STAT 460
Bayesian Statistics
Final Project

Steve Hof

11/12/2020

The data from this project was supplied by the course instructor, Dr. Michelle Miranda. We first read in the data, then use it in the second half of the project.

```
dat = readMat("Dataset_FinalProject_2.mat")
X = dat$X
y = dat$Y
n = length(y)
J = dim(X)[2]
```

Part I

Consider the multiple linear regression model:

$$y = \boldsymbol{X}\beta + \boldsymbol{E}$$

where y is a vector of size n containing the response variable, $\boldsymbol{X}$ is a matrix of size $n \times J$ of fixed covariates, and β is a vector of size J containing the coefficients that characterize the linear relationship between y and X. Let $\boldsymbol{E}$ be a vector of of size n of random noise terms. We assume $\boldsymbol{E} \sim N_n(0, \Sigma)$, with known $\Sigma = I_n$. Now assume that for each $j = 1, \ldots, J$

$$\begin{gathered} \beta_j \mid \delta_j, \tau, \epsilon \sim \delta_j N\left(0, \tau^2\right) + (1 - \delta_j) N(0, \epsilon) \\ \delta_j \mid \pi \sim \text{Bernoulli}(\pi) \\ \pi \mid a_\pi, b_\pi \sim \text{Beta}\left(\tfrac{a_\pi}{2}, \tfrac{b_\pi}{2}\right) \end{gathered}$$

Let $\theta = (\beta, \delta, \pi)$, then the prior distribution of θ is $p(\theta) = p\left(\pi \mid a_\pi, b_\pi\right) \prod_{j=1}^{J} p\left(\beta_j \mid \delta_j, \tau^2, \epsilon\right) p\left(\delta_j \mid \pi\right)$

Question (a)

Write down $p(\beta_j \mid \delta_j, \tau^2, \epsilon)$, the prior of β_j, up to a constant of proportionality.

solution:

$$\begin{aligned} p(\beta_j \mid \delta_j, \tau^2, \epsilon) &= \left(\frac{1}{\sqrt{2\pi\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\beta_j^2\right\}\right)^{\delta_j} \left(\frac{1}{\sqrt{2\pi\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\beta_j^2\right\}\right)^{(1-\delta_j)} \\ &\propto \left(\frac{1}{\tau} \exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}\right)^{\delta_j} \left(\frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}\right)^{(1-\delta_j)} \end{aligned}$$

Question (b)

Use (a) to find the full conditional distribution of β_j, i.e., $p\left(\beta_j \mid \delta_j, \tau^2, \epsilon, y\right)$.

Hint 1: consider two separate distributions, $p\left(\beta_j \mid \delta_j = 0, \tau^2, \varepsilon, y\right)$ and $p\left(\beta_j \mid \delta_j = 1, \tau^2, \varepsilon, y\right)$.

Hint 2: If it helps, use the fact that $y_i - \sum_{j=1}^{J} X_{ij}\beta_j = \tilde{y}_i - X_{ij}\beta_j$, where $\tilde{y}_i = y_i - \sum_{l \neq j} X_{il}\beta_l$

solution:

We start by using hint 1. First we focus on $p\left(\beta_j \mid \delta_j = 0, \tau^2, \varepsilon, y\right)$

$$\begin{aligned} p(\beta_j \mid \delta_j = 0, \tau^2, \varepsilon, y) &= p(\beta_j \mid \delta_j = 0, \tau^2, \varepsilon) \prod_{i=1}^{n} p(y_i|\beta_j) \\ &\propto \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\} \prod_{i=1}^{n} p(y_i|\beta_j) \end{aligned}$$

To keep things more organized we will calculate the likelihood $\prod_{i=1}^{n} p(y_i|\beta_j)$ separately now, then continue on by plugging it into the above.

$$\begin{aligned} \prod_{i=1}^{n} p(y_i|\beta_j) &= \prod_{i=1}^{n} \det(\Sigma)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\left(y_i - \sum_{j=1}^{J} X_{ij}\beta_j\right)^2\right\} \\ &\propto \exp\left\{\sum_{i=1}^{n}\left(-\frac{1}{2}\right)\left(y_i - \sum_{j=1}^{J} X_{ij}\beta_j\right)^2\right\} \\ &\propto \exp\left\{-\frac{1}{2}\sum_{i=1}^{n}\left(\tilde{y} - X_{ij}\beta_j\right)^2\right\} \text{ (Hint 2)} \end{aligned}$$

Now, plugging the likelihood back into the above, we have

$$
\begin{aligned}
p(\beta_j \mid \delta_j = 0, \tau^2, \varepsilon, y) &\propto \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\} \exp\left\{-\frac{1}{2}\sum_{i=1}^{n}(\tilde{y} - X_{ij}\beta_j)^2\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2}\left(\frac{\beta_j^2}{\varepsilon} + \sum_{i=1}^{n}(\tilde{y} - X_{ij}\beta_j)^2\right)\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(\beta_j^2 + \varepsilon\sum_{i=1}^{n}(\tilde{y} - X_{ij}\beta_j)^2\right)\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(\beta_j^2 + \varepsilon\sum_{i=1}^{n}\left[\tilde{y}_i^2 - 2\tilde{y}_i X_{ij}\beta_j + X_{ij}^2\beta_j^2\right]\right)\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(\beta_j^2 + \varepsilon\left[\sum_{i=1}^{n}\tilde{y}_i^2 - 2\sum_{i=1}^{n}\tilde{y}_i X_{ij}\beta_j + \sum_{i=1}^{n}X_{ij}^2\beta_j^2\right]\right)\right\} \\
&\propto \frac{1}{\sqrt{\varepsilon}} \exp\left\{\beta_j^2 + \varepsilon\left[-2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j + \sum_{i=1}^{n}X_{ij}\beta_j^2\right]\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(\beta_j^2 - 2\varepsilon\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j + \varepsilon\beta_j^2\sum_{i=1}^{n}X_{ij}^2\right)\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(\beta_j^2 + \varepsilon\beta_j^2\sum_{i=1}^{n}X_{ij}^2 - 2\varepsilon\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j\right)\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(\beta_j^2\left(1 + \varepsilon\sum_{i=1}^{n}X_{ij}^2\right) - 2\varepsilon\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j\right)\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(1 + \varepsilon\sum_{i=1}^{n}X_{ij}^2\right)\left[\beta_j^2 - \frac{2\varepsilon\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j}{1 + \varepsilon\sum_{i=1}^{n}X_{ij}^2}\right]\right\} \\
&= \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(1 + \varepsilon\sum_{i=1}^{n}X_{ij}^2\right)\left[\left(\beta_j^2 - \frac{\varepsilon\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j}{1 + \varepsilon\sum_{i=1}^{n}X_{ij}}\right)^2 - \left(\frac{\varepsilon\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j}{1 + \varepsilon\sum_{i=1}^{n}X_{ij}}\right)^2\right]\right\} \\
&\propto \frac{1}{\sqrt{\varepsilon}} \exp\left\{-\frac{1}{2\varepsilon}\left(\beta_j^2 - \frac{\varepsilon\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j}{1 + \varepsilon\sum_{i=1}^{n}X_{ij}}\right)^2\left(1 + \varepsilon\sum_{i=1}^{n}X_{ij}^2\right)\right\}
\end{aligned}
$$

The full conditional of β_j when $\delta_j = 0$ is, therefore, given by

$$
p(\beta_j \mid \delta_j = 0, \varepsilon, y) \sim \text{Normal}\left(\frac{\varepsilon\sum_{i=1}^{n}X_{ij}\tilde{y}_i}{1 + \varepsilon\sum_{i=1}^{n}X_{ij}^2}, \varepsilon\left(1 + \varepsilon\sum_{i=1}^{n}X_{ij}^2\right)^{-1}\right)
$$

