Key Tools for Experimental Research Design and Analysis in R

Estimation, ATE, SE

February 28, 2018

Estimation

- We have already talked about the fundamental problem of causal inference.
- We can use sample data (statistics) to estimate parameters (like the Average Treatment Effect).

A simulation in R: sample mean as an unbiased estimator of the population mean

First we will need to "create" a population, a *box of tickets*

```
population <- c(4,5,7,12,7,8,9,-3,5,8,9,3,2,3,4,6,10,4,6,7,8,9,2)

N <- length(population) # number of observations in the population
N
```

```
## [1] 23
```

```
pop_mean <- mean(population) # population mean
pop_mean
```

```
## [1] 5.869565
```

```
pop_sd <- sd(population) # population standard deviation
pop_sd
```

```
## [1] 3.293304
```

We will draw several random samples of 8 observations (m) each *without* replacement

```
s1 <- sample(population, size=8, replace = FALSE)

s2 <- sample(population, size=8, replace = FALSE)

s3 <- sample(population, size=8, replace = FALSE)

s4 <- sample(population, size=8, replace = FALSE)

samples <- rbind(s1, s2, s3, s4)

samples
```

```
##    [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## s1    5    8    7    6    2    5    9    4
## s2    7    7    9    4    3    8    7    2
## s3    4    4    9    2    7    8   12    2
## s4    4    7    8    8    2    7    6    6
```

Remember the population mean: 5.8695652

And the means of the samples

```
apply(samples, MARGIN=1, FUN=mean)
```

```
##    s1    s2    s3    s4
## 5.750 5.875 6.000 6.000
```

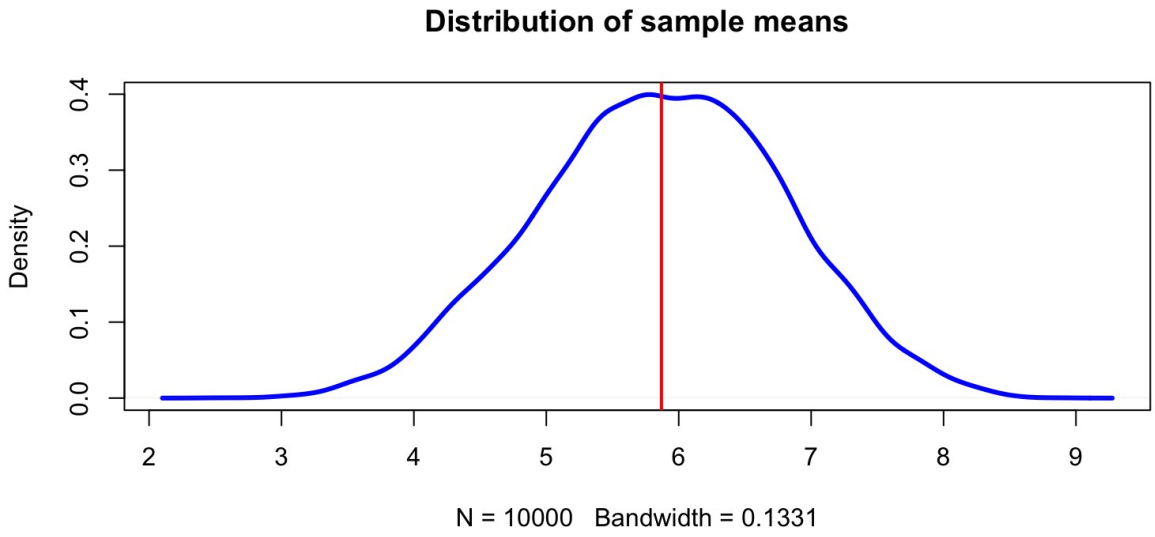
By chance each given sample mean may be a little higher or lower than the population mean.

How can we use R to show that the sample mean is an unbiased estimator of the population mean?

For this, we will write a *simulation*. We will repeat the sample process $10,000$ times.

```
sample_mean <- NA

for (i in 1:10000){

  sample <- sample(population, size=8, replace = FALSE)
  sample_mean[i] <- mean(sample)

}
```

```r
par(mfrow=c(1,1))
plot(density(sample_mean), col="blue", lwd=3,
     main="Distribution of sample means")
abline(v=pop_mean, col="red", lwd=2)
```




```r
average_sampling_distribution<- mean(sample_mean)
round(average_sampling_distribution,2)
```

```
## [1] 5.86
```

```r
round(pop_mean, 2)
```

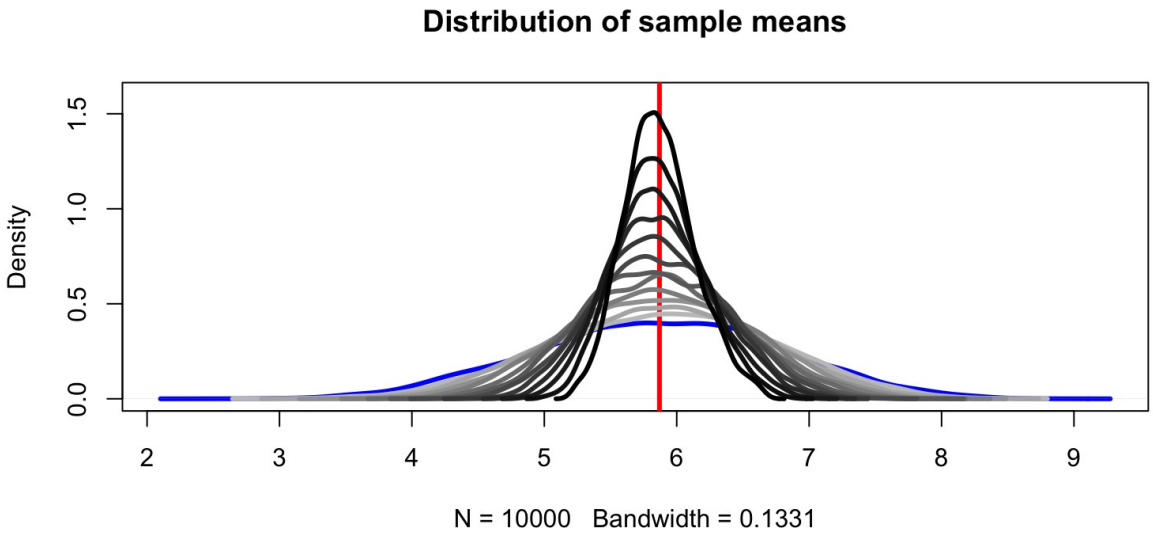
```
## [1] 5.87
```

Let's now look at the distribution of the sample mean as m gets closer to N.

