Week 8 Lab Solutions – MAST90125: Bayesian Statistical learning

Perform Gibbs sampling for linear models with proper priors for β.

In this week's lab, we discuss how to write Gibbs sampling code for linear models with proper priors. We consider the data in `USJudgeRatings.csv`, which is available on Canvas. We assume the variable `RTEN` is the response and the other variables as predictors.

Download `USJudgeRatings.csv` from Canvas.

Understand the code below that purports to perform Gibbs sampling for a variety of linear models. See if you can determine what the code is doing. You may find referring back to Lectures useful. Compare each other the posterior distributions obtained from the different priors.

Examples of Gibbs samplers for linear models

First, exercise the following two functions and see what they correspond to in Lecture?

```
Gibbs.lm1<-function(X,y,tau0,iter,burnin){
p <- dim(X)[2]
n <- dim(X)[1]
XTX <- crossprod(X)
XTXinv <-solve(XTX)
XTY <- crossprod(X,y)
betahat<-solve(XTX,XTY)
tau    <-tau0
library(mvtnorm)

par<-matrix(0,iter,p+1)
for( i in 1:iter){
  beta <- rmvnorm(1,mean=betahat,sigma=XTXinv/tau)
  beta <-as.numeric(beta)
  err  <- y-X%*%beta
  tau  <- rgamma(1,0.5*n,0.5*sum(err^2))
  par[i,] <-c(beta,tau)
}

par <-par[-c(1:burnin),]
return(par)
}
```

```
Gibbs.lm2<-function(X,y,tau0,iter,burnin){
p <- dim(X)[2]
n <-dim(X)[1]
svdX <-svd(X)
U    <-svdX$u
```

```r
Lambda<-svdX$d
V    <-svdX$v
Vbhat <- crossprod(U,y)/Lambda
tau <-tau0

vbeta<-rnorm(p)
par<-matrix(0,iter,p+1)
for( i in 1:iter){
  sqrttau<-sqrt(tau)
  vbeta <- rnorm(p,mean=Vbhat,sd=1/(sqrttau*Lambda) )
  beta <-V%*%vbeta
  err  <- y-X%*%beta
  tau  <- rgamma(1,0.5*n,0.5*sum(err^2))
  par[i,] <-c(beta,tau)
}

par <-par[-c(1:burnin),]
return(par)
}
```

Solution

```r
#Formatting data, and running chains.
#data<-read.csv('USJudgeRatings.csv')
data<-read.csv(file = './USJudgeRatings.csv',header=TRUE)
response<-data$RTEN  #response variable
n<-dim(data)[1]
intercept <-matrix(1,dim(data)[1],1) #Intercept (to be estimated without penalty)
Pred<-data[,2:12]         #Predictor variables.
Pred<-as.matrix(scale(Pred))
X   <-cbind(intercept,Pred)

system.time(chain1<-Gibbs.lm1(X=X,y=response,tau0=1,iter=10000,burnin=2000))
```

```
## Warning: package 'mvtnorm' was built under R version 4.3.1
```

```
##    user  system elapsed
##   1.244   0.026   1.342
```

```r
system.time(chain2<-Gibbs.lm1(X=X,y=response,tau0=5,iter=10000,burnin=2000))
```

```
##    user  system elapsed
##   1.209   0.019   1.229
```

```r
system.time(chain3<-Gibbs.lm1(X=X,y=response,tau0=0.2,iter=10000,burnin=2000))
```

```
##    user  system elapsed
##   1.217   0.019   1.236
```

```r
system.time(chain4<-Gibbs.lm2(X=X,y=response,tau0=1,iter=10000,burnin=2000))
```

```
##    user  system elapsed
##   0.052   0.004   0.056
```