We then repeat the process for the full conditional of β_j with $\delta_j = 1$.

$$
\begin{aligned}
p(\beta_j \mid \delta_j = 1, \tau^2, \tau^2, y) &\propto \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\} \exp\left\{-\frac{1}{2}\sum_{i=1}^{n}(\tilde{y} - X_{ij}\beta_j)^2\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2}\left(\frac{\beta_j^2}{\tau^2} + \sum_{i=1}^{n}(\tilde{y} - X_{ij}\beta_j)^2\right)\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(\beta_j^2 + \tau^2\sum_{i=1}^{n}(\tilde{y} - X_{ij}\beta_j)^2\right)\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(\beta_j^2 + \tau^2\sum_{i=1}^{n}\left[\tilde{y}_i^2 - 2\tilde{y}_i X_{ij}\beta_j + X_{ij}^2\beta_j^2\right]\right)\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(\beta_j^2 + \tau^2\left[\sum_{i=1}^{n}\tilde{y}_i^2 - 2\sum_{i=1}^{n}\tilde{y}_i X_{ij}\beta_j + \sum_{i=1}^{n}X_{ij}^2\beta_j^2\right]\right)\right\} \\
&\propto \frac{1}{\sqrt{\tau^2}} \exp\left\{\beta_j^2 + \tau^2\left[-2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j + \sum_{i=1}^{n}X_{ij}\beta_j^2\right]\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(\beta_j^2 - 2\tau^2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j + \tau^2\beta_j^2\sum_{i=1}^{n}X_{ij}^2\right)\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(\beta_j^2 + \tau^2\beta_j^2\sum_{i=1}^{n}X_{ij}^2 - 2\tau^2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j\right)\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(\beta_j^2\left(1 + \tau^2\sum_{i=1}^{n}X_{ij}^2\right) - 2\tau^2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j\right)\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(1 + \tau^2\sum_{i=1}^{n}X_{ij}^2\right)\left[\beta_j^2 - \frac{2\tau^2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j}{1 + \tau^2\sum_{i=1}^{n}X_{ij}^2}\right]\right\} \\
&= \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(1 + \tau^2\sum_{i=1}^{n}X_{ij}^2\right)\left[\left(\beta_j^2 - \frac{\tau^2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j}{1 + \tau^2\sum_{i=1}^{n}X_{ij}}\right)^2 - \left(\frac{\tau^2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j}{1 + \tau^2\sum_{i=1}^{n}X_{ij}}\right)^2\right]\right\} \\
&\propto \frac{1}{\sqrt{\tau^2}} \exp\left\{-\frac{1}{2\tau^2}\left(\beta_j^2 - \frac{\tau^2\sum_{i=1}^{n}\tilde{y}X_{ij}\beta_j}{1 + \tau^2\sum_{i=1}^{n}X_{ij}}\right)^2\left(1 + \tau^2\sum_{i=1}^{n}X_{ij}^2\right)\right\}
\end{aligned}
$$

The full conditional of β_j when $\delta_j = 1$ is, therefore, given by

$$
p(\beta_j \mid \delta_j = 1, \tau^2, y) \sim \text{Normal}\left(\frac{\tau^2\sum_{i=1}^{n}X_{ij}\tilde{y}_i}{1 + \tau^2\sum_{i=1}^{n}X_{ij}^2}, \tau^2\left(1 + \tau^2\sum_{i=1}^{n}X_{ij}^2\right)^{-1}\right)
$$

Question (c)

Show that the full conditional distribution of δ_j is Bernoulli$\left(\frac{p_1}{p_0 + p_1}\right)$ with $p_1 = \pi\exp\left\{-\frac{1}{2\tau^2}\beta_j^2\right\}$ and $p_0 = \frac{(1-\pi)\tau}{\sqrt{\varepsilon}}\exp\left\{-\frac{1}{2\varepsilon}\beta_j^2\right\}$.

solution:

$$
\begin{aligned}
p(\delta_j \mid \beta_j, \tau, \varepsilon) &= p(\beta_j|\delta_j, \tau, \varepsilon)p(\delta_j|\pi) \\
&\propto \left(\frac{1}{\tau}\exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}\right)^{\delta_j}\left(\frac{1}{\sqrt{\varepsilon}}\exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}\right)^{(1-\delta_j)}\pi^{\delta_j}(1-\pi)^{(1-\delta_j)} \\
&\propto \left(\frac{\pi}{\tau}\exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}\right)^{\delta_j}\left(\frac{1-\pi}{\sqrt{\varepsilon}}\exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}\right)^{(1-\delta_j)} \\
&\propto \left(\frac{\frac{\pi}{\tau}\exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}}{\frac{\pi}{\tau}\exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}+\frac{1-\pi}{\sqrt{\varepsilon}}\exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}}\right)^{\delta_j}\left(\frac{\frac{1-\pi}{\sqrt{\varepsilon}}\exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}}{\frac{\pi}{\tau}\exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}+\frac{1-\pi}{\sqrt{\varepsilon}}\exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}}\right)^{(1-\delta_j)} \\
&= \left(\frac{\pi\exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}}{\frac{\pi}{\tau}\exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}+\frac{1-\pi}{\sqrt{\varepsilon}}\exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}}\right)^{\delta_j}\left(\frac{\frac{(1-\pi)\tau}{\sqrt{\varepsilon}}\exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}}{\frac{\pi}{\tau}\exp\left\{-\frac{\beta_j^2}{2\tau^2}\right\}+\frac{1-\pi}{\sqrt{\varepsilon}}\exp\left\{-\frac{\beta_j^2}{2\varepsilon}\right\}}\right)^{(1-\delta_j)}
\end{aligned}
$$

We have now shown that the full conditional distribution of δ_j is $\text{Bernoulli}\left(\frac{p_1}{p_0+p_1}\right)$ with $p_1 = \pi\exp\left\{-\frac{1}{2\tau^2}\beta_j^2\right\}$ and $p_0 = \frac{(1-\pi)\tau}{\sqrt{\varepsilon}}\exp\left\{-\frac{1}{2\varepsilon}\beta_j^2\right\}$.

Question (d)

Write down the full conditional distribution of π.

solution:

$$
\begin{aligned}
p\left(\pi \mid \delta_j, \frac{a_\pi}{2}, \frac{b_\pi}{2}\right) &\propto p\left(\pi|\frac{a_\pi}{2}, \frac{b_\pi}{2}\right)\prod_{j=1}^{J}p(\delta_j|\pi) \\
&\propto \pi^{\frac{a_\pi}{2}-1}(1-\pi)^{\frac{b_\pi}{2}-1}\prod_{j=1}^{J}\pi^{\delta_j}(1-\pi)^{(1-\delta_j)} \\
&\propto \pi^{\frac{a_\pi}{2}-1}(1-\pi)^{\frac{b_\pi}{2}-1}\pi^{\sum_{j=1}^{J}\delta_j}(1-\pi)^{\sum_{j=1}^{J}(1-\delta_j)} \\
&\propto \pi^{\sum_{j=1}^{J}\delta_j+\frac{a_\pi}{2}-1}(1-\pi)^{\sum_{j=1}^{J}(1-\delta_j)+\frac{b_\pi}{2}-1}
\end{aligned}
$$

Which means that

$$
\pi \mid \delta_j, \frac{a_\pi}{2}, \frac{b_\pi}{2} \sim \text{Beta}\left(\sum\delta_j + \frac{a_\pi}{2}, \sum(1-\delta_j) + \frac{b_\pi}{2}\right)
$$

Question (e)

Write down a Gibbs sampler algorithm to sample from the joint posterior distribution of θ.

solution:

With Gibbs sampling the idea is to create a Markov Chain with stationary distribution equal to the full posterior so that we can generate posterior samples from it. We go back and forth updating the parameters one at a time using the current value of all the other parameters. We start by updating (sampling from) the distribution with the least number of dependencies and move toward the most dependencies.