So far, $m = 8$. We now need a new simulation that adds a new step: we need to vary the size of *m*. (Remember our population size, *N*, is 23)

```r
rep <- 10000

# The first loop varies m
for (m in 9:20){

  sample_mean <- NA #creating an object to store the results of the second loop

  # The second loop goes through the 10,000 simulations
  for (i in 1:rep){

    #we first get a random sample of size m from the population
    sample <- sample(population, size=m, replace = FALSE)
    #and then calculate and store the sample mean
    sample_mean[i] <- mean(sample)
  }

  #finally, we plot the distribution of the 10,000 sample means for the relevant m
  lines(density(sample_mean), lwd=3,
        #note that this next line of code varies the color of the line according to m
        #so that we can distinguish the different distributions
        col=paste0("grey",140-(7*m)))
}
```

What do we expect? Why?


Distribution of sample means
Density
0.0
0.5
1.0
1.5
2
3
4
5
6
7
8
9
N = 10000 Bandwidth = 0.1331

Remember the formula for the variance of the sample mean for the treatment group is:

$$\mathrm{Var}(Y^T) = \frac{\sigma^2}{m}$$

But we do not know σ^2 (it is a parameter). But we can estimate this quantity with the variance of the assigned-to-treatment sample by:

$$\hat{\sigma}^2 = (\frac{1}{m-1}) \sum_{i=1}^{m} (Y_i - \bar{Y^T})^2$$

Same with the variance of the sample mean for those units assigned to control. These are the analytic formulas… But we could estimate via bootstrapping.

1. Variance of the sample mean

2. The bootstrap

- 2a. CLT: Distribution of the bootstap mean as a function of sample size
- 2b. Comparing the estimated SE using the analytic formula and the boostrap

1. The variance of the sample mean

First we need a box to sample from.

```
pop1 <- rnorm(50, 0, 5) # draw 50 units from normal dist.
```

Let's see what is the average of this population

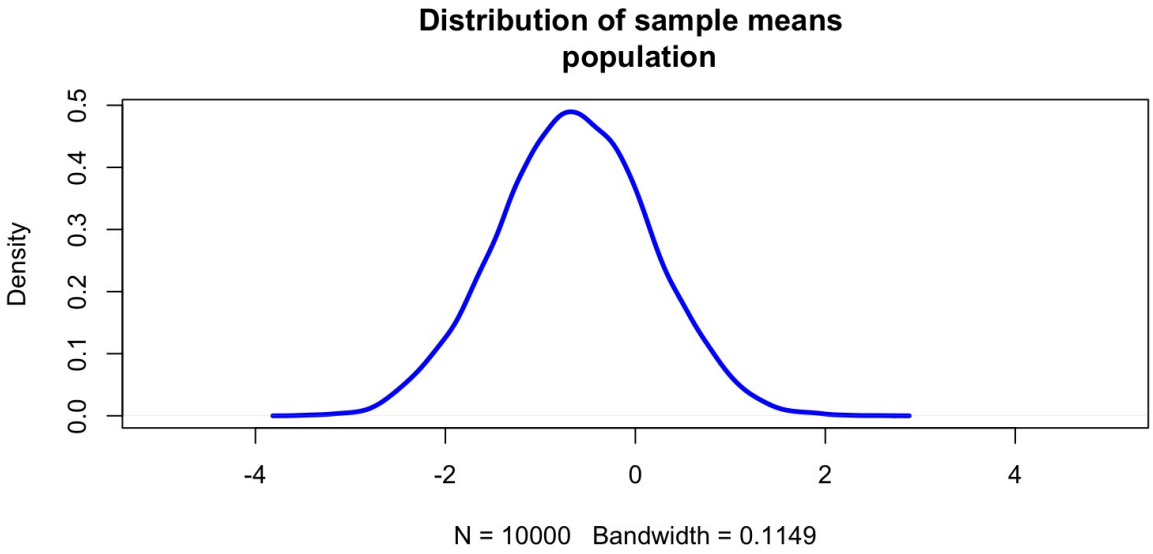
```
mean(pop1)
```

```
## [1] -0.6520966
```

- We will sample 25 units from this population without replacement (as in most experiments)
- We will repeat this process 10,000 times

```r
sample_means <- NA #Vector placeholder

for (i in 1:10000){
  sample_means[i] <- mean(sample(pop1, 25, replace=F))
}

head(sample_means)
```

```
## [1] -1.3181017  0.2245676 -1.6005098  0.1128294 -1.8807701 -1.7814493
```




```
mean(sample_means)
```

```
## [1] -0.6481124
```

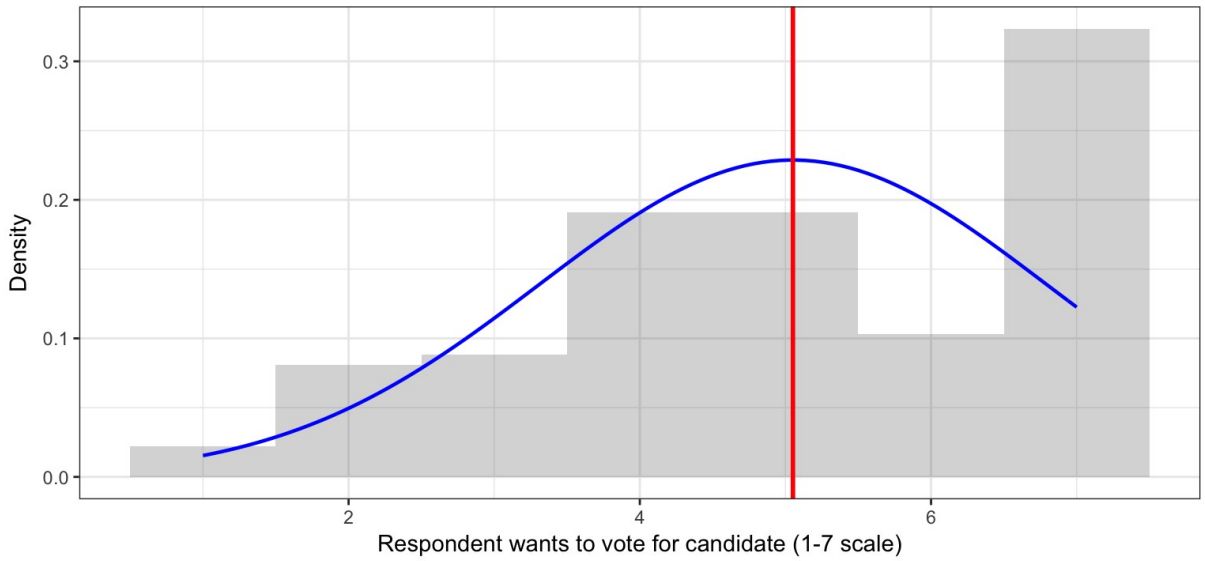
```
var(sample_means)
```

```
## [1] 0.6491997
```

