Assignment #9

Elliot Smith

10/28/2018

Problem 1

Part i

$p(y_h, \lambda_h|\alpha, \mu) = p(y_h|\lambda_h, \alpha, \mu) \times p(\lambda_h|\alpha, \mu)$

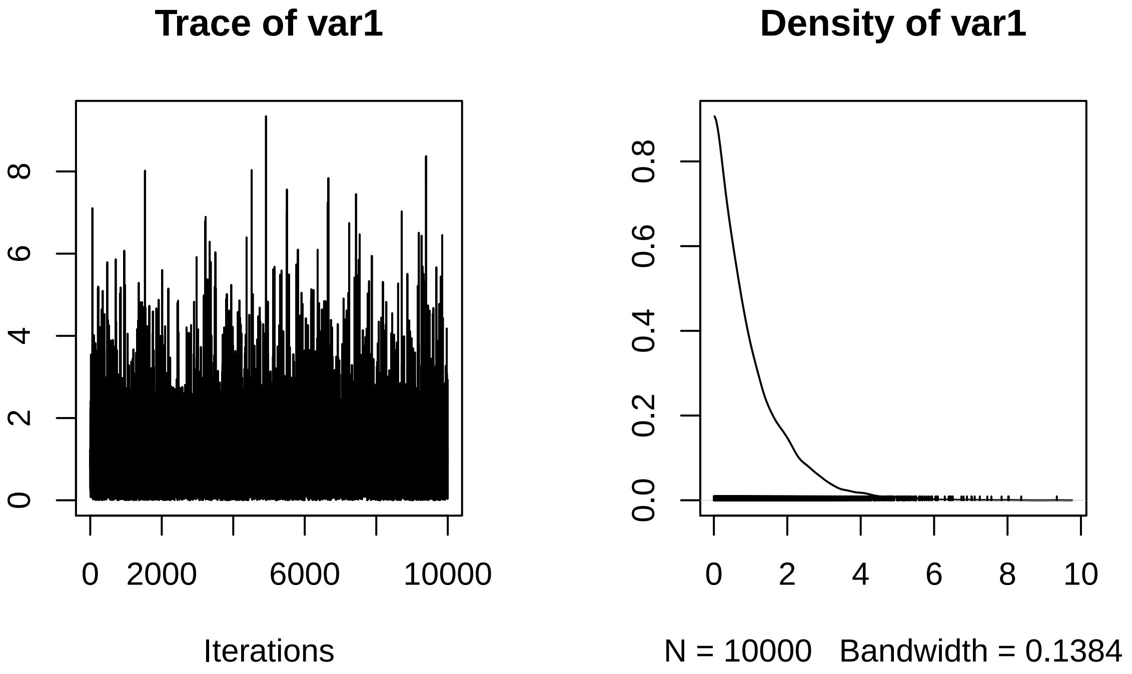
$$\begin{aligned}
p(y_h|\alpha, \mu) &= \int_\lambda p(y_h, \lambda_h|\alpha, \mu) d\lambda \\
&= \int_\lambda p(y_h|\lambda_h, \alpha, \mu) \times p(\lambda_h|\alpha, \mu) d\lambda \\
&= \int_\lambda \frac{1}{y_h!} (x_h \lambda_h)^{y_h} e^{-x_h \lambda_h} \times \frac{(\frac{\alpha}{\mu})^\alpha}{\Gamma(\alpha)} \lambda_h^{\alpha-1} e^{-(\frac{\alpha}{\mu})\lambda_h} d\lambda \\
&= \frac{x_h^{y_n}}{y_h!} \frac{(\frac{\alpha}{\mu})^\alpha}{\Gamma(\alpha)} \int_\lambda \lambda_h^{y_h+\alpha-1} e^{-x_h\lambda_h - (\frac{\alpha}{\mu})\lambda_h} d\lambda \\
&= \frac{x_h^{y_n}}{y_h!} \frac{(\frac{\alpha}{\mu})^\alpha}{\Gamma(\alpha)} \int_\lambda \lambda_h^{y_h+\alpha-1} e^{-(x_h + \frac{\alpha}{\mu})\lambda_h} d\lambda \\
&\sim \frac{x_h^{y_n}}{y_h!} \frac{(\frac{\alpha}{\mu})^\alpha}{\Gamma(\alpha)} \times Gamma(\alpha + y_h, \frac{\alpha}{\mu} + x_h) \\
&= \frac{(\frac{\alpha}{\mu})^\alpha}{\Gamma(\alpha)} \frac{\Gamma(\alpha + y_h)}{(\frac{\alpha}{\mu} + x_h)^{\alpha + y_h}} \frac{x_h^{y_h}}{y_h!} \\
&\propto Gamma(\alpha + y_h, \frac{\alpha}{\mu} + x_h)
\end{aligned}$$

Part ii

Please refer to the Code Appendix for the sampling algorithm. Note: I do not believe that I performed the sampling correctly as the results in subsequent steps seem odd and outside the realm of what would be acceptable (I had to deal with some scaling issues as well).

