# Stat 432 Homework 7

*Assigned: Mar 09, 2019; Due: 11:59pm Mar 15, 2019*

Before starting this homework, you should read the rlab file of BayRegLinReg on the course website carefully.

Question 1 (Bayesian updates) [5 points]

Consider a simple normal model for the height $y$ of students where $y|\theta \sim N(\theta, 16)$. Assume a conjugate prior $\theta \sim N(65, 36)$. We measure the height of two students and observe $y_1 = 68$ and $y_2 = 72$.

(a) Make use of the computed formulae on page 41 of the lecture note:

First, use $y_1$ to update $\theta$ (1 point):

$$\theta|y_1 \sim N(\mu_1, \sigma_1^2)$$
$$\mu_1 = \frac{\frac{65}{36} + \frac{68}{16}}{\frac{1}{36} + \frac{1}{16}} = 67.0769$$
$$\sigma_1^2 = \frac{1}{\frac{1}{36} + \frac{1}{16}} = 11.0769$$

Then we use $y_2$ to update $\theta|y_1$ (1 point):

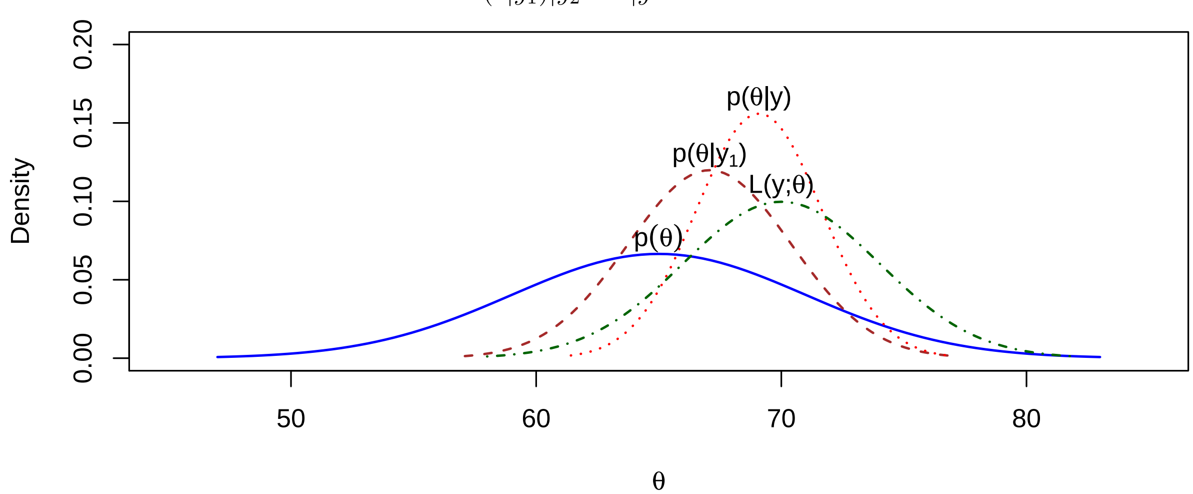
$$(\theta|y_1)|y_2 \sim N(\mu_2, \sigma_2^2)$$
$$\mu_2 = \frac{\frac{\mu_1}{\sigma_1^2} + \frac{72}{16}}{\frac{1}{\sigma_1^2} + \frac{1}{16}} = \frac{67.0769/11.0769 + 72/16}{1/11.0769 + 1/16} = 69.0909$$
$$\sigma_2^2 = \frac{1}{\frac{1}{\sigma_1^2} + \frac{1}{16}} = \frac{1}{11.0769} + \frac{1}{16} = 6.5454$$

(b) On the same time, if we use $y = \{68, 71\}$ to update $\theta$ simultaneously, then we have (1 point)

$$\theta|y \sim N(\tilde{\mu}_2, \tilde{\sigma}_2^2)$$
$$\tilde{\mu}_2 = \frac{\frac{65}{36} + \frac{68+72}{16}}{\frac{1}{36} + \frac{2}{16}} = \frac{65/36 + 70/8}{1/36 + 1/8} = 69.0909$$
$$\tilde{\sigma}_2^2 = \frac{1}{\frac{1}{36} + \frac{2}{16}} = \frac{1}{\frac{1}{36} + \frac{1}{8}} = 6.5455$$

