qbs120_ps7_gibran

Gibran Erlangga

10/25/2021

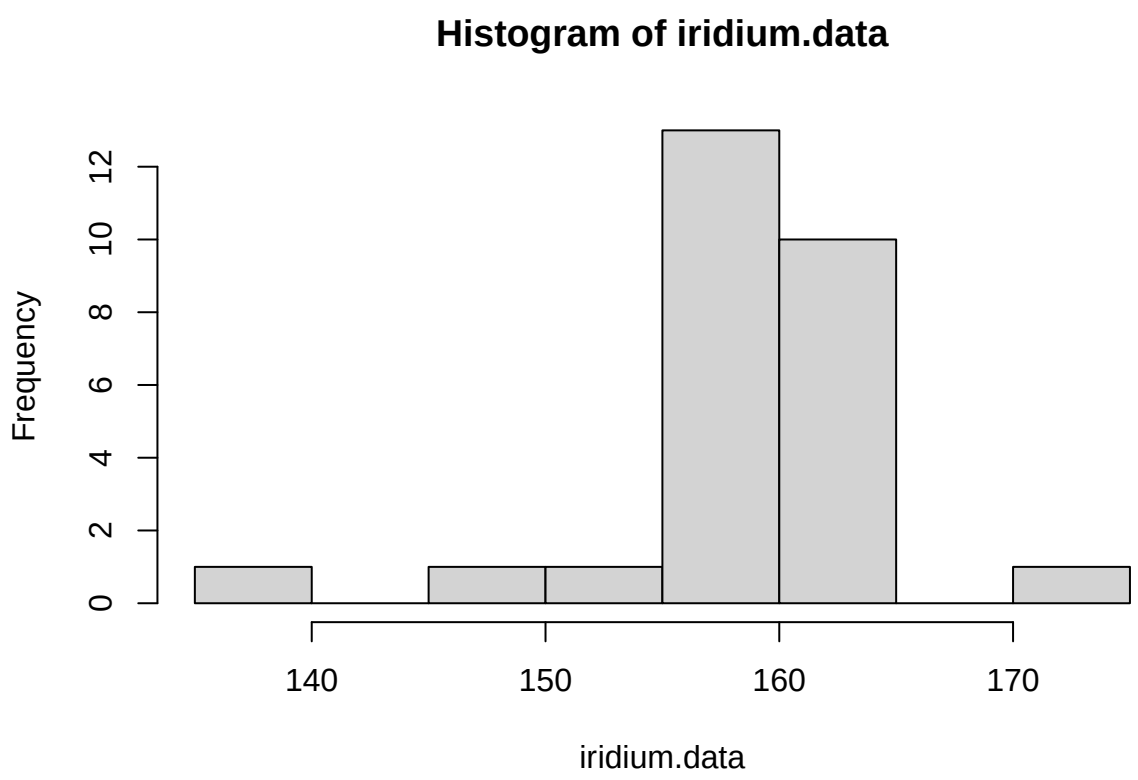
Import Libraries

```
library(ggplot2)
library(sfsmisc)
library(tidyverse)
library(reshape2)
```

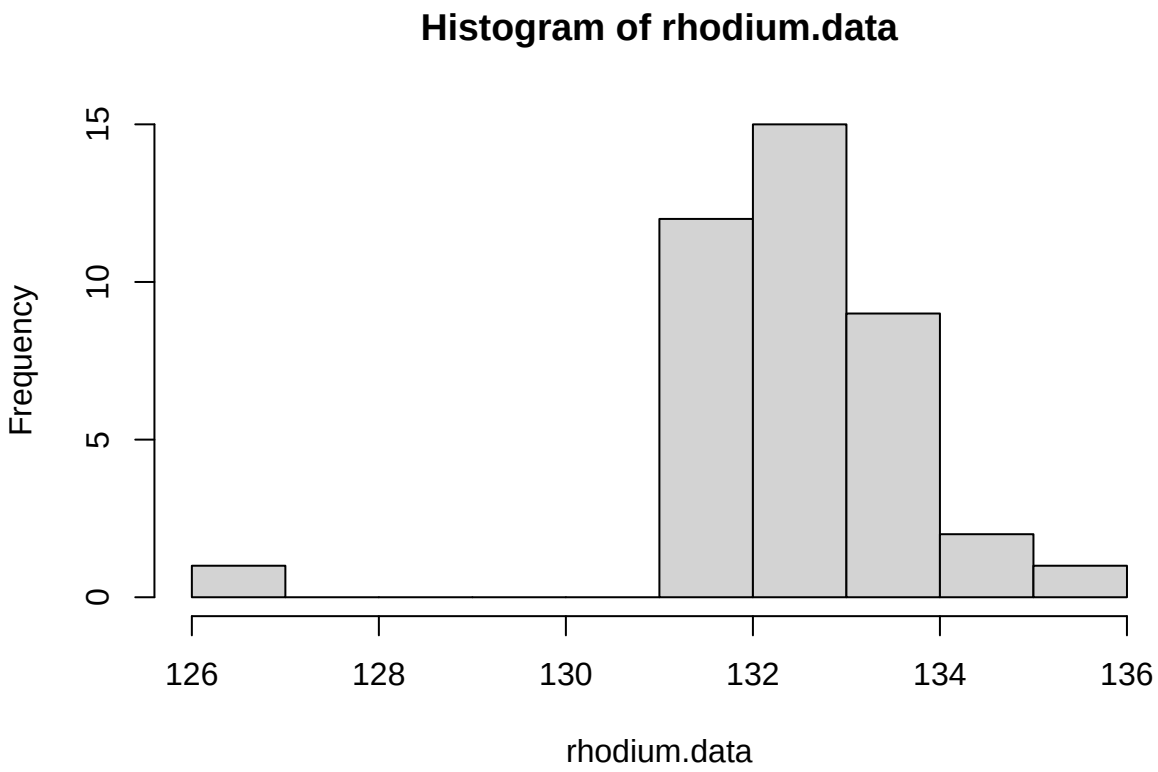
Question 1

(Based on Rice 10.26) Hampson and Walker also made measurements of the heats of sublimation of rhodium and iridium. Do the following calculations for each of the two given sets of data: (a) Plot a histogram of the data.

```
iridium.data = c(136.6, 145.2, 151.5, 162.7, 159.1, 159.8, 160.8, 173.9, 160.1, 160.4,
                 161.1, 160.6, 160.2, 159.5, 160.3, 159.2, 159.3, 159.6, 160.0, 160.2,
                 160.1, 160.0, 159.7, 159.5, 159.5, 159.6, 159.5)

rhodium.data = c(126.4, 135.7, 132.9, 131.5, 131.1, 131.1, 131.9, 132.7, 133.3, 132.5,
                 133.0, 133.0, 132.4, 131.6, 132.6, 132.2, 131.3, 131.2, 132.1, 131.1,
                 131.4, 131.2, 131.1, 131.1, 134.2, 133.8, 133.3, 133.5, 133.4, 133.5,
                 133.0, 132.8, 132.6, 133.3, 133.5, 133.5, 132.3, 132.7, 132.9, 134.1)

hist(iridium.data)
```

