Homework Three
Bayesian Statistical Methods

Akshay Umashankar
Department of Educational Psychology
The University of Texas at Austin
Email: akshayumashankar38@gmail.com

Vincent Indina
Department of Petroleum Engineering
The University of Texas at Austin
Email: vincentindina@utexas.edu

March 7, 2023

1 Develop a MCMC algorithm to fit the Bayesian regression model:

$$y_i \sim Beta(a_i, b_i)\ , for\ i = 1, ..., n\ , \tag{1}$$

$$a_i = \mu_i \tau\ , \tag{2}$$

$$b_i = (1 - \mu_i)\tau\ , \tag{3}$$

where $\tau > 0$ controls the precision, and the mean μ_i is linked to a set of covariates x_i using $logit(\mu_i) = \boldsymbol{x_i'\beta}$. For this model, assume the priors

$$\boldsymbol{\beta} \sim N(\boldsymbol{\mu}_\beta, \sigma^2_\beta \boldsymbol{I}) \tag{4}$$

$$\tau \sim Gamma(\gamma_1, \gamma_2) \tag{5}$$

Posterior distribution:

$$[\boldsymbol{\beta}, \tau | y_i] \propto [y_i | \boldsymbol{\beta}, \tau][\boldsymbol{\beta}][\tau]$$

Full conditional for β:

$$[\boldsymbol{\beta}, \tau | y_i] \propto [y_i | \boldsymbol{\beta}, \tau][\boldsymbol{\beta}]$$

$$[\boldsymbol{\beta}, \tau | y_i] \propto \prod_{i=1}^{n} (\frac{\Gamma(a_i + b_i)}{\Gamma(a_i)\Gamma(b_i)} y_i^{a_i - 1}(1 - y_i)^{b_i - 1})(2\pi\sigma_\beta^2)^{\frac{-1}{2}} \exp\{\frac{-1}{2}(\boldsymbol{\beta} - \boldsymbol{\mu_\beta})^T(\sigma_\beta^2)^- 1(\boldsymbol{\beta} - \boldsymbol{\mu_\beta})\}$$

Full conditional for τ:

$$[\boldsymbol{\beta}, \tau | y_i] \propto [y_i | \boldsymbol{\beta}, \tau][\boldsymbol{\tau}]$$

$$[\boldsymbol{\beta}, \tau | y_i] \propto \prod_{i=1}^{n} (\frac{\Gamma(a_i + b_i)}{\Gamma(a_i)\Gamma(b_i)} y_i^{a_i - 1}(1 - y_i)^{b_i - 1}) \frac{\gamma_2^{\gamma_2}}{\Gamma(\gamma_1)} \tau^{\gamma_1 - 1} \exp\{-\gamma_2 \tau\}$$

See the Appendix B for the full MCMC algorithm written in R.

The MCMC algorithm was created using two separate updates: one for beta and one for tau. Both were done using a Metropolis-Hastings (MH) acceptance criteria.

After setting up the hyper-parameters and subroutines for the logit and inverse logit functions, β was sampled using the normal distribution with a tuning parameter of 0.15. τ, μ_i, a_i, and b_i, were then calculated based off of the β proposal. If the MH criteria was accepted, then it would update the values accordingly.

τ was sampled similarly with the added criteria that the proposal must be greater than 0 in order to follow the constraints of the support. Once this constraint is met, then the MH algorithm can continue to run similar to the β parameter with a tuning parameter of 1.

After this, the posterior mean for the β parameters was calculated and the posterior p-value was calculated based on the MSE.

2 **Use the MCMC algorithm to fit the beta regression model to the gasoline data set using yield as the response variable and the three covariates below:**
(a) gravity
(b) pressure
(c) temp
You may want to standardize the covariates before fitting the model. Tune the algorithm as needed and check the trace plots for convergence.

