Lab Report in Bayesian Learning

Laboration 3

TDDE07

David Grönberg, Erik Lindgren
Davgr686, Erili040

LINKÖPINGS UNIVERSITET

17-05-2020

Contents

List of Figures

List of Tables

1. Assignments

1.1 Assignment 1

We are given a dataset containing daily measurements of precipitation between the years of 1948 to 1983. In this assignment we will analyse the data using first a normal model and then a mixture normal model. The results of the models will then be compared graphically.

1.1.1 a)

We assume that the daily precipitation are independent normally distributed, where $\mu \sim N(\mu_0, \tau_0^2)$ and $\sigma^2 \sim Inv - \chi^2(v_0, \sigma_0^2)$ The first task of the assignment is to implement a Gibbs sampler that simulates from the joint posterior $p(\mu_0, \sigma_0^2|y_1, ..., y_n)$.

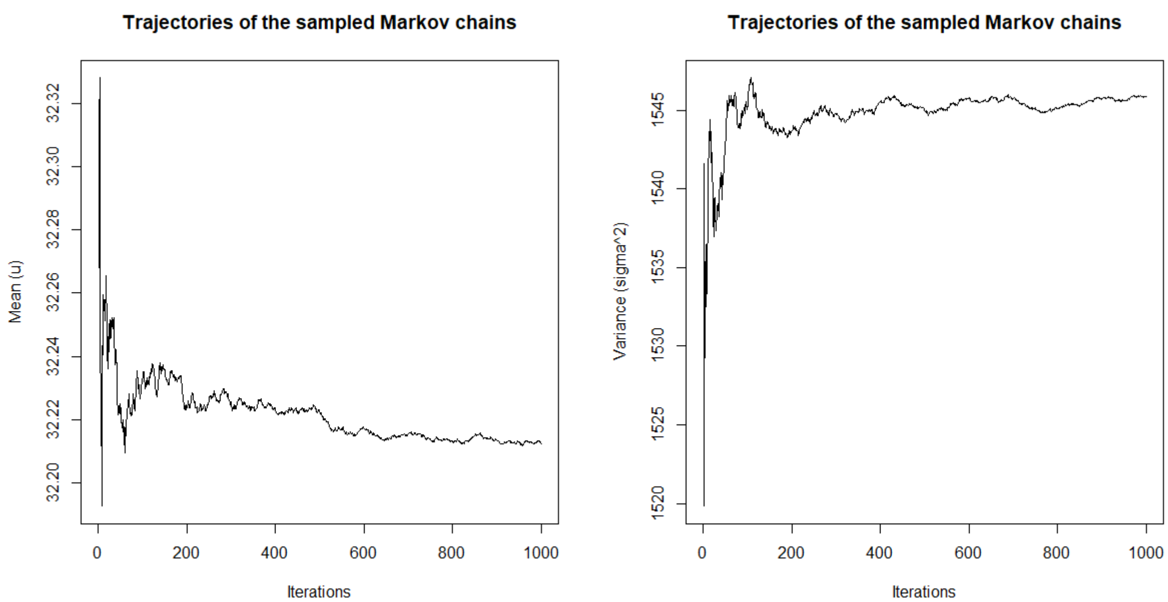

Figure 1.1: Trajectories of the sampled Markov chains for μ (left) and σ^2 (right).

In figure 1.1 we can see the trajectories of the sampled Markov chains for μ and σ^2. From the figure we can see that both of the trajectories converge, μ towards 32 and σ^2 towards 1546.

1.1.2 b)

In the next task we instead assume that the daily precipitation follow an iid two-component mixture of normals model:

$$p(y_i|\mu, \sigma^2, \pi) = \pi N(y_i|\mu_1, \sigma_1^2) + (1-\pi)N(y_i|\mu_2, \sigma_2^2)$$

where $\mu = (\mu_1, \mu_2)$ and $\sigma^2 = (\sigma_1^2, \sigma_2^2)$.