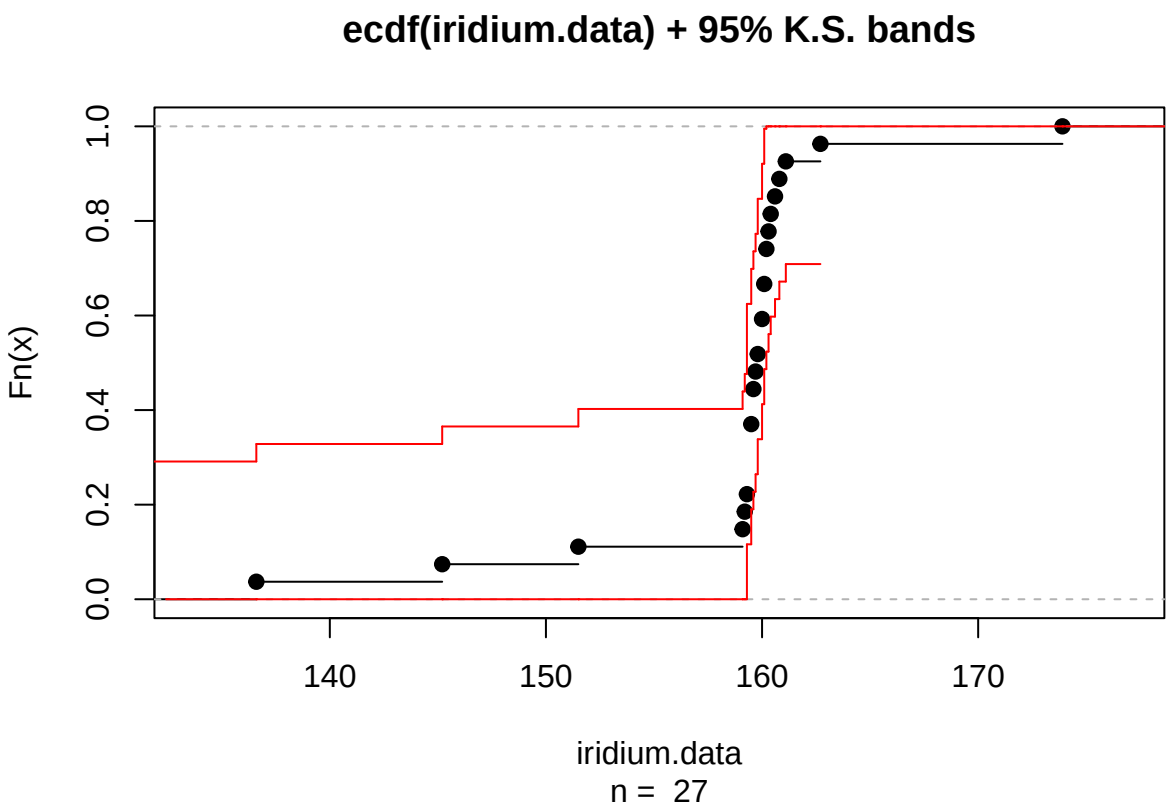



```
hist(rhodium.data)
```

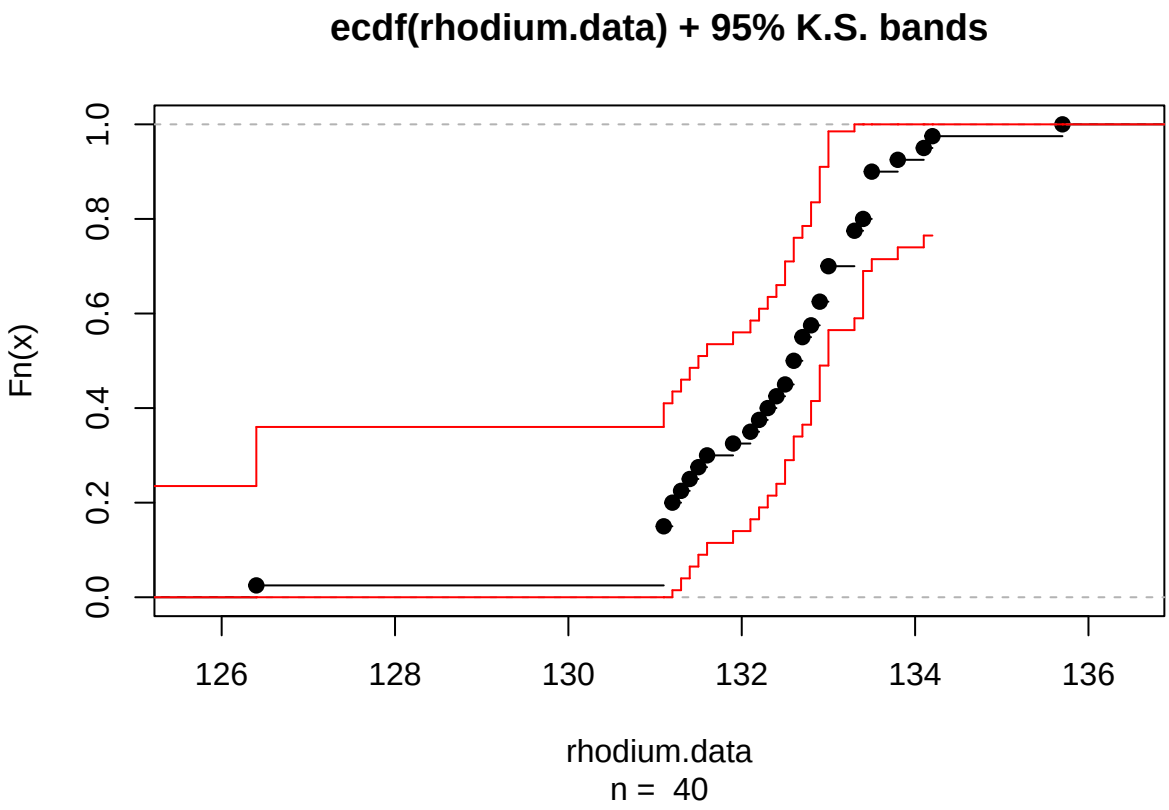


(b) Plot the eCDFs with 95% confidence bands (you may use the ecdf.ksC() function in the R sfsmisc library)

```
ecdf.ksCI(iridium.data)
```