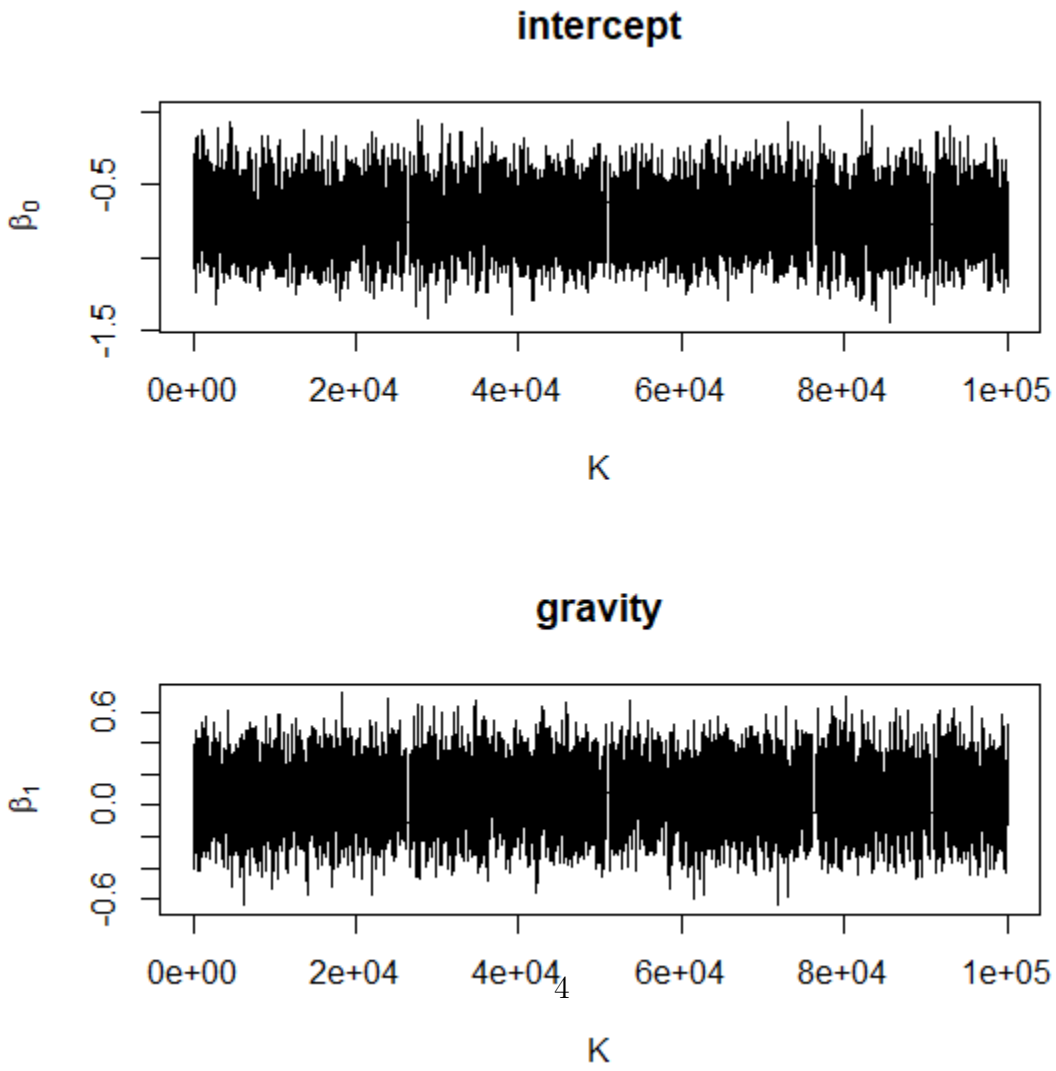

Figure 1: Traceplots of β_0 and β_1

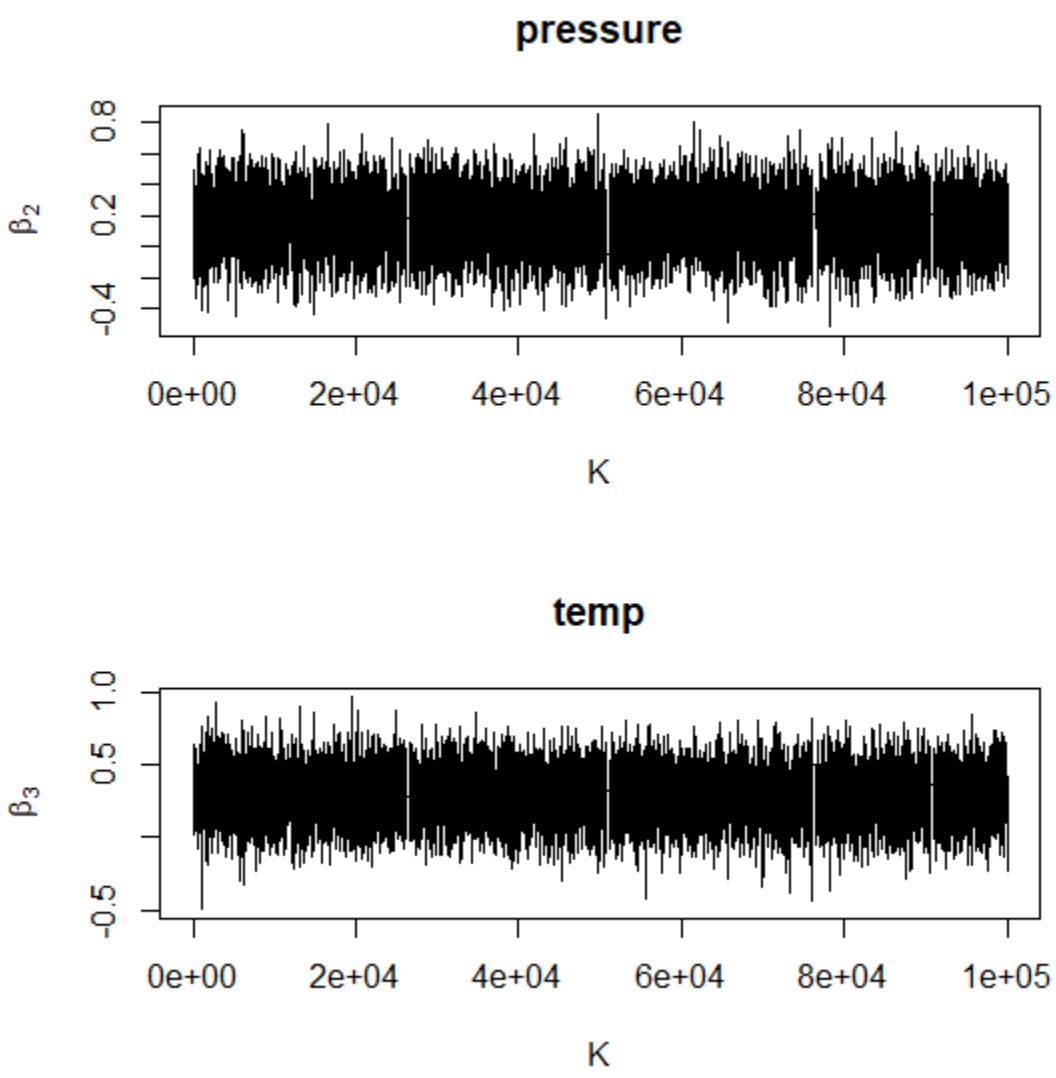

Figure 2: Traceplots of β_2 and β_3

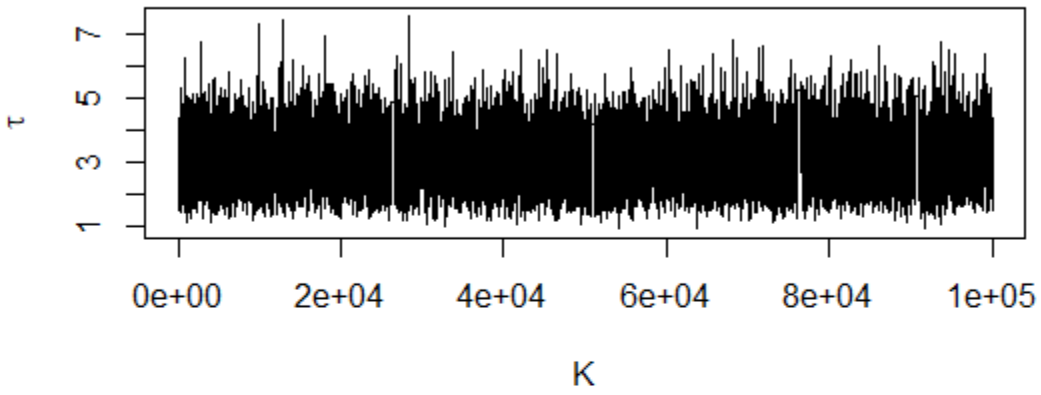

Figure 3: Traceplots of τ

Based on the trace plots for β and τ (***Figure* 1, *Figure* 2, *and Figure* 3**), the MCMC algorithm seems to have run well. The acceptance rate for β and τ are 0.33 and 0.39 respectively. The following table contains the posterior mean for β:

Parameters	Posterior μ
β_0	-0.7347155
β_1	0.05798116
β_2	0.1545265
β_3	0.2873034

3 Calculate the posterior predictive p-value using MSE as a statistic for this model and data set. Is this model appropriate for these data based on the p-value?

See the Appendix B for the full MCMC algorithm written in R.

The posterior predictive p-value using MSE came to be 0.69036. Based on this p-value, this model is appropriate for these data because the p-value is not at either extremes of 0 and 1.

4 If the model is appropriate for these data, what inference can you make about the predictors of yield proportion?

