ST559 Homework 6

Nick Sun

June 4, 2020

Abstract

From *Ch. 11, Bayesian Data Analysis*, do problems 2, 3, 4, and 7

1 Q2

Replicate the computations for the bioassay example of section 3.7 using the Metropolis algorithm. Be sure to define your starting points and your jumping rule. Compute the log-densities (see pg. 261). Run the simulations long enough for approximate convergence.

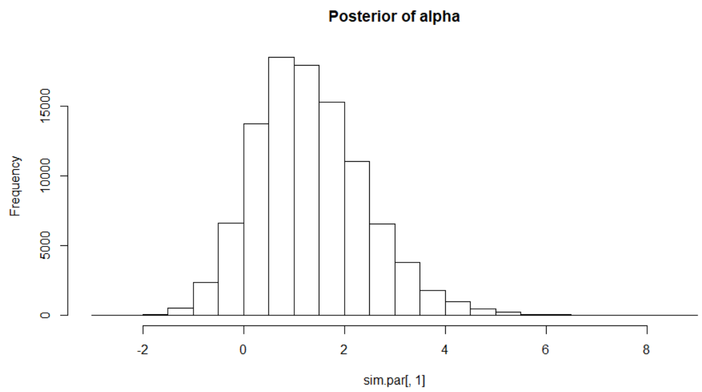
The bioassay data is

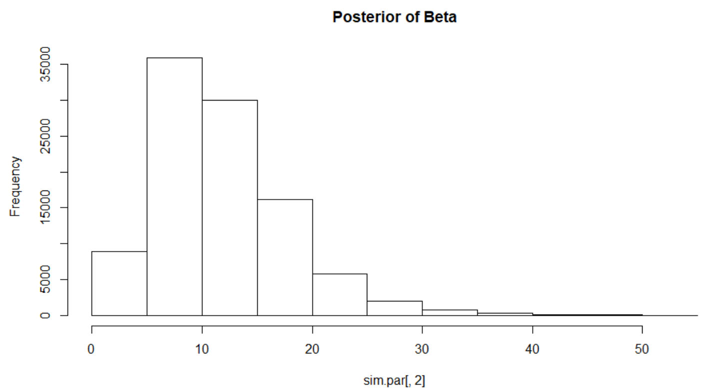
x	n	y
-.86	5	0
-.3	5	1
-.05	5	3
.73	5	5

answer We are given from the textbook that

$$p(y_i|\alpha,\beta,n,x) \propto \text{logit}^{-1}(\alpha+\beta x)^y \left(1-\text{logit}^{-1}(\alpha+\beta x)^{n-y}\right)$$
$$p(\alpha,\beta|y,n,x) \propto p(\alpha,\beta)\prod_{i=1}^{n} p(y_i|\alpha,\beta,n,x)$$

Using an initial guess for $(\alpha,\beta) = (.8, 7.7)$ and a bivariate normal proposal distribution $q(\alpha^*,\beta^*) = (N(\alpha_{i-1}, 9), N(\beta_{i-1}, 9))$, we get the following posterior histograms after $n = 100000$ draws from the Metropolis-Hastings algorithm.





The R code that I used for this problem is provided below.

Additionally, here are the posterior for the individual θ parameters.

```r
bioassay <- data.frame(
    x = c(-.86, -.3, -.05, .73),
    n = rep(5, 4),
    y = c(0, 1, 3, 5)
)

# Initial parameter guess from textbook pg. 76
ini.par <- c(.8, 7.7)

nsim <- 100000
sim.par <- matrix(0, nsim, 2)

# This function computes p(y | alpha, beta, n, x, y)
inv.logit <- function(alpha, beta, x, n , y) {
    (exp(alpha + beta*x)/(exp(alpha + beta*x) + 1))^(y) * (1 - exp(alpha +
        beta*x)/(exp(alpha + beta*x) + 1))^(n-y)
}

# This function computes the log of p(y | alpha, beta, n, x, y)
log.lik <- function(alpha, beta, x, n, y){
    sum(log((((exp(alpha+beta*x)/(1+exp(alpha+beta*x)))^y)*((1/(1+exp(alpha+beta*x)))^(n-y)))))
}

alpha.star <- alpha.prev <- ini.par[1]
beta.star <- beta.prev <- ini.par[2]
n.rej <- 0
set.seed(182)

for (i in 1:nsim) {
    # A reasonable proposal distribution is a bivariate normal which has nice
        Metropolis-Hastings properties
    alpha.star <- rnorm(1, mean = alpha.prev, sd = 3)
    beta.star <- rnorm(1, mean = beta.prev, sd = 3)

    # The M-H ratio is the exponent of the differences in the log likelihoods
    ratio <- exp(log.lik(alpha = alpha.star, beta = beta.star, x = bioassay$x, n
       = bioassay$n, y = bioassay$y) -
                log.lik(alpha = alpha.prev, beta = beta.prev, x = bioassay$x, n
                   = bioassay$n, y = bioassay$y))

    if (ratio >= 1) {
        sim.par[i,] <- c(alpha.star, beta.star)
        alpha.prev <- alpha.star
        beta.prev <- beta.star
    } else if ((0 < ratio) && (ratio < 1)) {
        tmp <- runif(1)
        if (tmp < ratio) {
            sim.par[i,] <- c(alpha.star, beta.star)
            alpha.prev <- alpha.star
            beta.prev <- beta.star
        } else {
            n.rej <- n.rej + 1
            sim.par[i,] <- c(alpha.prev, beta.prev)
        }
    }
}

par(mfrow = c(2,1))
```