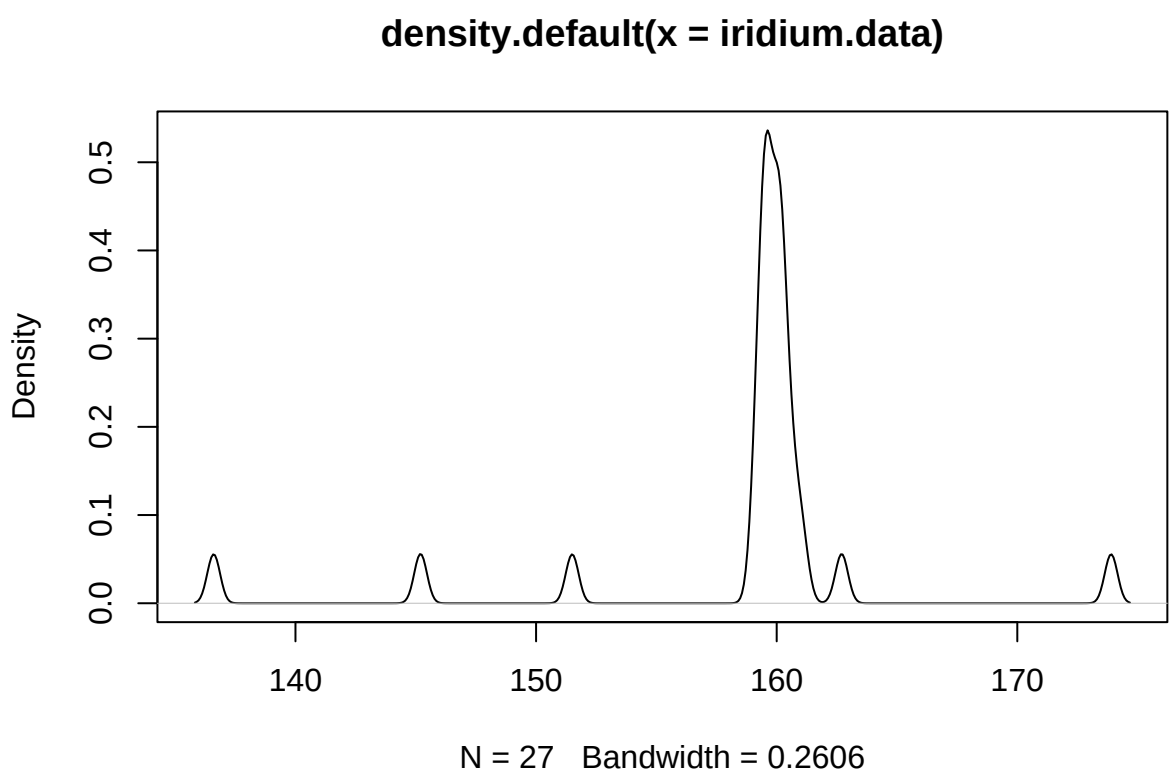



```
ecdf.ksCI(rhodium.data)
```

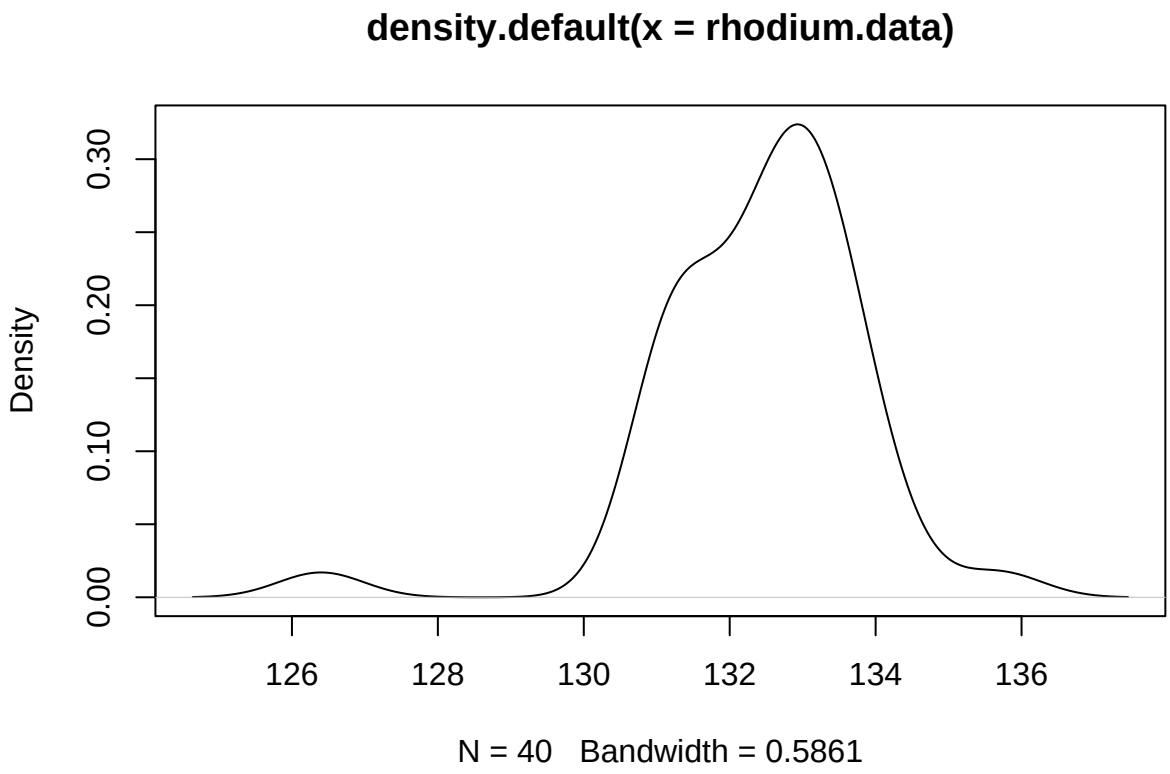


(c) Plot the kernel density estimate.

```
plot(density(iridium.data))
```

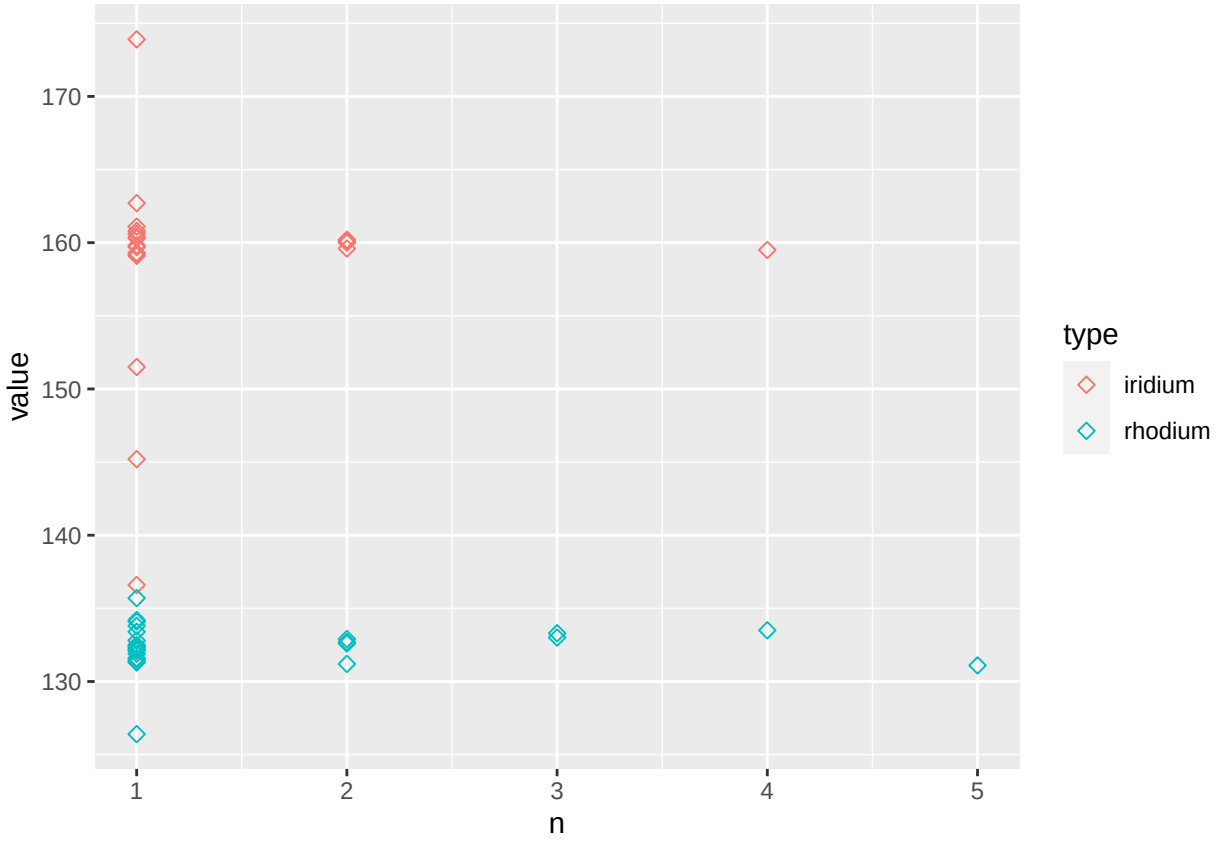



```
plot(density(rhodium.data))
```



(d) Plot the observations in the order of the experiment.

```
iridium_data <- data.frame(type = rep("iridium", length(iridium.data)), value = iridium.data)
rhodium_data <- data.frame(type = rep("rhodium", length(rhodium.data)), value = rhodium.data)

data <- rbind(iridium_data, rhodium_data)

df_plot <- data %>% group_by(type, value) %>% count(value)

df_plot %>% ggplot(aes(x=n, y=value, color=type)) +
  geom_point(size=2, shape=23)
```



(e) Does that statistical model of iid measurement errors seem reasonable? Explain.

From the observations, it seem that the data is iid. From the graph above (answer for d), we can see that there are some outliers in the Iridium dataset. If we see in the tabular format, the outlier seems to happen in the first measurement. This could probably because at first, the researcher assumed that the tool was already calibrated, which turned out it wasn't, and fixed it for the subsequent measurements after the first one.

(f) Find the mean, 10% and 20% trimmed means, and median and compare them.

```
# trimmed mean function
get_trimmed_data <- function(data, percent_trim) {
  data <- sort(data)
  n <- floor(length(data) * percent_trim)
  trimmed_data <- data[(n+1) : (length(data)-n)]
  return(trimmed_data)
}

# calculate results for iridium
iridium_mean <- round(mean(iridium.data), 3)
iridium_t_mean_10 <- round(mean(get_trimmed_data(iridium.data, 0.1)), 3)
iridium_t_mean_20 <- round(mean(get_trimmed_data(iridium.data, 0.2)), 3)
iridium_median <- round(median(iridium.data), 3)

# print results in words
```

```
paste0("For Iridium, mean is ", iridium_mean, "\n",
       "10% trimmed mean is ", iridium_t_mean_10, "\n",