2. The bootstrap

More often, we do not observe the full potential outcomes schedule for our population. We are now going to use the sample values for a study conducted by Dunnign and Harrison, where treatment is co-ethnic cousin ($treat\_assign = 1$). This will be our box, the "bootstrap population".

Let's plot the empirical distribution of responses in the treatment group.



We will use the bootstrap to investigate the properties of different sampling procedures.

1. Take the sample values as the population;
2. Draw a sample from this population (box) with replacement, using the sampling procedure we want to analyze; save the sample statistics (e.g. the mean). This is a "bootstrap replicate."
3. Repeat step (2) many times (say, 10,000 times).
4. Plot the distribution of the saved statistics across all the bootstrap replicates. This gives us a good glimpse of the sampling distribution of the statistic of interest.

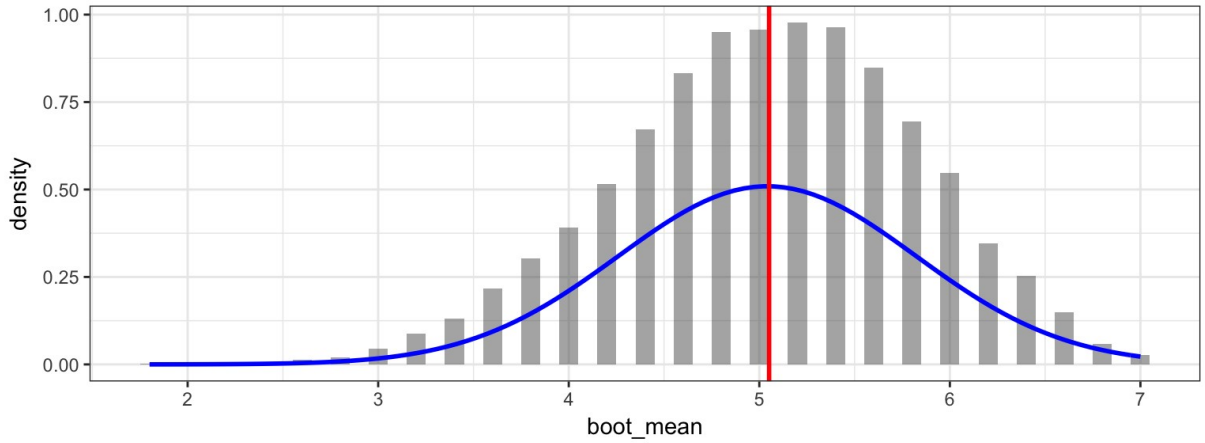
```r
# We first need to define the number of units we will draw from the box
# (with replacement).
# For the first example, let's take N=5
N <- 5

# If we wanted to do step (2) once, we would sample from the box N times with replacement
boot_sample <- sample(box$box, N, replace=T)
# And then take the statistic of interest for this bootstrap sample, here the mean.
mean(boot_sample)
```

```
## [1] 5
```

But we want to do this many times— 10,000 times! What can we do?

We write a for loop that repeats this sampling procedure 10,000 times:

```r
boot_reps <- 10000 # number of bootstrap replicates we will repeat step (2) for
boot_mean <- NA # placeholder vector for the results

for (i in 1:boot_reps){
  boot_sample <- sample(box$box, N, replace=T)
  boot_mean[i] <- mean(boot_sample)
}
```



We will now repeat this procedure, varying the size of the bootstrap sample, several times. We can write a function which does the bootstraps, and plots a histogram for their mean.

Our function will require - the data, - the number of bootstrap replicates, - the number of observations to be sampled from the box in each replicate - the binwidth to be used for the histogram.

