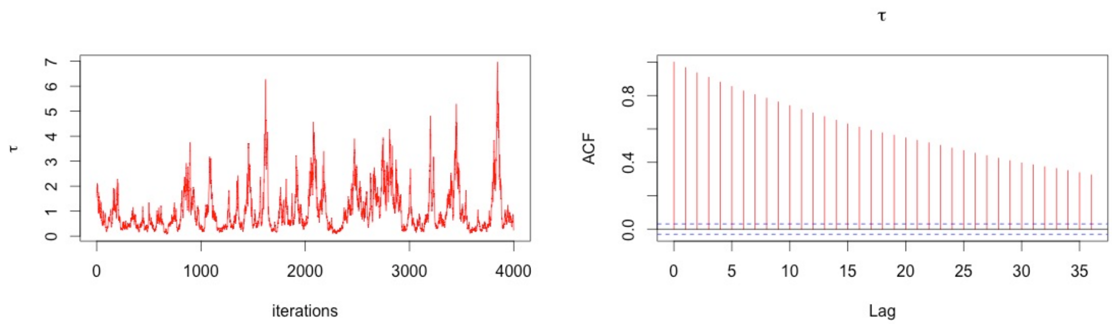
- generating random β_{1i} from $N(\mu_1, \tau_1)$ for each pair of μ_1, τ_1 that were sampled using Gibbs sampling.

- using generated sample of β_{1i}, approximate the probability by the share of $\beta_{1i} > 0.5$ in the sample, and obtain $Pr = 0.21925$.

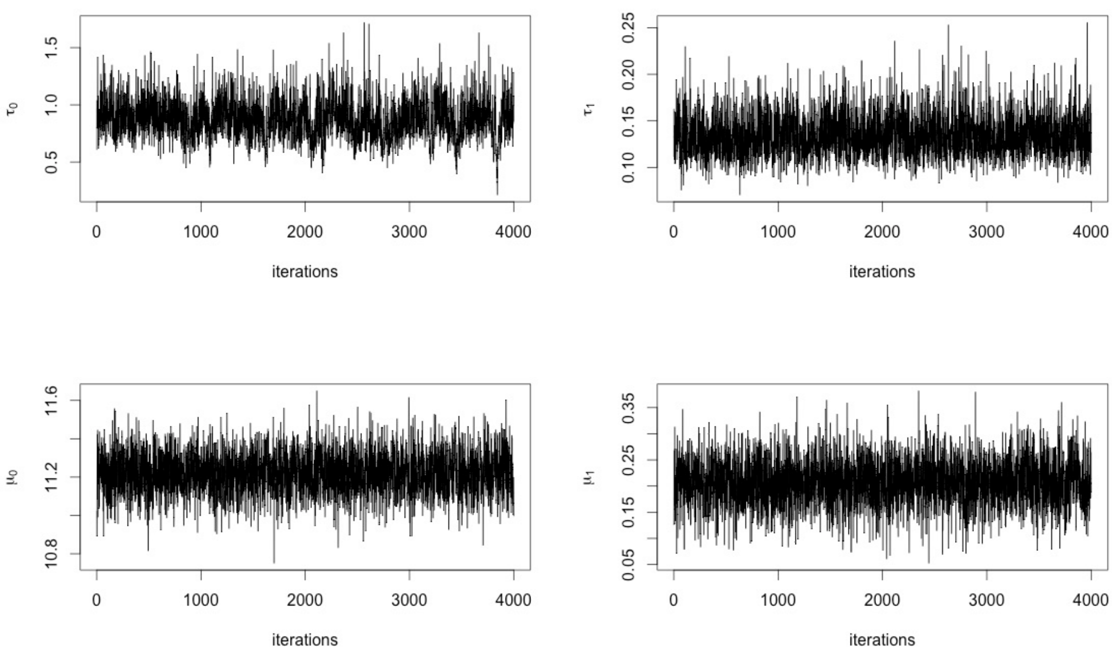
```
> p_beta1<-c()
> for(i in 1001:5000){
+   p_beta1[i-1000]=rnorm(1, mean=mu1[i], sd=sqrt(tau1[i]))
+ }
>
> mean(p_beta1>0.5)
[1] 0.21925
```