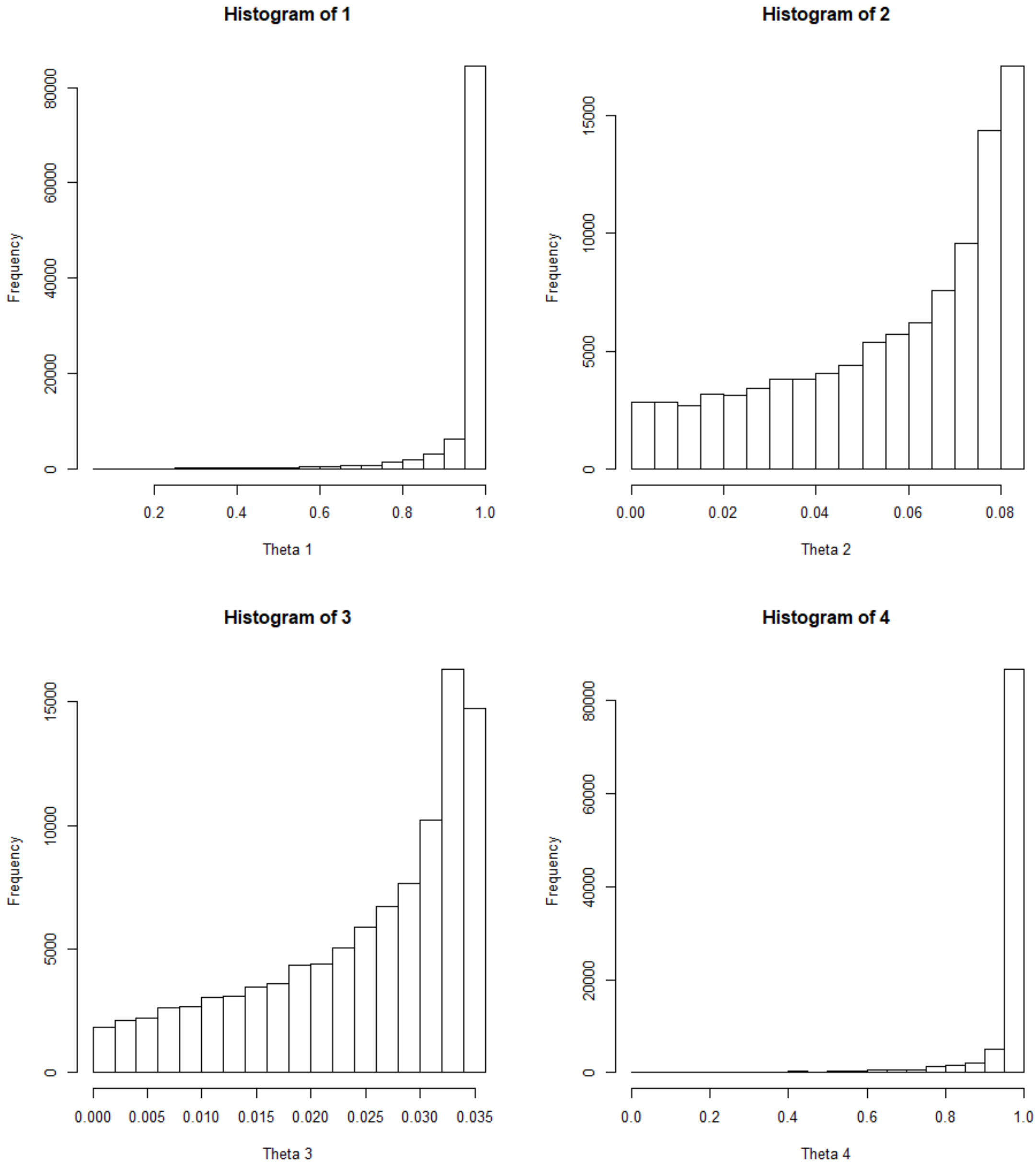


Figure 1: Posterior histograms of each theta parameter

```
summary(sim.par[,1])
summary(sim.par[,2])

hist(sim.par[,1], main = "Posterior of alpha")
hist(sim.par[,2], main = "Posterior of Beta")
```

