STATS551-HW2

Ningyuan Wang

2/15/2020

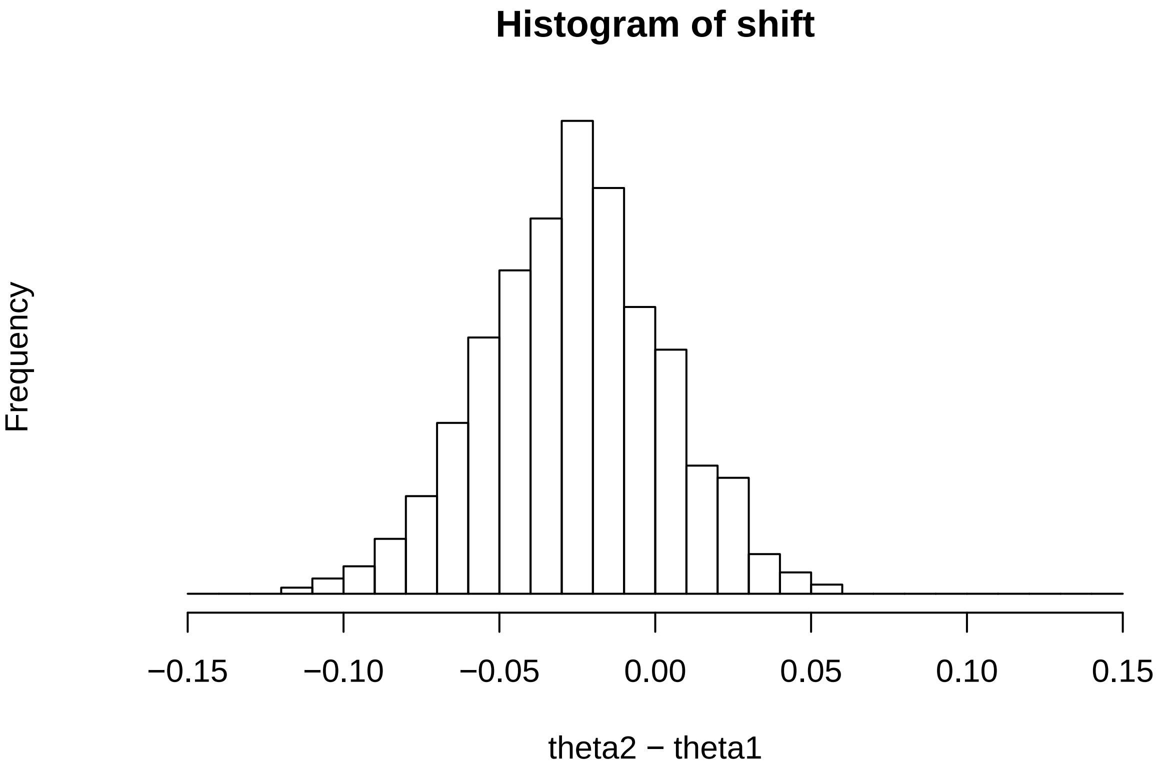
Dirichlet-Multinomial Model

For this problem, we use a non-informative uniform distribution by setting $\alpha_j = 1$ for all j. The resulting posterior distribution is proper if there is at least one observation in each of categories (refered to textbok p.69).

Therefore, with noninformative uniform prior distribution on θ, $\alpha_1 = \alpha_2 = \alpha_3$, the posterior distribution for pre-debate is Dirichlet(295, 308, 39). Similarly, the posterior distribution for post-debate is Dirichlet(289, 333, 20). With previous part, we already known the tramsformation between Dirichlet and Beta. Define u = $\theta_1/(\theta_1 + \theta_2)$, then posterior for pre-debate is Beta(295, 308), and posterior for post-debate is Beta(289, 333). Now, we estimate θ_2 - θ_1 and decide the bias of voting in two debates.