For our case, in particular, the algorithm is

1. Initialize β, δ, and π to given starting values
2. For each iteration, k up to number of iterations given:
   (a) Sample from $p(\pi \mid -)$ using the most recently sampled δ_j
   (b) Sample from $p(\delta_j \mid -)$ using the most recently sampled π and β_j's (or initial values of β_j if we are in iteration 1.
   (c) Sample from $p(\beta_j \mid -)$, calculated using the updated value of δ_j and the most recent values of $\beta_j, j = 1 \ldots, n$. Unless $j = 1$ or $j = n$, this will involve using β_1 to β_{j-1} from the current iteration of the algorithm, and β_{j+1} to β_n from the previous iteration of the algorithm.
3. Check convergence diagnostics by viewing trace plots, $\hat{R}$ measurements and comparing the posterior means to the β coefficients given by regular regression.
4. Remove burn-in, and re-check convergence diagnostics.
5. If necessary, apply thinning and re-check convergence diagnostics.

Part II

Question (f)

Let $\varepsilon = 10^{-4}$ and $\tau^2 = 10^2$. Explain heuristically how the spike-and-slab prior allows for variable selection in the multiple linear regression model context.

solution:

The spike-and-slab prior is a method for performing Bayesian regression and varialbe selection at the same time. It is especially useful when there are a large number of predictor varialbes and we think some of them are insignificant.

It is a mixture model consisting of, the spike and the slab. "Spike" refers to setting the prior of β to have mass at zero. "Slab'' refers to creating a relatively flat and diffuse prior by multiplying the variance-covariance matrix by a small scalar > 0.

In our case, we see that $\tau = 10$ corresponds to the spike and $\varepsilon = 10^{-4}$ corresponds to the slab.

Spike-and-slab also offerers efficiency because the Gibbs sampler does not have to explore the entire space each time.

Question (g)

Let $a_\pi = b_\pi = 1$, the bathtub prior distribution for π. Using ε and τ as in part (f), obtain posterior estimates for the coefficient β.

solution:

In order to make our code as readable as possible, we write helper functions to update π, δ_j, and β_j as well as an index generator function to assist with sampling from the proper past values of β_j and a function to apply the burn-in and thinning process.

```
burn.and.thin = function(post, bi, ti) {
  thin.indx = seq(from = bi, to = length(post[, 1]), by = ti)
  thin.post = post[thin.indx, ]
  colnames(thin.post) = beta.names
  return(thin.post)
}
```

```
indx_gen = function(j, M) {
  n = dim(M)[2]
  if (j == 1) return(M[1, ])
  if (j == n) return(M[2, ])
  return(c(M[2, 1:j-1], M[1, j:n]))
}

update_pi = function(delta_j, a_pi, b_pi) {
  shape1 = sum(delta_j) + a_pi/2
  shape2 = sum(1-delta_j) + b_pi/2
  rbeta(n = 1, shape1, shape2)
}

update_delta = function(bet, pi, tau, eps, j) {
  p0 = ((1 - pi) * tau / sqrt(eps)) * exp(-1/(2*eps) * bet[j]^2)
  p1 = pi * exp(-1/(2*tau^2) * bet[j]^2)
  p = p1 / (p0 + p1)
  return(rbinom(n = 1, size = 1, prob = p))
}

update_beta = function(delta_j, j, bet, tau, eps, mutop = 0, mubot = 0) {
  partial.mutop = mutop
  partial.mubot = mubot
  for(i in 1:dim(X)[1]) {
    partial.mutop = partial.mutop + X[i, j] *
      (y[i] - sum(setdiff(X[i,], X[i,j]) * setdiff(bet, bet[j])))
    partial.mubot = partial.mubot + X[i, j]^2
  }

  if(delta_j == 0) {
    bot = 1 + eps * partial.mubot
    mu = eps * partial.mutop / bot
    sd = sqrt(eps * bot^-1)
    return(rnorm(n = 1, mean = mu, sd = sd))

  }
  if(delta_j == 1) {
    bot = 1 + tau^2 * partial.mubot
    mu = tau^2 * partial.mutop / bot
    sd = sqrt(tau^2 * bot^-1)
    return(rnorm(n = 1, mean = mu, sd = sd))
  }
}
```

Our Gibbs sampler function follows the algorithm written for solution (e), except that we only maintain the most recent values for π and δ_j, rather than the entire vector of values.

```
gibbs = function(n_iter, init, priors) {
  beta.out = matrix(data = NA, nrow = n_iter, ncol = J)
  delta.curr = init$delta_j
  beta.curr = init$beta
  beta.out[1, ] = beta.curr

  for(k in 2:n_iter) {
```

```
    pi.curr = update_pi(delta_j = delta.curr, a_pi = priors$a_pi, b_pi = priors$b_pi)

    for(j in 1:length(beta.curr)) {
      delta.curr = update_delta(bet = beta.out[k-1,], pi = pi.curr,
                                tau = priors$tau, eps = priors$eps, j = j)

      betas = indx_gen(j = j, M = beta.out[(k-1) : k, ])
      beta.curr[j] = update_beta(delta_j = delta.curr, j = j, bet = betas,
                                 tau = priors$tau, eps = priors$eps)

      beta.out[k, j] = beta.curr[j]
    }
  }
  colnames(beta.out) = beta.names
  return(beta.out)
}
```

Here we set up our priors and initialize with the given starting values.

```
priors = list()
init = list()
n_iter = 10000

model = lm(y ~ X - 1)
init$beta = model$coefficients
init$delta_j = 0

priors$a_pi = 1
priors$b_pi = 1
priors$tau = 10
priors$eps = 10^-4
```

Finally, we run the algorithm and show the final 4 values for each β_j

```
# post = gibbs(n_iter = n_iter, init = init, priors = priors)
t(tail(post, 4))
```

```
##             [9997,]      [9998,]      [9999,]     [10000,]
## beta1   0.674105725  0.686811352  0.675436153  0.668077537
## beta2   2.516830539  2.514226928  2.526088636  2.524682562
## beta3   0.007226361  0.009772277  0.002389984  0.002928335
## beta4   1.840275294  1.830462241  1.825766257  1.830294375
## beta5   0.011299413  0.009155944  0.013899248  0.010601889
## beta6   1.304855759  1.310620245  1.311647826  1.313441688
## beta7  -0.015886620 -0.021002517 -0.008654419 -0.012761882
## beta8   2.006744150  2.011658949  1.996996852  2.004457475
## beta9   1.504224214  1.495770868  1.497623320  1.493147661
## beta10  0.748326231  0.762829665  0.769577084  0.771986787
```

Question (h)

Check for convergence of the MCMC chains using trace plots and compute $\hat{R}$.

solution:

First, we calculate $\hat{R}$ for our chain of sampled β values

```
calc.rhat = function(m, nchain, J, chain) {
  rhat = numeric(J)
  for (j in 1:J) {
    psi.mean = mean(chain[, j])
    psi.bar = numeric(m)
    aux.w = numeric(m)
    for (k in 1:m) {
      sub.chain = chain[seq((k - 1) * nchain + 1, k * nchain, 1), j]
      psi.bar[k] = mean(sub.chain)
      aux.w[k] = (1 / (nchain - 1)) * sum((sub.chain - mean(sub.chain))^2)
    }
    B = (nchain / (m - 1)) * (sum((psi.bar - psi.mean)^2))
    W = (1 / m) * sum(aux.w)
    VP = ((nchain - 1) / nchain) * W + (1 / nchain) * B
    rhat[j] = sqrt(VP / W)
    names(rhat) = beta.names
  }
  return(rhat)
}

rhat.post = calc.rhat(m = 5, nchain = 100, J = 10, chain = post)
rhat.post
```

```
##    beta1    beta2    beta3    beta4    beta5    beta6    beta7    beta8
## 2.857479 1.724200 1.050331 1.762468 2.050055 1.961335 1.041590 1.761001
##    beta9   beta10
## 1.264461 1.391706
```

The $\hat{R}$ values are not close enough to 1 for us to believe convergence has occurred, and our original chain shows a great deal of auto-correlation so we remove a burn-in of $2,000$ and thin the chain by a step-size of 10.

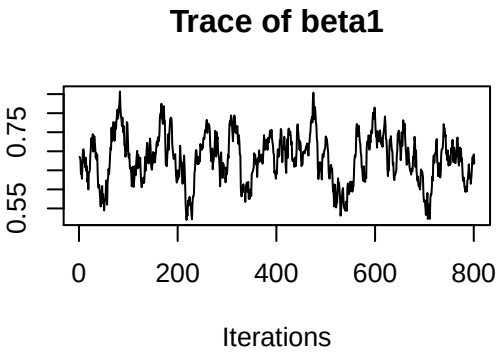
```
burn.in = 2000
thin_interval = 10

thin.post = burn.and.thin(post = post, bi = burn.in, ti = thin_interval)
rhat.thin.post = calc.rhat(m = 5, nchain = 100, J = 10, chain = thin.post)
rhat.thin.post
```

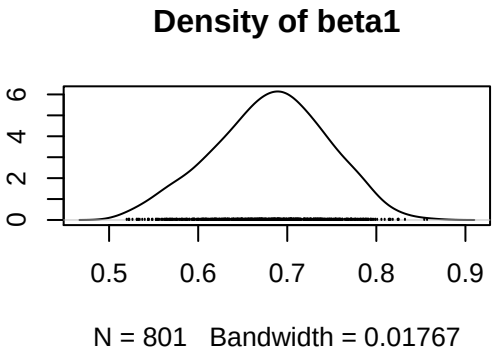
```
##     beta1     beta2     beta3     beta4     beta5     beta6     beta7     beta8
## 1.0442415 1.0931170 0.9989601 1.0400011 1.0284213 1.0431538 1.0633191 1.0734171
##     beta9    beta10
## 1.0470726 1.0023907
```

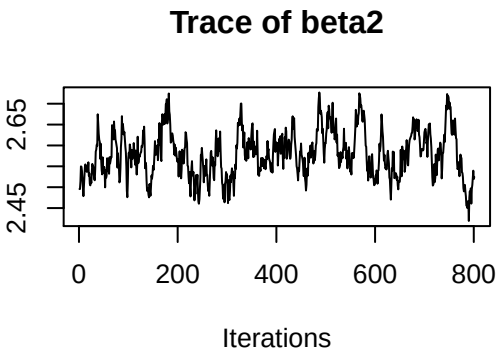
The $\hat{R}$ values for our thinned chain look much better. We now produce trace and density plots to further assess convergence.

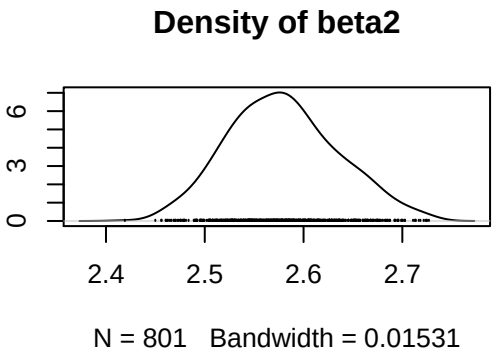
Trace plots and densities of β_j's:

```
plot(as.mcmc(thin.post[, 1:2]))
```

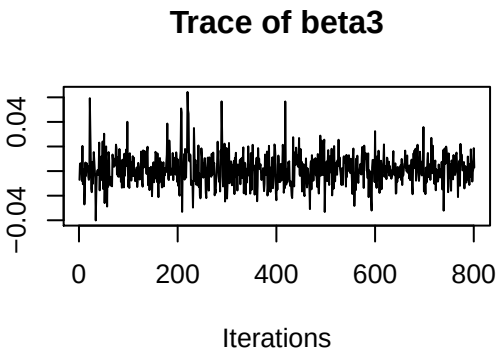


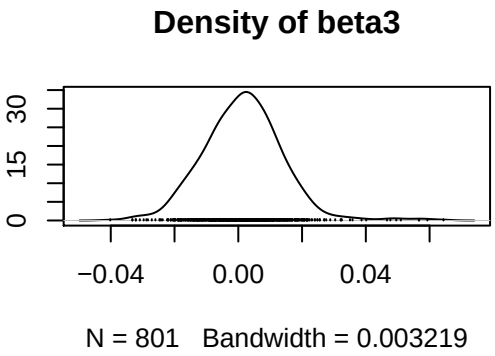


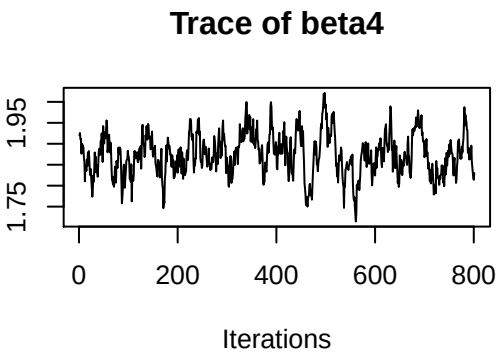


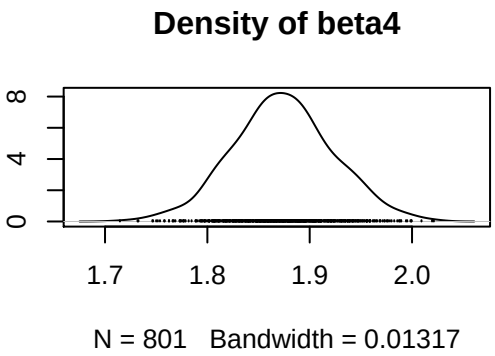



```
plot(as.mcmc(thin.post[, 3:4]))
```