The inference we can make about the predictors of yield proportion are that gravity, pressure, and temperature all positively relate to the yield proportion. Our conclusion makes sense because of the fact that oil gravity relates to the composition of hydrocarbon components in the crude oil. Moreover, the crude oil vapor pressure and temperature are also dependent on the composition of hydrocarbon components which have different vapor pressure and temperature. The amount of gasoline that can be extracted from the crude oil is dependent on the volume of the hydrocarbon component that can be processed to gasoline. Therefore, the amount of gasoline yield is also dependent to the crude oil gravity, vapor pressure, and temperature.

Author Contributions

- VI developed and fit the model using MCMC, performed priors tuning, tuned the posterior p-value, and wrote the homework solution document.

- AU derived the posterior p-value, checked the MCMC code, analyzed the data set, wrote and reviewed the homework solution document.

APPENDIX A: R Script

```
#Import data
library(betareg)
data("GasolineYield", package="betareg")
crude.df <- GasolineYield
crude.df

#Standardize covariates or dependent variables
crude.df$gravity=scale(crude.df$gravity)
crude.df$pressure=scale(crude.df$pressure)
crude.df$temp=scale(crude.df$temp)

crude.df

#Create data vector and design matrix
y=crude.df$yield
X=model.matrix(yield~gravity+pressure+temp,data=crude.df)

N=dim(crude.df)[1]
p=dim(X)[2]

#Fit Beta regression model
source("hw3.betareg.mcmc.R")
n.mcmc=100000

mu.beta=0
s2.beta=0.075
beta.tune=0.15
gamma1.tau=1
gamma2.tau=4

out.mcmc=hw3.betareg.mcmc(y,X,mu.beta,s2.beta,beta.tune,gamma1.tau,gamma2.tau,n.mcmc)

#Trace plots
layout(matrix(1:2,2,1))
plot(out.mcmc$beta.save[1,],main="intercept",ylab=bquote(beta[0]),xlab="K",type="l",lty=1)
plot(out.mcmc$beta.save[2,],main="gravity",ylab=bquote(beta[1]),xlab="K",type="l",lty=1)

layout(matrix(1:2,2,1))
plot(out.mcmc$beta.save[3,],main="pressure",ylab=bquote(beta[2]),xlab="K",type="l",lty=1)
plot(out.mcmc$beta.save[4,],main="temp",ylab=bquote(beta[3]),xlab="K",type="l",lty=1)

plot(out.mcmc$tau.save,ylab=bquote(tau),xlab="K",type="l",lty=1)
```