The posterior probability indicates that the probability of a shift toward Bush is 18.3%.

```r
set.seed(551)

# draw 1000 samples for pre and post debates, respectively
n_sample = 1000
theta1 = rbeta(n_sample, 295, 308)
theta2 = rbeta(n_sample, 289, 333)
shift = theta2 - theta1

hist (shift, xlab="theta2 - theta1", yaxt="n",
breaks=seq(-0.15,0.15,.01), cex=2)
```



```
mean(shift>0)
```

```
## [1] 0.183
```

Analysis of Proportions

Set Up

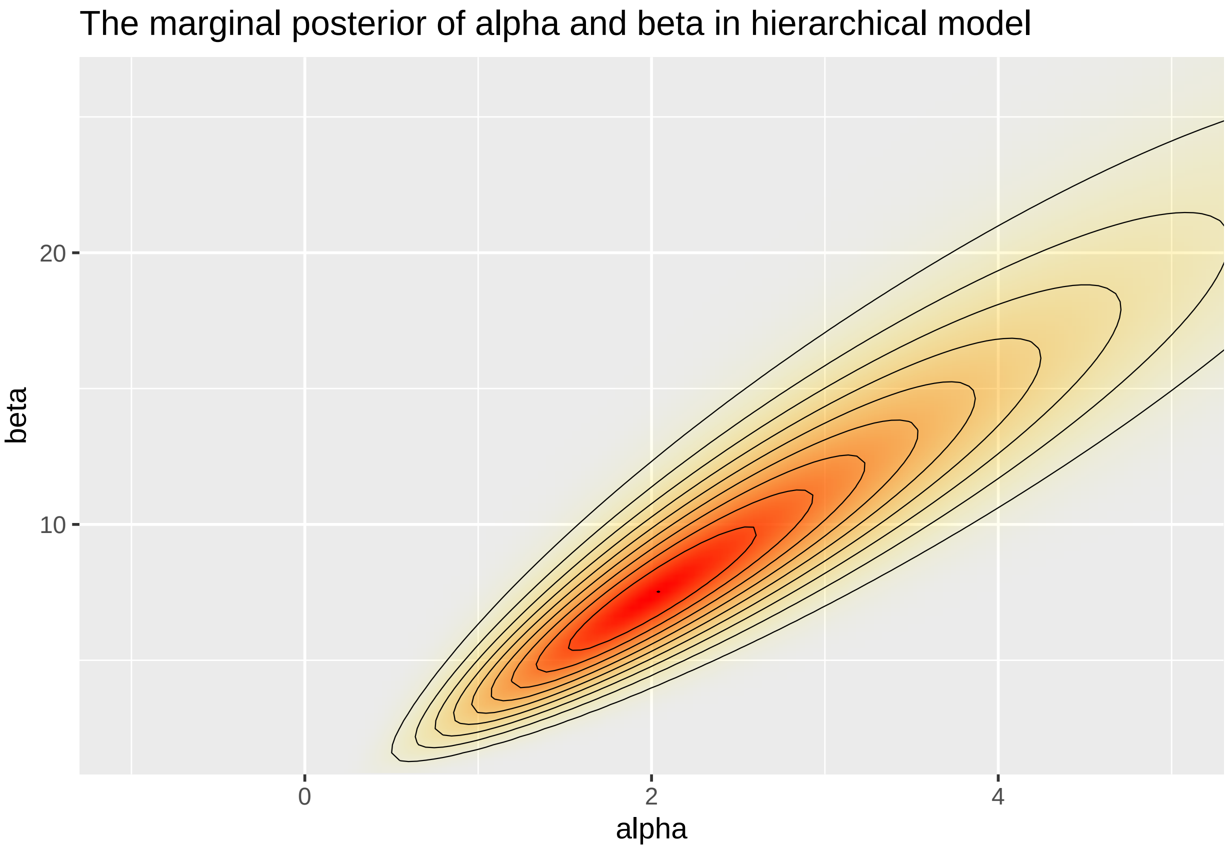
```
# load the library
library(ggplot2)
library(gridExtra)
library(tidyr)

# set up data
y = c(16, 9, 10, 13, 19, 20, 18, 17, 35, 55)
others = c(58, 90, 48, 57, 103, 57, 86, 112, 273, 64)
n = y + others
p = y/n
```

1.The joint posterior distribution was set up in analytical method on the attached paper.

