1. When conducting the work of Lab 11, we conducted the test that uses the Central Limit Theorem even though the sample size was "small" (i.e., $n < 30$). It turns out, that how "far off" the t-test is can be computed using a first-order Edgeworth approximation for the error. Below, we will do this for the the further observations.

   (a) Boos and Hughes-Oliver (2000) note that

$$P(T \leq t) \approx F_Z(t) + \underbrace{\frac{\text{skew}}{\sqrt{n}} \frac{(2t^2+1)}{6} f_Z(t)}_{\text{error}},$$

   where $f_Z(\cdot)$ and $F_Z(\cdot)$ are the Gaussian PDF and CDF and skew is the skewness of the data. What is the potential error in the computation of the p-value when testing $H_0 : \mu_X = 0; H_a : \mu_X < 0$ using the zebra finch further data?

```
#read the data file
dat.finches <- read_csv("zebrafinches.csv")
#separate further column
dat.further <- dat.finches$further

mu0 = 0
#compute t-statistics to get the t-value for the further data
t_furth <- t.test(dat.further, mu = mu0, alternative = "less")
t <- t_furth$statistic #extract t
n <- length(dat.further) #get n

#calculate skewness of data
skew <- skewness(dat.further)

#use Gaussian pdf and calculate the error
pdf.finches <- dnorm(t, mean = 0, sd = 1)
potential.error <- (skew/sqrt(n))*((2*t^2 +1)/6)*pdf.finches

potential.error

##             t
## -1.226006e-13
```

   The potential error in the computation of the p-value when testing $H_0 : \mu_X = 0; H_a : \mu_X < 0$ using the zebra finch further data is $-1.2260063 \times 10^{-13}$, which is extremely close to 0. Therefore, the t-test results on the zebra finch further data can be trusted even though the sample size is small. The Central Limit Theorem approximation works very well for the data. The error is negative, which means the true p-value is slightly lower than we assumed.

   (b) Compute the error for t statistics from -10 to 10 and plot a line that shows the error across t. Continue to use the skewness and the sample size for the zebra finch further data.

```
#generate a vector of t-values and compute pdf for each of them
t.vals <- seq(from = -10, to = 10, length.out = 1000)
pdf.t.vals <- dnorm(t.vals, mean = 0, sd = 1)

#calculate the error across of all t
t.potential.error <- (skew/sqrt(n))*((2*t.vals^2 +1)/6)*pdf.t.vals

#create a tibble to plot the errors for t-statistics
dat.error.plot <- tibble(t.vals, t.potential.error)

#plot the error for the t-statistics
error.plot <- ggplot(dat.error.plot)+
geom_line(aes(x= t.vals, y = t.potential.error))+ #plot the line for the errors
theme_bw()+
labs(title = "Edgeworth approximation for the error for p-value",
     x = "t-statistics",
     y = "Potential error")
```