2 Q3

Using this data of 5 observations from each of the six machines:

Machine	Measurements
1	83,92,92,46,67
2	117,109,114,104,87
3	101,93,92,86,67
4	105,119,116,102,116
5	79,97,103,79,92
6	57,92,104,77,100

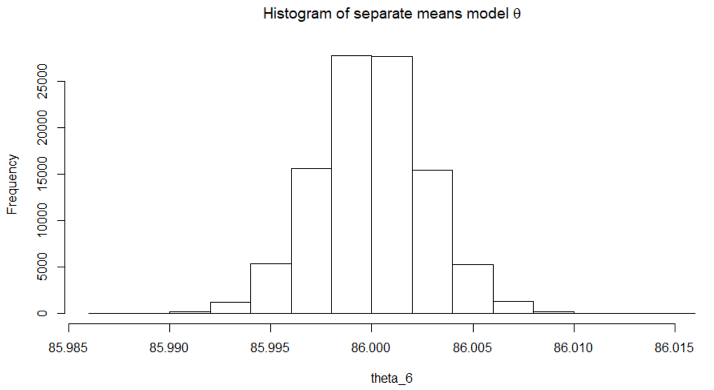
implement a separate, pooled, and hierarchical Gaussian model with common variance as detailed in section 11.6. We are interested in the quality of a seventh machine. Run the simulation long enough for approximate convergence.

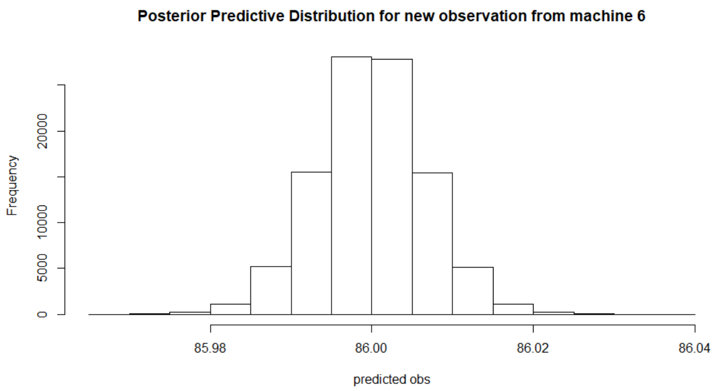
From each of the three models, report:

- the posterior distribution of the mean of the quality measurements of the sixth machine
- the predictive distribution for another quality measurement from the sixth machine
- the posterior distribution of the mean of the quality measurements of the seventh machine

separate means model Below is my calculated posterior histogram for θ_6, the posterior predictive distribution for a new observation, as well as the code I used. Note that the posterior distribution of the mean of the quality measurements of the seventh machine are going to be the same as the sixth machine because we did not define a prior with this model from which we should draw θ_7.

```
ini.par.separate <- c(dat.mean, MSW = MSW)
separate.par <- matrix(0, nsim, 7)

for (i in 1:nsim) {
  separate.par[i,7] <- 1/rgamma(n = 1, shape = (24/2), scale = (24*MSW)/2)
    for(j in 1:6){
      separate.par[i,j] <- rnorm(n = 1, mean = ini.par.separate[j], sd =
         sqrt(separate.par[i,7])/sqrt(n))
    }
}

hist(separate.par[,6], main = "Separate Means model", xlab = "theta_6")
```