       "10% trimmed mean is ", iridium_t_mean_20, "\n",
       "median is ", iridium_median)
```

```
## [1] "For Iridium, mean is 158.815\n10% trimmed mean is 159.548\n10% trimmed mean is 159.841\nmedian i
```

```
## calculate results for rhodium
rhodium_mean <- round(mean(rhodium.data), 3)
rhodium_t_mean_10 <- round(mean(get_trimmed_data(rhodium.data, 0.1)), 3)
rhodium_t_mean_20 <- round(mean(get_trimmed_data(rhodium.data, 0.2)), 3)
rhodium_median <- round(median(rhodium.data), 3)

# print results in words
paste0("For Rhodium, mean is ", rhodium_mean, "\n",
       "10% trimmed mean is ", rhodium_t_mean_10, "\n",
       "10% trimmed mean is ", rhodium_t_mean_20, "\n",
       "median is ", rhodium_median)
```

```
## [1] "For Rhodium, mean is 132.42\n10% trimmed mean is 132.478\n10% trimmed mean is 132.529\nmedian is
```

(g) Find the standard error of the sample mean and a corresponding 90% confidence interval. Overlay this CI on a density plot.

```
# standard error function
se <- function(x) round(sd(x)/sqrt(length(x)), 5)

# iridium
n <- length(iridium.data)
error <- qt(0.95,df=n-1)*se(iridium.data)
lower_bound <- round(iridium_mean-error, 3)
upper_bound <- round(iridium_mean+error, 3)

paste0("Confidence Interval for the sample mean of iridium: [", lower_bound, ", ", upper_bound, "]")
```

```
## [1] "Confidence Interval for the sample mean of iridium: [156.772, 160.858]"
```

```
# rhodium
n <- length(rhodium.data)
error <- qt(0.95,df=n-1)*se(rhodium.data)
lower_bound <- round(rhodium_mean-error, 3)
upper_bound <- round(rhodium_mean+error, 3)

paste0("Confidence Interval for the sample mean of rhodium: [", lower_bound, ", ", upper_bound, "]")
```

```
## [1] "Confidence Interval for the sample mean of rhodium: [132.037, 132.803]"
```

(h) Use the bootstrap to approximate the sampling distribution of the 10% and 20% trimmed means and median. Plot the kernel density estimates of these bootstrap distributions in a single plot. Compute the standard errors and compare.