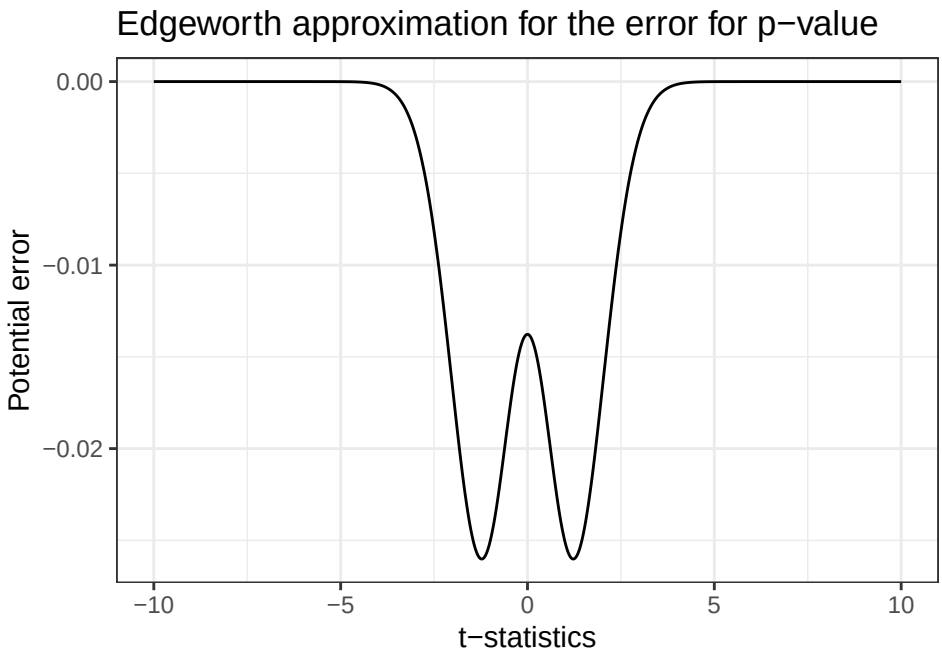

Figure 1: Edgeworth approximation for the error for p-value across t

The potential error in p-value estimation is minimal across all t values, which supports the validity of t-test approximation. As t gets closer to -10 and 10, the potential error is close to 0. The error becomes larger as t approaches 0. However, when t is very close to 0, the error decreases.

(c) Suppose we wanted to have a tail probability within 10% of the desired $\alpha = 0.05$. Recall we did a left-tailed test using the further data. How large of a sample size would we need? That is, we need to solve the error formula equal to 10% of the desired left-tail probability:

$$0.10\alpha \overset{set}{=} \underbrace{\frac{\text{skew}}{\sqrt{n}} \frac{(2t^2+1)}{6} f_Z(t)}_{\text{error}},$$

which yields

$$n = \left(\frac{\text{skew}}{6(0.10\alpha)} (2t^2+1) f_Z(t) \right)^2.$$

```
alpha <- 0.05
#calculate the critical value for the left-tailed test
t.crit <- qnorm(alpha, mean = 0, sd = 1)

#calculate pdf
pdf.sample <- dnorm(t.crit, mean = 0, sd = 1)

#calculate n
n.sample <- (skew/(6*(0.10*alpha))*(2*t.crit^2+1)*pdf.sample)^2
n.sample <- ceiling(n.sample) #round n to the closest positive integer
n.sample

## [1] 521
```

We would need a sample size of $n = 521$ to have a tail probability within 10% of the desired $\alpha = 0.05$. This large sample size compensates for the skewness in the data, controlling the approximation error.

2. Complete the following steps to revisit the analyses from lab 11 using the bootstrap procedure.

(a) Now, consider the zebra finch data. We do not know the generating distributions for the closer, further, and difference data, so perform resampling to approximate the sampling distribution of the T statistic:

$$T = \frac{\bar{x}_r - 0}{s/\sqrt{n}},$$

where $\bar{x}_r$ is the mean computed on the rth resample and s is the sample standard deviation from the original samples. At the end, create an object called `resamples.null.closer`, for example, and store the resamples shifted to ensure they are consistent with the null hypotheses at the average (i.e., here ensure the shifted resamples are 0 on average, corresponding to $t = 0$, for each case).

```
n.resamples <- 10000
mu0 <- 0

#separate closer and difference data
dat.closer <- dat.finches$closer
dat.diff <- dat.finches$diff

#get standard deviation
dat.closer.sd <- sd(dat.closer)
dat.further.sd <- sd(dat.further)
dat.diff.sd <- sd(dat.diff)

#store the approximation of T-statistics
t.stat.storage <- tibble(closer = rep(NA, n.resamples),
                         further = rep(NA, n.resamples),
                         diff = rep(NA, n.resamples))

#perform resampling and calculate T-statistics
for (i in 1:n.resamples){
#resample
resample.closer <- sample(dat.closer, size = n, replace = T)
resample.further <- sample(dat.further, size = n, replace = T)
resample.diff <- sample(dat.diff, size = n, replace = T)
#calculate T
t.stat.closer <- mean(resample.closer)/(dat.closer.sd/sqrt(n))
t.stat.further <- mean(resample.further)/(dat.further.sd/sqrt(n))
t.stat.diff <- mean(resample.diff)/(dat.diff.sd/sqrt(n))
#store the statistics
t.stat.storage$closer[i] = t.stat.closer
t.stat.storage$further[i] = t.stat.further
t.stat.storage$diff[i] = t.stat.diff
}

#store the shifted resamples in an object and shift the mean of the data to be 0 under null hypothesis
resamples.null.closer <- t.stat.storage$closer - mean(t.stat.storage$closer) + mu0
resamples.null.further <- t.stat.storage$further - mean(t.stat.storage$further) + mu0
resamples.null.diff <- t.stat.storage$diff - mean(t.stat.storage$diff) + mu0

#calculate the mean of resamples - should be 0 on average
mean.resample.closer <- mean(resamples.null.closer)
mean.resample.further <- mean(resamples.null.further)
mean.resample.diff <- mean(resamples.null.diff)

mean.resample.closer

## [1] -5.324852e-16

mean.resample.further

## [1] -1.477152e-16

mean.resample.diff

## [1] 2.280398e-16
```