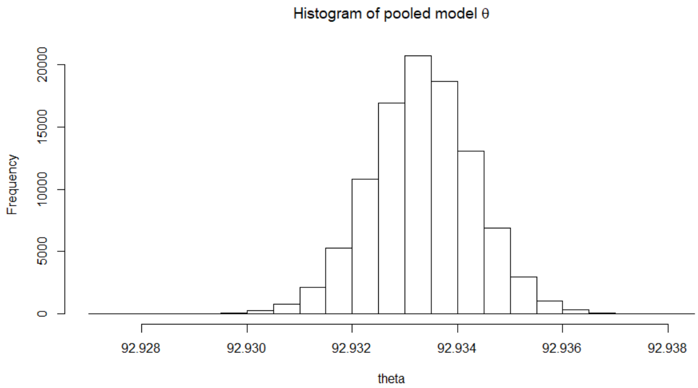


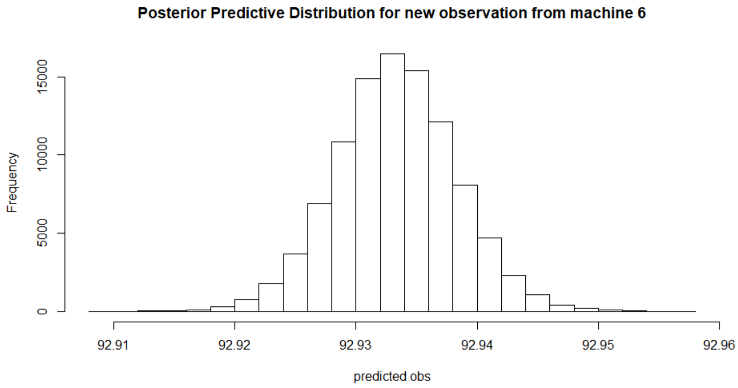


- Posterior distribution of $\theta_6 \sim N(86, 7.44 * 10^{-6})$
- Predictive distribution of new observation $\sim N(86, 4.45 * 10^{-5})$
- Posterior distribution of $\theta_7 \sim N(86, 7.44 * 10^{-6})$

pooled means model Below is the code and histograms for the pooled means model. Since there is only one θ parameter here, this code was pretty short. Note again that the posterior distribution for θ_7 is the same as the posterior distribution for $\theta_1, \ldots, \theta_6$ since all θ are assumed to be the same.

```
ini.par.pooled <- c(mean(dat.mean), var(as.vector(as.matrix(machine.data))))
pooled.par <- matrix(0, nsim, 2)

for(i in 1:nsim){
  pooled.par[i,2] <- 1/rgamma(n = 1, shape = 24/2, scale =
      (24*ini.par.pooled[2])/2)
  pooled.par[i,1] <- rnorm(n = 1, mean = ini.par.pooled[1], sd =
      sqrt(pooled.par[i,2]/24))
}

hist(pooled.par[,1], main = "Histogram of pooled mean", xlab = "theta")
```





- Posterior distribution of $\theta_6 \sim N(92.93, 9.74 * 10^{-7})$
- Predictive distribution of new observation $\sim N(92.93, 2.45 * 10^{-5})$
- Posterior distribution of $\theta_7 \sim N(92.93, 9.74 * 10^{-7})$

hierarchical means model This model is fairly complicated. I followed with the textbook process on page 289 which has the following conditional distributions to use in the Gibbs sampler:

- $\theta_j|\mu, \sigma, \tau, y \sim N(\hat{\theta}_j, V_{\theta_j})$
- $\mu|\theta, \sigma, \tau, y \sim N(\hat{\mu}, \tau^2/J)$
- $\sigma^2|\theta, \mu, \tau, y \sim \text{Inv-}\chi^2(n, \hat{\sigma^2})$
- $\tau^2|\theta, \mu, \sigma, y \sim \text{Inv-}\chi^2(J-1, \hat{\tau^2})$

In this model, we have 9 parameters to keep track of: $\theta_i, i \in (1, \ldots, 6)$, μ, σ^2, and τ^2. The first iteration of Gibbs sampling will be done by manually where θ_i^0 are just randomly sampled points from each of the machine samples.

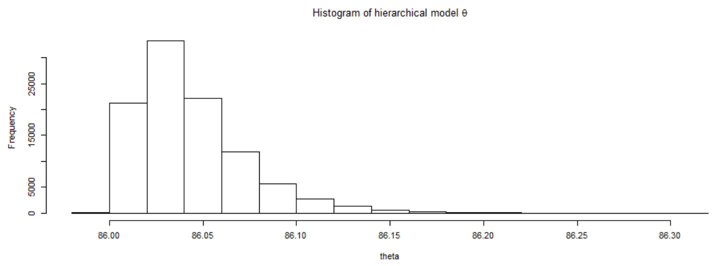
```
set.seed(182)
ini.y <- c(sample(machine.data$y1, 1),
           sample(machine.data$y2, 1),
           sample(machine.data$y3, 1),
           sample(machine.data$y4, 1),
           sample(machine.data$y5, 1),
           sample(machine.data$y6, 1))
hier.par <- matrix(0, nsim, 9)

# initialize with randomly sampled point for theta_j, their mean for mu, 0 for
    sigma2, 0 for tau
hier.par[1,] <- c(ini.y, mean(ini.y), 0, 0)
```