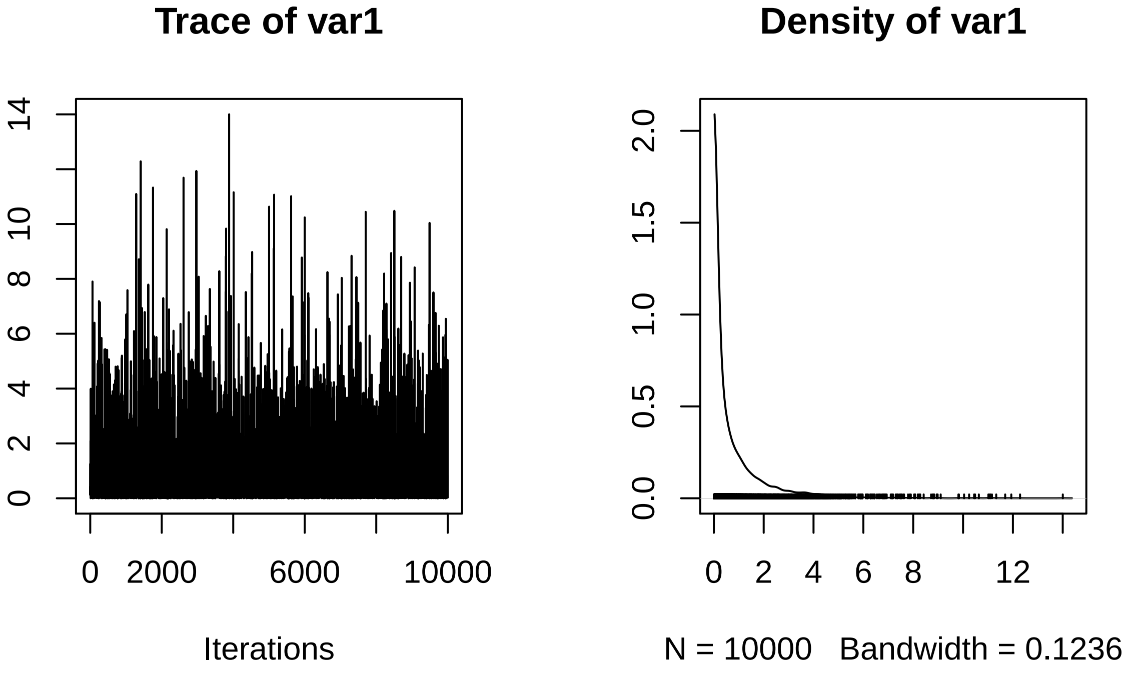
Part iii

Trace Plot - post_mu



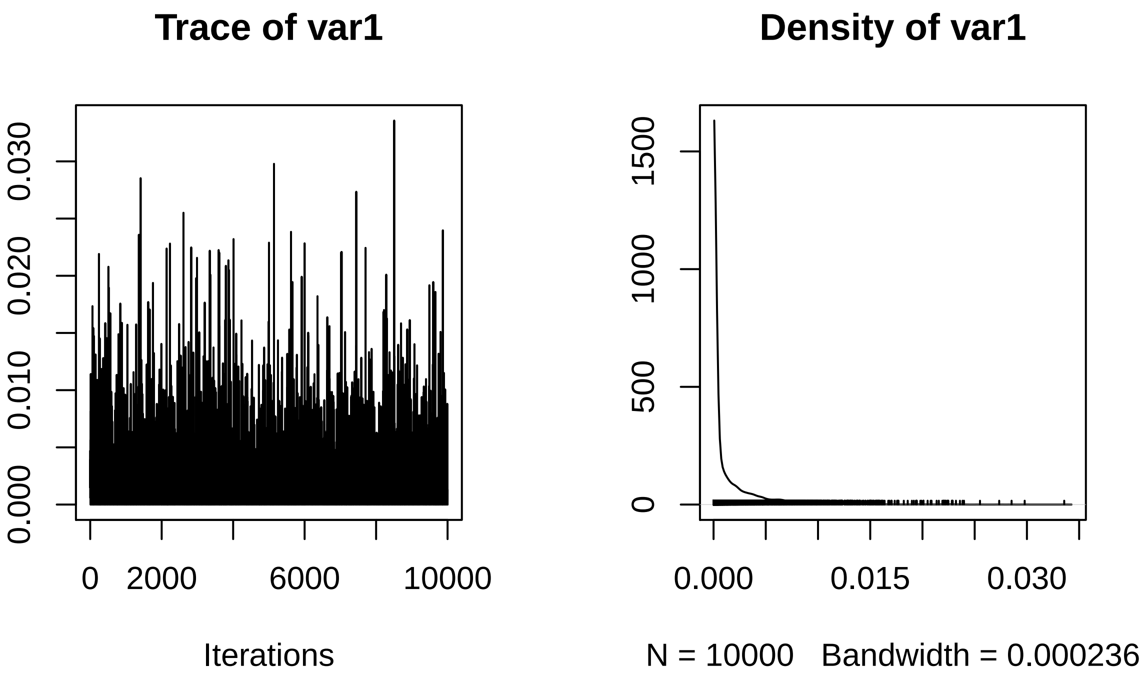
The MCMC appears efficient I suppose in that it may have converged to a state, however, the state does not appear very stable as can be noted by the lack of symmetric oscillation around a particular value and the lack of symmetry on the northern bound. This does not appear to be a very strong result for the MCMC.

Trace Plot - post_alpha



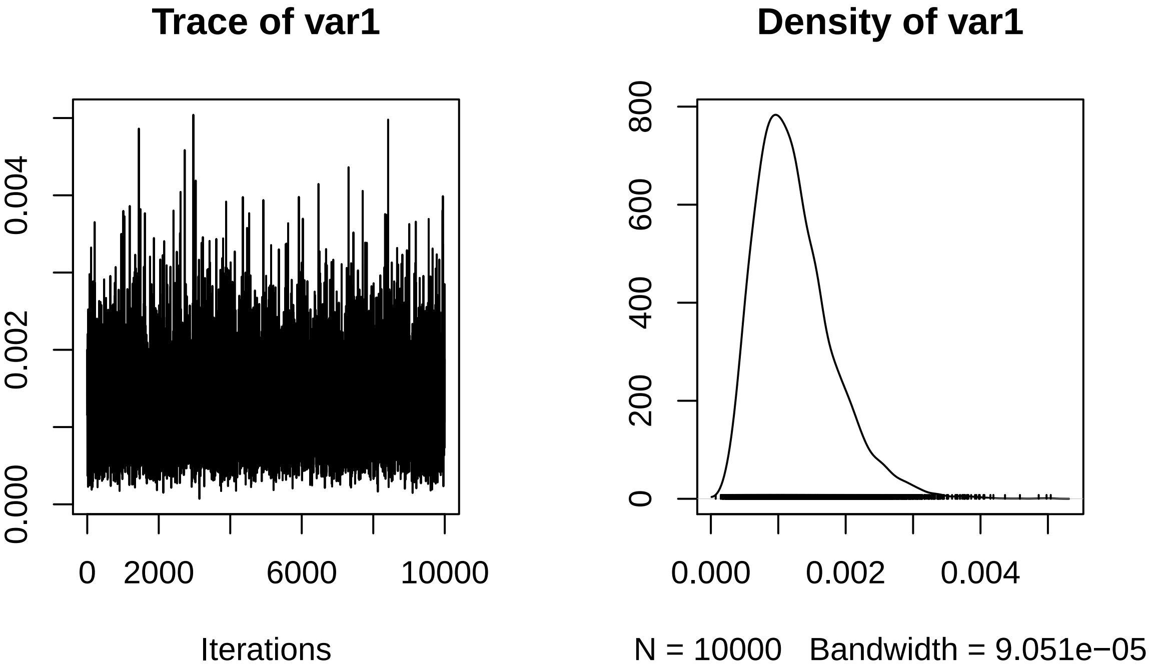
The same issues apply with α_{post} as existed with μ_{post}, the chain appears as though it may have converged, but the issues previously described are exacerbated here; the lack of chain symmetry is still present and the lack of consistency on the northern bound is exceptionally worse than it was for μ_{post}.

Trace Plot - post_lambda, h = 1



The exacerbated issues as with α_{post} apply here again so I will not list them again, however the issues with the northern bounds consistency is in an even worse state.

Trace Plot - post_lambda, h = 80



This is by far the best MCMC trace plot of the four variables in question. We can clearly see there is a central tendency for the chain (this was not present for the previous three chains) and that it has a reasonably tight oscillation around that value. There is still an element of asymmetry but certainly this results is much more positive than the previous three.