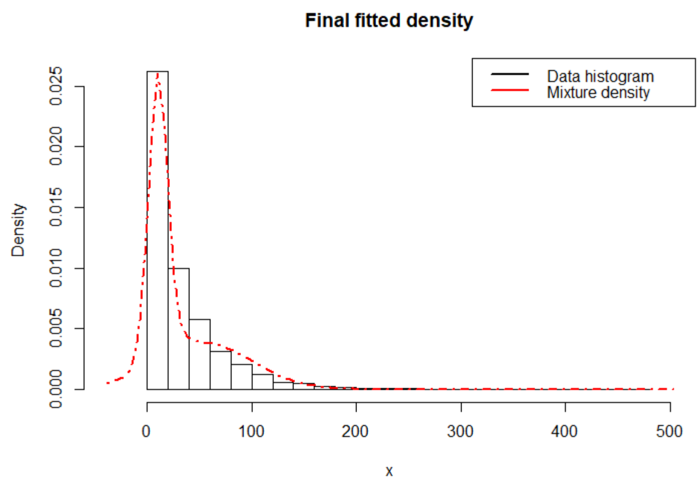

Figure 1.2: Fitted density of the Mixture of normal models(red) plotted over the precipitation data(black).

In figure 1.2 we can see that the mixture model fits nicely to the data. The model is created through 100 Gibbs sampling draws and with two mixture components. The prior mean and std of mu is set to 0 and 10. The Dirichlet(alpha) is set to 10 and prior sigma is set to the variance of the input data with 4 degrees of freedom.

1.1.3 c)

The last task in assignment 1 is to make a graphical comparison between the Normal model and the Mixture normal model. From figure 1.3 we can see that the fitted density of the Mixture normal model fits the data significantly better than the normal model.

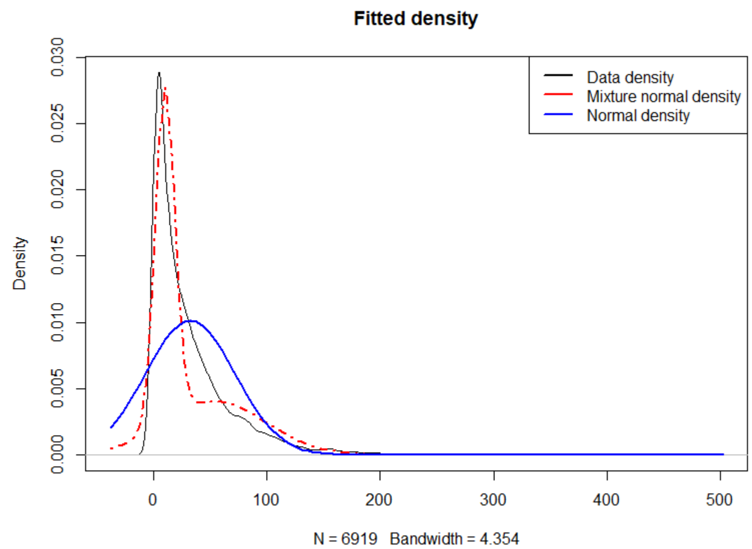

Figure 1.3: Graphical comparison between an normal density(blue) and Mixture normal density(red) on precipitation kernel density estimate(black)

1.2 Assignment 2

The dataset *eBayNumberOfBidderData.dat* contains observations from 1000 eBay auctions of coins. The response variable *nBids* denotes the number of bids in each auction. *nBids* can be modeled with the poisson regression model:

$$y_i|\beta \sim Poisson[exp(x_i^T\beta)], i = 1, ..., n$$

where y_i corresponds to the number of bids for the ith observation and x_i are the covariates for the ith observation.

1.2.1 a)

To obtain the maximum likelihood estimator of β in the Poisson regression model, we used the built in *glm()* function.

Figure 1.4 shows a barplot with the covariates beta values. As the plot shows, The covariates *Intercept, MinBidShare, Sealed, VerifyID* and *MajBlem* are significant in predicting the number of bids.