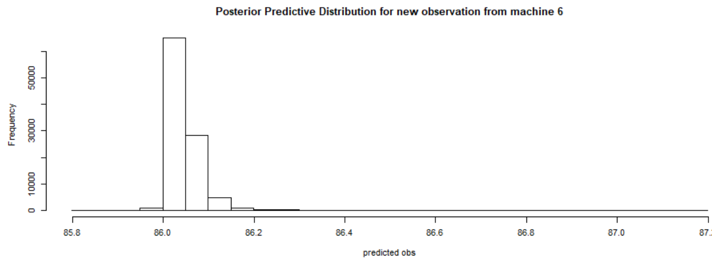
```
#Need to do first iteration by hand.
#update tau
hier.par[1,9] <- 1/rgamma(n = 1, shape = (J-1)/2,
                          scale = sum((hier.par[1,1:6] - hier.par[1,7])^2)/2)

#update sigma2
sigma_hat <- rep(0,6)
for(k in 1:6){
  sigma_hat[k] <- sum((t(machine.data)[k,]-hier.par[1,k])^2)
}
sigma_hat <- sum(sigma_hat)
hier.par[1,8] <- 1/rgamma(n = 1, shape = (n*J)/2, scale = sigma_hat/2)

#update mu
hier.par[1,7] <- rnorm(n = 1, mean = mean(hier.par[1,1:6]), sd =
    sqrt(hier.par[1,9]/J))

#update theta_i's
for(j in 1:6){
  vhat <- 1/(1/hier.par[1,9]+n/hier.par[1,8])
  theta_hat <-
      (hier.par[1,7]/hier.par[1,9]+(n/hier.par[1,8])*mean(t(machine.data)[j,]))/(1/hier.par[1,9
  hier.par[1,j] <- rnorm(n = 1, mean = theta_hat, sd = sqrt(vhat))
}
```

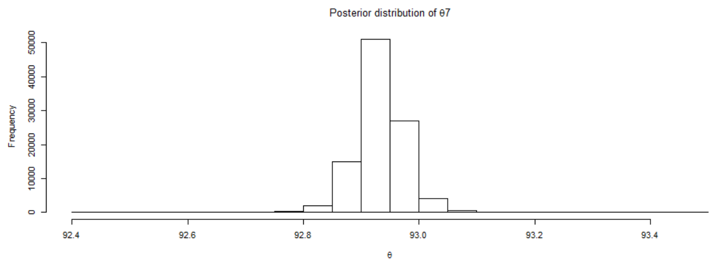
After we have run the initial iteration, we can now produce the next $n = 10000$ iterations of the Gibbs sampler. The main loop in the sampler is here.

```
for(i in 2:nsim){

  hier.par[i,9] <- 1/rgamma(n = 1, shape = (J-1)/2,
                            scale =
                                sum((hier.par[(i-1),1:6] - hier.par[(i-1),7])^2)/2)

  sigma_hat <- rep(0,6)
  for(k in 1:6){
    sigma_hat[k] <- sum((t(machine.data)[k,]-hier.par[(i-1),k])^2)
  }
  sigma_hat <- sum(sigma_hat)
  hier.par[i,8] <- 1/rgamma(n = 1, shape = (n*J)/2, scale = sigma_hat/2)

  hier.par[i,7] <- rnorm(n = 1, mean = mean(hier.par[(i-1),1:6]), sd =
      sqrt(hier.par[i,9]/J))

  for(j in 1:6){
    vhat <- 1/(1/hier.par[i,9]+n/hier.par[i,8])
    theta_hat <- (hier.par[i,7]/hier.par[i,9]+(n/hier.par[i,8])*
                  mean(t(machine.data)[j,]))/(1/hier.par[i,9]+n/hier.par[i,8])
    hier.par[i,j] <- rnorm(n = 1, mean = theta_hat, sd = sqrt(vhat))
  }
}
```

I found that after 10 or so iterations, the samples for the most part stabilized, so I excluded those initial “burn” samples as the histograms and summaries.







- Posterior distribution of $\theta_6 \sim N(86.04, .075)$
- Predictive distribution of new observation $\sim N(86.04, .0009)$
- Posterior distribution of $\theta_7 \sim N(92.93, .089)$

