Math 185 Midterm 1

Lucien Chen

2024-05-05

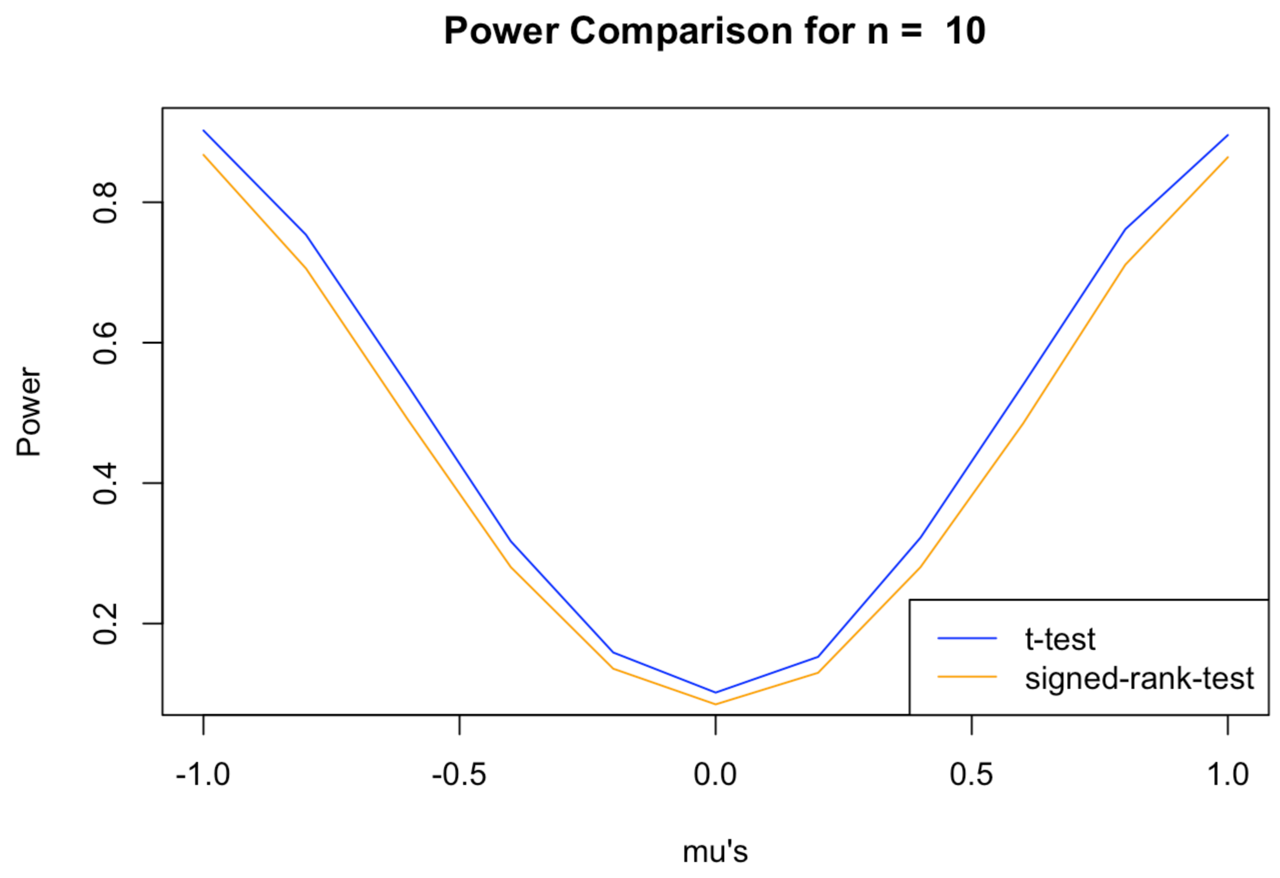
Midterm 1

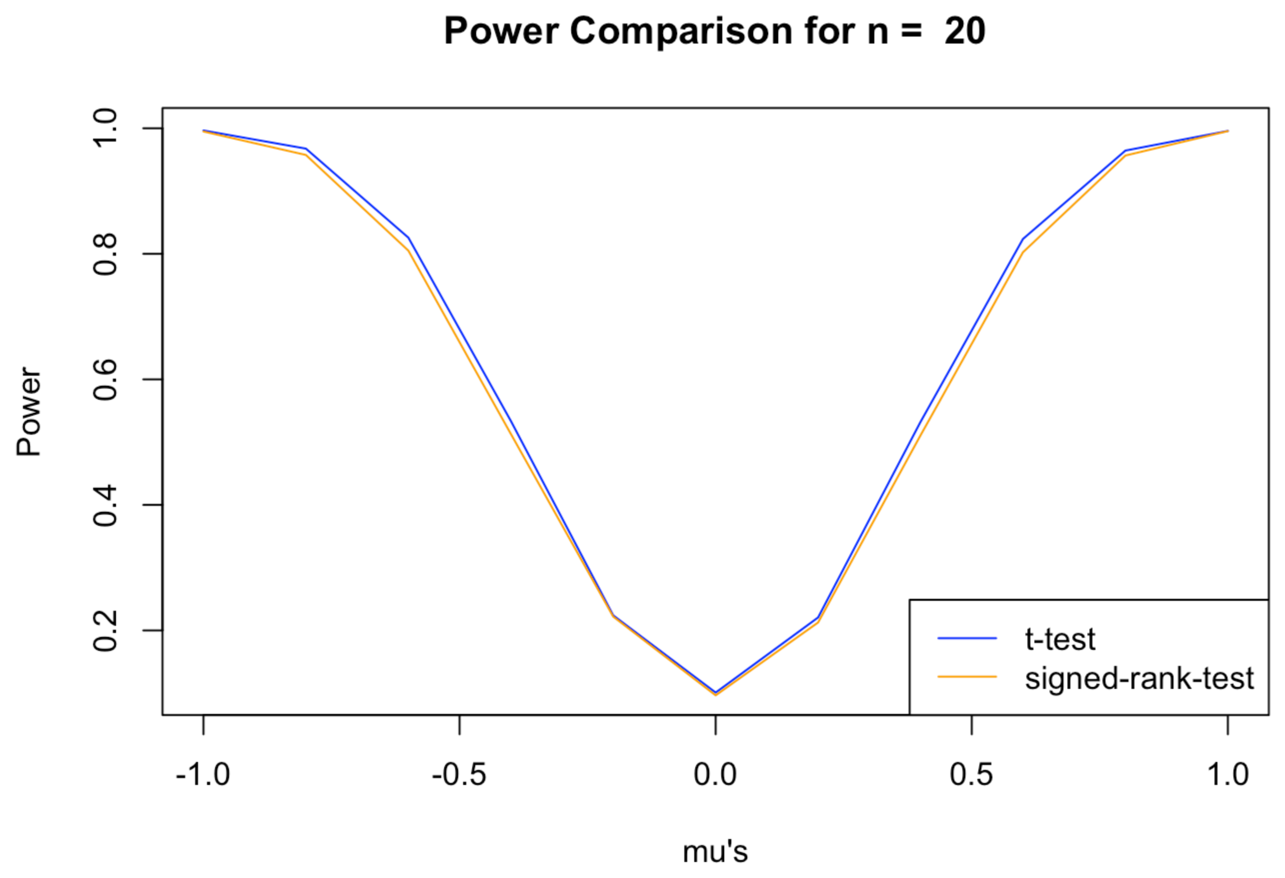