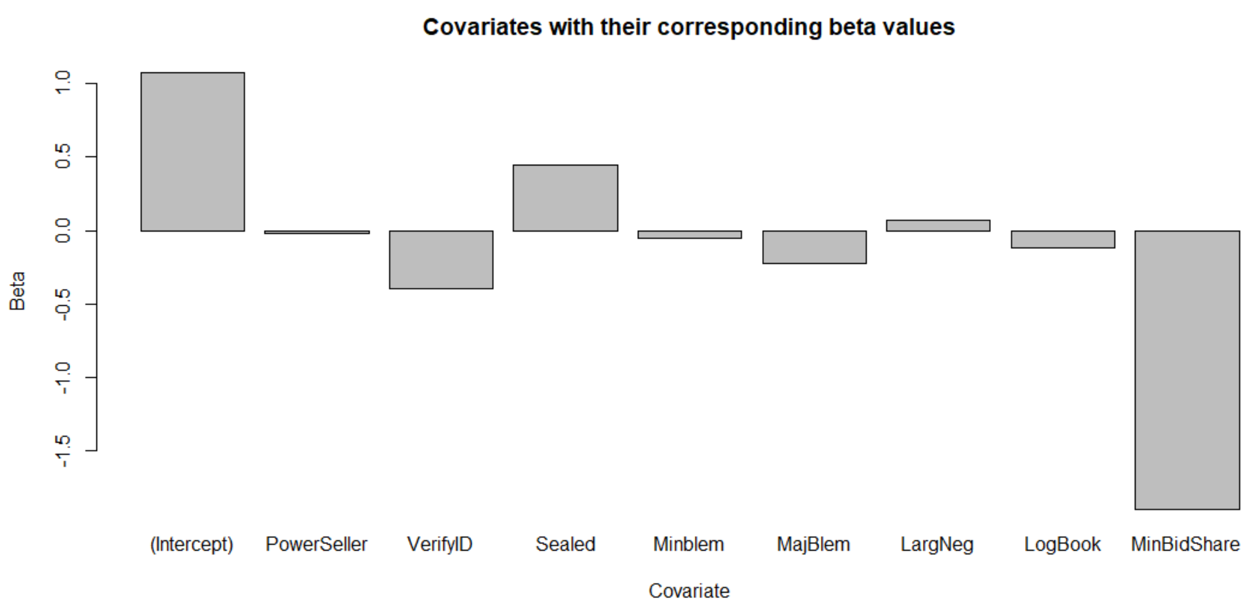

Figure 1.4: Barplot showing the β values for each covariate

1.2.2 b)

We conduct Bayesian analysis of the Poisson regression. We use the Zellners g-prior:

$$\beta \sim N[0, 100 * (X^TX)^{-1}]$$

where X is the nxp covariate matrix.

We assume that the posterior density is approximately multivariate normal

$$\beta|y \sim N(\hat{\beta}, J_y^{-1}(\hat{\beta}))$$

where $\hat{\beta}$ is the posterior mode and $J_y(\hat{\beta})$ is the negative Hessian at the posterior mode. These are obtained by numerical optimization using the *optim()* function in R. The log likelihood for the Poisson regression is expressed as:

$$\log L(\theta \mid X, Y) = \sum_{i=1}^{m} \left(y_i \theta' x_i - e^{\theta' x_i} - \log(y_i!) \right)$$

and because $\log(y_i!)$ does not contain θ, we may drop it from the expression. Leaving us with the following expression:

$$\ell(\theta \mid X, Y) = \sum_{i=1}^{m} \left(y_i \theta' x_i - e^{\theta' x_i} \right)$$

A function that calculates the posterior using the log likelihood and g-prior is used in the *optim()* function. Figure 1.5 shows the histogram of the β values when drawing from the multivariate normal model after obtaining values for $\hat{\beta}$ and $J_y(\hat{\beta})$.