Effective Samples Sizes

```
## Effective Size, post_mu:  10000
```

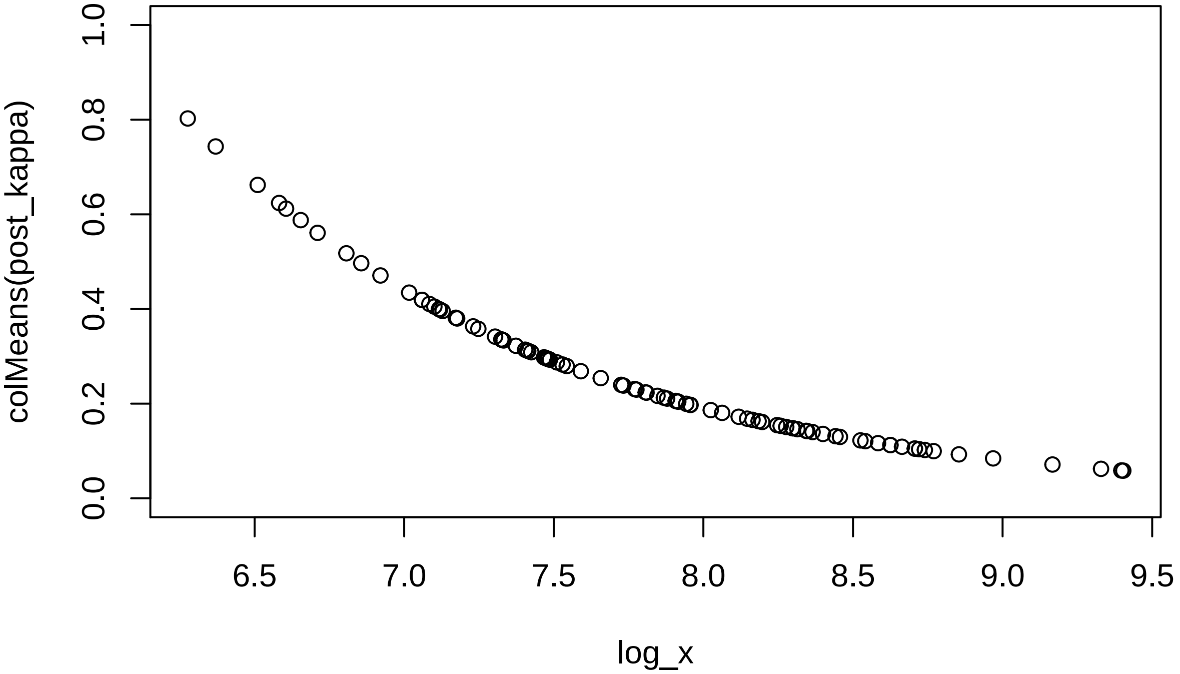
```
## Effective Size, post_alpha:  12879.73
```

```
## Effective Size, post_lambda, h = 1:  12256.09
```

```
## Effective Size, post_lambda, h = 80:  10894.78
```

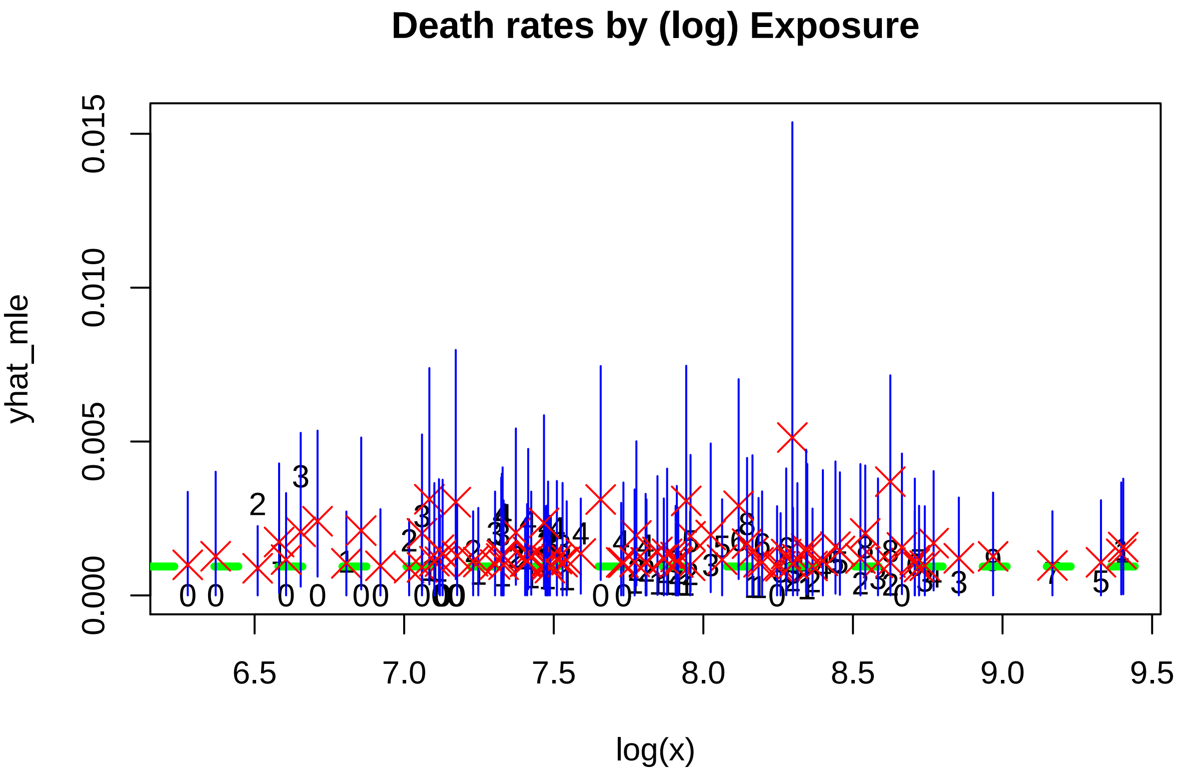
The effective sample sizes are high, which implies efficiency, however, I think this may be the result of a faulty analysis because I was not aware that the effective sample size could be greater than the number of samples present.

Part iv



There appears to be a clear negative relationship here, thus showing us that higher values of the log of the Exposure correspond directly to smaller values of the shrinkage parameter, κ.

Part v



The pooled MLE, as represented by the green dashed line, does a very good job of representing the middle of all of our data points divorced of any type of weighting. The unpooled MLE, as represented by the numbers representing the number of deaths, represent each hospital individually with no shared information. Our red X's, which represent the partial pooling, do a good job of pulling the more extreme unpooled data points closer to the pooled MLE. By taking some information from the hosptials overall, we can pull these unpooled values closer to a true mean.

Some strange results that I got (probably from faulty analysis), is that in some cases an extreme value is pulled passed the MLE (for example the first two 0 points all the way to the left of the plot). Again, probably related to issues with algorithm, unpooled data points are being pulled away from the MLE.