```r
library(dplyr)
library(readr)
```

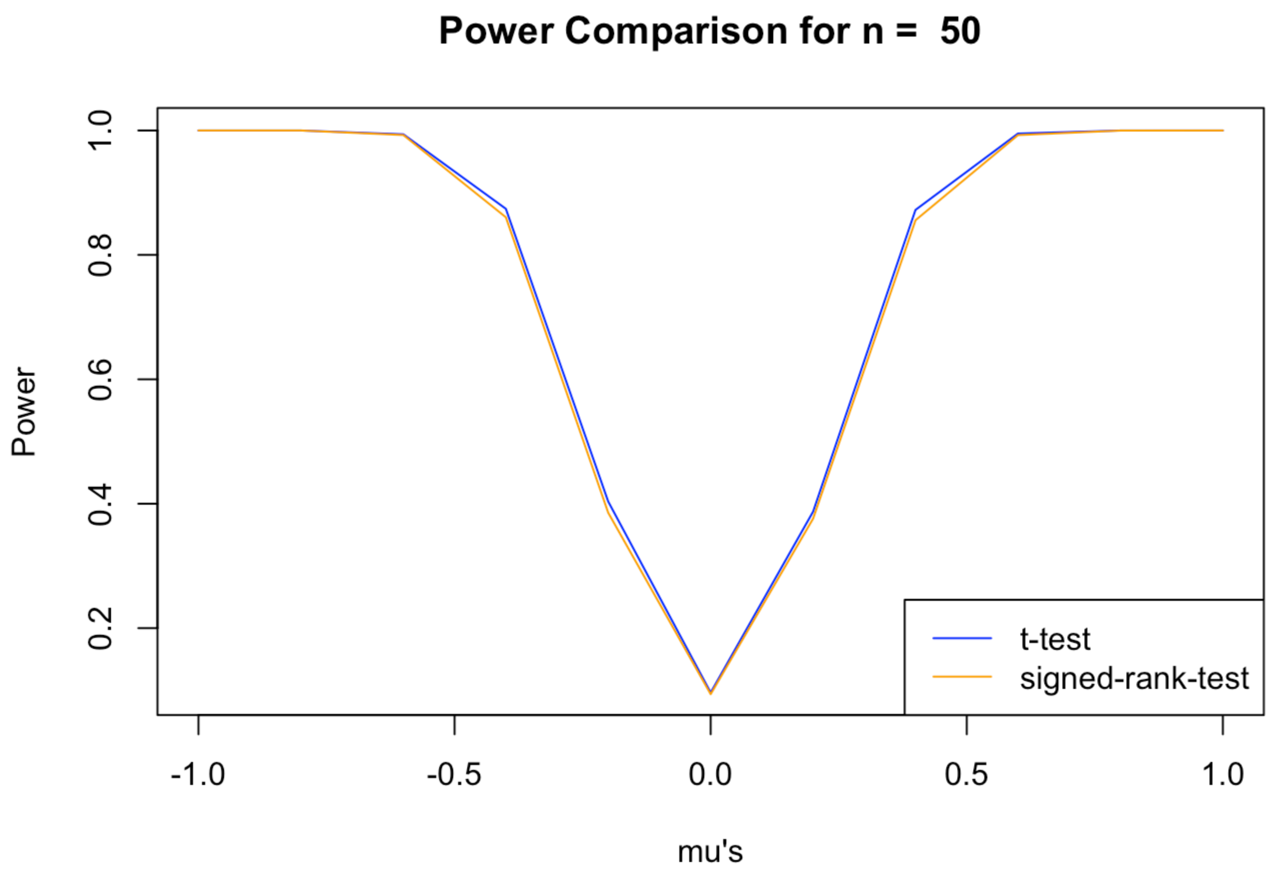
Problem 1

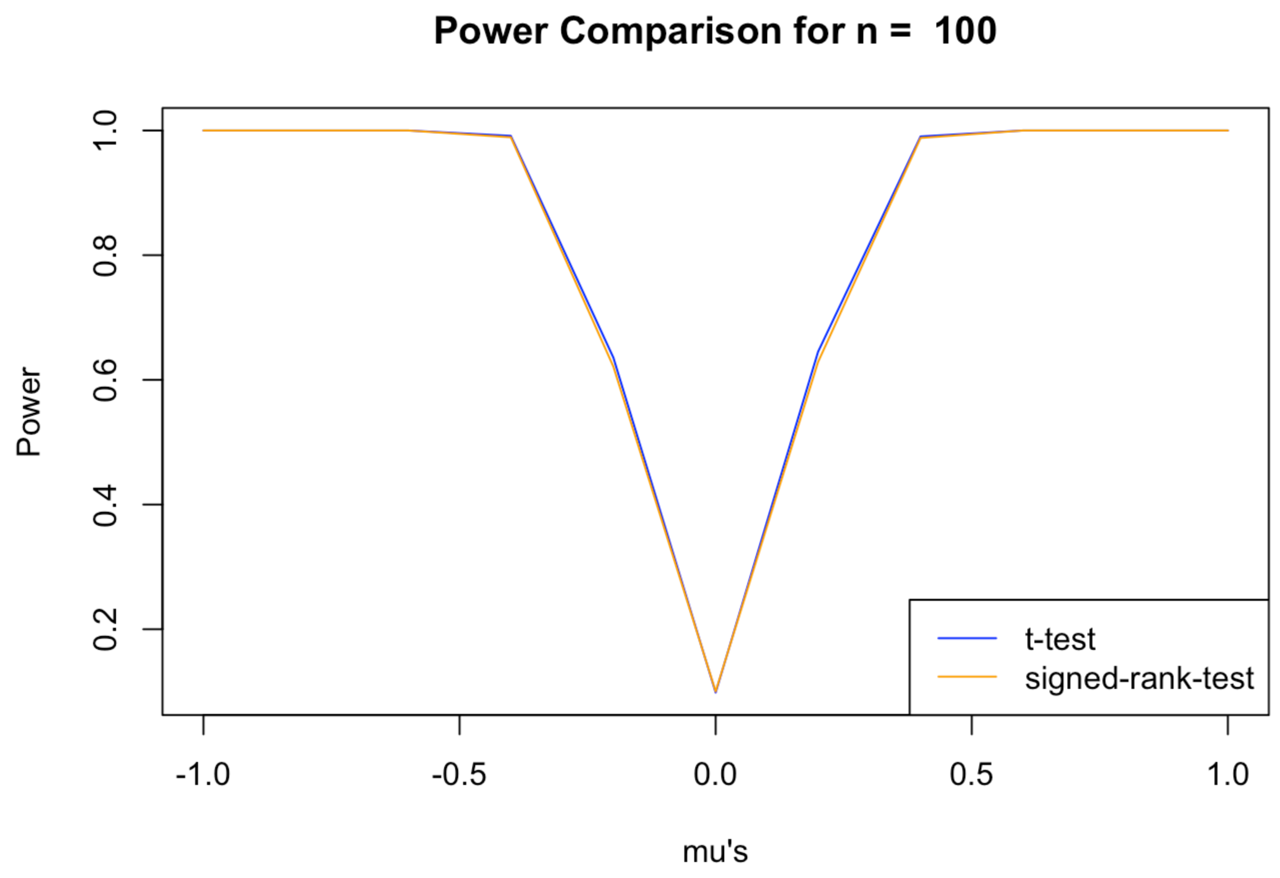
I simulate the scenario as follows. I first initialize all the necessary variables. Then, I create a helper function to generate # of rejections for the T test and signed-rank test accordingly. Finally, I repeat these simulations for each value of mu and n and generate plots for each n.