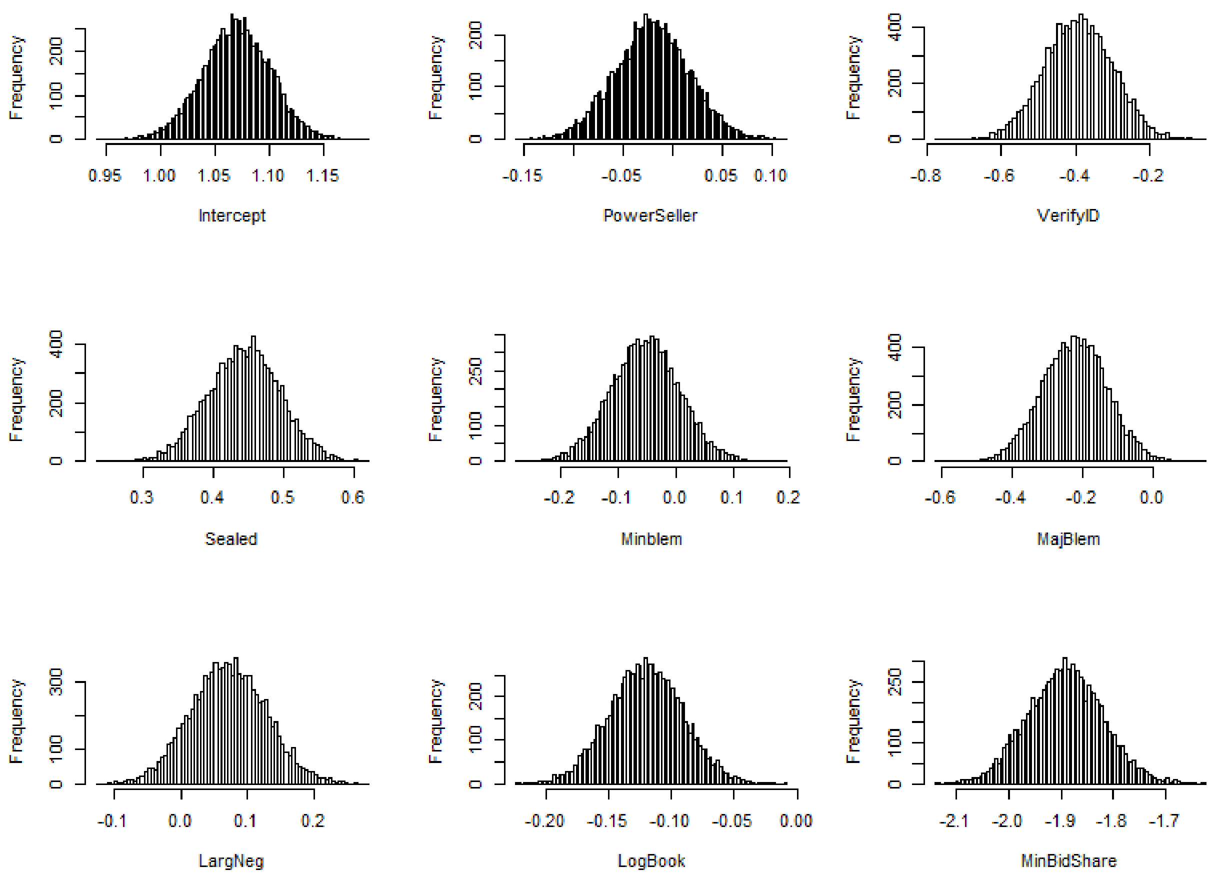

Figure 1.5: Histogram of β draws

1.2.3 c)

Next, we create a general function that takes an arbitrary posterior density and uses the Metropolis algorithm to generate random draws from the given posterior density. We let the proposal density be the multivariate normal density:

$$\theta_p|\theta^{(i-1)} \sim N(\theta^{(i-1)}, c*)$$

where c is a tuning parameter and $= J_y^{-1}(\hat{\beta})$. We use the created Metropolis function and input the log posterior of the Poisson regression to sample from the posterior of β

Figure 1.6 shows 10,000 draws of the β posterior using the Metropolis function. The draws reaches a stationary distribution after about 200 draws. Figure 1.7 shows the cumulative mean of the β posterior draws (represented by the black cureve), and the maximum likelihood estimate of each β, obtained by the *glm()* model. The cumulative mean converges to the MLE, however, because of the "burn-in", it takes a longer time, compared to calculating the cumulative mean from sample 200-10,000.