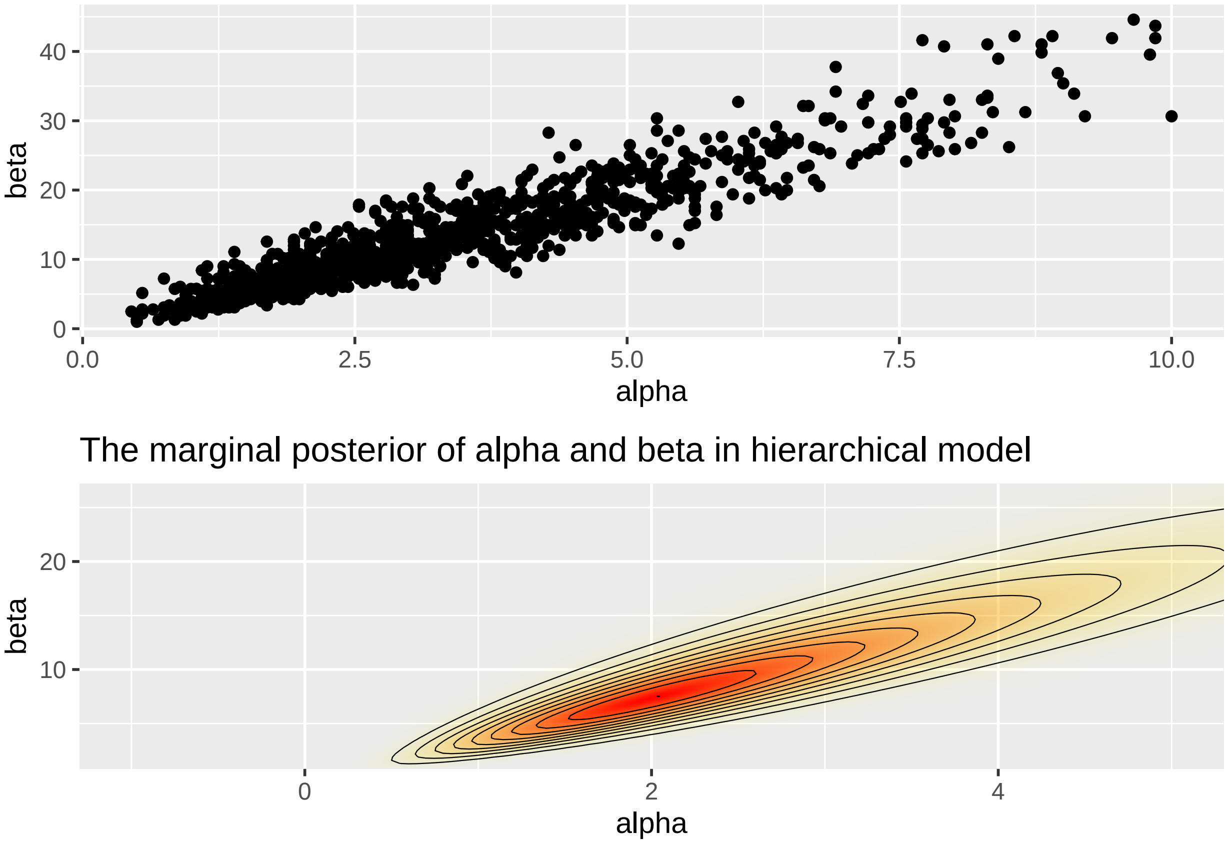
Hierarchicial Model (Beta - Binomial)