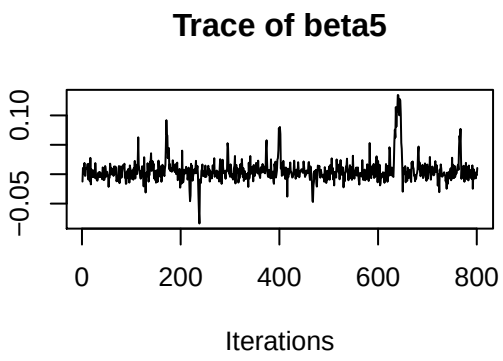


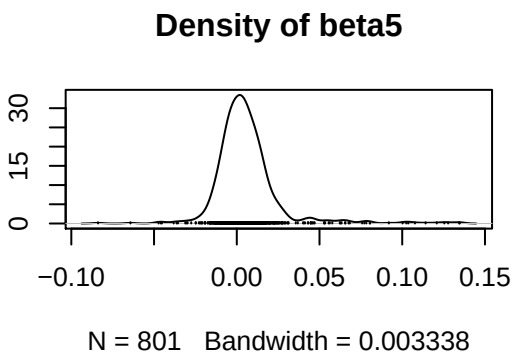


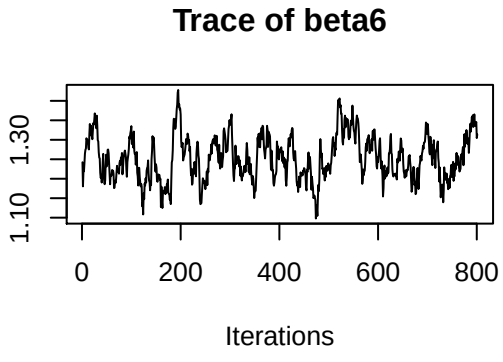


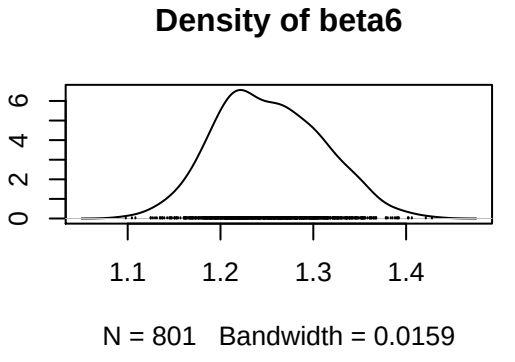


```
plot(as.mcmc(thin.post[, 5:6]))
```

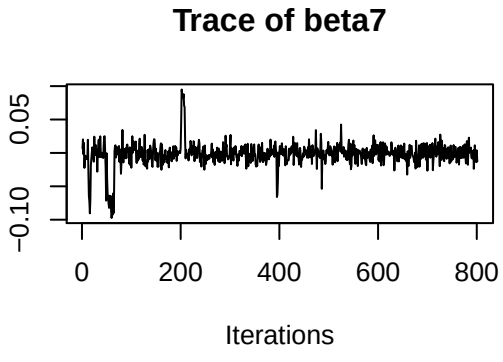


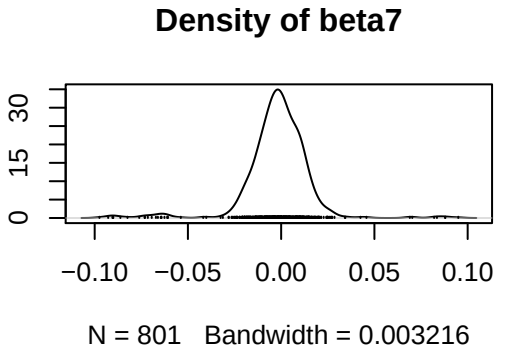


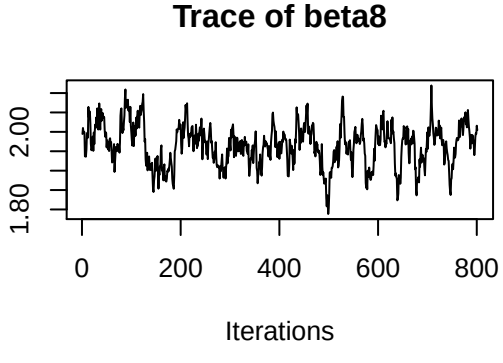


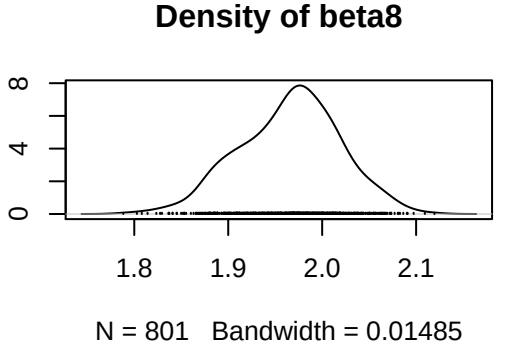



```
plot(as.mcmc(thin.post[, 7:8]))
```

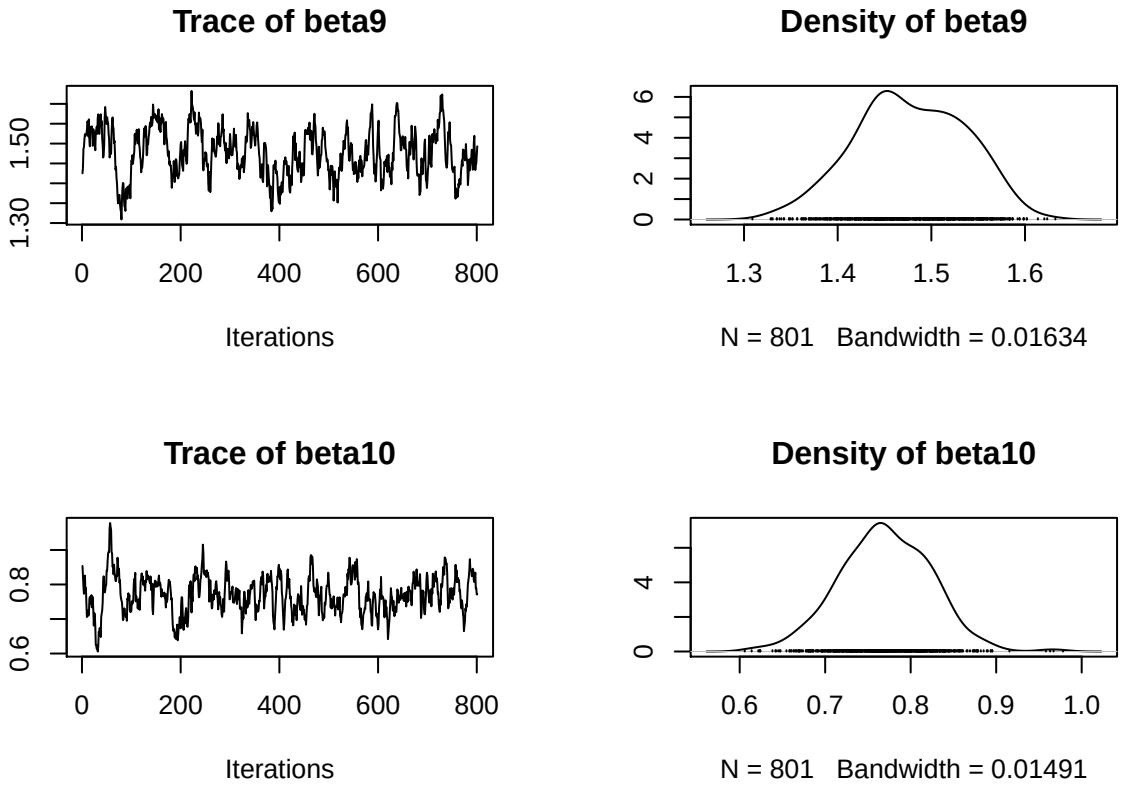








```
plot(as.mcmc(thin.post[, 9:10]))
```



The trace plots show us that our sampler seams to be traversing the entire space and that the thinned posterior Markov chain maintains a consistent mean. This, coupled with the $\hat{R}$ values all being close to 1, makes us believe convergence has been achieved.

