Exercise 4 - Sample distribution and Central Limit Theorem

12433732 - Stefan Merdian

2024-11-16

Task 1

1)

How many possible bootstrap samples are there, if each bootstrap sample has the same size as the original?

- For our dataset of 12 data points, each bootstrap sample also has 12 values, and we can pick the same point more than once. For each of the 12 values in a bootstrap sample, we have 12 choices. So, the number of possible bootstrap samples is 12^12, which is a huge number!

2)

```
sample_data = c(4.94, 5.06, 4.53, 5.07, 4.99, 5.16, 4.38, 4.43, 4.93, 4.72, 4.92, 4.96)
mean <- mean(sample_data)
median <- median(sample_data)

cat('Mean of the sample data: ',mean, '\n')
```

```
## Mean of the sample data:  4.840833
```

```
cat('Median of the sample data:',median)
```

```
## Median of the sample data: 4.935
```

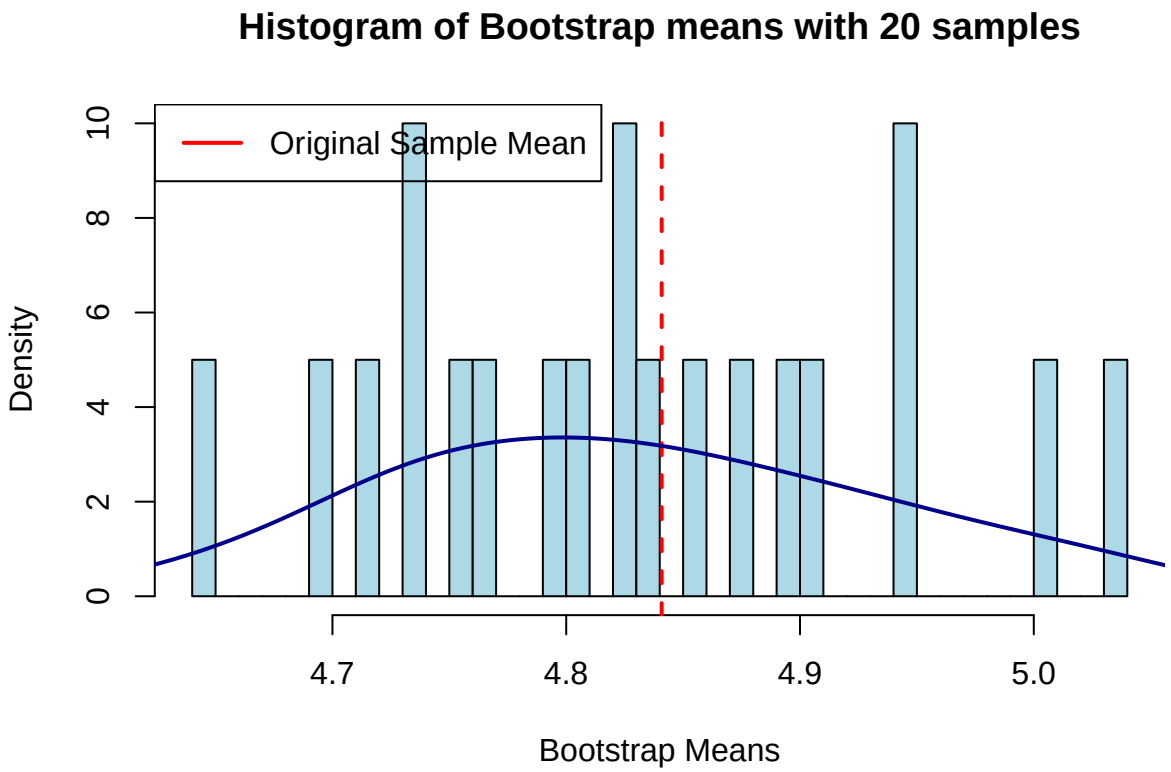
3)