2. The marginal posterior density was calculated on the attached paper. The 1000 simulations from the joint psterior ditribuiton was shown below, which was a matrix with dimensions 1000 * 10.

```
#2
A <- seq(0.1, 10, length.out = 200) ## alpha
B <- seq(1, 60, length.out = 200) ## beta do we really need change???

# make vectors that contain all pairwise combinations of A and B
cA <- rep(A, each = length(B))
cB <- rep(B, length(A))

# Use logarithms for numerical accuracy
lpfun <- function(a, b, y, n)
  log(a+b)*(-5/2) +
  sum(lgamma(a+b)-lgamma(a)-lgamma(b)+lgamma(a+y)+lgamma(b+n-y)-lgamma(a+b+n))

lp <- mapply(lpfun, cA, cB, MoreArgs = list(y, n))
df_marg <- data.frame(x = cA, y = cB, p = exp(lp - max(lp)))

# create a plot of the marginal posterior density
title1 <- 'The marginal posterior of alpha and beta in hierarchical model'
postdensityalphabeta = ggplot(data = df_marg, aes(x = x, y = y)) +
  geom_raster(aes(fill = p, alpha = p), interpolate = T) +
  geom_contour(aes(z = p), colour = 'black', size = 0.2) +
  coord_cartesian(xlim = c(-1,5), ylim = c(2, 26)) +
  labs(x = 'alpha', y = 'beta', title = title1) +
  scale_fill_gradient(low = 'yellow', high = 'red', guide = F) +
  scale_alpha(range = c(0, 1), guide = F)
postdensityalphabeta
```



```
# draw samples
# df_marg: first column (value of alpha)
#           second column (value of beta)
#           third column (value of p(alpha, beta | data))
# sample from the grid (with replacement)
nsamp <- 1000
samp_indices <- sample(length(df_marg$p), size = nsamp,
                       replace = T, prob = df_marg$p/sum(df_marg$p))

samp_A <- cA[samp_indices[1:nsamp]]
samp_B <- cB[samp_indices[1:nsamp]]

# visualization samples
samplesalphabeta = data.frame(alpha = samp_A , beta = samp_B)
scatteralphabeta = ggplot(samplesalphabeta, aes(x=alpha, y=beta)) + geom_point()
grid.arrange(scatteralphabeta, postdensityalphabeta)
```



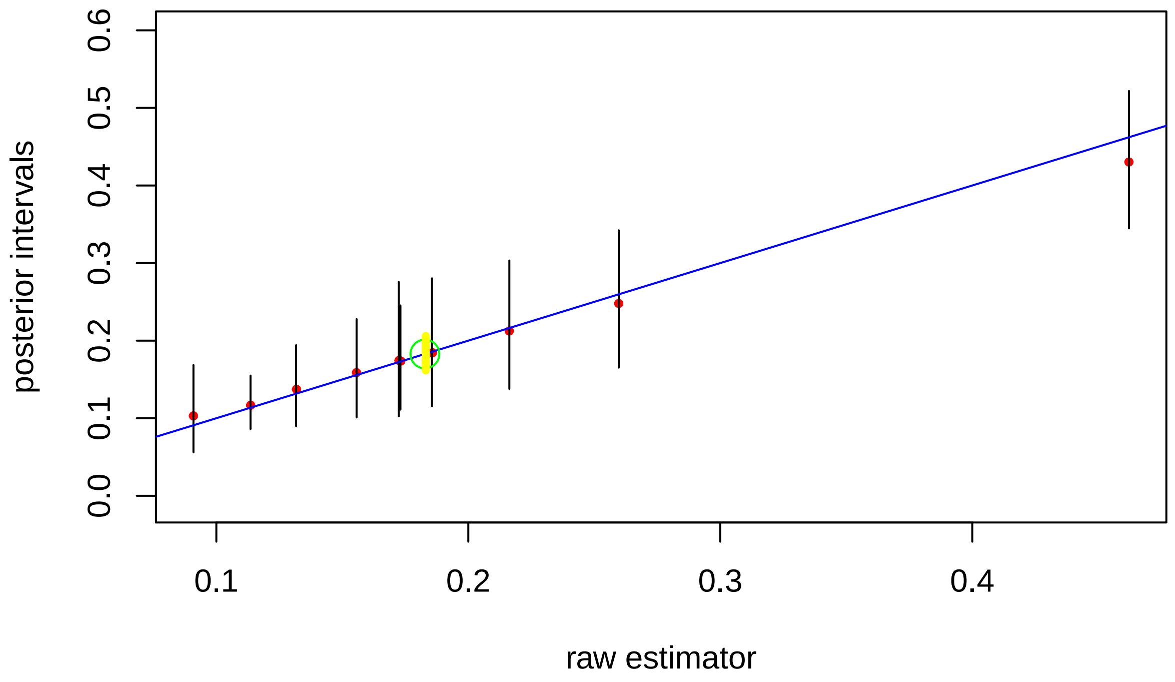
The marginal posterior of alpha and beta in hierarchical model

```
# make inference on theta's
samplestheta <- matrix(0, nsamp, length(y))
for(j in 1:length(y)){
  samplestheta[, j] = sapply(1:nsamp, function(k) rbeta(1, samp_A[k]+y[j], samp_B[k]+n[j]-y[j]))
} # dim = 1000 * 10

head(samplestheta, n = 5) # first 5 simulations
```

```
##             [,1]        [,2]        [,3]        [,4]        [,5]        [,6]
## [1,] 0.1975470 0.10985209 0.1826322 0.2081992 0.1506521 0.2216056
## [2,] 0.2205803 0.07058628 0.1537948 0.2254539 0.1846958 0.1594387
## [3,] 0.1973844 0.09326759 0.1888039 0.1687302 0.1228351 0.2018365
## [4,] 0.2246018 0.14854761 0.1976459 0.1700310 0.1317790 0.2085528
## [5,] 0.1972112 0.09537847 0.1633100 0.1259669 0.1520073 0.2274114
##            [,7]       [,8]       [,9]     [,10]
## [1,] 0.09899149 0.17788599 0.09328475 0.3691420
## [2,] 0.11600106 0.09327475 0.10601597 0.4966139
## [3,] 0.18778129 0.11373388 0.13578341 0.4124886
## [4,] 0.15001311 0.09304175 0.10697626 0.4565981
## [5,] 0.15531609 0.15128905 0.13263552 0.4576284
```