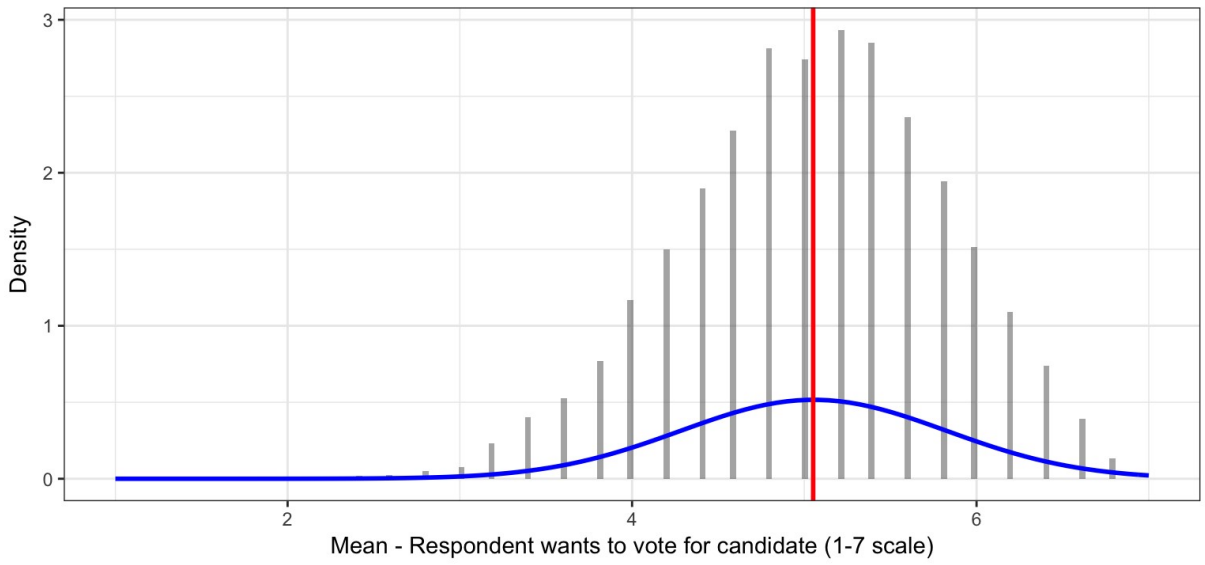
It will sample N units with replacement and get the mean for that sample

```r
bootstrap_mean <- function(data, replicates=10000, N, bin, title="",
                           xlab="", ylab="", xmax, xmin){

  boot_mean <- NA #plaaceholder vector
  for (i in 1:replicates){
    # we sample N units from the box with replacement
    boot_sample <- sample(data, N, replace=T)
    # and save their mean
    boot_mean[i] <- mean(boot_sample)
  }

  m <- ggplot(as.data.frame(boot_mean), aes(x=boot_mean))
  # First we plot a histogram with the results
  m + geom_histogram(aes(y = ..density..), alpha=.5, binwidth=bin) +
    # and overlay a line with the density of a normal distribution with mean equal to the mean
    # of the bootstrap means and sd equal to the sd of the bootstrap means.
    stat_function(fun=dnorm,
                  args=list(mean=mean(boot_mean), sd=sd(boot_mean)), col="blue", size=1) +
    # and we add a vertical line for the mean of the box
    geom_vline(xintercept = mean(box$box), col="red", size=1) +
    # limits for x axis
    scale_x_continuous(limits = c(xmin, xmax)) +
    # and labels
    labs(title=title,  x=xlab, y=ylab) + theme_bw()
}
```

Now we will run the bootstrap varying N.

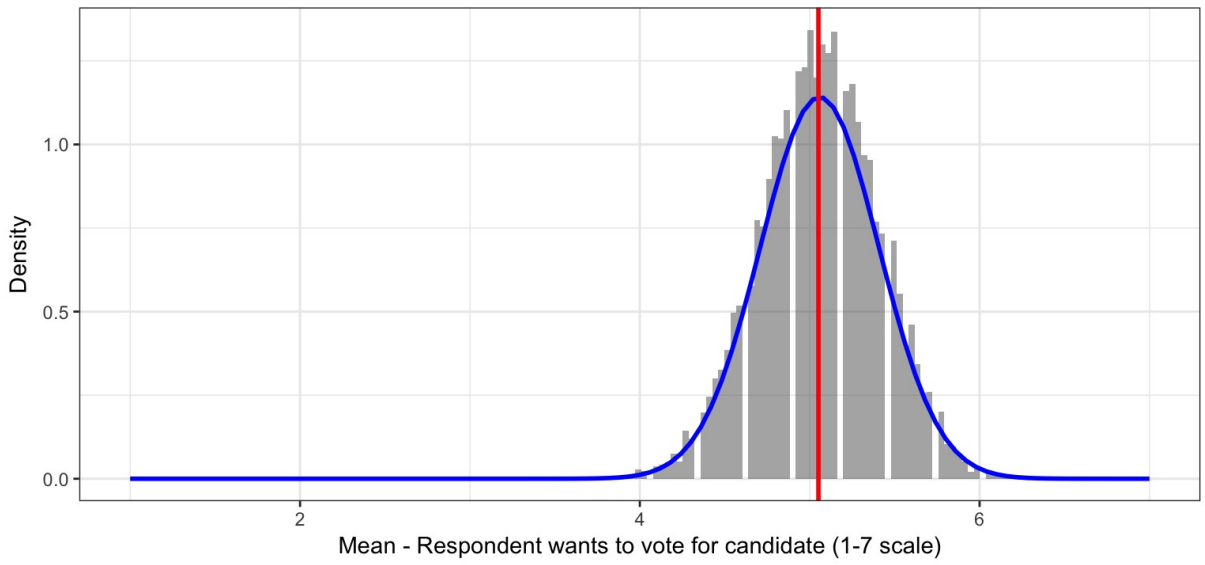
N = 5

```
bootstrap_mean(data=box$box, replicates=10000, N=5, bin=.035,
               # title="N=5",
               xmin=1, xmax=7,
               xlab="Mean - Respondent wants to vote for candidate (1-7 scale)",
               ylab="Density")
```



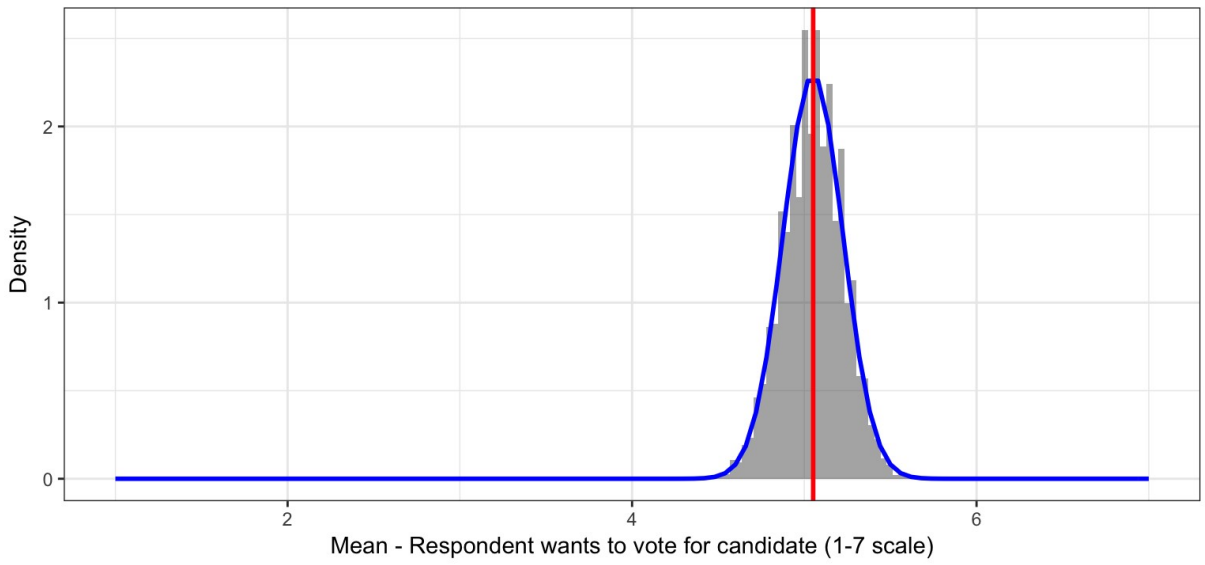
N = 25

```
bootstrap_mean(data=box$box, replicates=10000, N=25, bin=.035,
               # title="N=25",
               xmin=1, xmax=7,
               xlab="Mean - Respondent wants to vote for candidate (1-7 scale)",
               ylab="Density")
```