We see $\tilde{\mu}_2 = \mu_2$, $\tilde{\sigma}_2 = \sigma_2$, therefore

$$(\theta|y_1)|y_2 \stackrel{d}{=} \theta|y$$




(c)

The prior, the posterior with $y_1$, the posterior with all data $y$, and the scaled likelihood function are plotted in the above figure (1 point). From the plot we could find (1 point, has to include some commnets to earn 1 point; comments do not have to be same):

- The posteriors are the comprise between the prior and the likelihood.
- Updating the prior with data streamed sequentially results in the same posterior as the data come in simultaneously.
- With more and more data, the posterior becomes more concentrated towards the likelihood function.

  Question 2 (Bayesian linear regression) [5 points]

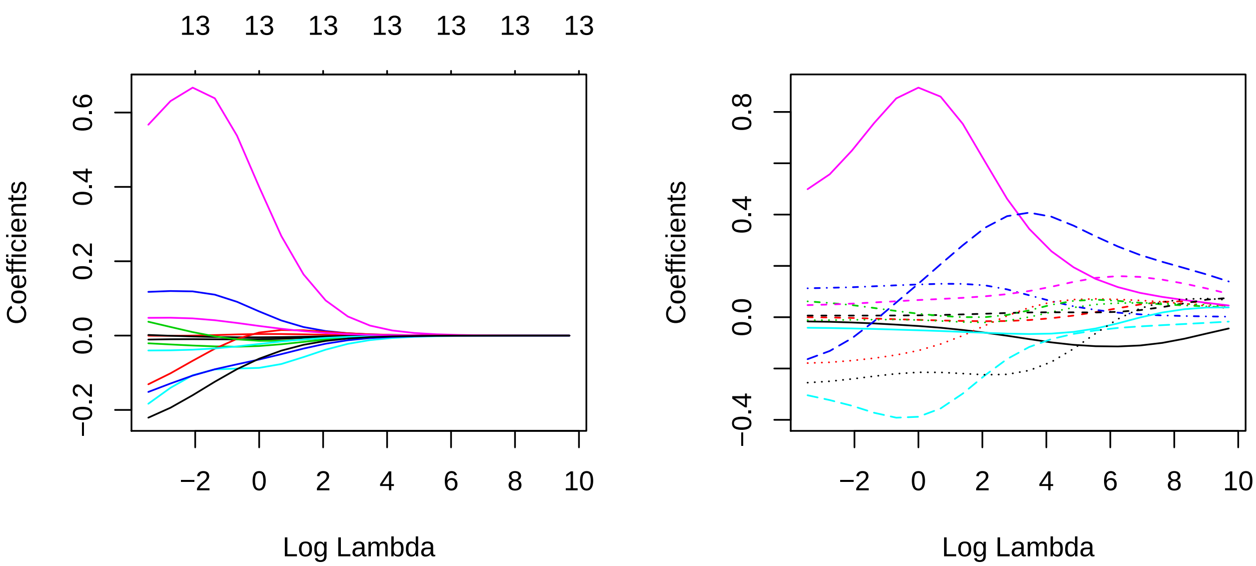
Now we use the same Boston housig data to do a Bayesian linear regression.

```
data(Boston, package="MASS")
# head(Boston)

useLog = c(1,3,5,6,8,9,10,14)
Boston[,useLog] = log(Boston[,useLog])
Boston[,2] = Boston[,2] / 10
Boston[,7] = Boston[,7]^2.5 / 10^4
Boston[,11] = exp(0.4 * Boston[,11])/1000
Boston[,12] = Boston[,12] / 100
Boston[,13] = sqrt(Boston[,13])
```