Question (i)

If the MCMC is converging, present the results including the posterior mean, posterior variance, and a 95% credible interval for each coefficient. Based on these results, which covariates are important to predict the response variable?

solution:

Summary statistics for the mean and standard deviation are given in the following output. (Since Variance, not standard deviation, was asked for in this question, **see Appendix for Variance**)

```
summary(as.mcmc(thin.post))$statistics
```

```
##                Mean         SD     Naive SE Time-series SE
## beta1   0.680954930 0.06348236 0.0022430391   0.0112590948
## beta2   2.579083001 0.05538289 0.0019568582   0.0086682010
## beta3   0.001531412 0.01271125 0.0004491299   0.0003980155
## beta4   1.873839462 0.04925460 0.0017403259   0.0067461212
## beta5   0.006118302 0.02046811 0.0007232053   0.0022559101
## beta6   1.252681849 0.05713013 0.0020185939   0.0095230229
## beta7  -0.002085532 0.01819278 0.0006428103   0.0017666417
## beta8   1.965439431 0.05333818 0.0018846118   0.0077130306
## beta9   1.477400824 0.05871491 0.0020745892   0.0093796801
## beta10  0.770230735 0.05358313 0.0018932670   0.0076445392
```

The quantiles of the distribution are given in the following output.

```
summary(as.mcmc(thin.post))$quantiles
```

```
##               2.5%          25%          50%         75%      97.5%
## beta1   0.55302303  0.639136820  0.684682727 0.726047631 0.79385343
## beta2   2.47596602  2.540815535  2.576730884 2.614505047 2.69528701
## beta3  -0.02117791 -0.006560103  0.001717724 0.008935030 0.02557962
## beta4   1.77518023  1.841573368  1.873406136 1.904996344 1.97278686
## beta5  -0.01947268 -0.004171821  0.003325691 0.011899930 0.06333445
## beta6   1.15030207  1.211114980  1.249663940 1.293718407 1.36491001
## beta7  -0.06075291 -0.008610987 -0.001206745 0.006871871 0.02472980
## beta8   1.86023068  1.929034416  1.969758241 2.001212745 2.06445246
## beta9   1.36342624  1.438233924  1.474661954 1.523222813 1.58213141
## beta10  0.66578391  0.733533673  0.768778398 0.808937205 0.87513289
```

The 95% Credible Intervals for each of the β_j's are given in the following output.

```
t(hdi(as.mcmc(thin.post)))
```

```
##              lower      upper
## beta1   0.55248556 0.79145413
## beta2   2.48888129 2.70275026
## beta3  -0.02215572 0.02407283
## beta4   1.77283852 1.96816436
## beta5  -0.02752419 0.04706820
## beta6   1.15308026 1.36761730
## beta7  -0.02662521 0.02742339
## beta8   1.86641698 2.06716034
## beta9   1.36895003 1.58400040
## beta10  0.65816610 0.86124517
## attr(,"credMass")
## [1] 0.95
```

Since the 95% Credible Intervals for β_3, β_5, and β_7 contain zero, we consider them insignificant to predict the response variable. All the other covariates are significant.

Finally, we compare our posterior means to those calculated with regular regression.

```
model$coefficients
```

```
##          X1          X2          X3          X4          X5          X6
##  0.65790941  2.57066350  0.02644729  1.87108529  0.06415213  1.24471526
##          X7          X8          X9         X10
## -0.02443434  1.93780057  1.48115716  0.77591824
```

All of the significant coefficients fall within our 95% Credible Intervals which gives us further proof that our Gibbs Sampling was successful.

Question (j)

Sensitivity Analysis. Consider four different prior distributions for π by choosing values of a_π and b_π that change the shape of the beta distribution. Plot the prior of π for each of these values. Is the posterior distribution of β sensitive to these new prior distributions?

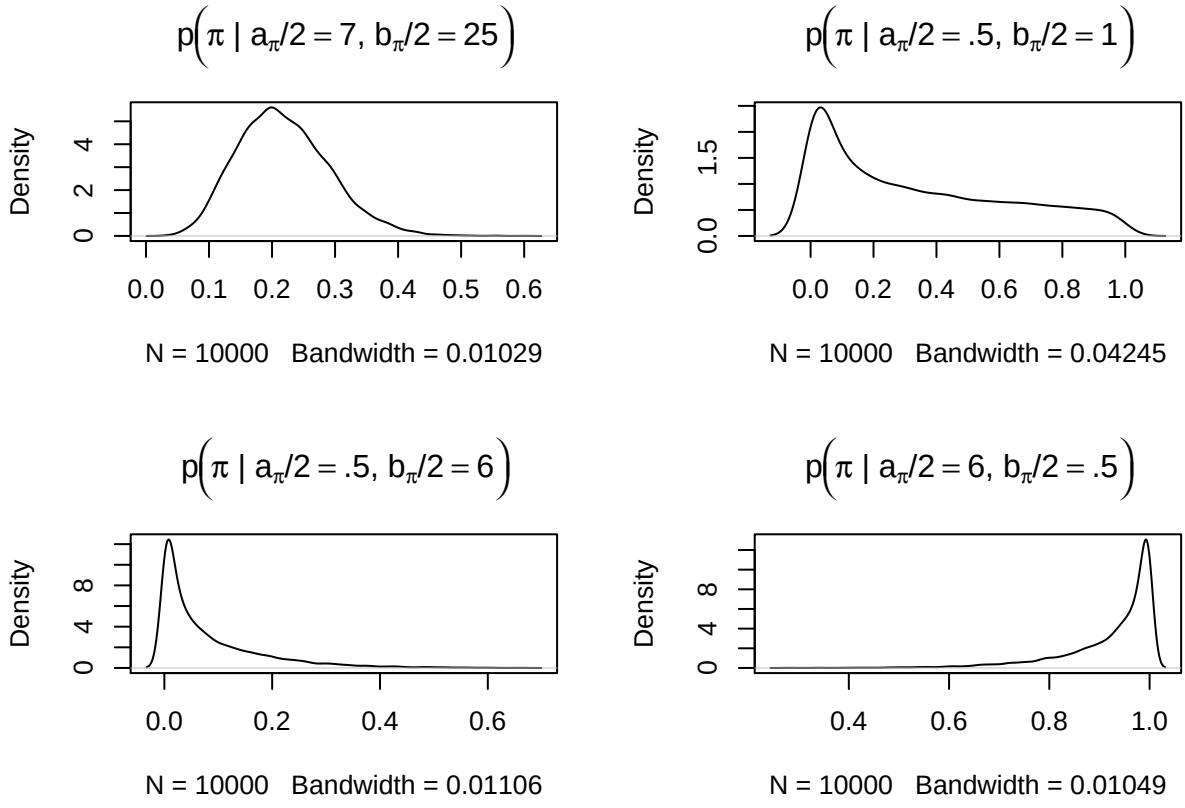
solution:

Here, we've chosen four new sets of values for (a_π, b_π). Each of these pairs gives a different shape for the distribution of the prior for π, as shown in the coming plots. Recall that, in our setup, both a_π and b_π are both divided by two, which is why the numbers in the code below are half what we state for the a_π and b_π pairs.