N = 100

```
bootstrap_mean(data=box$box, replicates=10000, N=100, bin=.035,
               # title="N=100",
               xmin=1, xmax=7,
               xlab="Mean - Respondent wants to vote for candidate (1-7 scale)",
               ylab="Density")
```



Now, let's compare the $\widehat{SE}$ that we obtain from the analytic formula and the bootstrap:

- The analytic formula of the $\widehat{SE}$ for the sample mean in the treatment group:

$$\widehat{SE} = \frac{\hat{\sigma}}{\sqrt{m}}$$

```
# We first need an unbiased estimator of the variance:
est.var.analy <- var(box) # denominator : (n-1)
# Then we plug it into the analytic formula of the SE (est.)
se_analy = sqrt(est.var.analy/(nrow(box)))
se_analy
```

```
##           box
## box 0.1495527
```

- Now, how can we get the SE from the bootstrap replicates?

```r
# Bootstrap:
N<-136
replicates <- 10000
boot_mean <- NA #plaaceholder vector

  for (i in 1:replicates){
    # we sample N units from the box with replacement
    boot_sample <- sample(box$box, N, replace=T)
    # and save their mean
    boot_mean[i] <- mean(boot_sample)
  }

boot.se<-sd(boot_mean)
ses<- c(se_analy,boot.se)
names(ses)<-c("Analytic Formula","Bootstrap")
ses
```

```
## Analytic Formula        Bootstrap
##        0.1495527        0.1469314
```

In this case, the bootstrap gives the same answer as the analytic formula because the latter is correct for i.i.d. sampling, and here we are building i.i.d. sampling into the bootstrap routine.

3. The difference of two means

We can write a function to estimate the difference in means. As I mentioned on Monday, a function is composed by: i) body, ii) arguments, and iii) environment (usually Global unless otherwise specified)

```
diff_means <- function(y, x){

  # Calculating difference in means
  mean1 <- mean(y[x==1], na.rm=T)
  mean0 <- mean(y[x==0], na.rm=T)
  diff <- mean1 - mean0

  # Calculating number of observations
  N <- length(na.omit(y))

  # Preparing output
  res <- c(mean1, mean0, diff, N)
  names(res) <- c("Mean 1", "Mean 0", "Difference", "N")

  return(c(res))
}
```

Now, let's explore the components of the function we just created:

```
body(diff_means)
```

```
## {
##     mean1 <- mean(y[x == 1], na.rm = T)
##     mean0 <- mean(y[x == 0], na.rm = T)
##     diff <- mean1 - mean0
##     N <- length(na.omit(y))
##     res <- c(mean1, mean0, diff, N)
##     names(res) <- c("Mean 1", "Mean 0", "Difference", "N")
##     return(c(res))
## }
```

```
formals(diff_means)
```

```
## $y
##
##
## $x
```

```
environment(diff_means)
```

```
## <environment: R_GlobalEnv>
```

To try our function, we will use the small dataset in Gerber & Green (2012)

```
gg_data <- as.data.frame(cbind(c(10,15,20,20,10,15,15),
                               c(15,15,30,15,20,15,30)))
names(gg_data) <- c("Y_i0", "Y_i1")
save(gg_data, file="gg_data.Rda")
```

We will need to “create” a treatment vector…

```
# let's fix m=3 (units in the treatment group)
treat <- c(1, 1, 1, 0, 0, 0, 0)
gg_data$treat <- sample(treat, 7, replace=F)
gg_data$treat
```

```
## [1] 1 0 0 1 0 1 0
```

…and a column with the “observed” outcomes

```
gg_data$observed <- ifelse(gg_data$treat==1, gg_data$Y_i1, gg_data$Y_i0)
```

Let's see how the complete data set looks now:

```
head(gg_data)
```

```
##   Y_i0 Y_i1 treat observed
## 1   10   15     1       15
## 2   15   15     0       15
## 3   20   30     0       20
## 4   20   15     1       15
## 5   10   20     0       10
## 6   15   15     1       15
```

```r
# mean of the treatment group
mean(gg_data$observed[gg_data$treat==1])
```

```
## [1] 15
```

```r
# mean of the control group
mean(gg_data$observed[gg_data$treat==0])
```

```
## [1] 15
```

```r
# difference of means
mean(gg_data$observed[gg_data$treat==1]) - mean(gg_data$observed[gg_data$treat==0])
```

```
## [1] 0
```

```r
# with our function
diff_means(gg_data$observed, gg_data$treat)
```

```
##     Mean 1     Mean 0 Difference          N
##         15         15          0          7
```

How can we get a distribution of the difference of means?