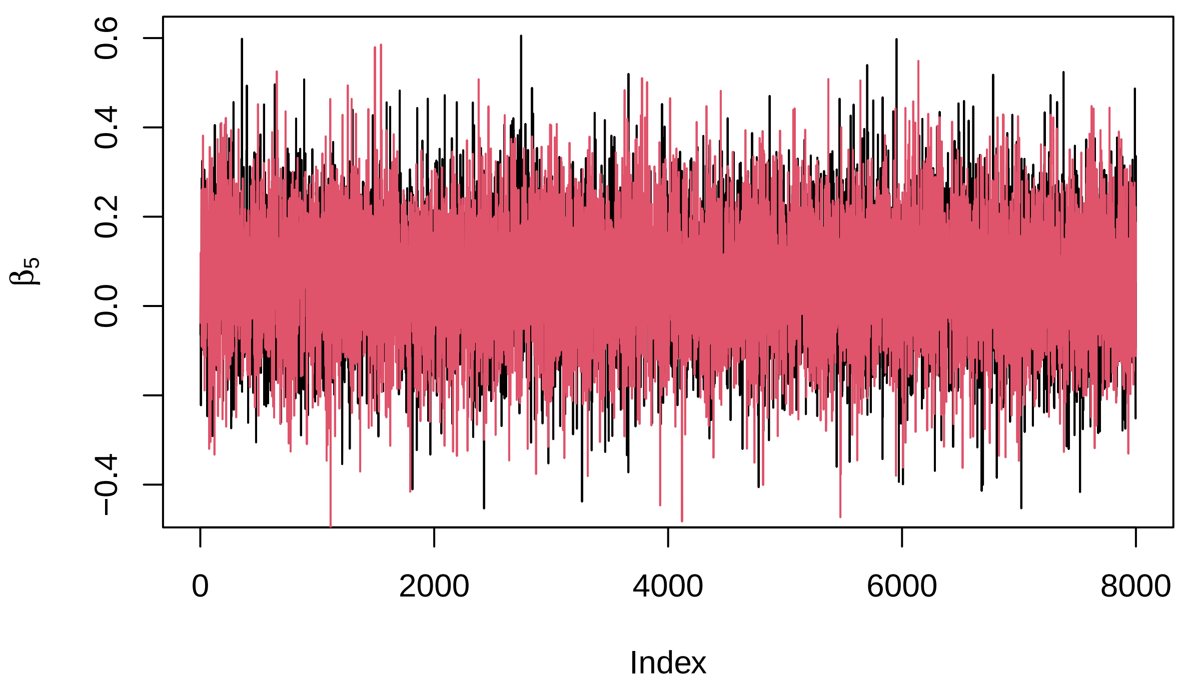
```r
system.time(chain5<-Gibbs.lm2(X=X,y=response,tau0=5,iter=10000,burnin=2000))
```

```
##    user  system elapsed
##   0.047   0.004   0.052
```

```r
system.time(chain6<-Gibbs.lm2(X=X,y=response,tau0=0.2,iter=10000,burnin=2000))
```

```
##    user  system elapsed
##   0.045   0.003   0.048
```

```r
#Comparing one co-efficient (the 5th)
plot(chain1[,5],type='l',ylab=expression(beta[5]))
lines(chain4[,5],type='l',col=2,ylab=expression(beta[5]))
```



What is the different between the above functions?

Then, we go on practising those tasks discussed in Lecture.