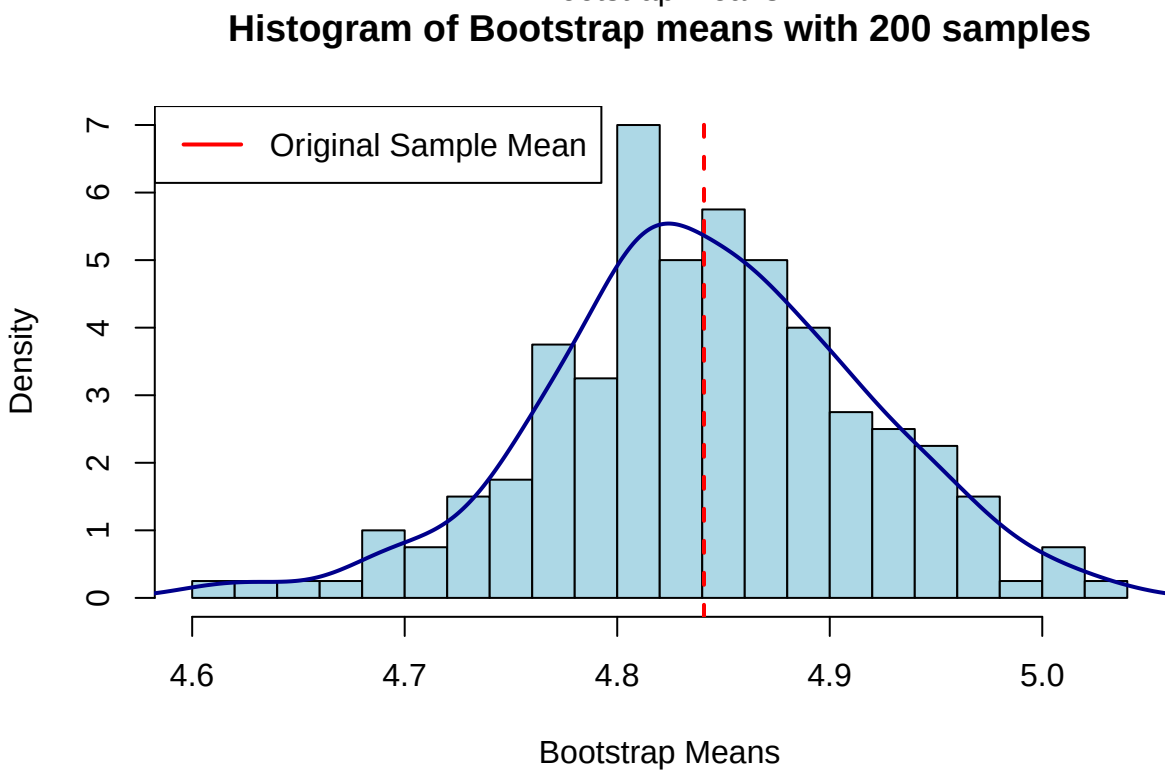
```
set.seed(12433732)
n_bootstrap <- 2000
bootstrap_means <- numeric(n_bootstrap)
bootstrap_samples <- vector("list", n_bootstrap)

for (i in 1:n_bootstrap) {
  bootstrap_sample <- sample(sample_data, size = length(sample_data), replace = TRUE)
  bootstrap_samples[[i]] <- bootstrap_sample
  bootstrap_means[i] <- mean(bootstrap_sample)
```