Problem 2

$[\bar{y}_{\cdot j}|\theta_j, \sigma_j] \overset{indep}{\sim} N(\theta_j, \sigma_j^2), \qquad j = 1, ..., J$
$[\theta_j|\mu, \sigma_\theta] \overset{iid}{\sim} N(\mu, \sigma_\theta^2), \qquad j = 1, ..., J$
$p(\mu, \theta) = p(\mu)p(\theta), \qquad p(\mu) \propto 1, \qquad [\sigma_\theta] \sim Uniform(0, A), \qquad A = 100$

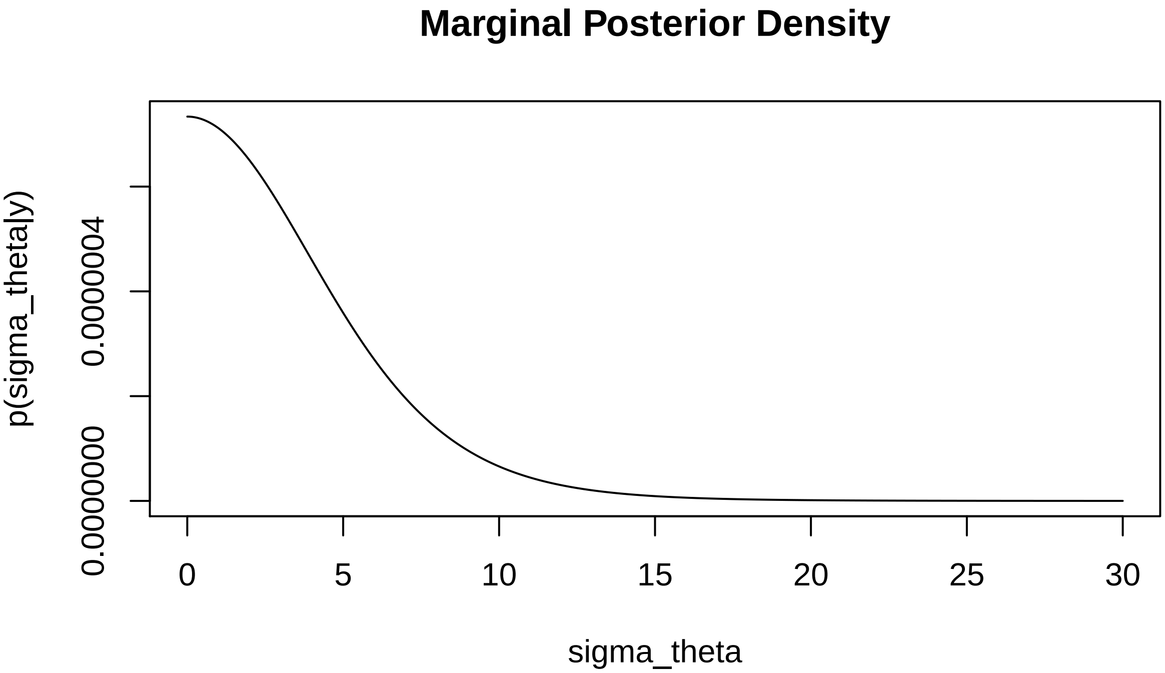
Part i - BDA Exercise 5.3

Part a

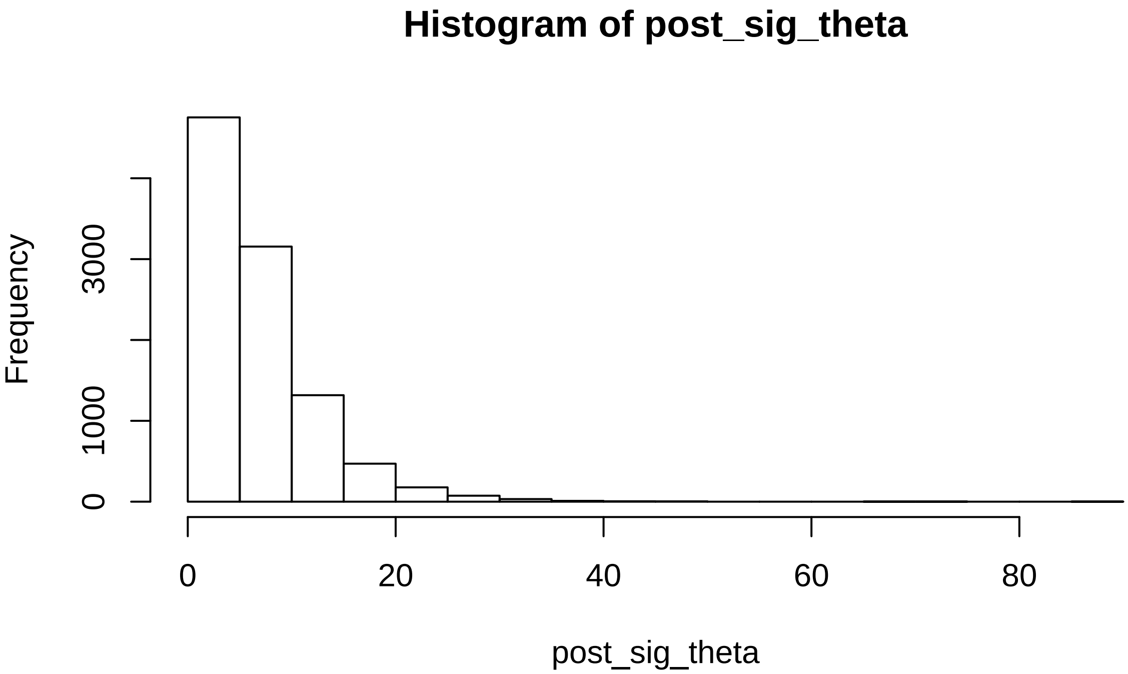
According to Formula 5.21 on BDA page 117, we have the following form for $p(\tau|y)$:
$p(\tau|y) \propto p(\tau)V_\mu^{-1/2}\prod_{j=1}^J(\sigma_j^2 + \tau^2)^{-1/2}exp\bigg(- \frac{(\bar{y}_{\cdot j} - \hat{\mu})^2}{2(\sigma_j^2 + \tau^2)}\bigg)$

The following will allow us to subsitute into the equation, from Formula 5.20 on BDA page 117:

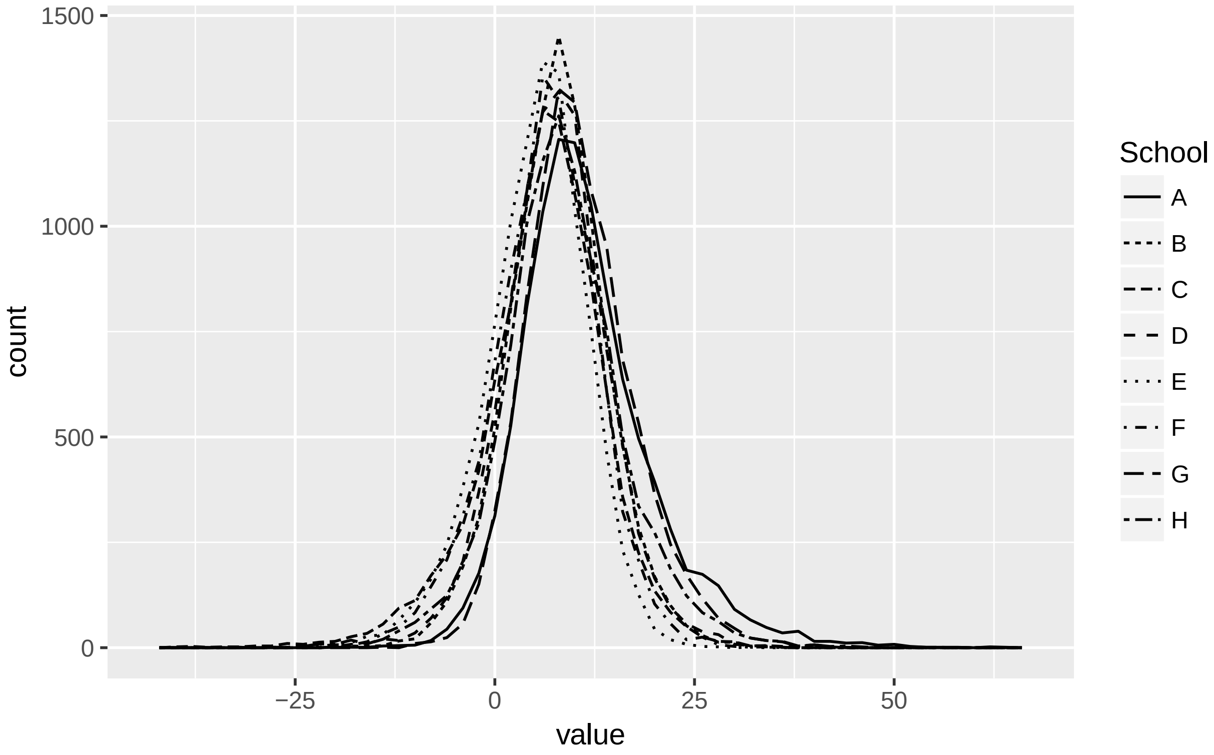
$\hat{\mu} = \frac{\sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2}\bar{y}_{\cdot j}}{\sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2}}$
$V_\mu^{-1} = \sum_{j=1}^J \frac{1}{\sigma_j^2 + \tau^2}$



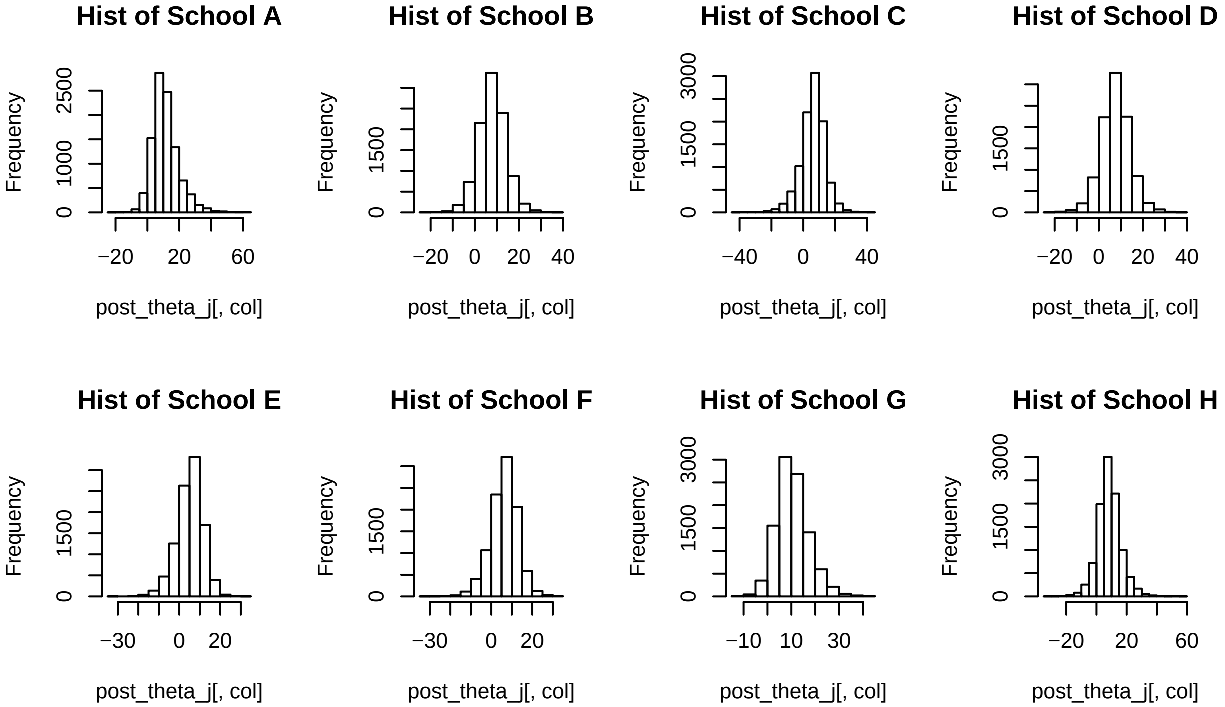
In regards to the plot of the posterior of $p(\sigma_\theta|y)$, we see a large amount of mass centered around 0, this implies that there may be little difference between our θ_j values; we will want to investiage this further.



The histogram is a re-hashing of our previous point; we are seeing a lot of mass centered around 0 ($\sigma_\theta = 0$), which implies that we are inclined to see less of a difference between the schools.



Above we are plotting simultaneous density plots of the densities for each school according to our draws from our posteriors on the same plot. Our inkling that there would be little difference between the schools proves correct; while there are slight differences between the means and the "fatness" of the tails, the densities appear to be very similar indeed.



For completeness, we also plot the individual histograms for each school. We again see very similar results for each school, as they all appear to be mostly Gaussian in shape and centered around a mean value of 10. School A, according to its histogram, appears to be the only case where perhaps the mean is larger than 10; we will want to investigate this further.

