STA 602 Lab 2

Yicheng Shen

12 September, 2022

Exercise 1

```
tumors <- read.csv(file = url("http://www.stat.columbia.edu/~gelman/book/data/rats.asc"),
                   skip = 2, header = T, sep = " ")[,c(1,2)]
y <- tumors$y
N <- tumors$N
n <- length(y)
```

```
plot(seq(0, 1, length.out = 1000),
     dbeta(seq(0, 1, length.out = 1000), 1, 1), type = 'l',
     xlab = expression(theta), ylab = "Density",
     main = "The Beta(1, 1) density")
```

```
stan_dat <- list(n = n, N = N, y =y, a = 1, b = 1)
fit_pool <- stan('lab-02-pool.stan',data = stan_dat,chains = 2,refresh = 0)
```

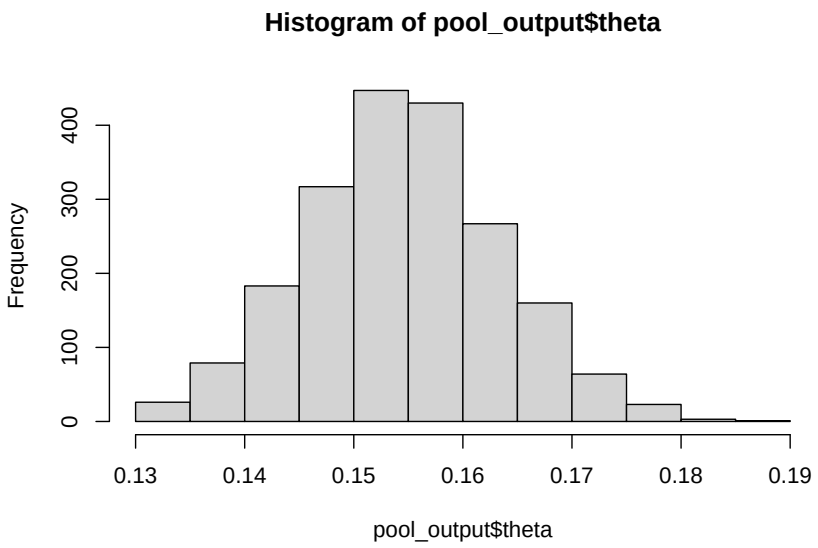
```
## Trying to compile a simple C file
```

```
pool_output <- rstan::extract(fit_pool)
```

```
mean(pool_output$theta)
```

```
## [1] 0.1545191
```

```
hist(pool_output$theta)
```



The distribution of θ seems approximately normal and centers around the mean of 0.155.

Exercise 2

```
stan_dat <- list(n = n, N = N, y =y, a = 1, b = 1)
fit_nopool <- stan('lab-02-nopool.stan',data = stan_dat,chains = 2,refresh = 0)
```