We can do this with a simulations. For each simulation,

- First: We will need to "create" a random treatment vector and generate the column with the associated observed outcomes.
- Second: We will have to calculate the difference between the treatment and control means (by hand or using our new function).

```r
# 1.
gg_data$treat <- sample(treat, 7, replace=F)
gg_data$observed <- ifelse(gg_data$treat==1, gg_data$Y_i1, gg_data$Y_i0)

# 2.
diff_means(gg_data$observed, gg_data$treat)
```

```
##     Mean 1     Mean 0 Difference          N
##  21.666667  16.250000   5.416667   7.000000
```

```r
# we should store this! so,
dm <- diff_means(gg_data$observed, gg_data$treat)
dm
```

```
##     Mean 1     Mean 0 Difference          N
##  21.666667  16.250000   5.416667   7.000000
```

```r
# but we only want the third element!
dm <- diff_means(gg_data$observed, gg_data$treat)[3]
dm
```

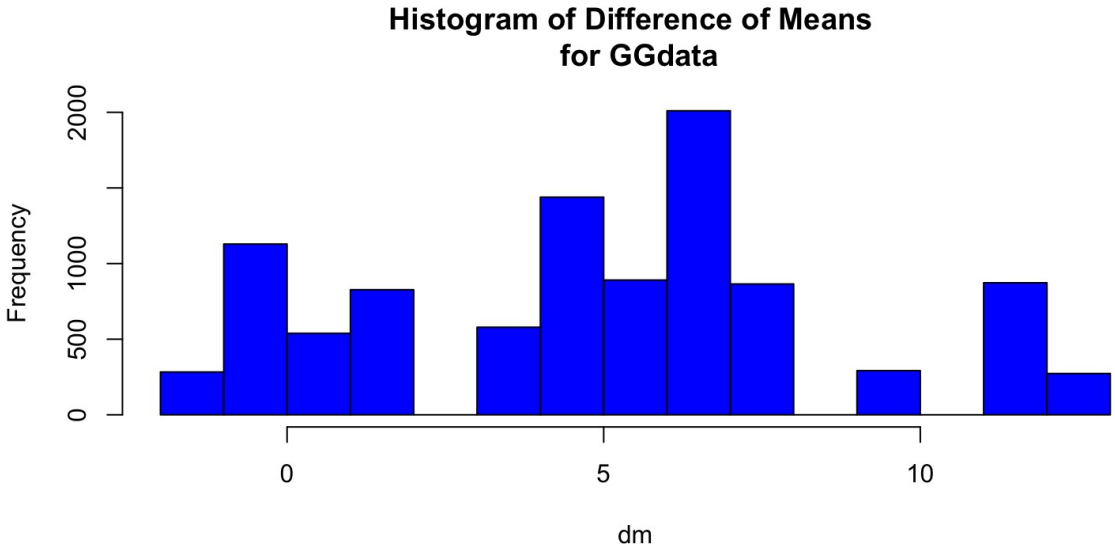
```
## Difference
##   5.416667
```

Now let's put this in a loop that allows us to repeat the process $10,000$ times (and saves the dom for each)…

```
dm <- NA #creating a placeholder to store all our doms...

for (i in 1:10000){

    # 1.
    gg_data$treat <- sample(treat, 7, replace=F)
    gg_data$observed <- ifelse(gg_data$treat==1, gg_data$Y_i1, gg_data$Y_i0)

    # 2.
    dm[i] <- diff_means(gg_data$observed, gg_data$treat)[3]

    }
```

Finally, let's plot the distribution

```
hist(dm, col="blue", main="Histogram of Difference of Means \n for GGdata")
```



4. Standard Error for the ATE and hypothesis testing

1. Standard error for the difference in means

2. Hypothesis testing

- 2a. T-test
- 2b. Randomization inference

1. Standard error for the difference in means

- The difference in means is an unbiased estimator of the true ATE. However, by chance, in some realizations of our sample that estimate might be off the true ATE.
- The SE tells us the likely size of the amount off.

A conservative formula for the $\widehat{SE}$ for the $\widehat{ATE}$

$$\widehat{SE}(\widehat{ATE}) = \sqrt{\frac{\widehat{\text{Var}}(Y_i(0))}{N-m} + \frac{\widehat{\text{Var}}(Y_i(1))}{m}}$$

This formula assumes independence and sampling with replacement.

We are going to estimate the SE for the difference in means using the data from Chattopadhyay and Duflo (2004) referenced in Gerber and Green (2012).

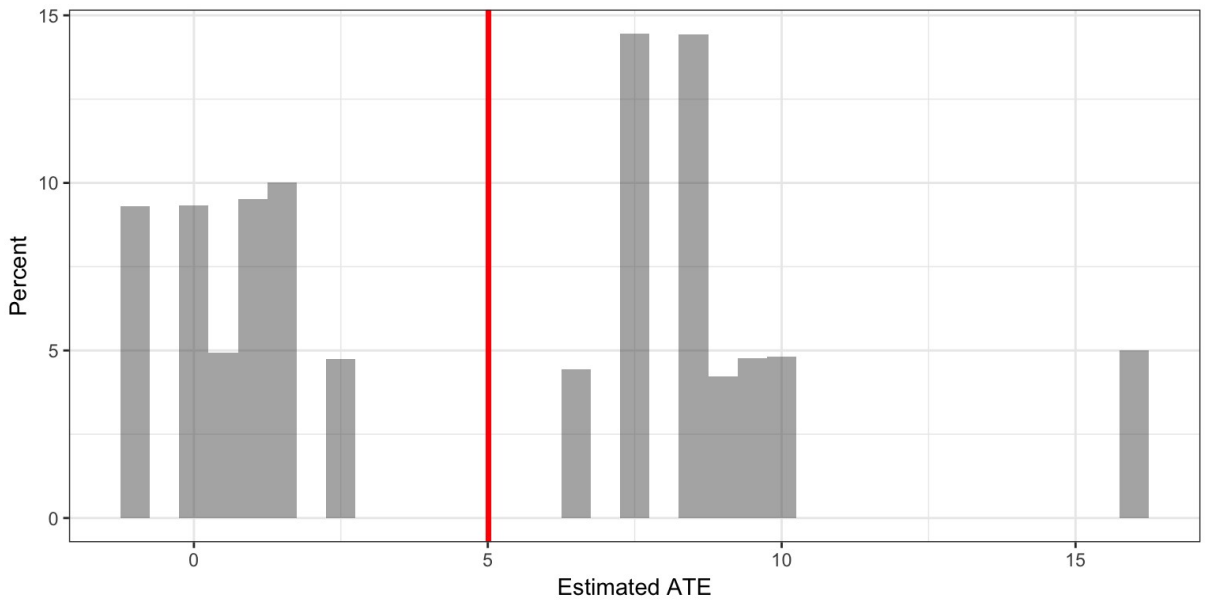
```
# the data
gg_data <- as.data.frame(cbind(c(10,15,20,20,10,15,15),
                               c(15,15,30,15,20,15,30)))
names(gg_data) <- c("Y_i0", "Y_i1")

true_ate<-mean(gg_data$Y_i1) - mean(gg_data$Y_i0)
```

```r
# generating empty dataframe to put the results
ate <- as.data.frame(matrix(NA, 10000, 2))
names(ate) <- c("estimated_ate", "estimated_se_ate")

# sampling
for (i in 1:10000){

  # generating treatment vector for this replicate
  gg_data$treat <- 0
  gg_data$treat[sample(1:7, 2, replace=F)]  <- 1

  treat_mean <- mean(gg_data$Y_i1[gg_data$treat==1])
  treat_var <- var(gg_data$Y_i1[gg_data$treat==1])

  control_mean <- mean(gg_data$Y_i0[gg_data$treat==0])
  control_var <- var(gg_data$Y_i0[gg_data$treat==0])

  ate[i,1] <- treat_mean - control_mean
  ate[i,2] <- sqrt(treat_var/2 + control_var/5)
}
```