APPENDIX B: MCMC Algorithm

```
hw3.betareg.mcmc<-function(y,X,beta.mn,beta.var,beta.tune,tau.gamma1,tau.gamma2,n.mcmc){

#Subroutines

logit <- function(theta){
  log(theta/(1-theta))
}

logit.inv <- function(z){
  exp(z)/(1+exp(z))
}

#Preliminary variables
n.burn=round(n.mcmc/10)
X=as.matrix(X)
y=as.vector(y)
n=length(y)
p=dim(X)[2]

beta.save=matrix(0,p,n.mcmc)
tau.save=rep(0,n.mcmc)
mse.y=rep(0,n.mcmc)
mse.ypred=rep(0,n.mcmc)
msediff.save=rep(0,n.mcmc)
ypred.save=rep(0,n.mcmc)
mse.save=rep(0,n.mcmc)



#Starting values
beta=rnorm(p,beta.mn,sqrt(beta.var))
tau=3 #positive number
tau.tune=1

mui=logit.inv(X%*%beta)
ai=mui*tau
bi=(1-mui)*tau

beta.acc=1
tau.acc=1

#MCMC Loop
for (k in 1:n.mcmc){
  if(k%%1000==0) cat(k," ")

  #Sample Beta
  beta.star=rnorm(p,beta,beta.tune)
  mui.star=logit.inv(X%*%beta.star)
  ai.star=mui.star*tau
```

```
  bi.star=(1-mui.star)*tau

  mh1=sum(dbeta(y,shape1= ai.star,shape2 = bi.star,log=TRUE))+sum(dnorm(beta.star,beta.mn,sqrt(beta.
  mh2=sum(dbeta(y,shape1= ai,shape2= bi,log=TRUE))+sum(dnorm(beta,beta.mn,sqrt(beta.var),log=TRUE))
  mh.beta=exp(mh1-mh2)

  #Updates beta
  if(mh.beta>runif(1)){
    beta=beta.star
    mui=mui.star
    ai=ai.star
    bi=bi.star
    beta.acc=beta.acc+1
  }

  #Sample tau
  tau.star=rnorm(1,tau,tau.tune)

  #Updates tau
  if(tau.star>0){
    ai.star=mui*tau.star
    bi.star=(1-mui)*tau.star

    mh1=sum(dbeta(y,shape1= ai.star,shape2= bi.star,log=TRUE))+sum(dgamma(tau.star,tau.gamma1,tau.ga
    mh2=sum(dbeta(y,shape1= ai,shape2 = bi,log=TRUE))+sum(dgamma(tau,tau.gamma1,tau.gamma2,log=TRUE)
    mh.tau=exp(mh1-mh2)

    if(mh.tau>runif(1)){
      tau=tau.star
      mui=mui.star
      ai=ai.star
      bi=bi.star
      tau.acc=tau.acc+1
    }

  }

  #Obtain predictions

  y.pred=rbeta(1,shape1=ai,shape2=bi)

  mse.y[k]=mean((y-mui)^2)
  mse.ypred[k]=mean((y.pred-mui)^2)
  msediff.save[k]=mse.ypred[k]-mse.y[k]
  mse.save[k]=mean((y.pred-y)^2)

  #Save Samples
  beta.save[,k]=beta
  tau.save[k]=tau
  ypred.save[k]=y.pred
}
```

```
cat("Posterior mean for beta:","\n")
cat(apply(beta.save[,-(1:n.burn)],1,mean),"\n")

#Calculate P-value based on MSE
p.value=sum(mse.ypred>mse.y)/n.mcmc
cat("Posterior p-value:", p.value)


#  Write output
list(y=y,X=X,n.mcmc=n.mcmc,beta.save=beta.save,tau.save=tau.save,beta.acc=beta.acc,tau.acc=tau.acc,m
}
```