We performed 10,000 resamples with replacement and computed t-statistics for each resample. Then, we shifted the resamples to make sure that they reflect the null hypothesis. We verified that the shifted resamples were properly centered around 0 by checking that their means were approximately zero: `mean.resample.closer` $= -5.3248517 \times 10^{-16}$, `mean.resample.further` $= -1.4771517 \times 10^{-16}$, and `mean.resample.diff` $= 2.2803981 \times 10^{-16}$.

(b) Compute the bootstrap p-value for each test using the shifted resamples. How do these compare to the t-test p-values?

```r
#calculate t-statistics on the original sample
t.obs.closer <- mean(dat.closer)/(dat.closer.sd/sqrt(n))
t.obs.further <- mean(dat.further)/(dat.further.sd/sqrt(n))
t.obs.diff <- mean(dat.diff)/(dat.diff.sd/sqrt(n))

# p-value: the proportion of observations that are at least as supportive of Ha
p.val.closer <- mean(resamples.null.closer >= t.obs.closer) #right-tailed test
p.val.further <- mean(resamples.null.further <= t.obs.further) #left-tailed test
p.val.diff <- mean(abs(resamples.null.diff) >= abs(t.obs.diff)) #two-tailed test

#compare with the t-test p-values
t.test.closer <- t.test(dat.closer, mu = 0, alternative = "greater")
t.test.further <- t.test(dat.further, mu = 0, alternative = "less")
t.test.diff <- t.test(dat.diff, mu = 0, alternative = "two.sided")

#extract their p-values
t.test.closer.p <- t.test.closer$p.value
t.test.further.p <- t.test.further$p.value
t.test.diff.p <- t.test.diff$p.value

#create a comparison table
comparison.table <-tibble(
"Case" = c("Closer", "Further", "Difference"),
"Bootstrap" = c(p.val.closer, p.val.further, p.val.diff),
"T-test" = c(t.test.closer.p, t.test.further.p, t.test.diff.p)
)
```

Case	Bootstrap	T-test
Closer	0.00	8.13e-09
Further	0.00	2.59e-08
Difference	0.00	1.04e-08

Table 1: Bootstrap and t-test p-values

The bootstrap p-values for all three cases were estimated as 0, while the corresponding t-test p-values, although extremely small, were nonzero. The difference arises because the t-test p-values are based on theoretical calculations assuming an approximately normal distribution, which can detect very small probabilities. In contrast, the bootstrapping is based on a finite number of resamples, and with our data it did not produce any extreme t-statistics as the theoretical version. We have statistically discernible evidence against the null hypothesis in each case.

(c) What is the 5^{th} percentile of the shifted resamples under the null hypothesis? Note this value approximates $t_{0.05,n-1}$. Compare these values in each case.

```r
#get 5th percentile of the shifted resamples
close.5th <- quantile(resamples.null.closer, probs = 0.05)
further.5th <- quantile(resamples.null.further, probs = 0.05)
diff.5th <- quantile(resamples.null.diff, probs = 0.05)

#calculate theoretical t-distribution value
df <- n-1
#approximate t0.05,n-1
t.critical.5th <- qt(0.05, df)

#comparison table
comparison.table.percentile <-tibble(
"Case" = c("Closer", "Further", "Difference"),
"Bootstrap" = c(close.5th, further.5th, diff.5th),
"Theoretical" = rep(t.critical.5th, 3)
)
```

Case	Bootstrap	Theoretical
Closer	-1.61	-1.71
Further	-1.70	-1.71
Difference	-1.60	-1.71

Table 2: Bootstrap and theoretical 5th percentile of the shifted resamples under the null hypothesis

For the left-tailed hypothesis test, which we use for further data, the critical value for rejecting

the null hypothesis is based on the 5th percentile of the shifted resamples. This value approximates $t_{0.05,n-1}$. Our approximation for $t_{0.05,n-1}$ (-1.7) is very close to the theoretical value -1.71. However, the theoretical value is slightly more extreme than the approximation. Therefore, the bootstrapping approximates the t-distribution well. However, we cannot draw concrete conclusions from comparing the right-tailed hypothesis test (closer data) and two-tailed hypothesis test (difference data) to $t_{0.05,n-1}$, as the value for rejecting for null hypothesis for right-tailed test is based on 95th percentile of the shifted resampes and the value for rejecting the null hypothesis for two-tailed test is based on 2.5th and 97.5 percentiles of the shifted resamples.