Let's explore how this matrix looks like:

```
head(ate)
```

```
##   estimated_ate estimated_se_ate
## 1          16.0         1.870829
## 2           9.5         7.599342
## 3           0.5         2.783882
## 4           7.5         7.826238
## 5           2.5         2.958040
## 6           0.0         1.581139
```

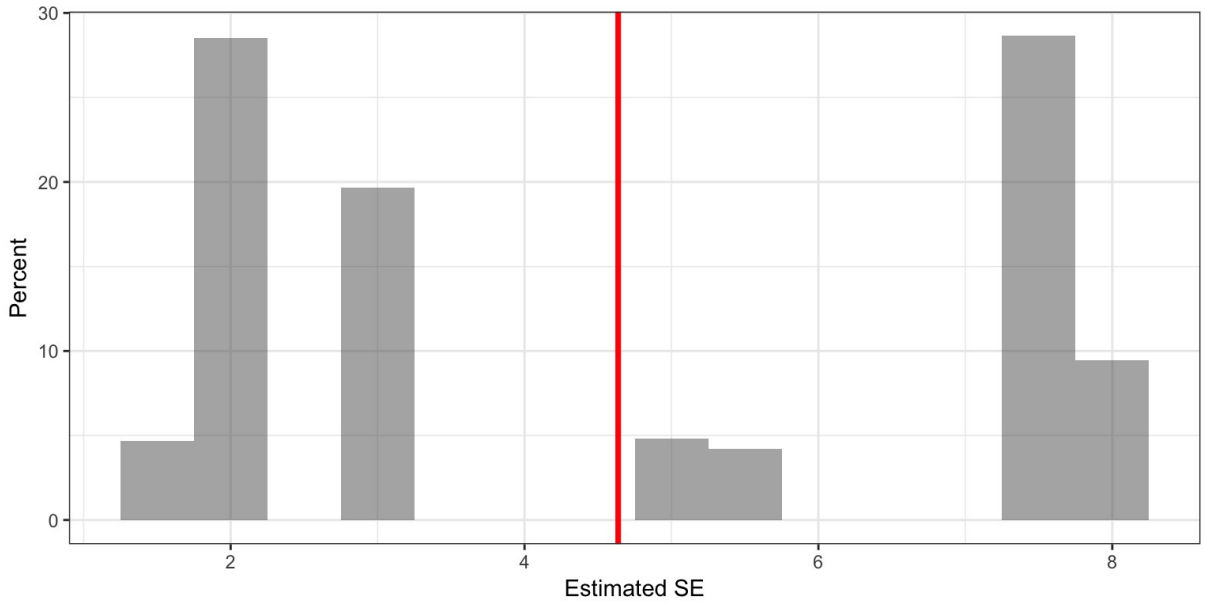


- What should be the title of this figure?
- How could we use this graph to get the SE of the estimated ATE?

```r
# The SE of the estimated ATE is the standard deviation of this sampling distribution:
se_sampling<-sd(ate[,1])
se_sampling
```

```
## [1] 4.627147
```

- Notice, the estimated SE will also have a distribution.



What should the average of the estimated SEs?

```r
# Comparing the true standard error to the conservative formula
se_est<-mean(ate[,2])
print(c(se_sampling, se_est))
```

```
## [1] 4.627147 4.637367
```

2. Hypothesis testing

```
# generating treatment vector for a given experiment
gg_data$treat <- c(1, 0, 0, 0, 0, 0, 1)

# getting observed outcomes
gg_data$observed <- ifelse(gg_data$treat==1, gg_data$Y_i1, gg_data$Y_i0)

# ate
ATE <- mean(gg_data$observed[gg_data$treat==1]) - mean(gg_data$observed[gg_data$treat==0])
ATE
```

```
## [1] 6.5
```

- What is a p-value?

2a. T-test

```
treated <- gg_data$observed[gg_data$treat==1]
treated
```

```
## [1] 15 30
```

```
var1 <- sum((treated - mean(treated))^2) / (length(treated) - 1)
var1
```

```
## [1] 112.5
```

```
not_treated <- gg_data$observed[gg_data$treat==0]
not_treated
```

```
## [1] 15 20 20 10 15
```

```
var0 <- sum((not_treated - mean(not_treated))^2) / (length(not_treated) - 1)
var0
```

```
## [1] 17.5
```

```
estimated_se <- sqrt(var1/length(treated) + var0/length(not_treated))
estimated_se
```

```
## [1] 7.729812
```