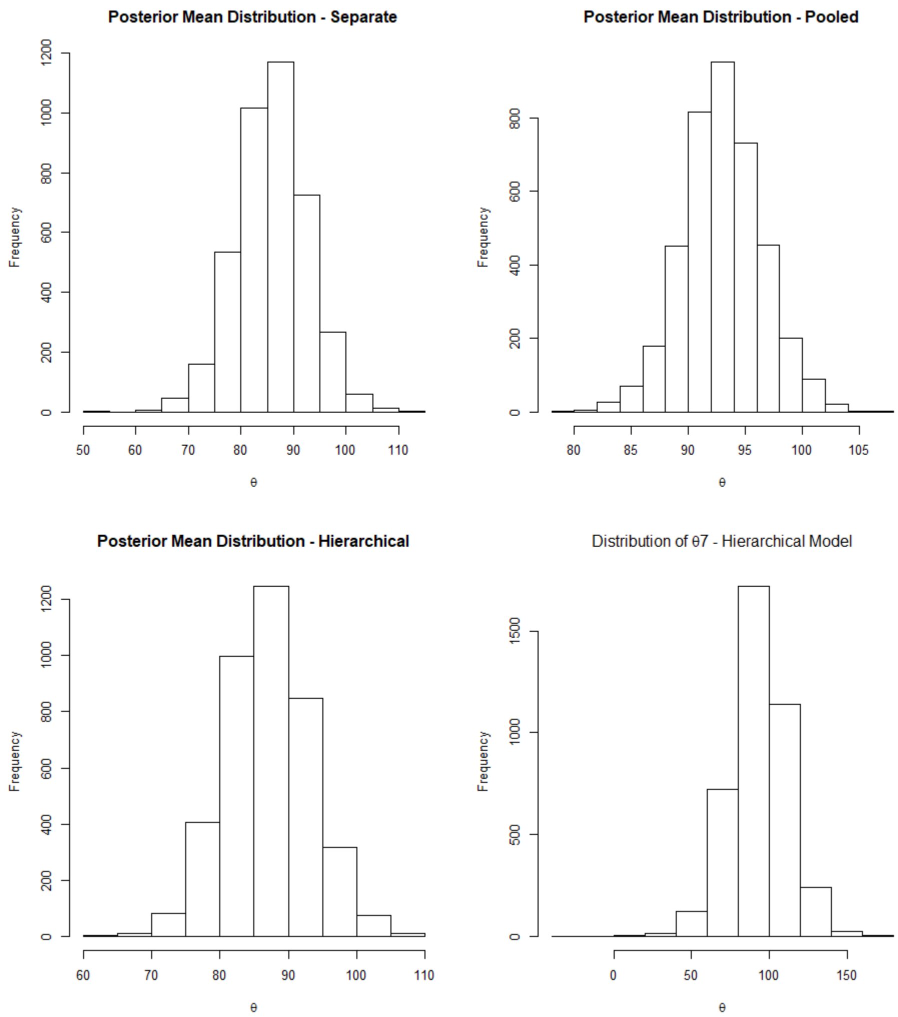
I believe there must be some error in my code, since the variance of the predictive distribution is smaller than the posterior distribution of θ_6.

3 Q4

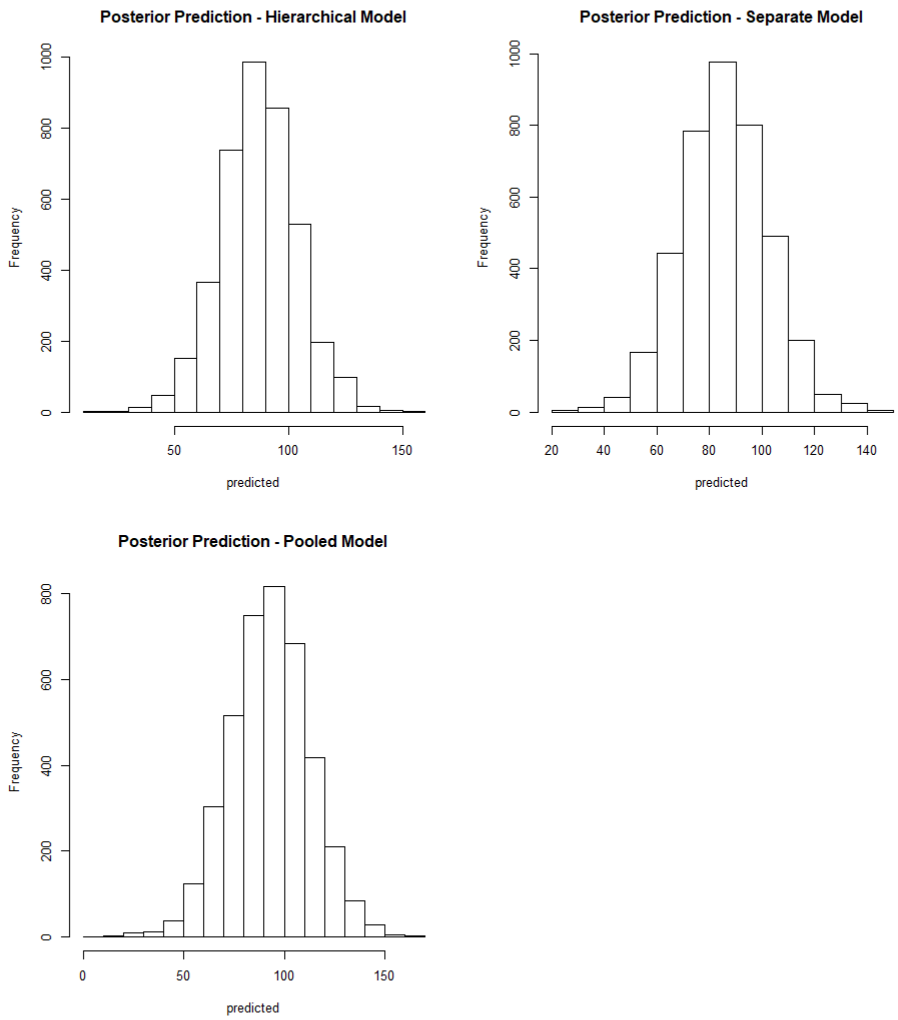
Extend the model from Q3 by adding a hierarchical model for the variances of the machine quality measurements. Use an Inv-χ^2 prior distribution for variances with unknown σ_0^2 and fixed degrees of freedom. The conditional distribution of σ_0^2 is not of simple form, but you can sample from its distribution, for example, using grid sampling.