(d) Compute the bootstrap confidence intervals using the resamples. How do these compare to the t-test confidence intervals?

```
mean.storage <- tibble(closer = rep(NA, n.resamples),
                       further = rep(NA, n.resamples),
                       diff = rep(NA, n.resamples))

#perform resampling
for (i in 1:n.resamples){
#resample
resample.closer <- sample(dat.closer, size = n, replace = T)
resample.further <- sample(dat.further, size = n, replace = T)
resample.diff <- sample(dat.diff, size = n, replace = T)
#calculate and store the mean
mean.storage$closer[i] = mean(resample.closer)
mean.storage$further[i] = mean(resample.further)
mean.storage$diff[i] = mean(resample.diff)
}
#confidence interval for closer data
ci.lower.closer <- quantile(mean.storage$closer, probs = 0.025)
ci.upper.closer <- quantile(mean.storage$closer, probs = 0.975)

#confidence interval for further data
ci.lower.further <- quantile(mean.storage$further, probs = 0.025)
ci.upper.further <- quantile(mean.storage$further, probs = 0.975)

#confidence interval for difference data
ci.lower.diff <- quantile(mean.storage$diff, probs = 0.025)
ci.upper.diff <- quantile(mean.storage$diff, probs = 0.975)

#compute t-test confidence intervals
t.test.closer <- t.test(dat.closer, mu = 0, alternative = "two.sided")
t.test.further <- t.test(dat.further, mu = 0, alternative = "two.sided")
t.test.diff <- t.test(dat.diff, mu = 0, alternative = "two.sided")

#extract t-test confidence intervals
ci.closer.t.test <- t.test.closer$conf.int
ci.closer.t.test.lower <- ci.closer.t.test[1] #get t-test CI for closer data
ci.closer.t.test.upper <- ci.closer.t.test[2]

ci.further.t.test <- t.test.further$conf.int
ci.further.t.test.lower <- ci.further.t.test[1] #get t-test CI for further data
ci.further.t.test.upper <- ci.further.t.test[2]

ci.diff.t.test <- t.test.diff$conf.int
ci.diff.t.test.lower <- ci.diff.t.test[1] #get t-test CI for diff data
ci.diff.t.test.upper <- ci.diff.t.test[2]

#create a comparison table
ci.comparison.table <- tibble(
"Case" = c("Closer", "Further", "Difference"),
"Bootstrap CI Lower" = c(ci.lower.closer, ci.lower.further, ci.lower.diff),
"T-test CI Lower" = c(ci.closer.t.test.lower, ci.further.t.test.lower, ci.diff.t.test.lower),
"Bootstrap CI Upper" = c(ci.upper.closer, ci.upper.further, ci.upper.diff),
"T-test CI Upper" = c(ci.closer.t.test.upper, ci.further.t.test.upper, ci.diff.t.test.upper)
)
```

Case	Bootstrap CI Lower	T-test CI Lower	Bootstrap CI Upper	T-test CI Upper
Closer	0.12	0.12	0.19	0.20
Further	-0.26	-0.26	-0.15	-0.15
Difference	0.28	0.27	0.44	0.45

Table 3: Bootstrap and t-test confidence intervals

Overall, the bootstrap and t-test confidence intervals are very similar in all three cases. The differences between the lower and upper bounds produced by the two methods are minimal, with discrepancies of approximately only 0.01.