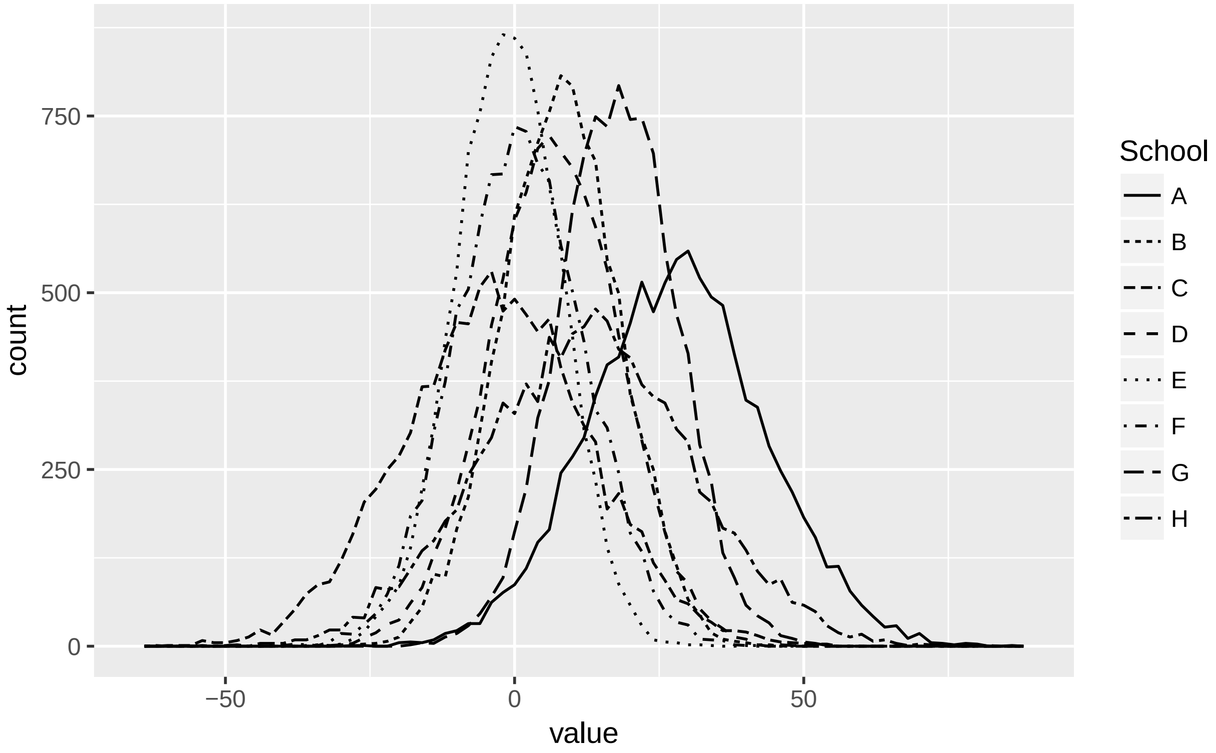
schools	prob_max	A	B	C	D	E	F	G	H
A	0.2559	—	0.6379	0.6895	0.6502	0.7371	0.6935	0.5258	0.6056
B	0.1017	0.3621	—	0.5725	0.5129	0.6315	0.5742	0.3722	0.4704
C	0.082	0.3105	0.4275	—	0.4359	0.5484	0.4998	0.3198	0.4097
D	0.0973	0.3498	0.4871	0.5641	—	0.6165	0.5603	0.3668	0.4666
E	0.0491	0.2629	0.3685	0.4516	0.3835	—	0.4509	0.2654	0.3591
F	0.072	0.3065	0.4258	0.5002	0.4397	0.5491	—	0.3184	0.4075
G	0.2017	0.4742	0.6278	0.6802	0.6332	0.7346	0.6816	—	0.5902
H	0.1403	0.3944	0.5296	0.5903	0.5334	0.6409	0.5925	0.4098	—

Above we find the probability tables defining certain data-related information. The first column represents the school we are considering, while the second column represents the probability that each of these schools has the best coaching program. The rest of the columns compares each school to each other in regards to, probabilistically, whether each has a better coaching program. For example, the first row displays how likely that School A has a better coaching program than any other school.

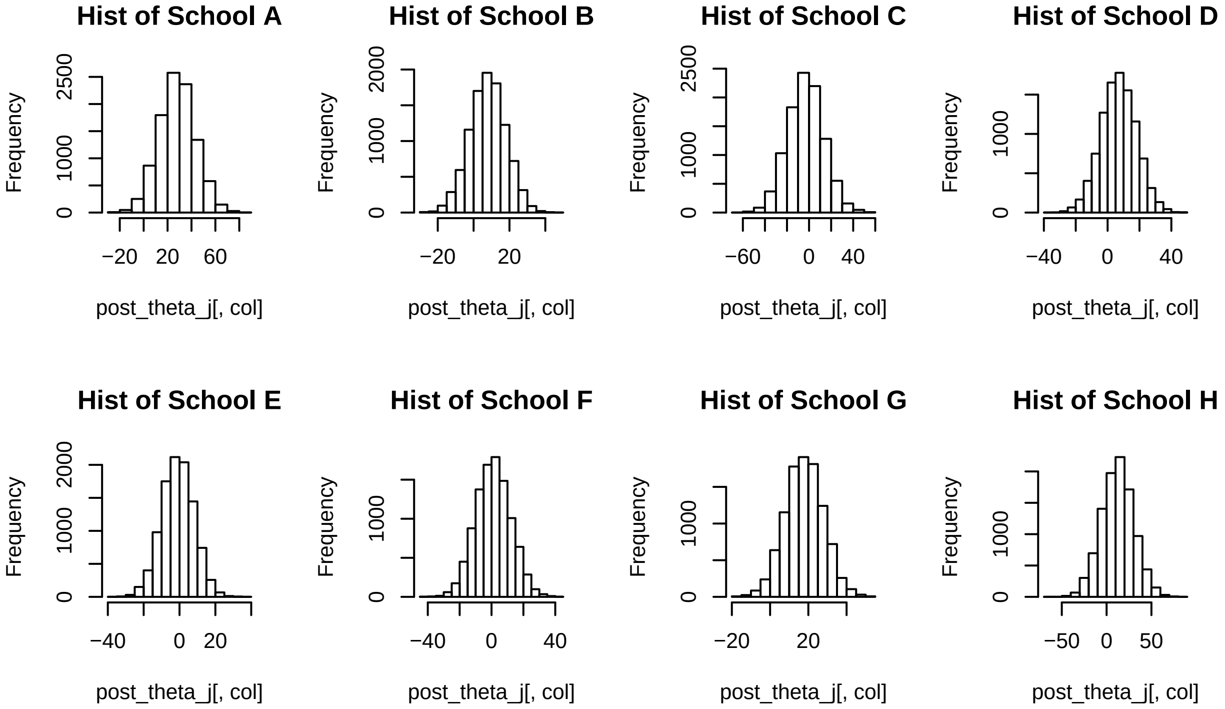
Some conclusions:

- School A does appear to have the best coaching program compared to any other school at ~25%, this gives credence to our previous conclusion
- School E appears to have the weakest coaching program; its chance of having the best program is ~5% and it does not compare favorably to any other school
- School G is our second strongest school at ~20% chance of having the best program; it compares favorably to all other schools aside for School A

Part b



As we can see, when we set our posterior draws of $\sigma_\theta \to \infty$, thus creating a situation where we account for no pooling, we see that the schools now have extremely different posterior distributions, since we are using no shared information; our belief that the schools share no information causes the posterior distributions to diverge. We again see that School A has separated itself as what appears to be the strongest coaching program.