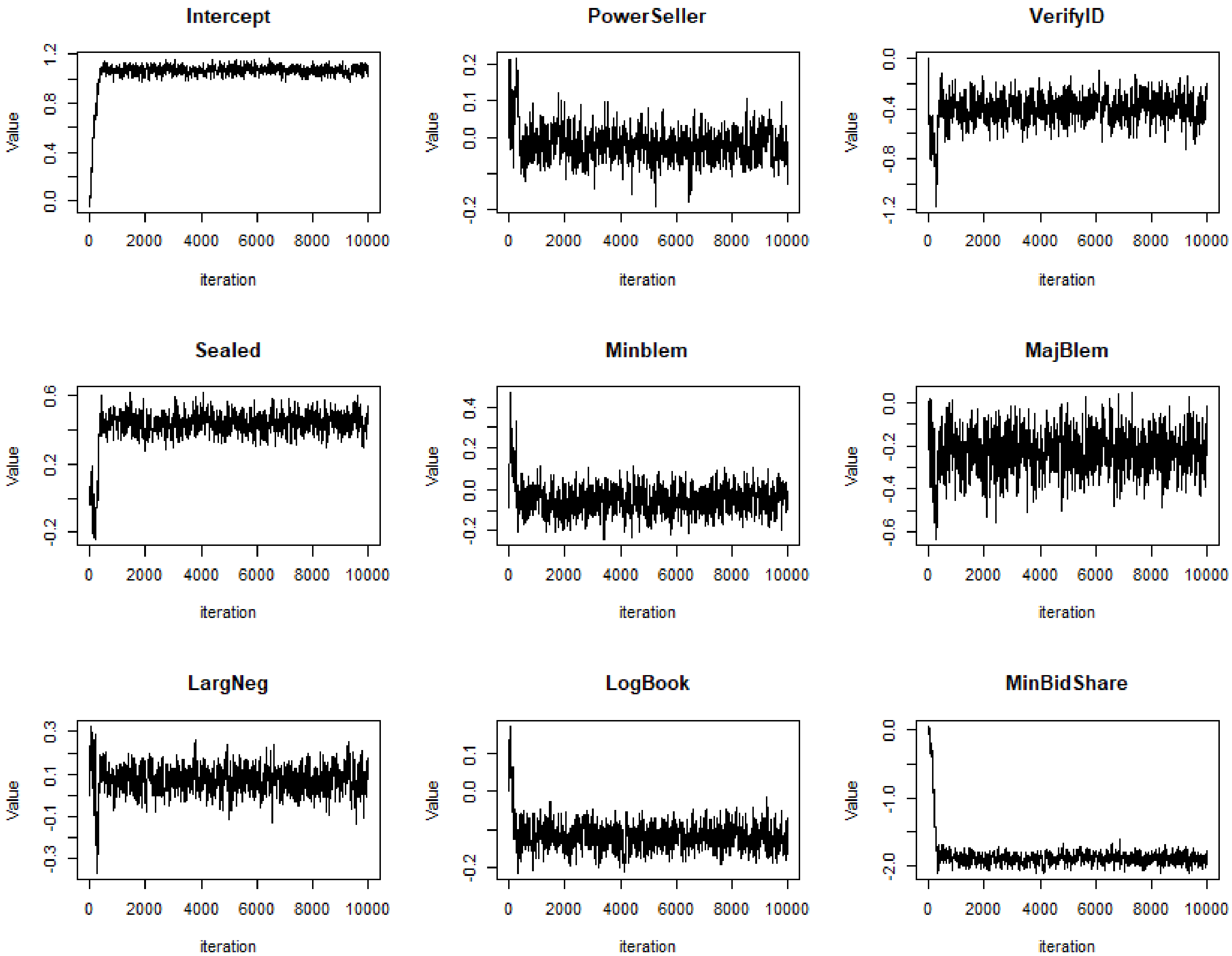

Figure 1.6: Draws from the posterior of β

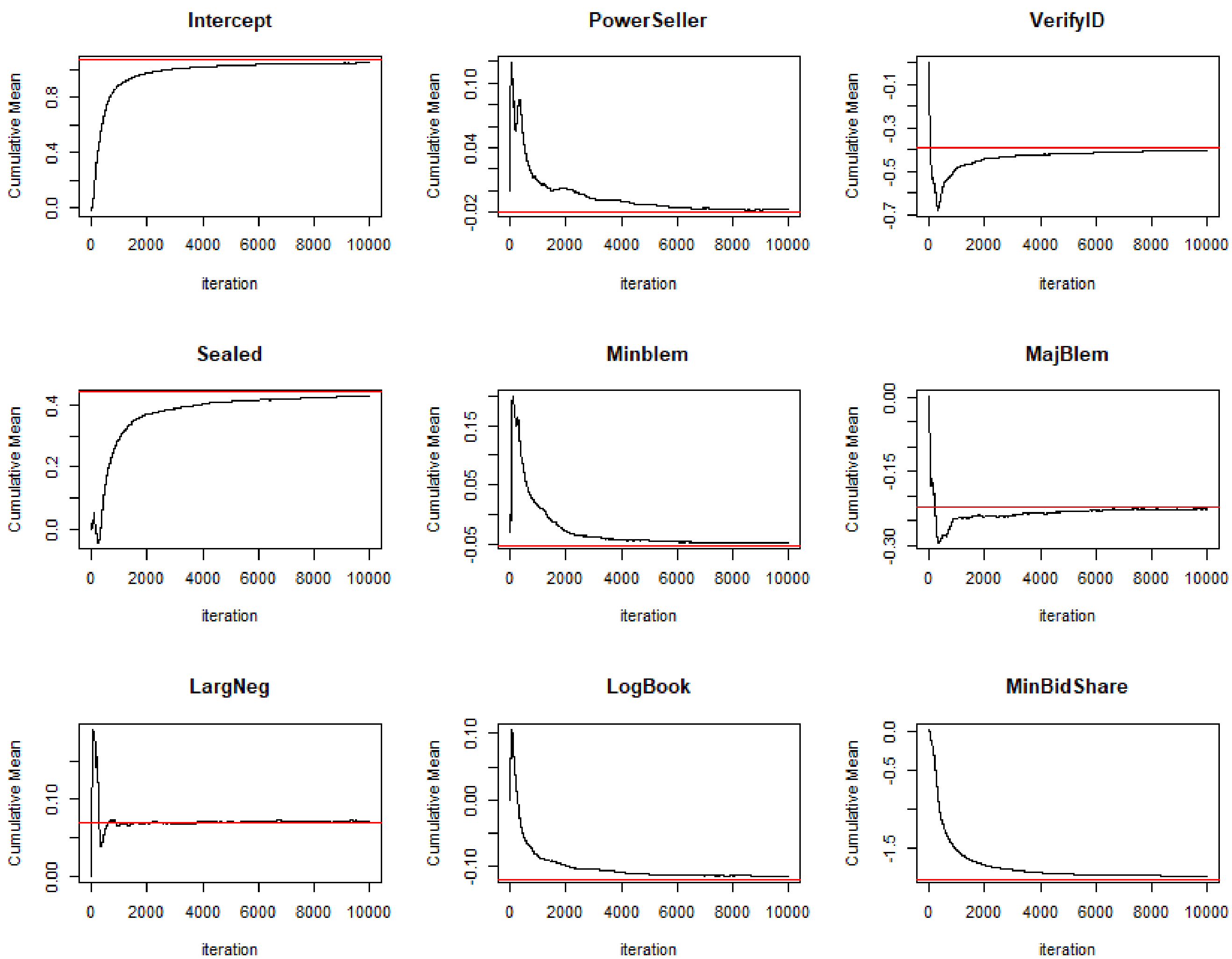

Figure 1.7: Cumulative mean of the draws from the posterior of β

1.2.4 d)

Next, we use the draws from c) to simulate from the predictive distribution of the number of bidders with the following characteristics:

Powerseller = 1

VerifyID = 1

Sealed = 1

MinBlem = 0

MajBlem = 0

LargeNeg = 0

LogBook = 1

MinBidShare = 0.5

Figure 1.8 shows a barplot of the predictive distribution of the number of bidders when a auction has the characteristics described above. The cumulative mean obtained from c) is used as the β. The probability of having zero bidders with the described characteristics is around 30%.