First, we fit ridge regression on a chosen series of $\lambda$'s using `cv.glmnet` function (2 points). Then obtain the Bayesian solution of the corresponding $\lambda$'s (2 points). Finally, we plot the regression coefficients $\beta$ versus $\lambda$ and compare their resuls side by side (1 point).

```
# specify lambdas
lambdas=2^(-5:14)
K=length(lambdas)
# frequestit solution
library(glmnet)
X=as.matrix(Boston[,-14]); y=Boston[,14]
ridge_freq=cv.glmnet(X,y, lambda=lambdas, nfolds = 10, alpha = 0)
# Bayesian solution
# # select lambdas
# lambdas=rev(ridge_freq$glmnet.fit$lambda[seq(1,length(ridge_freq$glmnet.fit$lambda),5)])
# K=length(lambdas)
# predictors
n=length(y); p=dim(X)[2]
X1=cbind(intercept=1,X)
XTX=t(X1)%*%X1; XTy=t(X1)%*%y
# prior parameters
mu0=rep(0,p+1);
nu0=1; sigma20=.5
nu_n=nu0+n
# define rinvchisq
rinvchisq=function(n,df,scale)(df *scale)/rchisq(n, df = df)
# generate samples using Gibbs sampler: similar as coordinate descent algorithm
Nsamp=1e4
beta_samp=matrix(NA,Nsamp,p+1); sigma2_samp=rep(NA,Nsamp)
library(mvtnorm)
set.seed(2019)
BETA_m=matrix(NA,K,p+1); BETA_q25=matrix(NA,K,p+1); BETA_q975=matrix(NA,K,p+1);
for(k in 1:K){
```

```r
  # obtain samples
  sigma2_samp[1]=rinvchisq(1,nu0,sigma20); Lambda0=sigma2_samp[1]/lambdas[k]*diag(p+1)
  beta_samp[1,]=rmvnorm(1,mu0,Lambda0);
  for(i in 2:Nsamp){
    # update beta, conditioned on sigma2
    Lambda_n=solve(XTX+lambdas[k]*diag(p+1))
    mu_n=Lambda_n%*%(XTy+lambdas[k]*mu0)
    Lambda_n=Lambda_n*sigma2_samp[i-1]
    beta_samp[i,]=rmvnorm(1,mu_n,Lambda_n)
    # update sigma2, conditioned on beta
    sigma2_n=(nu0*sigma20+sum((y-X1%*%beta_samp[i,])^2))/nu_n
    sigma2_samp[i]=rinvchisq(1,nu_n,sigma2_n)
  }
  # discard the first 2000 to reduce the autocorrelation for the estimates
  BETA_m[k,]=apply(beta_samp[-(1:2000),],2,mean)
  BETA_q25[k,]=apply(beta_samp[-(1:2000),],2,function(x)quantile(x,.025))
  BETA_q975[k,]=apply(beta_samp[-(1:2000),],2,function(x)quantile(x,.975))
}

# plot
par(mfrow = c(1, 2))
plot(ridge_freq$glmnet.fit, "lambda") # frequentist
# matplot(log(lambdas),t(ridge_freq$glmnet.fit$beta[,rev(seq(1,length(ridge_freq$glmnet.fit$lambda),5))
matplot(log(lambdas),BETA_m[,-1],type='l',xlab='Log Lambda',ylab='Coefficients')
```