Our results from our previous analysis are confirmed, we can see that the mass associated with posterior draws of θ_j for School A are drawn even further away from the rest of the schools thus separating itself as having the strongest program; its mass appears to sit at about 20-25, much larger than all other schools.

schools	prob_max	A	B	C	D	E	F	G	H
A	0.5462	—	0.8675	0.92	0.8733	0.9499	0.9285	0.7103	0.7485
B	0.0344	0.1325	—	0.7219	0.5273	0.7454	0.6726	0.2409	0.4173
C	0.0277	0.08	0.2781	—	0.3013	0.4537	0.4133	0.1327	0.2608
D	0.0371	0.1267	0.4727	0.6987	—	0.7093	0.6522	0.2307	0.4025
E	0.0025	0.0501	0.2546	0.5463	0.2907	—	0.4471	0.0771	0.2556
F	0.0115	0.0715	0.3274	0.5867	0.3478	0.5529	—	0.1322	0.2942
G	0.171	0.2897	0.7591	0.8673	0.7693	0.9229	0.8678	—	0.6087
H	0.1696	0.2515	0.5827	0.7392	0.5975	0.7444	0.7058	0.3913	—

Our probability table reinforces what we have learned so far, some notable conclusions and confirmations are listed below:

- School A is now clearly the strongest coaching program according to our analysis; it has a >50% chance of being the best program and compares favorably to all other schools
- Schools G and H have established themselves as potential second best schools, they are both sitting at ~17% chance of being the top schools
- The rest of the schools are each <5% and cannot be considered as having the best coaching program

Part c

To summarize what has already been discussed, when we set $\sigma_\theta \to \infty$ we get much more extreme values as there is no pooling. What this means is that we completely ignore the hierarchy of schools and share no information between them. As such, schools that are stronger or weakes will remain so, in contrast to the first example of partial pooling where weaker or stronger schools were pulled towards an overall mean. It is also important to note that the stronger and weaker schools become much more clear when there is no pooling. For example, School A's probability of having the best coaching program doubled during the second analysis, while other schools saw significant drops in their probability of being best.

Part d

If we set $\sigma_\theta \to 0$, then we treat all schools as the same and do complete pooling. This will cause all schools to have equal probability of being the best school since all of the information between them is shared.

Part ii

Setting our α to 0.05, for our partial pooling example we have no significant results and as such cannot reject any of our hypotheses. However, for our no pooling example, we may reject E > A as significant the following as they are below 0.05.

Part iii

Setting our α to 0.05, our Bonferroni-corrected p-value is now $\frac{\alpha}{J} = \frac{0.05}{8} = 0.00625$. At this Bonferroni-corrected p-value, we find no significant results, and as such, cannot reject any of our hypotheses.