- Linear mixed model/ ridge regression (flat prior for $\boldsymbol{\beta}_0$, $p(\tau) = \mathrm{Ga}(\alpha_e, \gamma_e)$, where $\tau = (\sigma^2)^{-1}$), $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \sigma_\beta^2\mathbf{I})$, $(\sigma_\beta^2)^{-1} = \tau_\beta \sim \mathrm{Ga}(\alpha_\beta, \gamma_\beta)$.

```
#Inputs: iter: no of iterations.
#Z: covariate matrix for parameters with normal prior.
#X: covariate matrix for parameters with flat prior.
#y: response vector.
#burnin: no of initial iterations to throw out.
#taue_0, tauu_0, initial values for tau, \tau_\beta
#a.e, b.e, a.u, b.u, hyper-parameters for priors for \tau, \tau_\beta.
normalmm.Gibbs<-function(iter,Z,X,y,burnin,taue_0,tauu_0,a.u,b.u,a.e,b.e){
  n   <-length(y) #no. observations
  p   <-dim(X)[2] #no of fixed effect predictors.
  q   <-dim(Z)[2] #no of random effect levels
  tauu<-tauu_0
  taue<-taue_0
  beta0<-rnorm(p) #initial value for \beta_0 (parameters with flat prior 'fixed effects')
  u0   <-rnorm(q,0,sd=1/sqrt(tauu))
  #intial value for u_0 (parameters with normal prior ,'random effects')

  #Building combined predictor matrix.
  W<-cbind(X,Z)
  WTW <-crossprod(W)
  library(mvtnorm)

  #storing results.
  par <-matrix(0,iter,p+q+2)
    #matrix for storing iterations, p fixed effects, q random effects, 2,
    #because two inverse variance components.

  #Create modified identity matrix for joint posterior.
  I0  <-diag(p+q)
  diag(I0)[1:p]<-0

  #Calculate WTy
  WTy<-crossprod(W,y)

  for(i in 1:iter){
    #Conditional posteriors.
    tauu <-rgamma(1,a.u+0.5*q,b.u+0.5*sum(u0^2)) #sampling tau_u from conditional posterior.
    #Updating component of normal posterior for beta,u
    Prec <-WTW + tauu*I0/taue
    P.mean <- solve(Prec)%*%WTy
    P.var  <-solve(Prec)/taue
    betau <-rmvnorm(1,mean=P.mean,sigma=P.var)
      #sample beta, u from joint full conditional posterior.
    betau <-as.numeric(betau)
    err   <- y-W%*%betau
    taue  <-rgamma(1,a.e+0.5*n,b.e+0.5*sum(err^2)) #sample tau_e from conditional posterior.
    #storing iterations.
```

```
    par[i,]<-c(betau,1/sqrt(tauu),1/sqrt(taue))
      #Note we are storing standard deviation, not precisions.
    beta0  <-betau[1:p]
    u0     <-betau[p+1:q]
  }

par <-par[-c(1:burnin),] #throw out initial observations.
colnames(par)<-c(paste('beta',1:p,sep=''),paste('u',1:q,sep=''),'sigma_b','sigma_e')
 return(par)
}
```

Solution

```
#Formatting data, and running chains.
data<-read.csv(file = './USJudgeRatings.csv',header=TRUE)
response<-data$RTEN  #response variable
n<-dim(data)[1]
intercept <-matrix(1,dim(data)[1],1) #Intercept (to be estimated without penalty)
Pred<-data[,2:12]          #Predictor variables.
Pred<-as.matrix(scale(Pred))
X    <-cbind(intercept,Pred)



system.time(chain10<-normalmm.Gibbs(iter=10000,Z=Pred,X=intercept, y=response,
  burnin=2000,taue_0=1,tauu_0=1,a.u=0.001,b.u=0.001,a.e=0.001,b.e=0.001))
```

```
##    user  system elapsed
##   1.567   0.017   1.587
```

```
system.time(chain11<-normalmm.Gibbs(iter=10000,Z=Pred,X=intercept,y=response,
  burnin=2000,taue_0=0.2,tauu_0=5,a.u=0.001,b.u=0.001,a.e=0.001,b.e=0.001))
```

```
##    user  system elapsed
##   1.552   0.018   1.571
```

```
system.time(chain12<-normalmm.Gibbs(iter=10000,Z=Pred,X=intercept,y=response,
     burnin=2000,taue_0=5,tauu_0=0.2,a.u=0.001,b.u=0.001,a.e=0.001,b.e=0.001))
```

```
##    user  system elapsed
##   1.539   0.021   1.561
```