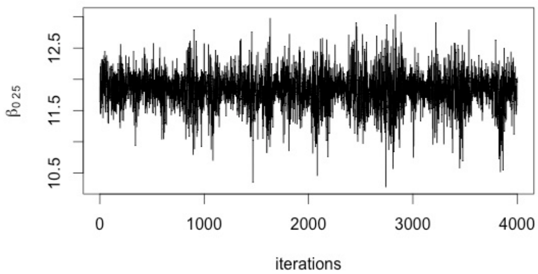
<u>Bonus</u>

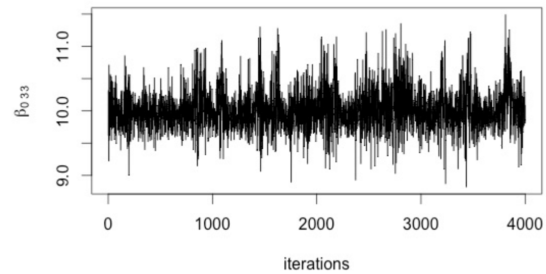
From the figures below we can see that the parameter τ is the one that has high autocorrelation:

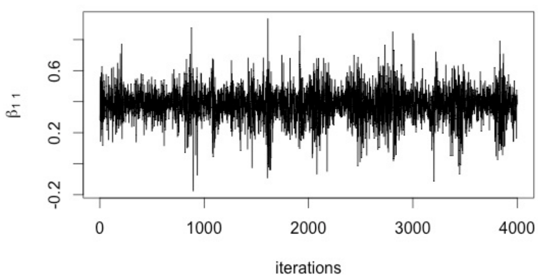


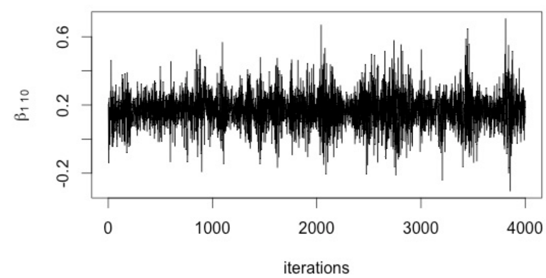
If we look at plots of other parameters, we can see that they don't have such high autocorrelation *(β_{0i} and β_{1i} have the same full conditional distributions for all i, so we look at plots for some i's only)*:





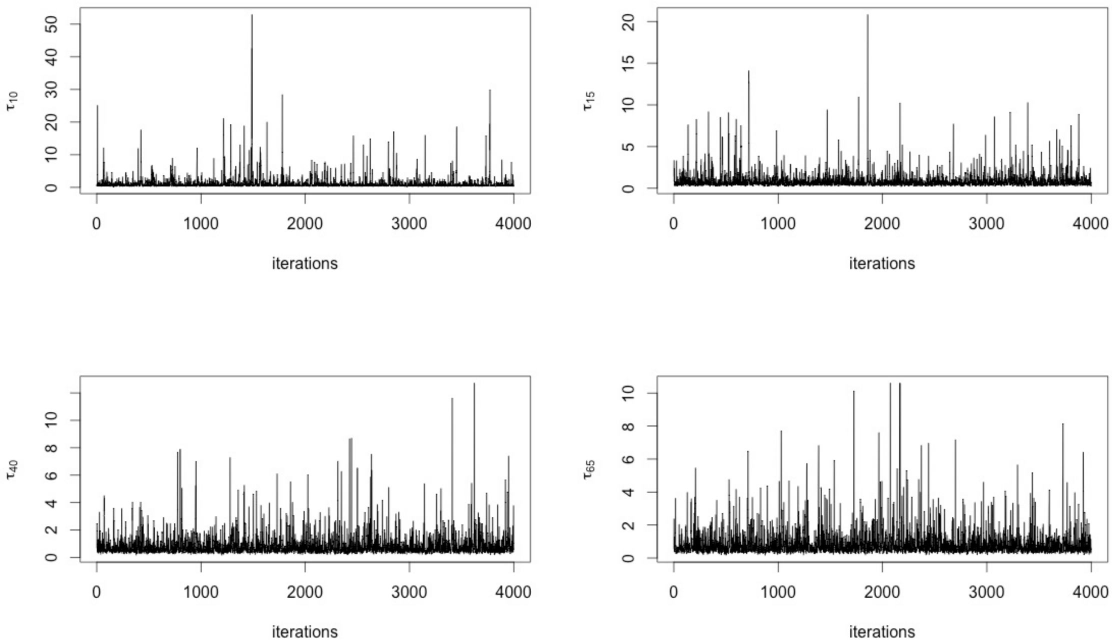






One way of dealing with high autocorrelation is *thinning* by saving every nth value in the sample. In this particular case, ACF function of τ illustrates that autocorrelation is very high even at lag 35. So, we need to choose every 50th or even 100th observation to decrease autocorrelation, which may be very inefficient. And in such case we need to sample more than 5000 observations with Gibbs, to be able to obtain good posterior approximations.