Code Appendix

```
########## Workspace Prep ##########



## Load in the necessary packages

library(ggplot2)
suppressMessages(
    suppressWarnings(
        library(truncdist)
    )
)
library(coda)
library(tidyr)
library(knitr)
suppressWarnings(
    library(LearnBayes)
)

## No scientific notation

options(scipen = 999)



########## Problem 1 ##########



## Load in the data

data("hearttransplants")
H = length(hearttransplants$y)
y = hearttransplants$y
x = hearttransplants$e



##### Part i #####



##### Part ii #####



## Set the number of iterations

S <- 10000

## Initialize values
```

```r
mu <- 2
alpha <- 1
z_0 <- 0.5

## Containers for the MCMC output

post_lambda_h <- array(0, c(S, H))
post_mu <- numeric(S)
post_alpha <- numeric(S)
post_kappa <- array(0, c(S, H))

## Run the MCMC

for (s in 1:S) {

    mu <- rgamma(n = 1, shape = y + 1, rate = 1)

    alpha <- rgamma(n = 1, shape = y + 1, rate = 1) * (z_0 / (z_0 + alpha)^2)

    lambda_h <- rgamma(n = H,
                       shape = alpha + y,
                       rate = (alpha / mu) + x)

    post_alpha[s] <- alpha
    post_mu[s] <- mu
    post_lambda_h[s,] <- lambda_h
    post_kappa[s,] = alpha / (alpha + (x * mu))

}

## Scale up post_kappa

post_kappa <- post_kappa * 100



##### Part iii #####



## Construct the trace plots

plot(as.mcmc(post_mu))
plot(as.mcmc(post_alpha))
plot(as.mcmc(post_lambda_h[,1]))
plot(as.mcmc(post_lambda_h[,80]))

## Calculate the effective sample sizes

eff_mu <- as.vector(effectiveSize(post_mu))
eff_alpha <- as.vector(effectiveSize(post_alpha))
eff_lambda_1 <- as.vector(effectiveSize(post_lambda_h[,1]))
eff_lambda_80 <- as.vector(effectiveSize(post_lambda_h[,80]))
```

```
## Output the results

cat("Effective Size, post_mu: ", eff_mu)
cat("Effective Size, post_alpha: ", eff_alpha)
cat("Effective Size, post_lambda, h = 1: ", eff_lambda_1)
cat("Effective Size, post_lambda, h = 80: ", eff_lambda_80)



##### Part iv #####



## Calculate the log-exposure

log_x <- log(x)

## Plot posterior means as a function of log_x

plot(x = log_x, y = colMeans(post_kappa), ylim = c(0,1),
     xlab = 'Log Exposure', ylab = 'Post_Kappa')



##### Part v #####



## Generate the plot from class

yhat_mle = (y / x)
yhat_mle_pool = (sum(y) / sum(x))

ci_h <- array(0, c(H, 3))

for (i in 1:H) {

    ci_h[i,1:2] <- round(as.vector(HPDinterval(as.mcmc(post_lambda_h[i,]), prob = 0.95)[1, 1:2]), 6)
    ci_h[i,3] <- round(mean(post_lambda_h[i,]), 6)

}

plot(log(x), yhat_mle, ylim = range(ci_h, (y /  x)),
     pch = as.character(y), cex = 1,
     main = 'Death rates by (log) Exposure')
abline(h = yhat_mle_pool, lwd=4, col='green', lty=2)

for(h in 1:H) {

    lines(rep(log_x[h], 2),
          ci_h[h,1:2], col='blue', lwd=1)

    lines(log_x[h], ci_h[h,3],
          type='p', pch=4, lwd=1, cex = 2, col='red')

}



########## Problem 2 ##########



## Input the known data and parameters

y_j <- c(28, 8, -3, 7, -1, 1, 18, 12)
sigma_j <- c(15, 10, 16, 11, 9, 11, 10, 18)
sigma_j_sq <- sigma_j^2
names(y_j) <- c("A", "B", "C", "D", "E","F", "G", "H")
names(sigma_j) <- c("A", "B", "C", "D", "E","F", "G", "H")

J <- length(y_j)



##### Part i #####



### A = 100 ###



## Set some intial values

theta_j <- y_j
mu <- mean(theta_j)
sigma_theta <- sd(y_j)

A <- 100
S <- 10000

## Create the containers for the posterior draws

post_theta_j <- array(0, c(S, J))
post_mu <- numeric(S)
post_sig_theta <- numeric(S)
post_kappa_j <- array(0, c(S,J))

## Run the MCMC

for(s in 1:S) {

    ## Sample from mu

    Q_mu <- sum(1/(sigma_j^2 + sigma_theta^2))
    ell_mu <- sum(y_j/(sigma_j^2 + sigma_theta^2))
    mu <- rnorm(n = 1,
                mean = Q_mu^-1*ell_mu,
                sd = sqrt(Q_mu^-1))

    ## Sample from theta_j

    Q_theta <- 1/sigma_j^2 + 1/sigma_theta^2
    ell_theta <- y_j/sigma_j^2 + mu/sigma_theta^2
    theta_j <- rnorm(n = J,
                     mean = Q_theta^-1*ell_theta,
                     sd = sqrt(Q_theta^-1))

    ## Sample from sigma_theta

    eta_theta <- rtrunc(n = 1,
                        'gamma',
                        a = 1/A^2,
                        b = A,
                        shape = J/2 - 1/2,
                        rate =  sum((theta_j-mu)^2)/2)
    sigma_theta <- 1/sqrt(eta_theta)

    post_mu[s] <- mu
    post_theta_j[s,] <- theta_j
    post_sig_theta[s] <- sigma_theta

    # Get the shrinkage parameter

    post_kappa_j[s,] <- sigma_j^2/(sigma_theta^2 + sigma_j^2)

}

## Function for V inverse

v_inv <- function(sigma_j_sq, sigma_theta) {

    sum(1 / (sigma_j_sq + sigma_theta^2))

}

## Function for mu hat

mu_hat <- function(y_j, sigma_j_sq, sigma_theta) {

    sum(1 / (sigma_j_sq + sigma_theta^2) * y_j) / v_inv(sigma_j_sq, sigma_theta)

}

## Function for post sigma_theta

post_sigma_theta <- function(y_j, sigma_j_sq, sigma_theta, mu_h) {
    v_inv(sigma_j_sq, sigma_theta)^(-(1 / 2)) * prod((sigma_j_sq + sigma_theta^2)^(-(1 / 2) * exp(-(y_j

}

## Set a sequence for sigma_theta

seq_sigma_theta <- seq(from = 0, to = 30, length.out = S)

## Set a container for plotting the posterior of sigma_theta

post_sigma_theta_plot <- numeric()

## Loop to sample from post sigma_theta and plot the results

for (i in 1:length(seq_sigma_theta)) {

    mu_h <- mu_hat(y_j, sigma_j_sq, seq_sigma_theta[i])
    post_sigma_theta_plot[i] <- post_sigma_theta(y_j, sigma_j_sq, seq_sigma_theta[i], mu_h)

}

plot(x = seq_sigma_theta, y = post_sigma_theta_plot, type = 'l',
     xlab = 'sigma_theta', ylab = 'p(sigma_theta|y)', main = 'Marginal Posterior Density')

## Plot the posterior draws of sigma_theta

hist(post_sig_theta)

## Plot the posterior draws of mu

labels = c("A", "B", "C", "D", "E", "F", "G", "H")

ggplot(gather(data.frame(post_theta_j)), aes(value, linetype=key)) + geom_freqpoly(binwidth=2) +
    scale_linetype_discrete(name = "School", labels = labels)

par(mfrow = c(2, 4))

for (col in 1:ncol(post_theta_j)) {

    hist(post_theta_j[,col], main = paste("Hist of School" , labels[col]))

}

par(mfrow = c(1, 1))

## Calculate the probability table

max_vals <- apply(post_theta_j, 1, which.max)
prob_max <- table(max_vals) / S

prob_table <- matrix(NA, J, J)

for (i in 1:(J-1)) {
    for (j in (i+1):J) {

        prob_table[i,j] = sum(post_theta_j[,i] > post_theta_j[,j]) / S
        prob_table[j,i] = 1 - prob_table[i,j]

    }

}

prob_table[is.na(prob_table)] = "---"
schools = c("A", "B", "C", "D", "E", "F", "G", "H")
colnames(prob_table) <- schools
kable(cbind(schools, prob_max, prob_table))



### sigma_theta = Infinity ###



## Set some intial values

theta_j <- y_j
mu <- mean(theta_j)
sigma_theta <- sd(y_j)

S <- 10000

## Create the containers for the posterior draws

post_theta_j <- array(0, c(S, J))
post_mu <- numeric(S)
post_sig_theta <- numeric(S)
post_kappa_j <- array(0, c(S,J))

## Run the MCMC

for(s in 1:S) {

    ## Sample from mu

    Q_mu <- sum(1/(sigma_j^2 + sigma_theta^2))
    ell_mu <- sum(y_j/(sigma_j^2 + sigma_theta^2))
    mu <- rnorm(n = 1,
                mean = Q_mu^-1*ell_mu,
                sd = sqrt(Q_mu^-1))

    ## Sample from theta_j

    Q_theta <- 1/sigma_j^2 + 1/sigma_theta^2
    ell_theta <- y_j/sigma_j^2 + mu/sigma_theta^2
    theta_j <- rnorm(n = J,
                     mean = Q_theta^-1*ell_theta,
                     sd = sqrt(Q_theta^-1))

    ## Sample from sigma_theta

    # eta_theta <- rtrunc(n = 1,
    #                      'gamma',
    #                      a = 1/A^2,
    #                      b = A,
    #                      shape = J/2 - 1/2,
    #                      rate =  sum((theta_j-mu)^2)/2)
    # sigma_theta <- 1/sqrt(eta_theta)

    sigma_theta <- 99999

    post_mu[s] <- mu
    post_theta_j[s,] <- theta_j
    post_sig_theta[s] <- sigma_theta

    # Get the shrinkage parameter

    post_kappa_j[s,] <- sigma_j^2/(sigma_theta^2 + sigma_j^2)

}

## Function for V inverse

v_inv <- function(sigma_j_sq, sigma_theta) {

    sum(1 / (sigma_j_sq + sigma_theta^2))

}

## Function for mu hat

mu_hat <- function(y_j, sigma_j_sq, sigma_theta) {

    sum(1 / (sigma_j_sq + sigma_theta^2) * y_j) / v_inv(sigma_j_sq, sigma_theta)

}

## Function for post sigma_theta

post_sigma_theta <- function(y_j, sigma_j_sq, sigma_theta, mu_h) {

    v_inv(sigma_j_sq, sigma_theta)^(-(1 / 2)) * prod((sigma_j_sq + sigma_theta^2)^(-(1 / 2) * exp(-(y_j

}

## Plot the posterior draws of mu

labels = c("A", "B", "C", "D", "E", "F", "G", "H")

ggplot(gather(data.frame(post_theta_j)), aes(value, linetype=key)) + geom_freqpoly(binwidth=2) +
    scale_linetype_discrete(name = "School", labels = labels)

par(mfrow = c(2, 4))

for (col in 1:ncol(post_theta_j)) {

    hist(post_theta_j[,col], main = paste("Hist of School" , labels[col]))

}

par(mfrow = c(1, 1))

## Calculate the probability table

max_vals <- apply(post_theta_j, 1, which.max)
prob_max <- table(max_vals) / S

prob_table <- matrix(NA, J, J)

for (i in 1:(J-1)) {

    for (j in (i+1):J) {

        prob_table[i,j] = sum(post_theta_j[,i] > post_theta_j[,j]) / S
        prob_table[j,i] = 1 - prob_table[i,j]

    }

}

prob_table[is.na(prob_table)] = "---"
schools = c("A", "B", "C", "D", "E", "F", "G", "H")
colnames(prob_table) <- schools
kable(cbind(schools, prob_max, prob_table))



##### Part ii #####



##### Part iii #####
```