- LASSO. $\boldsymbol{\beta}_j$ drawn from Laplace prior with parameter λ assumed fixed. Implicit prior is $p(\beta_j|\sigma_j^2) = \mathcal{N}(0, \sigma_j^2)$, $p(\sigma_j^2) = \text{Exp}(\gamma^2/2)$.

```
#Inputs: iter: no of iterations.
#Z: covariate matrix for parameters with Lasso (Laplace) prior.
#X: covariate matrix for parameters with flat prior.
#y: response vector.
#burnin: no of initial iterations to throw out.
#taue_0, initial values for tau
#a.e, b.e, hyper-parameters for priors for \tau,
#lambda. Hyper-parameter for the Laplace prior. (assumed fixed)


normallasso.Gibbs<-function(iter,Z,X,y,burnin,taue_0,lambda,a.e,b.e){
  library(LaplacesDemon)
  n   <-length(y) #no. observations
  p   <-dim(X)[2] #no of fixed effect predictors.
  q   <-dim(Z)[2] #no of random effect levels
  taue<-taue_0   #initial chain for tau_e
  tauu <-rinvgaussian(q,lambda/abs(rnorm(q)),lambda^2)
  #augmenting by make sampling \beta_j ~ N(0,\sigma^2_j),1/\sigma^j ~Exp(\gamma^2/2).

  #Building combined predictor matrix.
  W<-cbind(X,Z)
  WTW <-crossprod(W)
  #Calculate WTy.
  WTy<-crossprod(W,y)
  library(mvtnorm)

  #storing results.
  par <-matrix(0,iter,p+q+1)

  for(i in 1:iter){
    #Conditional posteriors.

    #Updating component of normal posterior for beta,u
    Kinv  <-diag(p+q)
    diag(Kinv)[1:p]<-0
    diag(Kinv)[p+1:q]<-tauu

    Prec <-taue*WTW + Kinv
    P.var  <-solve(Prec)
    P.mean <- taue*P.var%*%WTy
    betau <-rmvnorm(1,mean=P.mean,sigma=P.var)
    #sampling \beta (with flat prior), u (With laplace prior) from joint full conditional.
    betau <-as.numeric(betau)
    err   <- y-W%*%betau
    taue  <-rgamma(1,a.e+0.5*n,b.e+0.5*sum(err^2)) #sample tau_e from conditional posterior.
    tauu <-rinvgaussian(q,lambda/abs(betau[-c(1:p)]),lambda^2)
       #sample 1/sigma^2=j = \tauu_j from full conditional posterior.

    #storing iterations.
    par[i,]<-c(betau,1/sqrt(taue))
      #note despite sampling tau_e, we store the standard deviation sigma_e instead.
```

```r
  }

par <-par[-c(1:burnin),]#throw out initial iterations.
colnames(par)<-c(paste('beta',1:p,sep=''),paste('u',1:q,sep=''),'sigma_e')
 return(par)
}
```

Solution

```r
system.time(chain13<-normallasso.Gibbs(iter=10000,Z=Pred,X=intercept,y=response,
  burnin=2000,taue_0=1,lambda=0.4,a.e=0.001,b.e=0.001))
```

```
##
## Attaching package: 'LaplacesDemon'

## The following objects are masked from 'package:mvtnorm':
##
##     dmvt, rmvt

##    user  system elapsed
##   1.615   0.024   1.704
```

```r
system.time(chain14<-normallasso.Gibbs(iter=10000,Z=Pred,X=intercept,y=response,
  burnin=2000,taue_0=0.2,lambda=0.4,a.e=0.001,b.e=0.001))
```

```
##    user  system elapsed
##   1.565   0.017   1.583
```