3. Complete the following steps to revisit the analyses from lab 11 using the randomization procedure.

   (a) Now, consider the zebra finch data. We do not know the generating distributions for the closer, further, and difference data, so perform the randomization procedure

```
R <- 10000
rand <- tibble(t.stat.closer = rep(NA, R), #place to store statistics
               t.stat.further = rep(NA, R),
               t.stat.diff = rep(NA, R))

#shift the data to be mean 0 under H0
closer.shifted <- dat.closer - mu0
further.shifted <- dat.further - mu0
diff.shifted <- dat.diff - mu0

#Randomize / Shuffle
for(i in 1:R){
  #randomize data
  curr.rand.closer <- closer.shifted *
    sample(x = c(-1, 1),
           size = n,
           replace = T)
  curr.rand.further <- further.shifted *
    sample(x = c(-1, 1),
           size = n,
           replace = T)
  curr.rand.diff <- diff.shifted *
    sample(x = c(-1, 1),
           size = n,
           replace = T)
  #calculate and store t-statistics
  rand$t.stat.closer[i] <- mean(curr.rand.closer)/(dat.closer.sd/sqrt(n))
  rand$t.stat.further[i] <- mean(curr.rand.further)/(dat.further.sd/sqrt(n))
  rand$t.stat.diff[i] <- mean(curr.rand.diff)/(dat.diff.sd/sqrt(n))
}
#shift randomized statistics back
rand <- rand |>
  mutate(t.stat.closer = t.stat.closer + mu0)|>
  mutate(t.stat.further = t.stat.further + mu0)|>
  mutate(t.stat.diff = t.stat.diff + mu0)
```

   We performed a non-parametric randomization test. It allows us to approximate the null distribution of the t-statistics based purely on the observed data, without relying on normality. First, we shifted each dataset (closer, further, and difference) so that their means aligned with the null hypothesis. Then, we multiplied each data point by either +1 or -1. For each randomized sample, we computed the t-statistic and repeated this process 10,000 times to approximate the null distribution without assuming normality.

   (b) Compute the randomization test p-value for each test.

```
#calculate deltas
delta.closer <- abs(mean(dat.closer) - mu0)/(dat.closer.sd/sqrt(n))
delta.further <- abs(mean(dat.further) - mu0)/(dat.further.sd/sqrt(n))
delta.diff <- abs(mean(dat.diff) - mu0)/(dat.diff.sd/sqrt(n))

#calculate p-value for closer data
high.closer <- mu0 + delta.closer
p.close.rand <- mean(rand$t.stat.closer >= high.closer)

#calculate p-value for further data
low.further <- mu0 - delta.further
p.further.rand <- mean(rand$t.stat.further <= low.further)

#calculate p-value for difference data
low.diff <- mu0 - delta.further
high.diff <- mu0 + delta.further
p.diff.rand <- mean(rand$t.stat.diff <= low.diff) + mean(rand$t.stat.diff >= high.diff)

p.close.rand

## [1] 0
```

```
p.further.rand

## [1] 0

p.diff.rand

## [1] 0
```

In each scenario, the p-values are 0, indicating that none of the randomized t-statistics were as extreme as theoretical statistics. This provides statistically discernible evidence against the null hypothesis in each case.