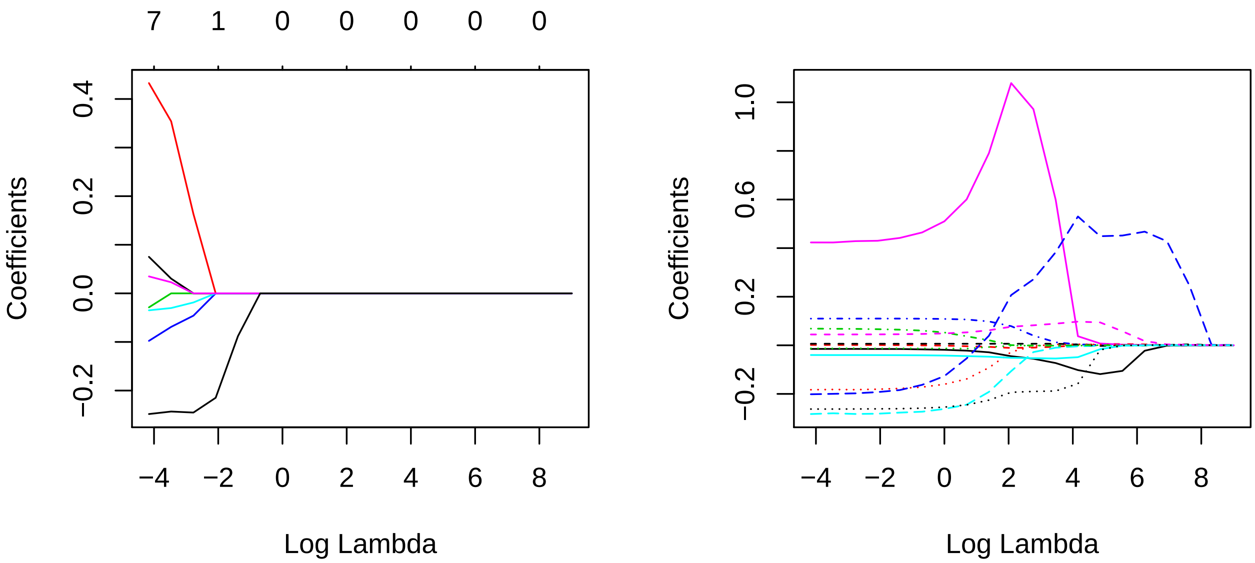

Both solutions have shrinking magnitude towards zero in the regression coefficient vector $\beta$ as $\lambda$ gets bigger. However, the shrinking effect of Bayesian solution is smaller than the frequentist solution. This is due to the presence of the other parameter $\sigma^2$.

Extra-Credit Question [5 points]

Now we repeart the similar procedure to compare the frequentist (2 points) and Bayesian solutions (2 points) for lasso.

```r
# specify lambdas
lambdas=2^(-6:13)
K=length(lambdas)
```

```r
# frequestit solution
lasso_freq=cv.glmnet(X,y, lambda=lambdas, nfolds = 10, alpha = 1)
# Bayesian solution
# generate samples using Gibbs sampler: similar as coordinate descent algorithm
Nsamp=1e4
beta_samp=matrix(NA,Nsamp,p+1); sigma2_samp=rep(NA,Nsamp); tau2_samp=matrix(NA,Nsamp,p+1)
library(SuppDists)
set.seed(2019)
BETA_m=matrix(NA,K,p+1); BETA_q25=matrix(NA,K,p+1); BETA_q975=matrix(NA,K,p+1);
for(k in 1:K){
  # obtain samples
  beta_samp[1,]=rep(.01,p+1);
  sigma2_samp[1]=100;
  tau2_samp[1,]=1/rinvGauss(rep(1,p+1),
                          sqrt(sigma2_samp[1])*abs(lambdas[k]/beta_samp[1,]),lambdas[k]^2)
  for(i in 2:Nsamp){
    # update beta, conditioned on sigma2
    Lambda_n=solve(XTX+diag(1/tau2_samp[i-1,]))
    mu_n=Lambda_n%*%XTy
    Lambda_n=Lambda_n*sigma2_samp[i-1]
    beta_samp[i,]=rmvnorm(1,mu_n,Lambda_n)
    # update sigma2, conditioned on beta and tau2
    sigma2_n=(sum((y-X1%*%beta_samp[i,])^2)+sum(beta_samp[i,]^2/tau2_samp[i-1,]))/2
    sigma2_samp[i]=1/rgamma(1,(n+p)/2,sigma2_n)
    # update tau2, conditioned on beta and sigma2
    tau2_samp[i,]=1/rinvGauss(rep(1,p+1),
                            sqrt(sigma2_samp[i])*abs(lambdas[k]/beta_samp[i,]),lambdas[k]^2)
  }
  # discard the first 2000 to reduce the autocorrelation for the estimates
  BETA_m[k,]=apply(beta_samp[-(1:2000),],2,mean)
  BETA_q25[k,]=apply(beta_samp[-(1:2000),],2,function(x)quantile(x,.025))
  BETA_q975[k,]=apply(beta_samp[-(1:2000),],2,function(x)quantile(x,.975))
}

# plot
par(mfrow = c(1, 2))
plot(lasso_freq$glmnet.fit, "lambda") # frequentist
matplot(log(lambdas),BETA_m[,-1],type='l',xlab='Log Lambda',ylab='Coefficients')
```




Again we see the similar shinking effect of both solutions with the stronger shrinking effect in the frequentist solution in the above figure (1 point).