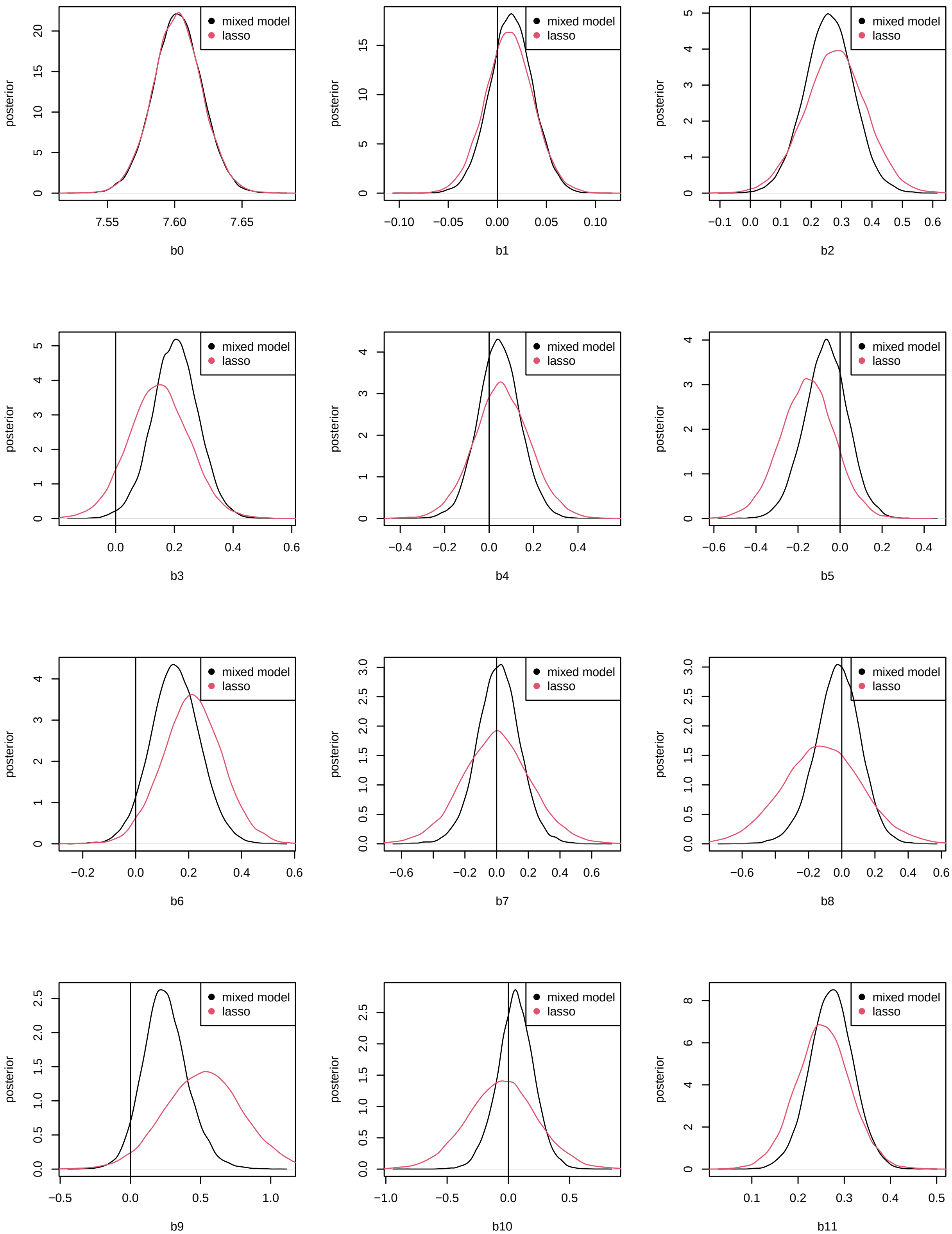
```r
system.time(chain15<-normallasso.Gibbs(iter=10000,Z=Pred,X=intercept,y=response,
  burnin=2000,taue_0=5,lambda=0.4,a.e=0.001,b.e=0.001))
```

```
##    user  system elapsed
##   1.577   0.020   1.599
```

- LASSO to linear mixed model posteriors for $\boldsymbol{\beta}$.

```r
chainmm<-rbind(chain10,chain11,chain12)
chainlasso<-rbind(chain13,chain14,chain15)

par(mfrow = c(4,3))
for(i in 1:12){
plot(density(chainmm[,i]),xlab=paste('b',i-1,sep=''),ylab='posterior',main='')
lines(density(chainlasso[,i]),col=2)
legend('topright',legend=c('mixed model','lasso'),col=1:2,pch=19)
abline(v=0)
}
```


mixed model
lasso
posterior
b0
b1
b2
b3
b4
b5
b6
b7
b8
b9
b10
b11
7.55
7.60
7.65
−0.10
−0.05
0.00
0.05
0.10