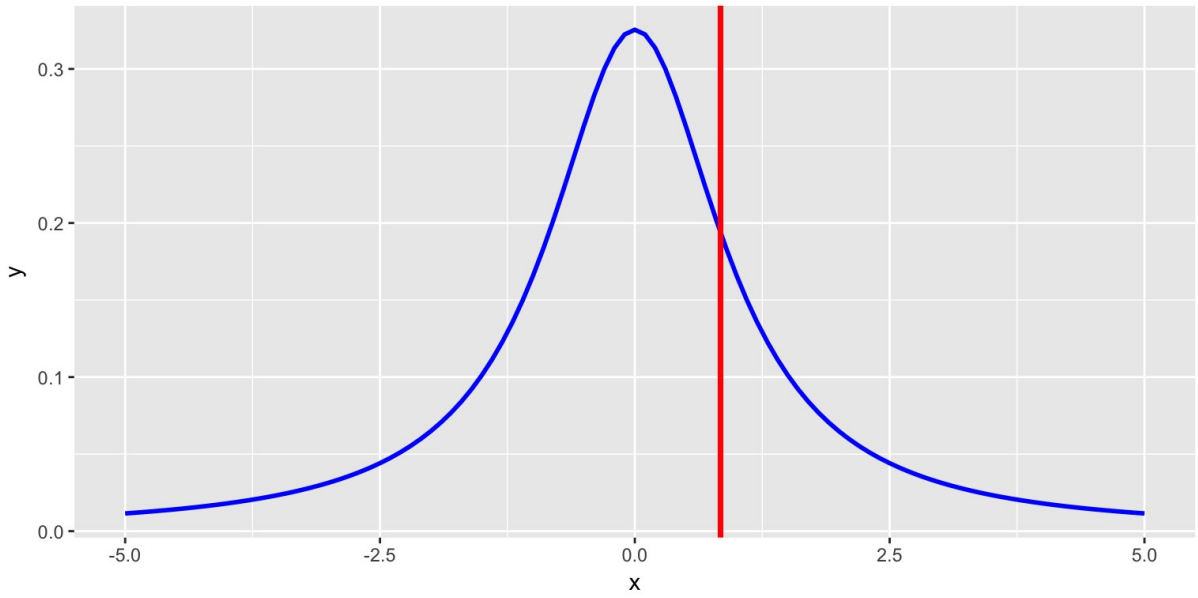
```r
# converting to standard units: Why is it ATE - 0?
t_stat <- (ATE - 0) / estimated_se
t_stat
```

```
## [1] 0.8409001
```

```r
# To be able to get the right Student t Distribution, we need to calculate
# the degrees of freedom (Satterthwaite)
df <- (var1/length(treated) + var0/length(not_treated))^2 /
          ((var1/length(treated))^2 / (length(treated) - 1) +
          (var0/length(not_treated))^2 / (length(not_treated) - 1))
df
```

```
## [1] 1.127225
```

```r
# Overlaying the t_stat to the student t distribution
ggplot(data.frame(x = c(-5, 5)), aes(x)) +
  stat_function(fun=dt, args=list(df=df, ncp=0), col="blue", size=1) +
  geom_vline(xintercept = mean(t_stat), col="red", size=1.25)
```



```r
# One tailed p-value
pt(t_stat, df=df, ncp=0, lower.tail=F)
```

```
## [1] 0.2708243
```

```r
# Two tailed p-value
pt(-t_stat, df=df, ncp=0, lower.tail=T) + pt(t_stat, df=1.12, ncp=0, lower.tail=F)
```

```
## [1] 0.5419946
```

2b. Randomization inference

To get all the possible treatment vectors, we will generate 10000 differente ones…

```
fake_treats <- matrix(NA, 10000, 7)
for (i in 1:10000){
fake_treats[i,] <- sample(gg_data$treat, 7, replace=F)
}
```

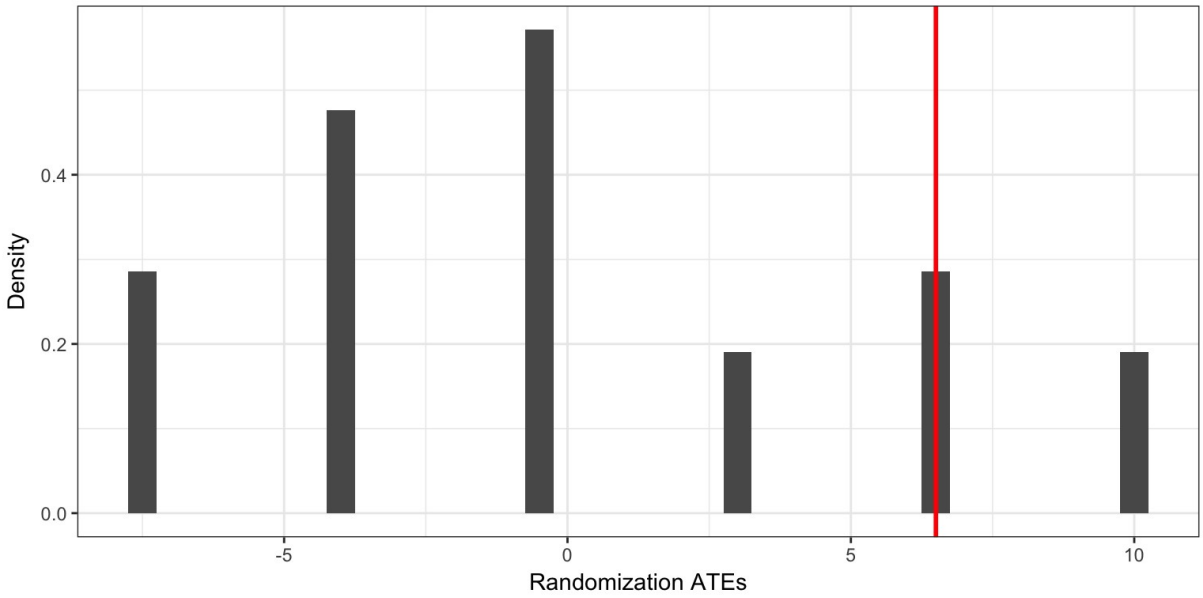
… and then only keep the unique ones

```
fake_treats <- unique(fake_treats)
```

Now we need to calculate the ATE for each of these possible randomizations. For that, we will need a loop

```
rand_ate <- NA # placeholder vector for results

for (i in 1:nrow(fake_treats)){ # for each of the fake treatment vectors

  mean_treat <- mean(gg_data$observed[fake_treats[i,]==1])

  mean_control <- mean(gg_data$observed[fake_treats[i,]==0])

  # calculating ATE for this randomization
  rand_ate[i] <- mean_treat - mean_control

}
```

Now we can plot the distribution of the randomization ATEs



Distribution of randomization ATEs

And we can get the p-value

```
# One tailed
sum(rand_ate>=ATE)/length(rand_ate)
```

```
## [1] 0.2380952
```

```
# Two tailed
sum(abs(rand_ate)>=ATE)/length(rand_ate)
```

```
## [1] 0.3809524
```