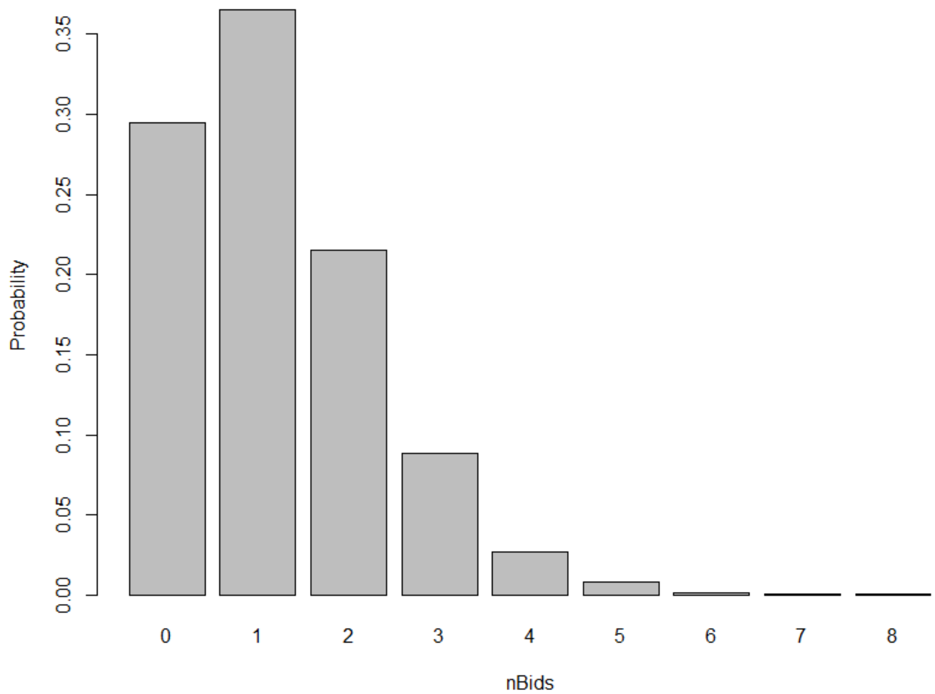

Figure 1.8: Predictive distribution of *nBids*

2. Code Appendix

2.1 Assignment 1

2.1.1 a

```
library(mvtnorm)
df = read.table("rainfall.dat", header = TRUE)
n = length(df$X136)

#prior
tau_0 = 1
u_0= 32
v_0 = 1
sigma_0 = 1
sigma = 1

x = mean(df$X136)
v_n = v_0 + n
draws = matrix(0,1000,2)

for (j in 1:1000){
  tau_n = 1/((n/sigma)+(1/tau_0))
  w = (n/sigma)*tau_n
  u_n = w*x + (1-w)*u_0

  u = rnorm(1,u_n,tau_n)
  sigma = (v_n * (v_0*sigma_0 + sum((df$X136 - u )^2))/(n*v_0)) / rchisq(1,df=v_n)

  draws[j,1] = u
  draws[j,2] = sigma
}
hist(draws[,1],50) #u
hist(draws[,2],50) #sigma^"

plot(draws[,1],type='l')
plot(draws[,2],type='l')

TrajectoryData =  cumsum(draws[,1])/seq(1,1000)
plot(1:1000, TrajectoryData, type = "l", xlab ="Iterations",ylab="Mean (u)", main ="
    Trajectories of the sampled Markov chains")

TrajectoryData =  cumsum(draws[,2])/seq(1,1000)
plot(1:1000, TrajectoryData, type = "l",xlab = "Iterations",ylab="Variance (sigma^2)", main ="
    Trajectories of the sampled Markov chains")

mean(draws[,1])
mean(draws[,2])
```

2.1.2 b

See NormalMixtureGibbs.R on course page. Changed nComp to 2.

2.1.3 c

```
u = 32.2129
sigma= 1546.929

plot(density(x), breaks = 20, freq = FALSE, xlim = c(xGridMin,xGridMax), main = "Fitted
    density")
lines(xGrid, mixDensMean, type = "l", lwd = 2, lty = 4, col = "red")
lines(xGrid, dnorm(xGrid, mean = u, sd = sqrt(sigma)), type = "l", lwd = 2, col = "blue")
legend("topright", box.lty = 1, legend = c("Data histogram","Mixture normal density","Normal
    density"), col=c("black","red","blue"), lwd = 2)
```