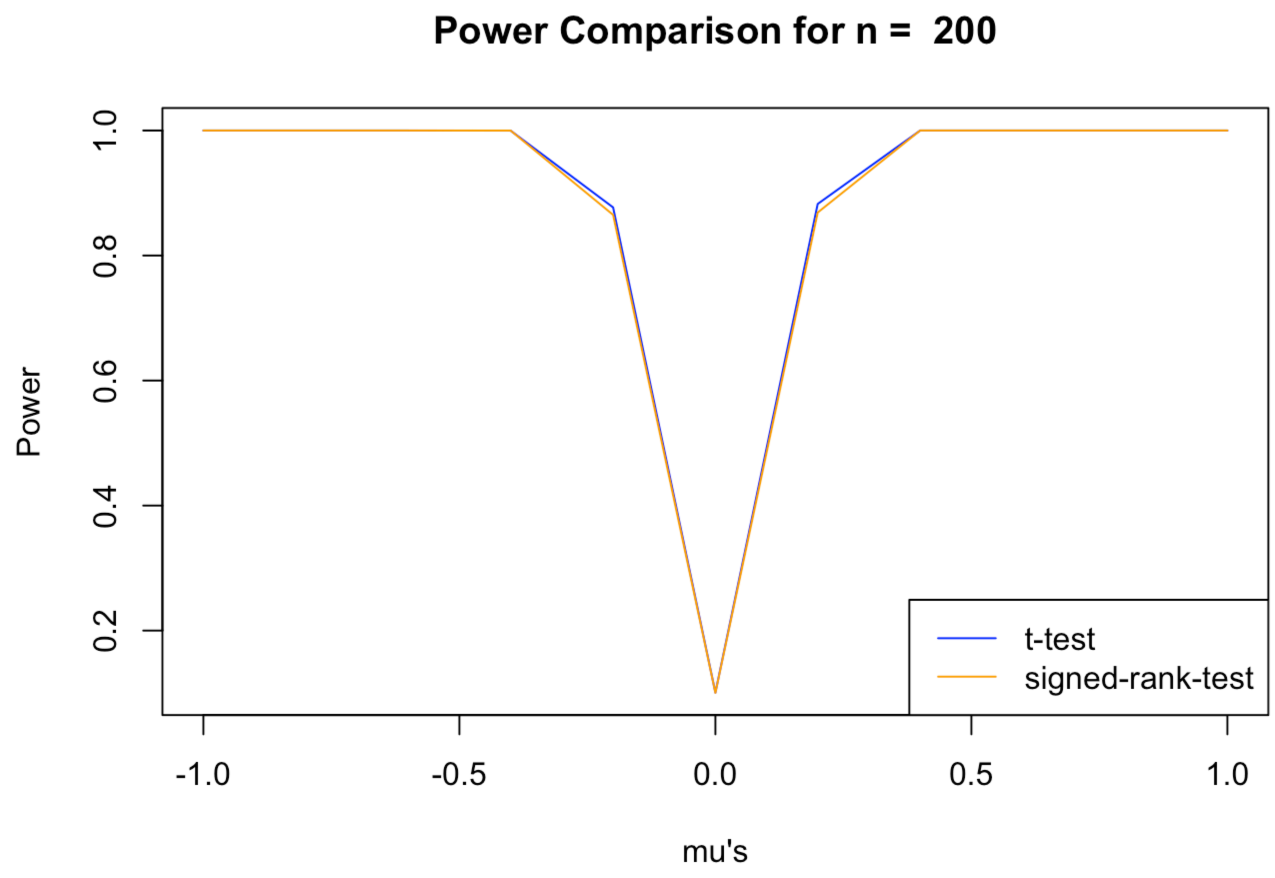
```r
N <- c(10, 20, 50, 100, 200, 500)
B <- 1e4
Mu <- seq(-1, 1, length.out=11)
sigma <- 1
alpha <- 0.1
simulate <- function(n, mu) {
  t_rejects <- 0
  signed_rank_rejects <- 0
  for (b in 1:B) {
    x <- rnorm(n, mean=mu, sd=sigma)
    T <- t.test(x, alternative="two.sided", mu=0)
    D <- wilcox.test(x, alternative="two.sided", mu=0)

    if (T$p.value < alpha) {
      t_rejects <- t_rejects + 1
    }

    if (D$p.value < alpha) {
      signed_rank_rejects <- signed_rank_rejects + 1
    }
  }
  t_power <- t_rejects / B
  signed_rank_power <- signed_rank_rejects / B
  return(c(t_power, signed_rank_power))
}

for (n in N) {
  t_powers <- numeric(length(Mu))
  signed_rank_powers <- numeric(length(Mu))
  for (i in 1:length(Mu)) {
    mu <- Mu[i]
    sim <- simulate(n, mu)
    t_powers[i] = sim[1]
    signed_rank_powers[i] = sim[2]
  }
  plot(Mu, t_powers, type="l", col="blue", xlab = "mu's", ylab="Power",
       main=paste("Power Comparison for n = ", n))
  lines(Mu, signed_rank_powers, col="orange")
  legend("bottomright", legend=c("t-test", "signed-rank-test"), col=c("blue", "orange"), lty=1)
}
```