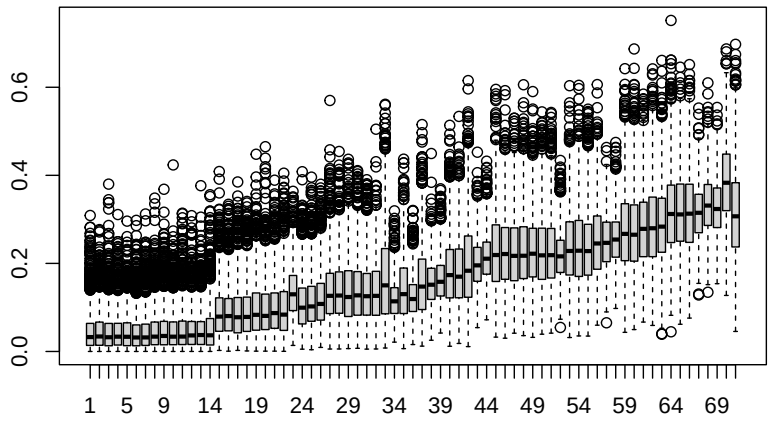
```
## Trying to compile a simple C file
```

```
nopool_output <- rstan::extract(fit_nopool)
```

```
apply(nopool_output$theta, 2, mean)
```

```
##  [1] 0.04579826 0.04715404 0.04596269 0.04516189 0.04532554 0.04422680
##  [7] 0.04513279 0.04784227 0.04895462 0.04721587 0.04860876 0.04897792
## [13] 0.04892574 0.05252159 0.08996093 0.09048885 0.09014098 0.09094530
## [19] 0.09709948 0.09471619 0.09900744 0.09897390 0.13688813 0.10989081
## [25] 0.11351762 0.11925748 0.13738411 0.13882967 0.13662708 0.13726870
## [31] 0.13586695 0.13743237 0.16874974 0.11805714 0.14325795 0.12422127
## [37] 0.15949923 0.15608609 0.16351185 0.18273913 0.18123662 0.19881137
## [43] 0.19995903 0.21387065 0.22883300 0.22803254 0.22579150 0.22800440
## [49] 0.22812856 0.22625953 0.22705701 0.21841529 0.23821747 0.23849031
## [55] 0.23650854 0.25101042 0.25034637 0.25532303 0.27326701 0.27322986
## [61] 0.28106067 0.28706659 0.28903718 0.31787549 0.31748145 0.31714851
## [67] 0.31542333 0.33389483 0.32686804 0.38586191 0.31362283
```

```
boxplot(nopool_output$theta)
```



We have 71 groups, therefore for each group we have a θ_i. The visualization above shows posterior distribution of each of the 71 θ_i. Each point is a point from the posterior density of θ_i.

Exercise 3

pool stan code:

```
parameters {
  real<lower=0, upper=1> theta;  // chance of success (pooled)
}
```

nopool stan code:

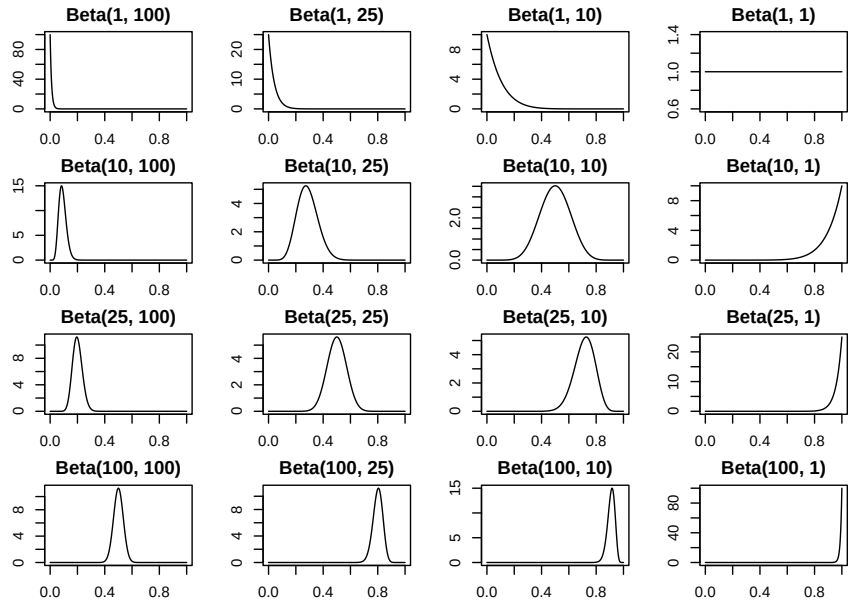
```
parameters {
  vector<lower=0, upper=1>[n] theta;  // chance of success (unpooled)
}
```

The pool model uses `real` while the no pool code uses `vector` and `[n]`. The difference affects whether we view the groups behave independently (each group has varying parameter) or they share the same parameter.