3. In the plot below, the red points indicate raw estimators of proportion of bikes in 10 city blocks, and the vertical bars indicate 95% posterior interval for each block. Since the estimators followed the line very well, and the posterior intervals caught the estimators well, we say the posterior is good estimator of raw proportions.

```
postintervals_sample <- apply(samplestheta, 2, function(x) c(median(x), c(quantile(x, c(0.025, 0.975)))

# compare raw and prediction
rawest = jitter(p)
plot(jitter(p), postintervals_sample[1, ], pch = 19, col = 'red', cex = 0.5, ylim = c(-0.01, 0.6), ylab

for(k in 1:length(y)){
  lines(cbind(rep(rawest[k], 2), postintervals_sample[2:3, k]))
}
lines(seq(0, 0.6, by = 0.01), seq(0, 0.6, by = 0.01), col = 'blue')
phatpool = sum(y)/sum(n)
points(phatpool, phatpool, cex = 2, col = 'green')
lines(cbind(rep(qbeta(0.5, sum(y)+1, sum(n)-sum(y)+1), 2), c(qbeta(0.025, sum(y)+1, sum(n)-sum(y)+1), q
```



```
# how to read the plot??? blue line, green circle and yellow
```

4. 95% posterior interval for the average underlying porportionn of traffic is between 0.172 and 0.222.

```
sample_mean = rowMeans(samplestheta)
quantile(sample_mean, c(0.025, 0.975))
```

```
##      2.5%     97.5% 
## 0.1720071 0.2218828
```

5. The 95% posterior interval of those bicycles out of 100 vehicles is (3, 50). Since the interval is pretty large, the interval is trustful in application, but it may not offer a lot of information.

```
#
n_new = 100
theta_new = c()
y_new = c()
for(j in 1:nsamp){
  theta_new[j] = rbeta(1,samp_A[j],samp_B[j]) # theta ~ beta
  y_new[j] = rbinom(1,n_new,theta_new[j]) # y ~binomial
}
quantile(y_new,c(.025,.975)) #( 3, 50 )
```

```
##  2.5% 97.5% 
##     3    50
```

6.

I think the beta prior is a fair choice, since beta-binomial is a conjugate model and it is a natural choice for easier using and computing. Also, beta prior is always considered as psuedoaccounts, which makes sense in the problem context.