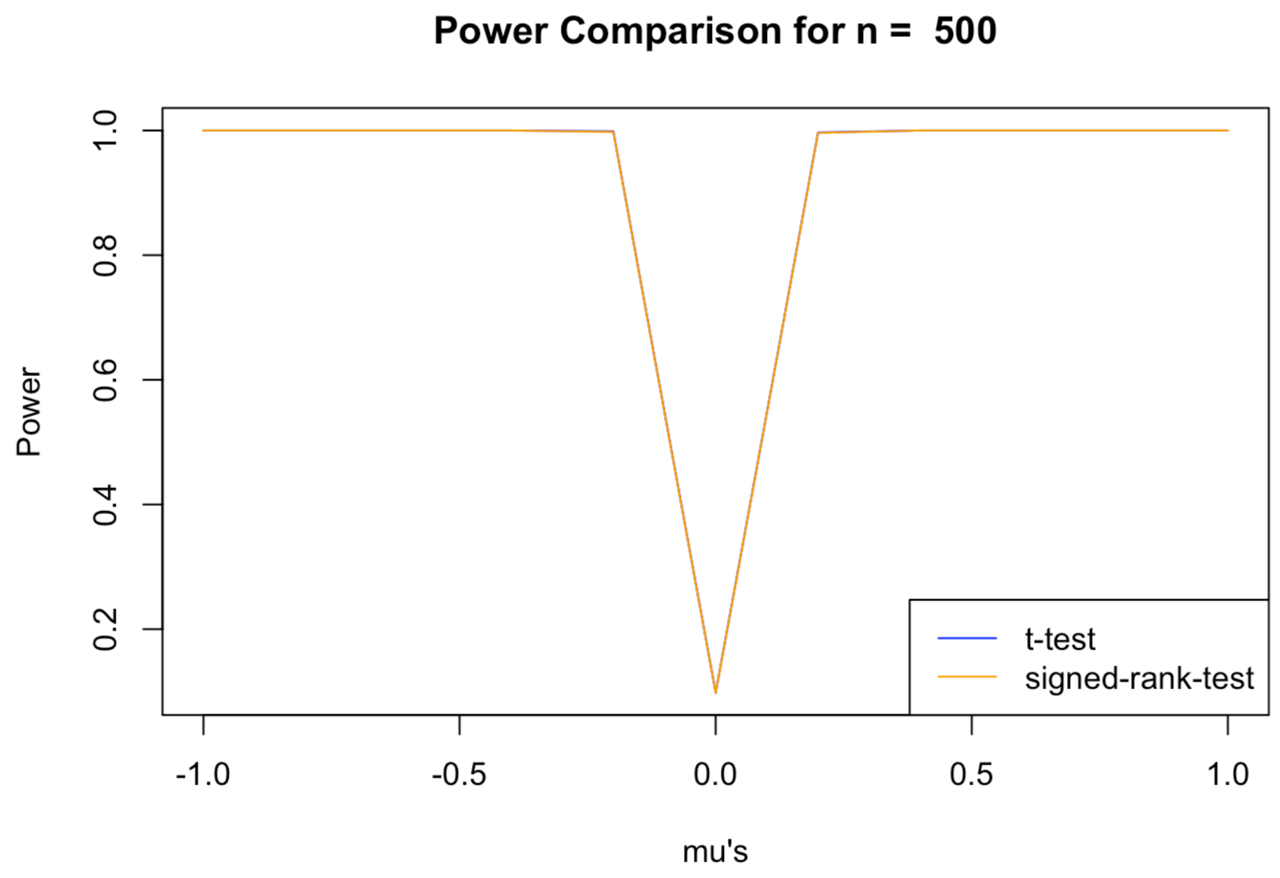

Power Comparison for n = 10
Power
0.2
0.4
0.6
0.8
-1.0
-0.5
0.0
0.5
1.0
mu's
t-test
signed-rank-test


Power Comparison for n = 20
Power
1.0
0.8
0.6
0.4
0.2
-1.0
-0.5
0.0
0.5
1.0
mu's
t-test
signed-rank-test


Power Comparison for n = 50
Power
1.0
0.8
0.6
0.4
0.2
-1.0
-0.5
0.0
0.5
1.0
mu's
t-test
signed-rank-test


Power Comparison for n = 100
Power
1.0
0.8
0.6
0.4
0.2
-1.0
-0.5
0.0
0.5
1.0
mu's
t-test
signed-rank-test


Power Comparison for n = 200
Power
mu's
1.0
0.8
0.6
0.4
0.2
-1.0
-0.5
0.0
0.5
1.0
t-test
signed-rank-test



As we can see from the plots, as we increase n, the difference in power between the t-test and the signed-rank test becomes smaller and smaller. We also observe that we get higher power for values near 0 and lower power for values closer to 0. This makes sense because we are testing for $\mu = 0$ and our test would have higher power when the true mean further away from 0.

Problem 2

First, let's load in our data. I will be working with scores of students in a classes with podcasts versus those without podcasts (taken from Math 181B).

```
dat <- read.csv("../data/podcast.csv", colClasses=c("numeric", "logical"))
head(dat)
```

```
##          x podcast
## 1 79.74648   FALSE
## 2 72.90180   FALSE
## 3 74.32638   FALSE
## 4 76.29433   FALSE
## 5 74.50803   FALSE
## 6 77.42243   FALSE
```