(c) Compute the randomization confidence interval by iterating over values of μ_0.
Hint: You can "search" for the lower bound from Q_1 and subtracting by 0.0001, and the upper bound using Q_3 and increasing by 0.0001. You will continue until you find the first value for which the two-sided p-value is greater than or equal to 0.05.

```
#confidence interval for closer data
R <- 1000
mu0.iterate <- 0.0001
starting.point <- mean(dat.closer)

#get the lower bound
mu.lower <- starting.point
repeat{
rand <- tibble(xbar = rep(NA, R))

# PREPROCESSING: shift the data to be mean 0 under H0
dat.shift <- dat.closer - mu.lower
# RANDOMIZE / SHUFFLE
for(i in 1:R){
  curr.rand <- dat.shift *
    sample(x = c(-1, 1),
           size = n,
           replace = T)

  rand$xbar[i] <- mean(curr.rand)
}
rand <- rand |>
  mutate(xbar = xbar + mu.lower) # shifting back

# p-value
delta <- abs(mean(dat.closer) - mu.lower)
high<- mu.lower + delta
p.val <- mean(rand$xbar >= high)

if(p.val < 0.05){
  break
}else{
  mu.lower <- mu.lower - mu0.iterate
}
}
#get the upper bound
mu.upper <- mean(dat.closer)
repeat{
rand <- tibble(xbar = rep(NA, R))

# PREPROCESSING: shift the data to be mean 0 under H0
dat.shift <- dat.closer - mu.upper
# RANDOMIZE / SHUFFLE
for(i in 1:R){
  curr.rand <- dat.shift *
    sample(x = c(-1, 1),
           size = n,
           replace = T)

  rand$xbar[i] <- mean(curr.rand)
}
rand <- rand |>
  mutate(xbar = xbar + mu.upper) # shifting back

# p-value
delta <- abs(mean(dat.closer) - mu.upper)
high<- mu.upper + delta
p.val <- mean(rand$xbar >= high)

if(p.val < 0.05){
  break
```

```
}else{
  mu.upper <- mu.upper + mu0.iterate
}
}
ci.close.rand <- c(mu.lower, mu.upper)

#confidence interval for further data
#get the lower bound
starting.point <- mean(dat.further)
mu.lower <- starting.point
repeat{
rand <- tibble(xbar = rep(NA, R))

# PREPROCESSING: shift the data to be mean 0 under H0
dat.shift <- dat.further - mu.lower
# RANDOMIZE / SHUFFLE
for(i in 1:R){
  curr.rand <- dat.shift *
    sample(x = c(-1, 1),
           size = n,
           replace = T)

  rand$xbar[i] <- mean(curr.rand)
}
rand <- rand |>
  mutate(xbar = xbar + mu.lower) # shifting back

# p-value
delta <- abs(mean(dat.further) - mu.lower)
low <- mu.lower - delta
p.val <- mean(rand$xbar <= low)

if(p.val < 0.05){
  break
}else{
  mu.lower <- mu.lower - mu0.iterate
}
}
#get the upper bound
mu.upper <- mean(dat.further)
repeat{
rand <- tibble(xbar = rep(NA, R))

# PREPROCESSING: shift the data to be mean 0 under H0
dat.shift <- dat.further - mu.upper
# RANDOMIZE / SHUFFLE
for(i in 1:R){
  curr.rand <- dat.shift *
    sample(x = c(-1, 1),
           size = n,
           replace = T)

  rand$xbar[i] <- mean(curr.rand)
}
rand <- rand |>
  mutate(xbar = xbar + mu.upper) # shifting back

# p-value
delta <- abs(mean(dat.further) - mu.upper)
low <- mu.upper - delta
p.val <- mean(rand$xbar <= low)

if(p.val < 0.05){
  break
}else{
  mu.upper <- mu.upper + mu0.iterate
}
}
ci.further.rand <- c(mu.lower, mu.upper)

#confidence interval for difference data
#get the lower bound
starting.point <- mean(dat.diff)
mu.lower <- starting.point
repeat{
rand <- tibble(xbar = rep(NA, R))

# PREPROCESSING: shift the data to be mean 0 under H0
dat.shift <- dat.diff - mu.lower
# RANDOMIZE / SHUFFLE
```

```r
for(i in 1:R){
  curr.rand <- dat.shift *
    sample(x = c(-1, 1),
           size = n,
           replace = T)

  rand$xbar[i] <- mean(curr.rand)
}
rand <- rand |>
  mutate(xbar = xbar + mu.lower) # shifting back

# p-value
delta <- abs(mean(dat.diff) - mu.lower)
low <- mu.lower - delta
high <- mu.lower + delta
p.val <- mean(rand$xbar <= low) + mean(rand$xbar >= high)

if(p.val < 0.05){
  break
}else{
  mu.lower <- mu.lower - mu0.iterate
}
}
#get the upper bound
mu.upper <- mean(dat.diff)
repeat{
rand <- tibble(xbar = rep(NA, R))

# PREPROCESSING: shift the data to be mean 0 under H0
dat.shift <- dat.diff - mu.upper
# RANDOMIZE / SHUFFLE
for(i in 1:R){
  curr.rand <- dat.shift *
    sample(x = c(-1, 1),
           size = n,
           replace = T)

  rand$xbar[i] <- mean(curr.rand)
}
rand <- rand |>
  mutate(xbar = xbar + mu.upper) # shifting back

# p-value
delta <- abs(mean(dat.diff) - mu.upper)
low <- mu.upper - delta
high <- mu.upper + delta
p.val <- mean(rand$xbar <= low) + mean(rand$xbar >= high)

if(p.val < 0.05){
  break
}else{
  mu.upper <- mu.upper + mu0.iterate
}
}
ci.diff.rand <- c(mu.lower, mu.upper)

ci.close.rand

## [1] 0.1251231 0.1876231

ci.further.rand

## [1] -0.2456244 -0.1613244

ci.diff.rand

## [1] 0.2776475 0.4431475
```

These confidence intervals (close data: 0.1251231, 0.1876231; further data: -0.2456244, -0.1613244; and difference data: 0.2776475, 0.4431475) closely match the confidence intervals obtained using the bootstrap and t-test procedures, suggesting that all three methods provide consistent estimates for these data.

References

Boos, D. D. and Hughes-Oliver, J. M. (2000). How large does n have to be for z and t intervals? *The American Statistician*, 54(2):121–128.