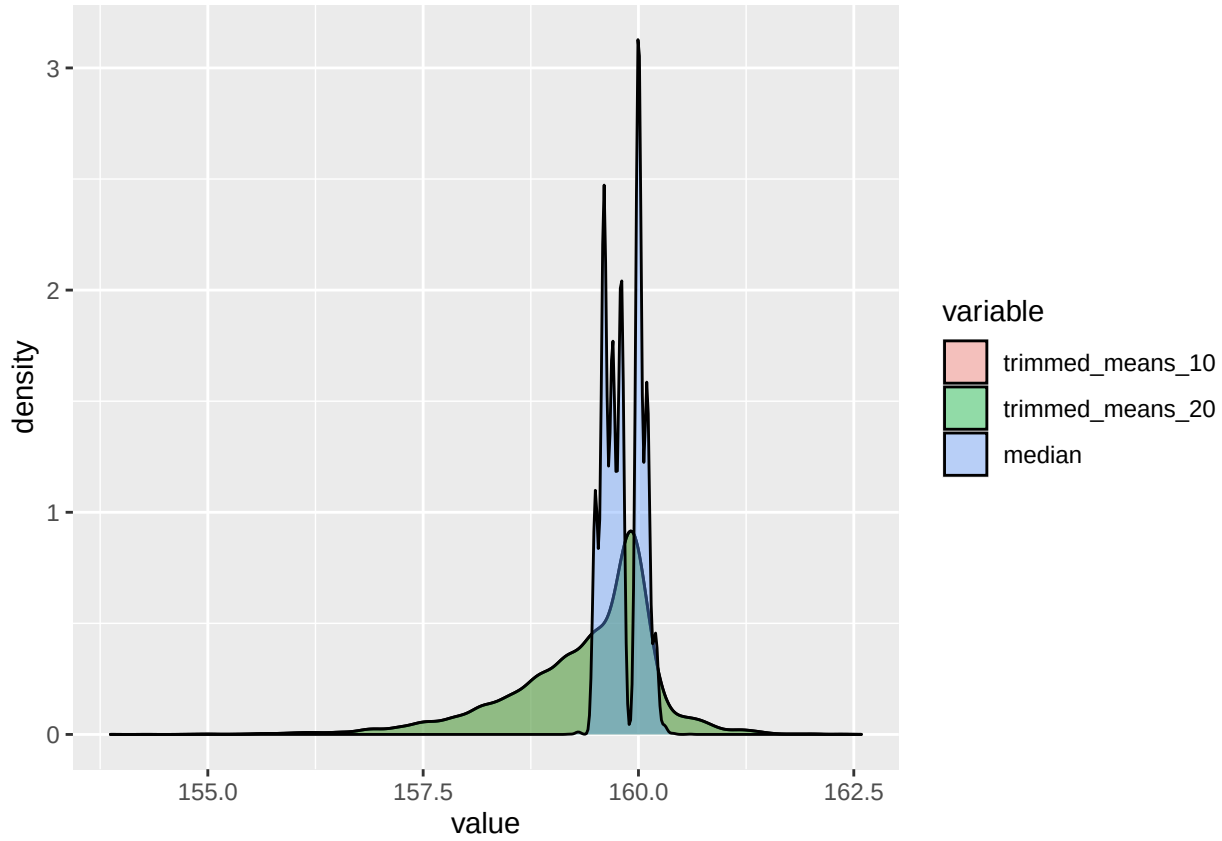
```
get_trimmed_data_plot = function(data, n) {
  len = length(data)
  bootstrap = matrix(sample(data, n*len, replace=T), nrow=n, ncol=len)
  t_means_10 = apply(bootstrap, 1, function(x) {mean(x, trim=0.1)})
  t_means_20 = apply(bootstrap, 1, function(x) {mean(x, trim=0.1)})
  t_median = apply(bootstrap, 1, function(x) {median(x)})
  df = data.frame(trimmed_means_10=t_means_10, trimmed_means_20=t_means_20, median=t_median)
  return(melt(df))
}

iridium_plot <- get_trimmed_data_plot(iridium.data, 10000)
```

```
## No id variables; using all as measure variables
```