answer After some suggestions during office hours, I decided to do this problem using `RStan`. My three separate Stan files will be provided below my histograms for this problem.

The first figure displays the posterior distribution of θ_6 for each model, as well as the distribution of θ_7 for the hierarchical model.



The second figure displays the posterior predictive distribution of a new observation from machine 6 for each model.



Stan code for separate means

```
data {
    int<lower=0> J; // Number of machines
    int<lower=0> n; // number of samples per machine
    matrix[J, n] X; // a JxN matrix containing all the data points
    int nu; // degrees of freedom for the inv-Achisq prior for sigma2
}

parameters {
    vector[J] theta; // vector of the means for the individual machines
    real<lower=0> sigmasq; // the variance of the thetas
    real<lower=0> sigmasq0; // the scale parameter for the prior of the variance
    // since we don't know what sigma0 is, we will use a uniform prior for it
    // as per the stan documentation, this is just declaring it in the
        parameters block
}

model {
    sigmasq ~ scaled_inv_chi_square(nu, sigmasq0);
    for (i in 1:J) {
        X[i] ~ normal(theta[i], sqrt(sigmasq));
    }
}
```

Stan code for pooled means

```
data {
    int<lower=0> J;
    int<lower=0> n;
    matrix[J, n] X;
    int<lower=0> nu;
}

parameters {
    real theta;
    real<lower=0> sigmasq;
    real<lower=0> sigmasq0;
}

model {
    sigmasq ~ scaled_inv_chi_square(nu, sigmasq0);
    for (i in 1:J) {
        X[i] ~ normal(theta, sqrt(sigmasq));
    }
}
```

Stan code for hierarchical model

```
data {
    int<lower=0> J;
    int<lower=0> n;
    matrix[J, n] X;
    int<lower=0> nu;
}

parameters {
    vector[J] theta;
    real mu;
    real<lower=0> sigmasq;
    real<lower=0> sigmasq0;
    real<lower=0> tausq;
}

model {
    sigmasq ~ scaled_inv_chi_square(nu, sigmasq0);
    theta ~ normal(mu, sqrt(tausq));
    for (i in 1:J) {
        X[i] ~ normal(theta[i], sqrt(sigmasq));
    }
}
```

4 Q7

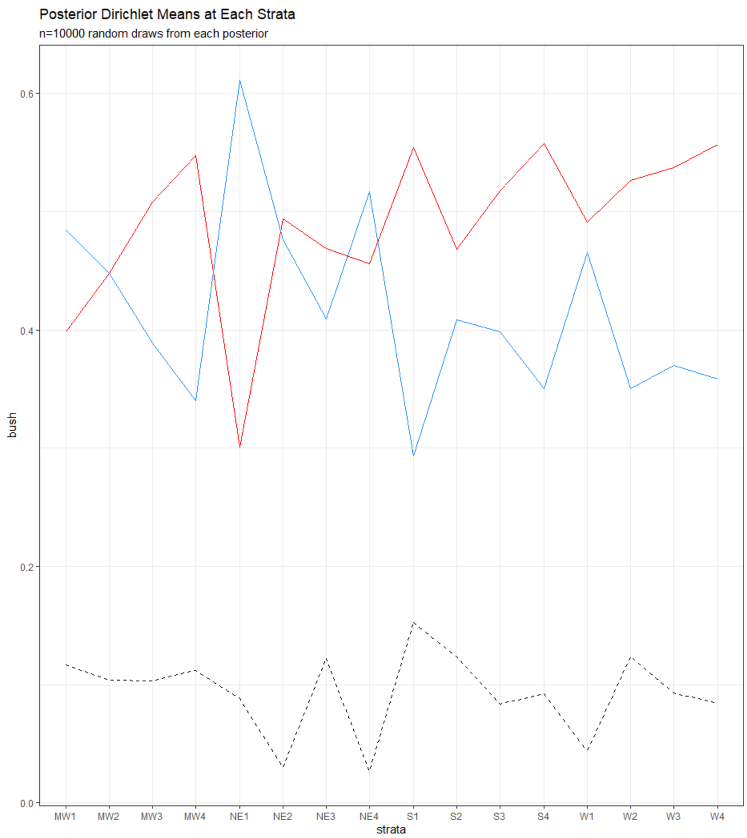
Section 8.3 presents an analysis of a stratified sample survey using a hierarchical model on the stratum probabilities.