Exercise 4

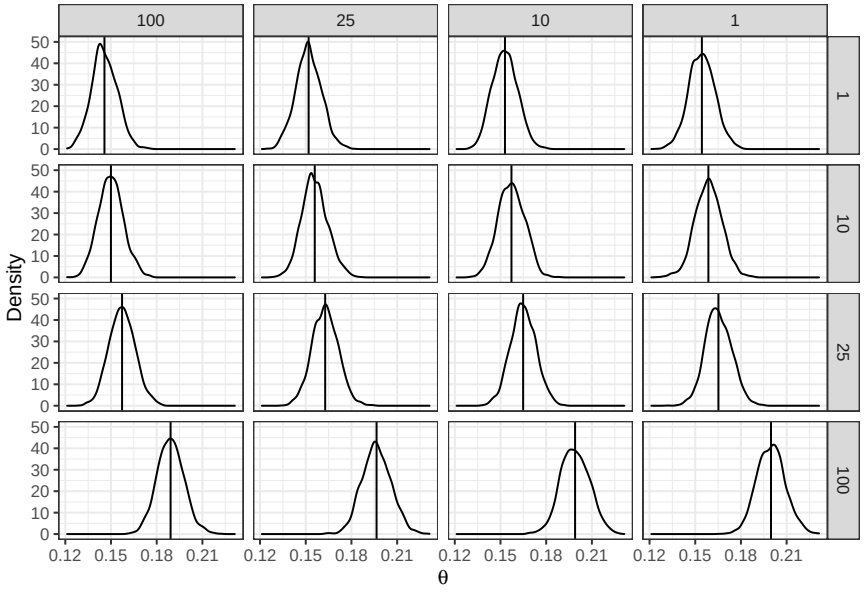
```
par(mfrow = c(4, 4))
par(mar=c(2,2,2,2))
for(a_val in c(1, 10, 25, 100)){
  for(b_val in rev(c(1, 10, 25, 100))){
    plot(seq(0, 1, length.out = 1000),
     dbeta(seq(0, 1, length.out = 1000), a_val, b_val),
     type = 'l',
     xlab = expression(theta), ylab = "Density",
```

```
    main = paste0("Beta(", a_val, ", ", b_val, ")"))
  }
}
```




```
output_list <- list()
for(a_val in c(1, 10, 25, 100)){
  for(b_val in c(1, 10, 25, 100)){
    stan_dat <- list(n = n, N = N, y = y, a = a_val, b = b_val)
    fit_pool <- stan('lab-02-pool.stan', data = stan_dat, chains = 2, refresh = 0)
    output_list[[paste0("a_", a_val, ":b_", b_val)]] <- rstan::extract(fit_pool)[["theta"]]
  }
}
```

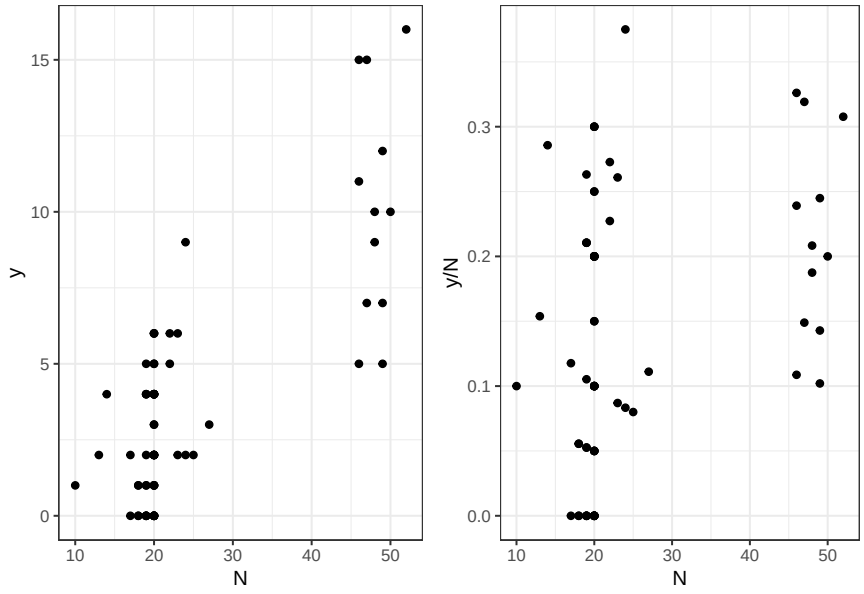
```
output_list %>%
  plyr::ldply(function(theta){
    reshape2::melt(theta) %>%
      dplyr::mutate(post_mean = mean(theta))
  }, .id = "prior") %>%
  tidyr::separate("prior", into = c("a", "b"), sep = ":") %>%
  dplyr::mutate(a = as.numeric(gsub("._", "", a)),
                b = as.numeric(gsub("._", "", b))) %>%
  ggplot2::ggplot() +
  geom_density(aes(x = value)) +
  geom_vline(aes(xintercept = post_mean)) +
  facet_grid(a~factor(b, levels = rev(c(1, 10, 25, 100)))) +
  scale_colour_brewer(palette = "Set1") +
  labs(x = expression(theta), y = "Density")
```