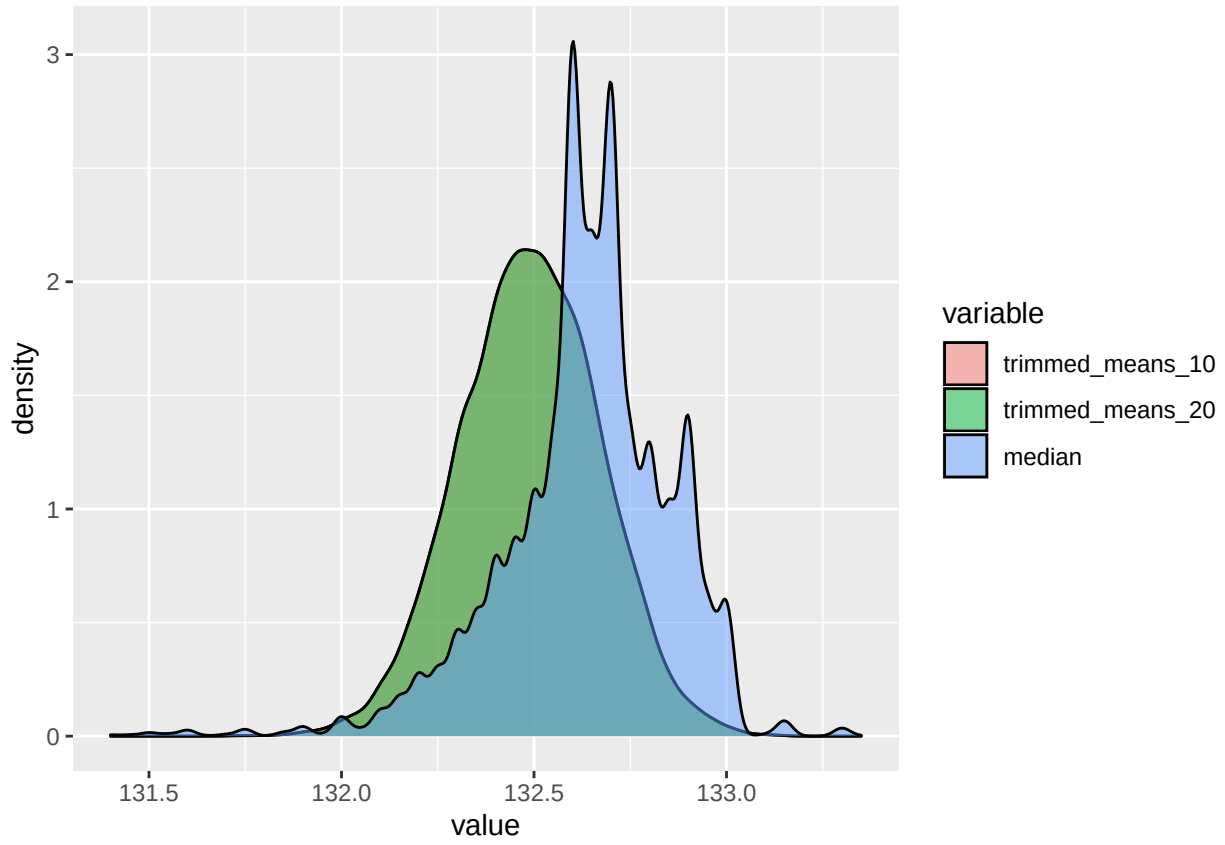
```
rhodium_plot <- get_trimmed_data_plot(rhodium.data, 10000)
```

```
## No id variables; using all as measure variables
```

```
# plot
ggplot(iridium_plot, aes(x=value, fill=variable)) + geom_density(alpha=0.4)
```




```
ggplot(rhodium_plot, aes(x=value, fill=variable)) + geom_density(alpha=0.5)
```




```
irid_10_means <- iridium_plot %>% filter(variable == "trimmed_means_10") %>% select(value)
irid_20_means <- iridium_plot %>% filter(variable == "trimmed_means_20") %>% select(value)
irid_med <- iridium_plot %>% filter(variable == "median") %>% select(value)

rhod_10_means <- rhodium_plot %>% filter(variable == "trimmed_means_10") %>% select(value)
rhod_20_means <- rhodium_plot %>% filter(variable == "trimmed_means_20") %>% select(value)
rhod_med <- rhodium_plot %>% filter(variable == "median") %>% select(value)

se_iridium_10_means <- se(irid_10_means[, 1])
se_iridium_20_means <- se(irid_20_means[, 1])
se_iridium_median <- se(irid_med[, 1])

se_rhodium_10_means <- se(rhod_10_means[, 1])
se_rhodium_20_means <- se(rhod_20_means[, 1])
se_rhodium_median <- se(rhod_med[, 1])

labels <- c("se_iridium_trimmed_means_10", "se_iridium_trimmed_means_20",
            "se_iridium_median",
            "se_rhodium_trimmed_means_10", "se_rhodium_trimmed_means_20",
            "se_rhodium_median")
se_values <- c(se_iridium_10_means, se_iridium_20_means,
               se_iridium_median,
               se_rhodium_10_means, se_rhodium_20_means,
               se_rhodium_median)

data.frame(label=labels, value=se_values)
```

```
##                          label   value
## 1 se_iridium_trimmed_means_10 0.00847
## 2 se_iridium_trimmed_means_20 0.00847
## 3           se_iridium_median 0.00209
## 4 se_rhodium_trimmed_means_10 0.00182
## 5 se_rhodium_trimmed_means_20 0.00182
## 6           se_rhodium_median 0.00217
```

(i) Find approximate 90% CIs based on the trimmed means and median and compare to the intervals for the mean.

```
group = c()
lower_bounds = c()
upper_bounds = c()

## trimmed means
# 10% trim
iridium_t_data_10 <- get_trimmed_data(iridium.data, 0.1)
n <- length(iridium_t_data_10)
error <- qt(0.95,df=n-1)*se(iridium_t_data_10)
lower_bound <- round(iridium_t_mean_10-error, 3)
upper_bound <- round(iridium_t_mean_10+error, 3)

group <- append(group, "10% trimmed mean of iridium")
lower_bounds <- append(lower_bounds, lower_bound)
upper_bounds <- append(upper_bounds, upper_bound)

rhodium_t_data_10 <- get_trimmed_data(rhodium.data, 0.1)
n <- length(rhodium_t_data_10)
error <- qt(0.95,df=n-1)*se(rhodium_t_data_10)
lower_bound <- round(rhodium_t_mean_10-error, 3)
upper_bound <- round(rhodium_t_mean_10+error, 3)

group <- append(group, "10% trimmed mean of rhodium")
lower_bounds <- append(lower_bounds, lower_bound)
upper_bounds <- append(upper_bounds, upper_bound)

# 20% trim
iridium_t_data_20 <- get_trimmed_data(iridium.data, 0.2)
n <- length(iridium_t_data_20)
error <- qt(0.95,df=n-1)*se(iridium_t_data_20)
lower_bound <- round(iridium_t_mean_20-error, 3)
upper_bound <- round(iridium_t_mean_20+error, 3)

group <- append(group, "20% trimmed mean of iridium")
lower_bounds <- append(lower_bounds, lower_bound)
upper_bounds <- append(upper_bounds, upper_bound)

rhodium_t_data_20 <- get_trimmed_data(rhodium.data, 0.2)
n <- length(rhodium_t_data_20)
error <- qt(0.95,df=n-1)*se(rhodium_t_data_20)
```

```r
lower_bound <- round(rhodium_t_mean_20-error, 3)
upper_bound <- round(rhodium_t_mean_20+error, 3)

group <- append(group, "20% trimmed mean of rhodium")
lower_bounds <- append(lower_bounds, lower_bound)
upper_bounds <- append(upper_bounds, upper_bound)

## median
# iridium
n <- length(iridium.data)
error <- qt(0.95,df=n-1)*se(iridium.data)
lower_bound <- round(iridium_median-error, 3)
upper_bound <- round(iridium_median+error, 3)

group <- append(group, "median of iridium")
lower_bounds <- append(lower_bounds, lower_bound)
upper_bounds <- append(upper_bounds, upper_bound)

# rhodium
n <- length(rhodium.data)
error <- qt(0.95,df=n-1)*se(rhodium.data)
lower_bound <- round(rhodium_median-error, 3)
upper_bound <- round(rhodium_median+error, 3)

group <- append(group, "median of rhodium")
lower_bounds <- append(lower_bounds, lower_bound)
upper_bounds <- append(upper_bounds, upper_bound)

data.frame(group, lower_bounds, upper_bounds)
```

```
##                         group lower_bounds upper_bounds
## 1 10% trimmed mean of iridium      158.894      160.202
## 2 10% trimmed mean of rhodium      132.235      132.721
## 3 20% trimmed mean of iridium      159.697      159.985
## 4 20% trimmed mean of rhodium      132.309      132.749
## 5           median of iridium      157.757      161.843
## 6           median of rhodium      132.267      133.033
```