Our chosen values for a_π and b_π are: $(a_\pi = 14, \beta_\pi = 50)$, $(a_\pi = 1, \beta_\pi = 2)$, $(a_\pi = 1, \beta_\pi = 12)$ and $(a_\pi = 12, \beta_\pi = 1)$

Here are plots of the prior for π given the different parameter values.

```
par(mfrow = c(2, 2))
plot(density(rbeta(10000, 7, 25)),
     main = TeX("$p\\left(\\pi \\, |\\, a_{\\pi}/2 = 7, \\, b_{\\pi}/2 = 25\\right)$"))
plot(density(rbeta(10000, .5, 1)),
     main = TeX("$p\\left(\\pi \\, |\\, a_{\\pi}/2 = .5, \\, b_{\\pi}/2 = 1\\right)$"))
plot(density(rbeta(10000, .5, 6)),
     main = TeX("$p\\left(\\pi \\, |\\, a_{\\pi}/2 = .5, \\, b_{\\pi}/2 = 6\\right)$"))
plot(density(rbeta(10000, 6, .5)),
     main = TeX("$p\\left(\\pi \\, |\\, a_{\\pi}/2 = 6, \\, b_{\\pi}/2 = .5\\right)$"))
```



We run our Gibbs sampler for the full $10,000$ iterations for each of the new chosen values of a_π, b_π and get the following models.

```
post1450 = readRDS(file = "postone1450.Rds")
post12 = readRDS(file = "post_1_2.Rds")
postone12 = readRDS(file = "postone12.Rds")
post12one = readRDS(file = "postone12one.Rds")
all.posts.new.pi = list(post1450 = post1450, post12 = post12,
                        postone12 = postone12, post12one = post12one)
```

We then perform the same burin-in and thinning operations on each of these models.

```
thin.all.post.new = list(4)
for (i in 1:4) {
  thin.all.post.new[[i]] = burn.and.thin(post = all.posts.new.pi[[i]], bi = 2000, ti = 10)
}
thin.post1450 = thin.all.post.new[[1]]
thin.post12 = thin.all.post.new[[2]]
thin.postone12 = thin.all.post.new[[3]]
thin.post12one = thin.all.post.new[[4]]
```

Let's now compare the quantiles with the new parameters to those from our original parameters.

The quantiles for our posterior distribution with parameters $a_\pi = 14, b_\pi = 50$ are

```
quant.thin.post1450
```

```
##               2.5%          25%          50%         75%      97.5%
## beta1   0.54458131  0.625052153 6.740538e-01 0.715749676 0.78947670
## beta2   2.46285439  2.543308803 2.577317e+00 2.606713463 2.66321652
## beta3  -0.01911539 -0.006356512 2.722249e-04 0.007738956 0.01889322
## beta4   1.80143309  1.854477930 1.889232e+00 1.922664619 1.98197544
## beta5  -0.01736754 -0.004325402 2.440048e-03 0.009452145 0.02226830
## beta6   1.15425016  1.211103054 1.246143e+00 1.288174008 1.35626376
## beta7  -0.02143331 -0.006938310 3.023007e-05 0.006798781 0.01832722
## beta8   1.85861337  1.922751105 1.953421e+00 1.989942884 2.06384602
## beta9   1.37682499  1.442091606 1.479397e+00 1.514743578 1.58575175
## beta10  0.68215894  0.749849978 7.859712e-01 0.821535807 0.88752274
```

The quantiles for our posterior distribution with parameters $a_\pi = 1, b_\pi = 2$ are

```
quant.thin.post12
```

```
##               2.5%          25%           50%         75%      97.5%
## beta1   0.55261349  0.638564183  0.6820146282 0.721507608 0.81297871
## beta2   2.47303404  2.532020057  2.5700676456 2.610842141 2.67369090
## beta3  -0.01876954 -0.005624993  0.0005657306 0.007822439 0.02030390
## beta4   1.78345574  1.836929770  1.8721084133 1.912144677 1.99356110
## beta5  -0.01804513 -0.004396286  0.0023314618 0.010192948 0.02628661
## beta6   1.14891170  1.216012835  1.2529093709 1.291879405 1.36757757
## beta7  -0.02270787 -0.007018645 -0.0006828245 0.006171006 0.01735220
## beta8   1.85482556  1.916775955  1.9596522924 2.001511356 2.07854863
## beta9   1.38132785  1.446244392  1.4863590600 1.523652505 1.59480484
## beta10  0.66581334  0.732275682  0.7739980772 0.816136329 0.87516052
```

The quantiles for our posterior distribution with parameters $a_\pi = 1, b_\pi = 12$ are

```
quant.thin.postone12
```

```
##               2.5%          25%           50%         75%      97.5%
## beta1   0.53508221  0.604761423  0.6451261179 0.685308005 0.76894950
## beta2   2.49353353  2.554492505  2.5876985405 2.623863058 2.69496356
## beta3  -0.01906633 -0.005141766  0.0021137028 0.009086543 0.02200877
## beta4   1.75040536  1.830079876  1.8709100246 1.904507551 1.98428712
## beta5  -0.01788164 -0.004393974  0.0023102314 0.009364124 0.02190172
## beta6   1.15366381  1.221492971  1.2645092726 1.306890208 1.38625818
## beta7  -0.01956855 -0.007086872 -0.0006986127 0.006054905 0.01944690
## beta8   1.85686358  1.918118082  1.9552897924 1.993640299 2.07133634
## beta9   1.38828172  1.449024234  1.4818215023 1.516005626 1.57744187
## beta10  0.68698868  0.749042997  0.7887096101 0.820756760 0.89930267
```

The quantiles for our posterior distribution with parameters $a_\pi = 12, b_\pi = 1$ are

```
quant.thin.post12one
```

```
##               2.5%          25%          50%         75%      97.5%
## beta1   0.55344776  0.617935230  0.657984920 0.702721667 0.82586663
## beta2   2.45401787  2.542733773  2.581184495 2.617297945 2.71008660
## beta3  -0.04154475 -0.006544857  0.001720934 0.010785596 0.05419263
## beta4   1.75658861  1.833715016  1.864948662 1.901436493 1.96697310
## beta5  -0.02233044 -0.001629989  0.008192762 0.044533480 0.16274178
## beta6   1.15132920  1.220500873  1.260449367 1.294997879 1.37698114
## beta7  -0.11324190 -0.009422286 -0.001711592 0.008239628 0.06065891
## beta8   1.83565800  1.906266110  1.937431371 1.977541158 2.07220474
## beta9   1.38637354  1.448791257  1.486479235 1.518787331 1.59798828
## beta10  0.67595195  0.737043941  0.776439441 0.822162759 0.90431084
```

As the quantiles show, there is very little difference between the posterior distributions when we change the values of a_π and b_π in the prior for π. This tells us that the posterior distribution of β is not sensitive to changes in the prior distribution of π.

Question (k)

Model Checking. Generate $10,000$ replications of the data y^{rep} using the same x_i as the original data. Compare the posterior mean and median. Based on that, does the model generate predicted results similar to the observed data in the study?

solution:

We first create our vector of y^{rep}'s, where each $y^{\text{rep}} \in \mathbb{R}^n$.

```
y.rep = matrix(NA, nrow = n, ncol = n_iter)
Sig = diag(n)
mu = X %*% t(post)

# for (i in 1:n_iter) {
#   y.rep[, i] = mvrnorm(mu = mu[, i], Sigma = Sig)
# }

y.rep = readRDS(file = "yrep_most_recent.Rds")
```

We then compare the posterior mean and median to that of the observed data, y.

```
plot(density(apply(X = y.rep, MARGIN = 2, FUN = mean)), main = "Distribution of Posterior Mean")
abline(v = mean(y), col = "red")
```

```
plot(density(apply(X = y.rep, MARGIN = 2, FUN = median)), main = "Distribution of Posterior Median")
abline(v = median(y), col = "red")
```

As seen in the figures below, the distributions of both the posterior mean and median have their peaks almost exactly in line with the mean / median of the observed data, y. This leads us to believe that our model does a great job of generating predicted results similar to the observed data in the study.