a represents the number of successes in prior observations, b represents the number of failures in the prior observations.

Exercise 5

```
a <- ggplot(tumors) + geom_point(aes(x = N, y = y))
b <- ggplot(tumors) + geom_point(aes(x = N, y = y / N))
grid.arrange(a, b, nrow = 1)
```



```
tumors %>% summarise(proportion = mean(y/N))
```

```
##   proportion
## 1  0.1381151
```

In the `tumors` data, we observe a relatively low proportion of successes compared with failures. The overall average $\frac{y}{N}$ is 0.1381151.

Exercise 6

Most prior beliefs (represented by a and b) are not very close to the data, but a Beta(1,10) or Beta(10,100) prior may be a close match to the data.

Exercise 7

```
# approach 1
mle.1 <- sum(y)/sum(N)
mle.1
```

```
## [1] 0.1535365
```

```
mean(pool_output$theta)
```

```
## [1] 0.1545191
```

```
# approach 2
mle.2 <- y/N
mle.2
```

```
##  [1] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##  [7] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [13] 0.00000000 0.00000000 0.05000000 0.05000000 0.05000000 0.05000000
## [19] 0.05263158 0.05263158 0.05555556 0.05555556 0.11111111 0.08000000
## [25] 0.08333333 0.08695652 0.10000000 0.10000000 0.10000000 0.10000000
## [31] 0.10000000 0.10000000 0.10000000 0.10204082 0.10526316 0.10869565
## [37] 0.11764706 0.14285714 0.14893617 0.15000000 0.15000000 0.15384615
## [43] 0.18750000 0.20000000 0.20000000 0.20000000 0.20000000 0.20000000
## [49] 0.20000000 0.20000000 0.20000000 0.20833333 0.21052632 0.21052632
## [55] 0.21052632 0.22727273 0.23913043 0.24489796 0.25000000 0.25000000
## [61] 0.26086957 0.26315789 0.27272727 0.30000000 0.30000000 0.30000000
## [67] 0.30769231 0.32608696 0.31914894 0.37500000 0.28571429
```

```
apply(nopool_output$theta, 2, mean)
```

```
##  [1] 0.04579826 0.04715404 0.04596269 0.04516189 0.04532554 0.04422680
##  [7] 0.04513279 0.04784227 0.04895462 0.04721587 0.04860876 0.04897792
## [13] 0.04892574 0.05252159 0.08996093 0.09048885 0.09014098 0.09094530
## [19] 0.09709948 0.09471619 0.09900744 0.09897390 0.13688813 0.10989081
```

```
## [25] 0.11351762 0.11925748 0.13738411 0.13882967 0.13662708 0.13726870
## [31] 0.13586695 0.13743237 0.16874974 0.11805714 0.14325795 0.12422127
## [37] 0.15949923 0.15608609 0.16351185 0.18273913 0.18123662 0.19881137
## [43] 0.19995903 0.21387065 0.22883300 0.22803254 0.22579150 0.22800440
## [49] 0.22812856 0.22625953 0.22705701 0.21841529 0.23821747 0.23849031
## [55] 0.23650854 0.25101042 0.25034637 0.25532303 0.27326701 0.27322986
## [61] 0.28106067 0.28706659 0.28903718 0.31787549 0.31748145 0.31714851
## [67] 0.31542333 0.33389483 0.32686804 0.38586191 0.31362283
```

As mentioned above, the posterior mean is a weighted average of the prior mean and the MLE. When our prior mean is determined by Beta(1,1), the posterior from Bayesian approach is close to the MLE.