On the other hand, we can change our model a bit. In original model we were assuming that $Y_{ij} \sim N(\beta_{0i} + \beta_{1i}x_j, \tau)$, i.e. that each patient has the same variance τ, which is not generally true. So if we instead assume that $Y_{ij} \sim N(\beta_{0i} + \beta_{1i}x_j, \tau_i)$, and run Gibbs sampling that is very similar to the previous one, with only exception that we now draw τ_i for each parient, the resulting samples will have low autocorrelation *(see figures below)*. If needed we can also use thinning, but in this case it will be enough to save at most every 10th observation. *Complete R code for this case is given below*




```
setwd('/Users/akbota/Documents/STA601/Labs/lab7')
#setwd('/net/nfs1/s/grad/dnv2/Desktop')
data=read.table('hier.txt')
dim(data)


Y = as.matrix(na.omit(data[,2:11]))
X = 1:10 - mean(1:10)

#looking at some data, which is technically cheating
plot(X,Y[1,])
plot(X,Y[2,])
plot(X,Y[67,])

#install.packages('pscl')
library(pscl)
hist(invGammaDraws<-rigamma(10000,5,4))

#setting hyperparameters
a=5
```

```
lambda=4
m0 = 12
m1 = 1
s0.sq = 1
s1.sq = 1



N=nrow(Y)
T=ncol(Y)
M=5000


beta1 = beta0 = matrix(nrow=M, ncol=nrow(Y))

##setting starting values
tau=matrix(nrow=M, ncol=nrow(Y))

tau0=tau1=mu0=mu1=rep(NA,M)

beta1[1,]=1
beta0[1,]=10
tau[1,]=.3
tau0[1]=tau1[1]= mu1[1] = .3; mu0[1] = 11
mui0 = mui1 = rep(NA,N)
sigmai0 = sigmai1 = rep(NA,N)
SUMX.sq = sum(X^2)


for(k in 2:M){

  for(i in 1:N){
   tau[k,i] =rigamma(1, T/2+a, 1/2*(sum(Y[i,] - beta0[k-1,i]-beta1[k-1,i]*X)^2)+lambda)
  }

  sum.for.lambda0 =0
  for(i in 1:N){
    sum.for.lambda0 = sum.for.lambda0 + sum(beta0[k-1,i] - mu0[k-1])^2
  }
  tau0[k] =rigamma(1, N/2+a, 1/2*sum.for.lambda0+lambda)


  sum.for.lambda1 =0
  for(i in 1:N){
    sum.for.lambda1 = sum.for.lambda1 + sum(beta1[k-1,i] - mu1[k-1])^2
```

```
  }
  tau1[k] =rigamma(1, N/2+a, 1/2*sum.for.lambda1+lambda)


  sigstar<-(N/tau0[k] + 1/s0.sq)^(-1/2)
  mustar<- (sum(beta0[k-1,])/tau0[k] + m0/s0.sq )*sigstar^2
  mu0[k] = rnorm(1, mustar, sigstar)


  sigstar<-(N/tau1[k] + 1/s1.sq)^(-1/2)
  mustar<- (sum(beta1[k-1,])/tau1[k] + m1/s1.sq )*sigstar^2
  mu1[k] = rnorm(1, mustar, sigstar)



  ### Fill in the blanks


  for(i in 1:N){
    sigmai0[i]= (10*(tau[k,i])^(-1)+(tau0[k])^(-1))^(-1/2)
    mui0[i] = (sum(Y[i,])-beta1[k-1,i]*sum(X))/tau[k,i]+mu0[k]/tau0[k]
  }
  beta0[k,] = rnorm(N, mui0*sigmai0^2, sigmai0)

  sigmai1 = (SUMX.sq/tau[k]+ 1/tau1[k])^(-1/2)
  # SUMX.sq = sum(X^2) is defined above (outside of loop)
  for(i in 1:N){
    sigmai1[i] = (SUMX.sq/tau[k,i]+ 1/tau1[k])^(-1/2)
    mui1[i] = (sum(Y[i,]*X)-beta0[k,i]*sum(X))/tau[k,i] + mu1[k]/tau1[k]
  }
  beta1[k,] = rnorm(N, mui1*sigmai1^2, sigmai1)

  if(k%%1000==0){print(k)}
}
plot(tau[1001:5000,10], type='l', ylab=expression(tau[10]), xlab='iterations')
plot(tau[1001:5000,15], type='l', ylab=expression(tau[15]), xlab='iterations')
plot(tau[1001:5000,40], type='l', ylab=expression(tau[40]), xlab='iterations')
plot(tau[1001:5000,65], type='l', ylab=expression(tau[65]), xlab='iterations')
```