2.2 Assignment 2

```

set.seed(1234)
data <- read.table("eBayNumberOfBidderData.dat", header = T)

# a

model <- glm(nBids ~ . - Const, family="poisson", data=data)
barplot(model$coefficients, ylab = "Beta", xlab="Covariate", main="Covariates with their
    corresponding beta values")
summary(model)

# b

library(mvtnorm)
y <- data$nBids
X <- as.matrix(data[,-c(1)])

Sigma <- 100 * solve(t(X)%*%X)
mu <- matrix(0, nrow=1, ncol=9)


LogPoisson <- function(betas,y,X,mu,Sigma){
  linPred <- betas %*% t(X)

  logLik <- sum(linPred * y - exp(linPred))
  if (abs(logLik) == Inf) logLik = -20000

  logPrior <- dmvnorm(betas, matrix(0, length(betas), 1), Sigma, log=TRUE) ## zellners prior
  return(logLik + logPrior)
}
initVal <- as.vector(rep(0,dim(X)[2]));

OptimResults<-optim(initVal,LogPoisson,gr=NULL,y,X,mu,Sigma,method=c("BFGS"),control=list(
    fnscale=-1),hessian=TRUE)

beta_tilde <- OptimResults$par
j_inv <- -solve(OptimResults$hessian)
beta_draws <- rmvnorm(10000,beta_tilde, j_inv )
```

```r

par(mfrow=c(3,3))

hist(beta_draws[, 1], 100, xlab = "Intercept", main = "")
hist(beta_draws[, 2], 100, xlab = "PowerSeller", main = "")
hist(beta_draws[, 3], 100, xlab = "VerifyID", main = "")
hist(beta_draws[, 4], 100, xlab = "Sealed", main = "")
hist(beta_draws[, 5], 100, xlab = "Minblem", main = "")
hist(beta_draws[, 6], 100, xlab = "MajBlem", main = "")
hist(beta_draws[, 7], 100, xlab = "LargNeg", main = "")
hist(beta_draws[, 8], 100, xlab = "LogBook", main = "")
hist(beta_draws[, 9], 100, xlab = "MinBidShare", main = "")

par(mfrow=c(1,1))

# c

Metropolis <- function(c, N, summa, postFunction, ...) {
  proposal_last <- as.vector(rep(0, dim(X)[2]))
  thetas <- matrix(0, N, dim(X)[2])
  acceptance <- 0
  for (x in 1:N) {
    proposal <- as.vector(rmvnorm(1, mean = proposal_last, sigma = c * summa))
    p_theta <- postFunction(proposal, ...)
    p_theta_last <- postFunction(proposal_last, ...)
    alpha <- min(1, exp(p_theta - p_theta_last))
    rand <- runif(1)
    if (alpha > rand) {
      proposal_last <- proposal
      acceptance <- acceptance + 1
    }
    thetas[x, ] <- proposal_last

  }
  cat("Acceptance Ratio: ", acceptance / N)
  return (thetas)
}
n <- 10000

draws <- Metropolis(0.5, n, j_inv, LogPoisson, y, X, mu, Sigma)

cum_mean <- matrix(0, n, dim(X)[2])

for (i in 1:dim(draws)[2]) {
  cum_mean[,i] <- cumsum(draws[,i]) / seq_along(draws[,i])
}

par(mfrow=c(3,3))

plot(draws[,1], main="Intercept", xlab="iteration", ylab="Value", type="l")
plot(draws[,2], main="PowerSeller", xlab="iteration", ylab="Value", type="l")
plot(draws[,3], main="VerifyID", xlab="iteration", ylab="Value", type="l")
plot(draws[,4], main="Sealed", xlab="iteration",ylab="Value", type="l")
plot(draws[,5], main="Minblem", xlab="iteration",ylab="Value", type="l")
plot(draws[,6], main="MajBlem", xlab="iteration", ylab="Value", type="l")
plot(draws[,7], main="LargNeg", xlab="iteration", ylab="Value", type="l")
plot(draws[,8], main="LogBook", xlab="iteration", ylab="Value", type="l")
plot(draws[,9], main="MinBidShare", xlab="iteration", ylab="Value", type="l")

plot(cum_mean[,1],  main="Intercept", xlab="iteration", ylab="Cumulative Mean", type="l")
abline(h=model$coefficients[1], col="red")
plot(cum_mean[,2], main="PowerSeller", xlab="iteration", ylab="Cumulative Mean", type="l")
```

```
abline(h=model$coefficients[2], col="red")
plot(cum_mean[,3], main="VerifyID", xlab="iteration", ylab="Cumulative Mean", type="l")
abline(h=model$coefficients[3], col="red")
plot(cum_mean[,4], main="Sealed", xlab="iteration", ylab="Cumulative Mean", type="l")
abline(h=model$coefficients[4], col="red")
plot(cum_mean[,5], main="Minblem", xlab="iteration", ylab="Cumulative Mean", type="l")
abline(h=model$coefficients[5], col="red")
plot(cum_mean[,6], main="MajBlem", xlab="iteration", ylab="Cumulative Mean", type="l")
abline(h=model$coefficients[6], col="red")
plot(cum_mean[,7], main="LargNeg", xlab="iteration", ylab="Cumulative Mean", type="l")
abline(h=model$coefficients[7], col="red")
plot(cum_mean[,8], main="LogBook", xlab="iteration", ylab="Cumulative Mean", type="l")
abline(h=model$coefficients[8], col="red")
plot(cum_mean[,9], main="MinBidShare", xlab="iteration", ylab="Cumulative Mean", type="l")
abline(h=model$coefficients[9], col="red")

par(mfrow=c(1,1))

betas_mean <- as.vector(draws[n-1,])
#d
cov = as.vector(c(1,1,1,1,0,0,0,1,0.5))

lambda = exp(cov%*%betas_mean)
nBids_sim = rpois(10000, lambda)

barplot(table(nBids_sim)/n, ylab="Probability", xlab="nBids")
```