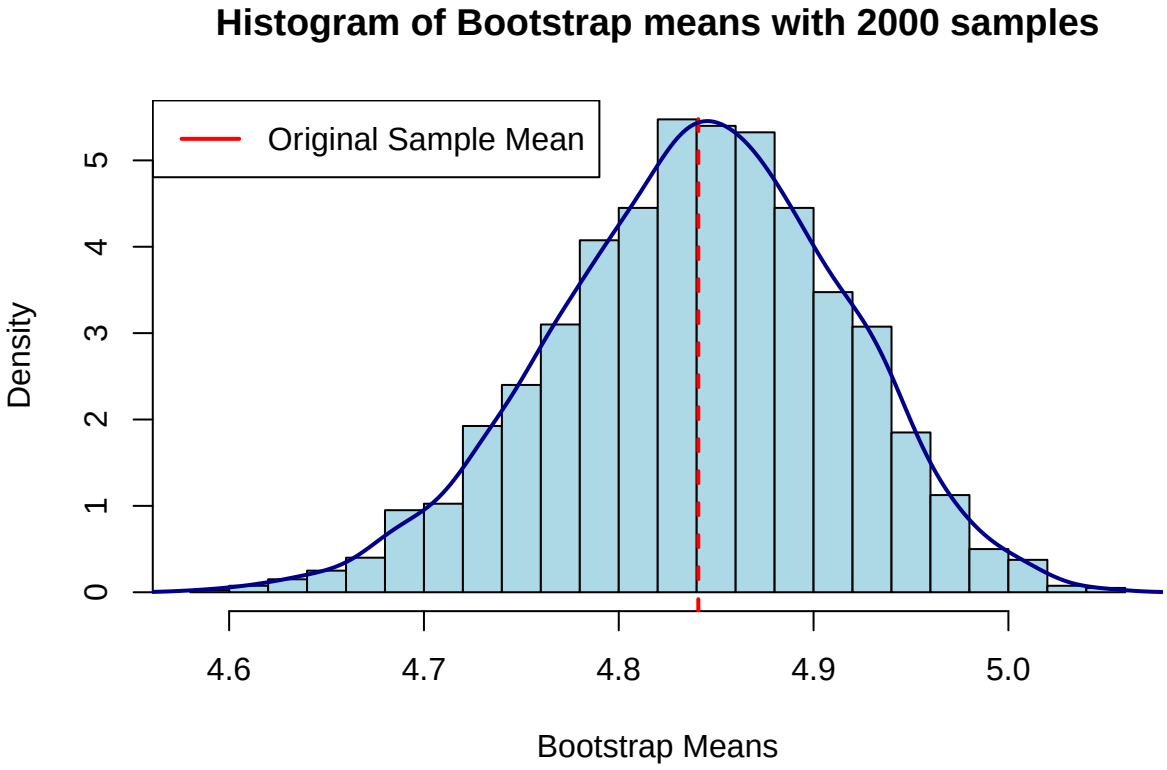
```r
}

bootstrap_samples_list <- list(
  subset_20 = bootstrap_means[1:20],
  subset_200 = bootstrap_means[1:200],
  subset_2000 = bootstrap_means[1:2000]
)

for (i in 1:length(bootstrap_samples_list)) {
  current_sample <- bootstrap_samples_list[[i]]

  cat(paste("Mean of the ", length(current_sample) , " bootstrap means:", mean(current_sample), "\n"))
}
```

```
## Mean of the  20  bootstrap means: 4.829625
## Mean of the  200  bootstrap means: 4.841325
## Mean of the  2000  bootstrap means: 4.84010708333333
```

```r
for (i in 1:length(bootstrap_samples_list)) {
  hist_title <- paste("Histogram of Bootstrap means with", length(bootstrap_samples_list[[i]]), "samples
  hist(
    bootstrap_samples_list[[i]],
    breaks = 30,
    probability = TRUE,
    main = hist_title,
    xlab = "Bootstrap Means",
    col = "lightblue",
    border = "black"
  )

  # original sample mean
  abline(
    v = mean,
    col = "red",
    lwd = 2,
    lty = 2
  )

  # density curve
  lines(density(bootstrap_samples_list[[i]]),
        col = "darkblue",
        lwd = 2)

  legend(
    "topleft",
    legend = c("Original Sample Mean"),
    col = c("red"),
    lwd = 2
  )
}
```


Histogram of Bootstrap means with 20 samples
Original Sample Mean
Density
0
2
4
6
8
10
4.7
4.8
4.9
5.0
Bootstrap Means


Histogram of Bootstrap means with 200 samples
Original Sample Mean
Density
0
1
2
3
4
5
6
7
4.6
4.7
4.8
4.9
5.0
Bootstrap Means