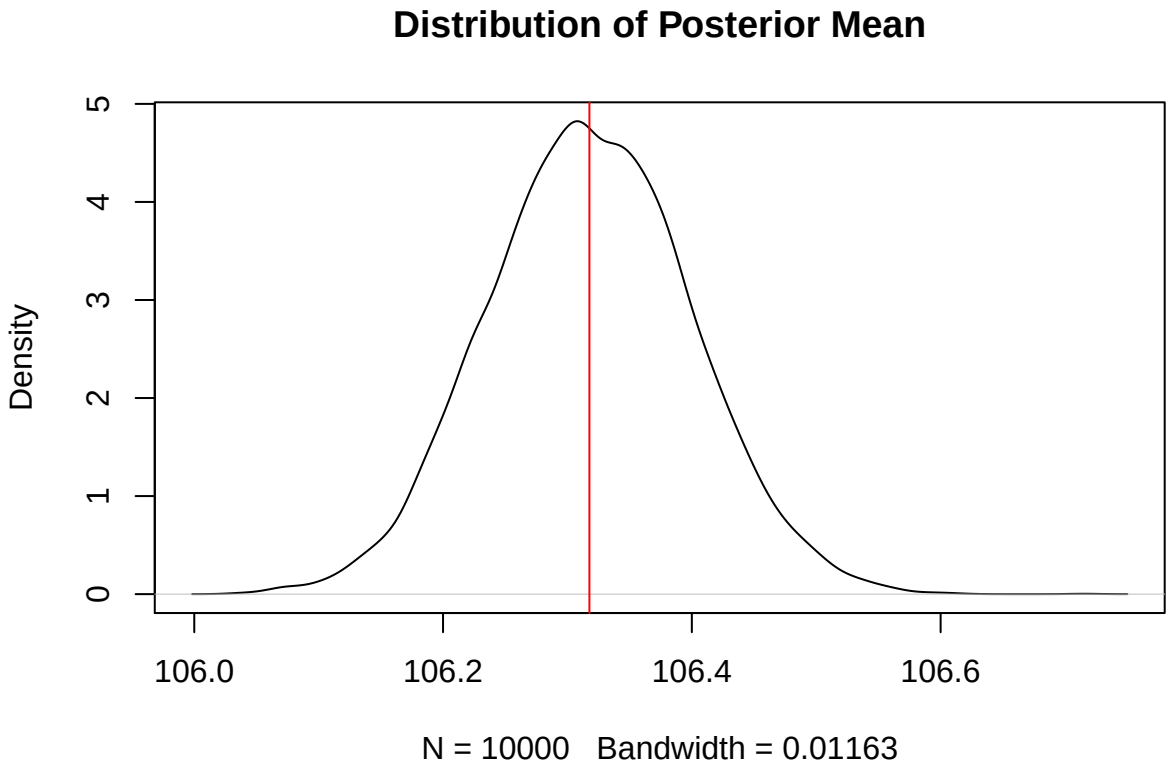


Figure 1: Vertical red line shows the mean of the observed data for comparison

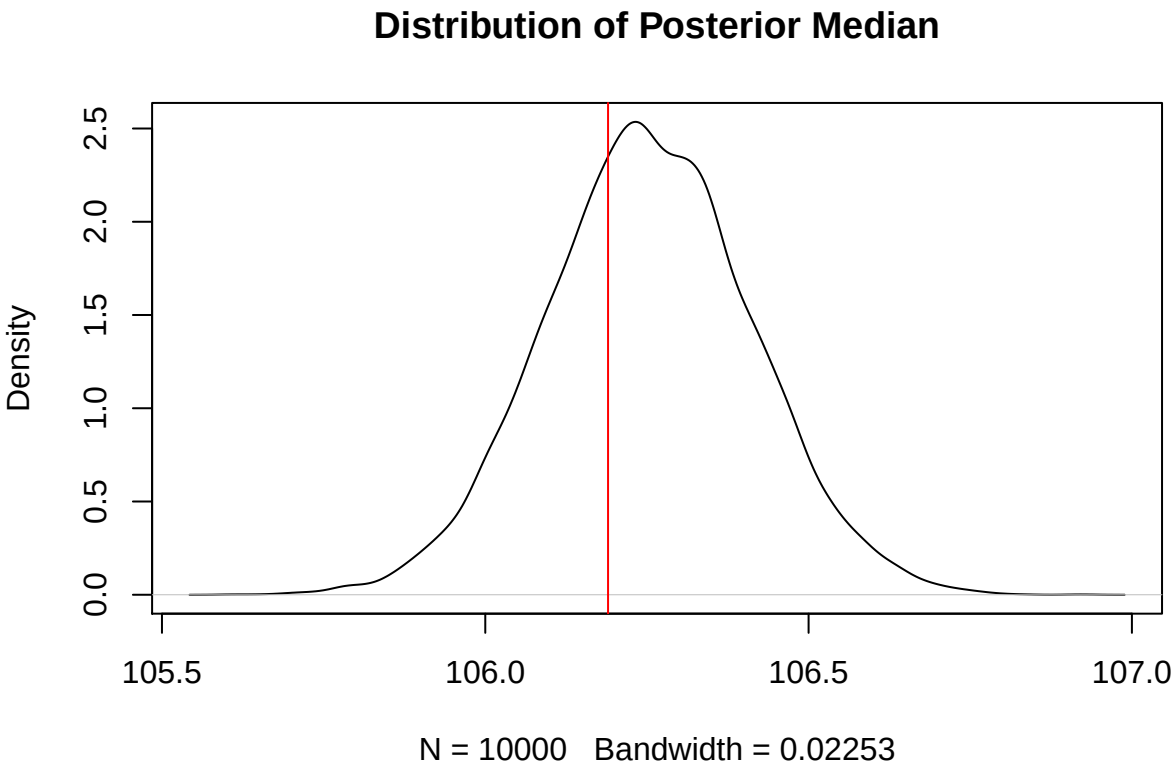


Figure 2: Vertical red line shows the median of the observed data for comparison

Appendix

The Variances of the β_j's is given in the following output

```
ThinnedPosterior = as.data.frame(thin.post)

thinned.var.summary <-
  list("Variance of Thinned Posterior" =
         list("Variance beta1:" = ~ var(beta1),
              "Variance beta2:" = ~ var(beta2),
              "Variance beta3:" = ~ var(beta3),
              "Variance beta4:" = ~ var(beta4),
              "Variance beta5:" = ~ var(beta5),
              "Variance beta6:" = ~ var(beta6),
              "Variance beta7:" = ~ var(beta7),
              "Variance beta8:" = ~ var(beta8),
              "Variance beta9:" = ~ var(beta9),
              "Variance beta10:"= ~ var(beta10)))

summary_table(ThinnedPosterior, thinned.var.summary)
```

```
##
##
## |                                 |ThinnedPosterior (N = 801) |
## |:--------------------------------|:--------------------------|
## |**Variance of Thinned Posterior**|                 |
## |   Variance beta1:     |0.00403001059030698        |
## |   Variance beta2:     |0.00306726462643133        |
## |   Variance beta3:     |0.000161575876725779       |
## |   Variance beta4:     |0.00242601611044185        |
## |   Variance beta5:     |0.000418943729365998       |
## |   Variance beta6:     |0.00326385191683435        |
## |   Variance beta7:     |0.000330977277781282       |
## |   Variance beta8:     |0.00284496100142858        |
## |   Variance beta9:     |0.00344744021196545        |
## |   Variance beta10:    |0.00287115226912168        |
```