Question 2

(Based on Rice 11.21) A study was done to compare the performances of engine bearings made of different compounds. Ten bearings of each type were tested. The following table gives the times until failure (in millions of cycles):

```r
type.I.failure.times = c(3.03, 5.53, 5.6, 9.3, 9.92, 12.51, 12.95, 15.21, 16.04, 16.84)
type.II.failure.times = c(3.19, 4.26, 4.47, 4.53, 4.67, 4.69, 12.78, 6.79, 9.37, 12.75)

data.frame(type.I.failure.times, type.II.failure.times)
```

```
##    type.I.failure.times type.II.failure.times
## 1                  3.03                 3.19
## 2                  5.53                 4.26
```

```
## 3                  5.60                  4.47
## 4                  9.30                  4.53
## 5                  9.92                  4.67
## 6                 12.51                  4.69
## 7                 12.95                 12.78
## 8                 15.21                  6.79
## 9                 16.04                  9.37
## 10                16.84                 12.75
```

(a) Use normal theory to test the hypothesis that there is no difference between the type types of bearings (you can use pt() but not t.test()).

```
type_1_mean <- mean(type.I.failure.times)
type_2_mean <- mean(type.II.failure.times)
mean_diff <- type_1_mean - type_2_mean
type_1_var <- var(type.I.failure.times)
type_2_var <- var(type.II.failure.times)

type_1_length <- length(type.I.failure.times)
type_2_length <- length(type.II.failure.times)
df <- type_1_length + type_2_length - 2

var_p <- ((type_1_length-1)*type_1_var + (type_2_length-1)*type_2_var) / df
test_statistic <- mean_diff / (sqrt(var_p)*sqrt((1/type_1_length)+(1/type_2_length)))
rejection_threshold <- qt(0.995, df)
p_value <- 2*(1-pt(test_statistic, df))

paste("test statistic: ", test_statistic)
```

```
## [1] "test statistic:  2.07230889085455"
```

```
paste("rejection threshold: ", rejection_threshold)
```

```
## [1] "rejection threshold:  2.87844047273861"
```

```
paste("p-value: ", p_value)
```

```
## [1] "p-value:  0.0528756527405685"
```

(b) Test the same hypothesis using a nonparametric method (use just pnorm() to evaluate using the normal approximation for the rank sum and compare that result to the exact distribution using wilcox.test()).

```
merge <- c(type.I.failure.times, type.II.failure.times)
rank <- rank(merge, ties.method = "average")

type_1_rank <- rank[1:type_1_length]
type_1_total <- sum(type_1_rank)
type_2_rank <- rank[(type_1_length+1):(type_1_length+type_2_length)]
type_2_total <- sum(type_2_rank)

e_r <- (type_2_length*(type_2_length+type_1_length+1))/2
```

```
stdev_r <- sqrt(type_1_length*type_2_length*(type_1_length+type_2_length+1)/12)
z <- (type_2_total-e_r)/stdev_r
p_value <- 2*pnorm(z)
p_value_wilcoxon <- wilcox.test(type.I.failure.times, type.II.failure.times,
                                conf.int = T, exact = T)


paste("calculated p-value is ", round(p_value, 5),
      " and wilcoxon p-value is ", round(p_value_wilcoxon$p.value, 5))
```

```
## [1] "calculated p-value is  0.05878  and wilcoxon p-value is  0.06301"
```

(c) Which of the methods, parametric or nonparametric, do you think is better in this case? Nonparametric, because the resulting p-value is closer to wilcoxon p-value.

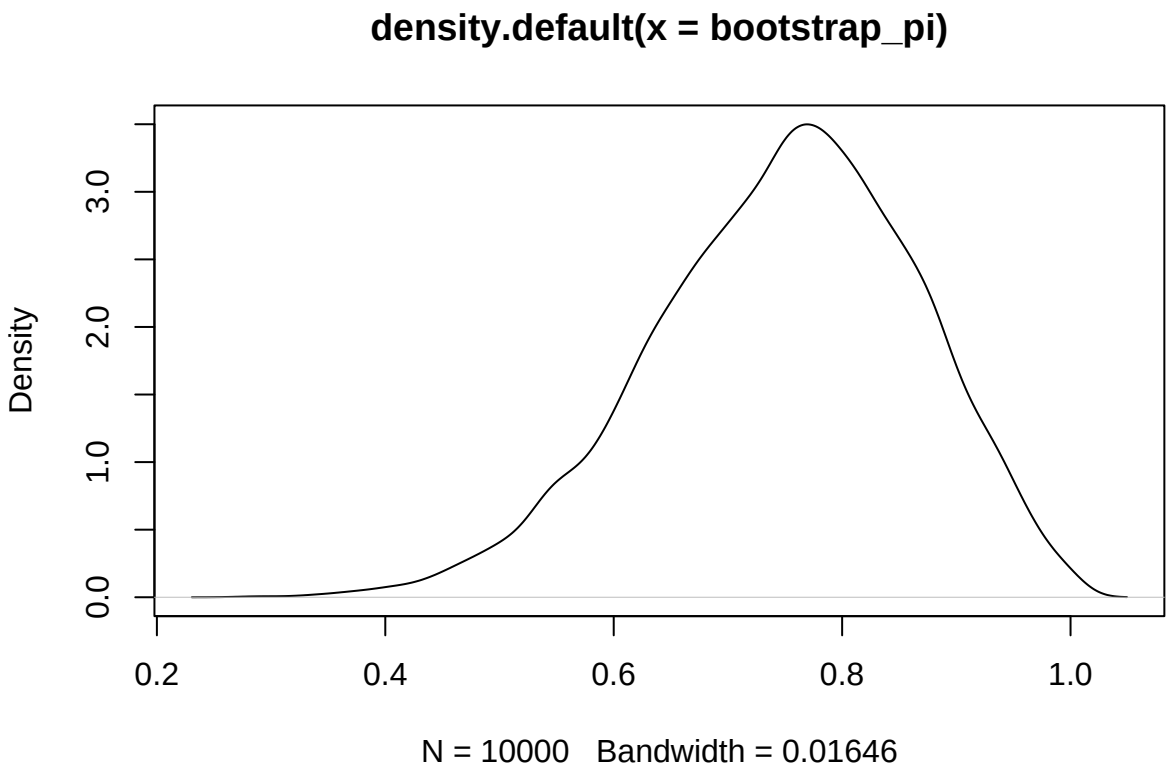
(d) Estimate π, the probability that a type I bearing will outlast a type II bearning?

```
pi <- (1/(type_1_length*type_2_length))*(type_1_total - (type_2_length*(type_2_length+1)/2))
print(pi)
```

```
## [1] 0.75
```