4.1 part a.

Perform the computations for the simple nonhierarchical model described in the example.

This problem is (relatively) straightforward. We are intersted in modeling the proportions of Bush, Dukakis, and No opinion voters in each strata. The easiest way to do this that we have covered in class is with the Dirichlet prior, which will yield a Dirichlet posterior after conditioning with multinomial distributed data. I will use a noninformative Dirichlet prior $\pi(p_1, p_2, p_3) \propto Dirichlet(1/2, 1/2, 1/2)$.

The first part of this problem involves figuring out the counts in each of the strata. This is easy enough, since the textbook tells us that 1447 total adults were in the sample. After this, we can compute the posterior probability distributions for each strata. For ease of plotting purposes, I computed the mean of each of the probability distributions so that we could plot the strata against each other with their respective posterior means for Dukakis and Bush.




```
> post.df
bush      dukakis         noops     l
NE1  0.3006309  0.6113988  0.08797031   1
NE2  0.4940330  0.4760971  0.02986992   2
NE3  0.4689885  0.4090034  0.12200809   3
NE4  0.4563083  0.5168835  0.02680825   4
MW1  0.3986051  0.4847569  0.11663798   5
MW2  0.4479915  0.4480324  0.10397609   6
MW3  0.5083175  0.3887653  0.10291722   7
MW4  0.5477800  0.3403804  0.11183961   8
S1   0.5542165  0.2932569  0.15252658   9
S2   0.4679640  0.4088113  0.12322471  10
S3   0.5178651  0.3986478  0.08348714  11
S4   0.5577930  0.3501038  0.09210324  12
W1   0.4915228  0.4652081  0.04326913  13
W2   0.5262533  0.3502351  0.12351161  14
W3   0.5372355  0.3699020  0.09286241  15
W4   0.5569620  0.3585072  0.08453078  16
```

The code to produce these posterior distributions and the plot is provided below.

```
library(gtools)

election.data <- data.frame(
    bush = c(.3, .5, .47, .46, .40, .45, .51, .55, .57, .47, .52, .56, .5, .53,
        .54, .56),
    dukakis = c(.62, .48, .41, .52, .49, .45, .39, .34, .29, .41, .40, .35, .47,
        .35, .37, .36),
    noop = c(.08, .02, .12, .02, .11, .10, .10, .11, .14, .12, .08, .09, .03,
        .12, .09, .08),
```

```
    stat.p = c(.032, .032, .115, .048, .032, .065, .08, .1, .015, .066, .068,
        .126, .023, .053, .086, .057)
)

N <- 1447
election.data$n <- N*election.data$stat.p

nsim <- 10000
posts <- list()

for (i in 1:nrow(election.data)) {
    posts[[i]] <- apply(rdirichlet(n = nsim, alpha =
        as.numeric(election.data$n[i]*election.data[i,c(1:3)] + .5)), 2, mean)
}

post.df <- as.data.frame(do.call(rbind, posts))
colnames(post.df) <- c("bush", "dukakis", "noops")
rownames(post.df) <- c("NE1", "NE2", "NE3", "NE4",
                        "MW1", "MW2", "MW3", "MW4",
                        "S1", "S2", "S3", "S4",
                        "W1", "W2", "W3", "W4")
post.df$l <- 1:16
post.df
library(tidyverse)
post.df %>%
    rownames_to_column(var = "strata") %>%
    ggplot() +
    geom_line(mapping =aes(x = strata, y = bush, group = 1),
              color = "red") +
    geom_line(mapping =aes(x = strata, y = dukakis, group = 1),
              color = "dodgerblue1") +
    geom_line(mapping =aes(x = strata, y = noops, group = 1),
              linetype = 2) +
    labs(
        title = "Posterior Dirichlet Means at Each Strata",
        subtitle = "n=10000 random draws from each posterior"
    ) +
    theme_bw()
```

4.2 part b.

Using the Metropolis algorithm, perform the computations for the hierarchical model, using the results from part a. as a starting distribution. Check by comparing your simulations to the results in Figure 8.1b.

answer Honestly I have no idea how to start this problem haha