I define the bootCombinedTest function below. I first compute the observed treatment sum of squares from the original data. Then for B iterations, I create a bootstrap sample from the combined data and generate treatment sum of squares for each bootstrap sample. Then I calculate the bootstrap MC p-value accordingly.

```r
bootCombinedTest <- function(dat, B = 1e4) {
  vals <- dat %>% pull(1)
  groups <- dat %>% pull(2)
  obs_tss <- sum(tapply(vals, groups, function(x) length(x) * (mean(x) - grand_mean)^2))
  bootstrap_stats <- numeric(B)
  for (b in 1:B) {
    bootstrap_sample <- sample(vals, replace=TRUE)
    bootstrap_tss <- sum(tapply(bootstrap_sample, groups,
                          function(x) length(x) * (mean(x) - grand_mean)^2))
    bootstrap_stats[b] <- bootstrap_tss
  }
  bootstrap_p_value <- (sum(bootstrap_stats >= obs_tss) + 1) / (B + 1)
  return(bootstrap_p_value)
}
```

Now, I will test the function on the data I loaded in previously.

```r
bootCombinedTest(dat)
```

```
## [1] 0.00189981
```

This variant of the bootstrap does not produce a p-value that is valid under the null hypothesis that the groups come from populations that share the same mean because we treat the data as coming from one singular distribution versus grouping from each sample and as such we lose independence. Instead, this p-value is approximately valid for the null hypothesis that there is no difference in means between both groups.