(e) Use the bootstrap to estimate the sampling distribution sof $\hat{\pi}$ and its SE (visualize the bootstrap distribution using both a kernel density plot probability plot relative to normal distribution and comment on the bootstrap distribution.)

```
bootstrap_pi = rep(NA, 10000)

for (i in 1:10000) {
  bootstrap_1 = sample(type.I.failure.times, type_1_length, replace=T)
  bootstrap_2 = sample(type.II.failure.times, type_2_length, replace=T)
  bootstrap_result = matrix(NA, 10, 10)
  for (j in 1:type_1_length) {
    for (k in 1:type_2_length) {
      if (bootstrap_1[j] > bootstrap_2[k]) {
        bootstrap_result[j, k] = 1
      } else {
        bootstrap_result[j, k] = 0
      }
    }
  }
  bootstrap_pi[i] = sum(bootstrap_result)/(type_1_length*type_2_length)
}

plot(density(bootstrap_pi))
```



(f) Use the bootstrap to find an approximate 90% CI for π (compute using both the basic and percentile bootstrap CI methods).

```
upper_bound <- mean(bootstrap_pi) + qnorm(0.95)*sd(bootstrap_pi)
lower_bound <- mean(bootstrap_pi) - qnorm(0.95)*sd(bootstrap_pi)

paste("(", round(lower_bound, 4), ",", round(upper_bound, 4), ")")
```

```
## [1] "( 0.5593 , 0.9389 )"
```

Question 3

(Based on Rice 11.25) Referring to Example A in Section 11.2.1: (a) If the smallest observation for method B is made arbitrarily small, will the t test still reject?
Test statistic is defined as:

$$T = \frac{\bar{X} - \bar{Y}}{s_{\bar{X}-\bar{Y}}}$$

Reducing the smallest observation for method B will decrease the value of $\bar{Y}$, which resulting in an increase the value of the numerator (bigger difference). Similarly, the value of the denominator will also increase as the overall spread of the second sample increases, which also increase the pooled standard deviation, which ultimately increasing the value of the standard error $s_{\bar{X}-\bar{Y}}$.

In conclusion, the t-statistic value will decrease. Therefore, we will not reject the null hypothesis (the test rejects for large value of T).

(b) If the largest observation for method B is made arbitrarily large, will the t test still reject?

Increasing the largest observation for method B will increase the value of $\bar{Y}$, which resulting in a decrease the value of the numerator (smaller difference). On the other hand, the value of the denominator will increase as the overall spread of the second sample increases, which also increase the pooled standard deviation, which ultimately increasing the value of the standard error $s_{\bar{X}-\bar{Y}}$.

In conclusion, the t-statistic value will decrease. Therefore, we will not reject the null hypothesis (the test rejects for large value of T).

(c) Answer the same questions for the Mann-Whitney test. Mann-Whitney test only cares about the rank and not the values. Therefore, for the first question, it will not change the conclusion of the test as reducing the value will not change the rank. For second question, increasing the value of the largest observation for method B will make the rank goes up to 21, which will change the sum of the rank of method B to 56.5. Referencing appendix B in rice book, z value for 2-sided test with $\alpha = 60$.

Therefore, we can reject the null hypothesis as the value of z is higher than the new t-statistic.

Question 4

(Based on Rice 11.36) Lin, Sutton and Qurashi compared microbiological and hydroxylamine methods for the analysis of ampicillin dosages. In one series of experiments, pairs of tablets were analyzed by the two methods. The data in the following table give the percentages of the claimed amount of ampicillin found by the two methods in several pairs of tablets.

```
data = data.frame(micro=c(97.2, 105.8, 99.5, 100, 93.8, 79.2, 72,
                          72, 69.5, 20.5, 95.2, 90.8, 96.2, 96.2, 91),
                  hydro=c(97.2, 97.8, 96.2, 101.8, 88, 74, 75, 67.5, 65.8,
                          21.2, 94.8, 95.8, 98, 99, 100.2))
data
```

```
##    micro hydro
## 1   97.2  97.2
## 2  105.8  97.8
## 3   99.5  96.2
## 4  100.0 101.8
## 5   93.8  88.0
## 6   79.2  74.0
## 7   72.0  75.0
## 8   72.0  67.5
## 9   69.5  65.8
## 10  20.5  21.2
## 11  95.2  94.8
## 12  90.8  95.8
## 13  96.2  98.0
## 14  96.2  99.0
## 15  91.0 100.2
```

(a) What are $X^- - Y^-$ and $sX^- - Y^-$?

$\bar{X} - \bar{Y}$ is the difference of the sample mean, and $s_{\bar{X}-\bar{Y}}$ is the estimated standard error of the sample mean differences, which can be obtained through:

```r
data['difference'] <- data['micro'] - data['hydro']

diff_mean <- mean(data$difference)
sd_diff <- sd(data$difference)
se_diff <- sd_diff / sqrt(length(data$difference))
paste0("sample mean of differences: ", diff_mean)
```

```
## [1] "sample mean of differences: 0.440000000000001"
```

```r
paste0("standard error of differences: ", se_diff)
```

```
## [1] "standard error of differences: 1.19565881421081"
```

(b) If the pairing had been erroneously ignored and it had been assumed that the two samples were independent, what would have been the estimate of the SD of $X^- - Y^-$?

```r
var_micro <- var(data$micro)
var_hydro <- var(data$hydro)

se_diff <- sqrt((var_micro+var_micro)/length(data$micro))
paste0("standard error of differences: ", se_diff)
```

```
## [1] "standard error of differences: 7.73972806216023"
```

(c) Analyze the data to determine if there is a systematic difference between the two methods.
It is a two-sided t-test. We want to check:

$$H_0 : \mu_{micro} - \mu_{hydro} = 0$$

$$H_A : \mu_{micro} - \mu_{hydro} \neq 0$$

with $\alpha = 0.05$. Test statistic is defined as:

$$T = \frac{\bar{D}}{s_{\bar{D}}} = \frac{0.44}{1.195} = 0.368$$

Referencing Appendix B Percentiles of the t Distribution with degree of freedom (df) = n - 1 = 15 - 1 = 14 and T = 0.368, we get p-value to be:

```r
(p_val <- 2*(1-pt(0.368, df=14)))
```

```
## [1] 0.7183775
```

Since the p-value is more than predefined α value, we cannot reject the null hypothesis and therefore there is no difference between the two methods.