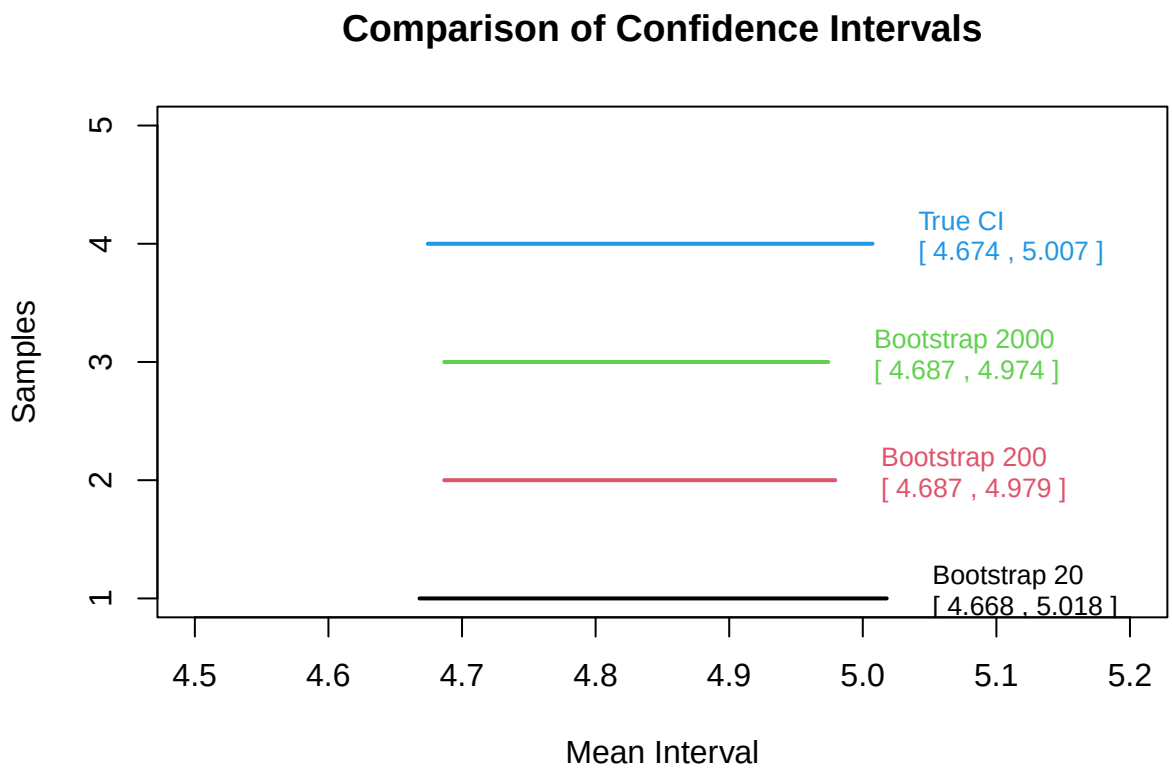
4) Does the Central Limit Theorem kick in?

The Central Limit Theorem says that if we take a lot of samples, the average of those samples will start to look like a **normal distribution** (a bell curve), even if the original data isn't normally distributed. In the first histogram with 20 bootstrap samples, the distribution looks uneven and doesn't have a bell shape. When we increase to 200 samples, the shape starts becoming more like a normal distribution, but it's not quite there yet. By the time we get to 2000 samples, the histogram clearly looks like a bell curve, showing that the Central Limit Theorem kicks in as we take more samples. This tells us that with enough data, the average of our bootstrap means will always approach a normal distribution.

4)

```
ci_list <- list()
# calc interval for 0.025, 0.975 quantiles
for (i in 1:length(bootstrap_samples_list)) {
  current_sample <- bootstrap_samples_list[[i]]

  quantiles <- quantile(current_sample, probs = c(0.025, 0.975))

  ci_list[[i]] <- c(quantiles[1], quantiles[2])
}

# t-test confidence interval for original sample
ci_list[[length(ci_list) + 1]] <- t.test(sample_data, conf.level = 0.95)$conf.int

labels <- c("Bootstrap 20", "Bootstrap 200", "Bootstrap 2000", "True CI")
```

```r
plot(1, type = "n", xlim = c(4.5, 5.2), ylim = c(1, length(ci_list) + 1),
     xlab = "Mean Interval", ylab = "Samples",
     main = "Comparison of Confidence Intervals")

for (i in 1:length(ci_list)) {
  # interval itself for each i
  segments(x0 = ci_list[[i]][1], x1 = ci_list[[i]][2], y0 = i, y1 = i, lwd = 2, col = i)

  # labels
  text(x = ci_list[[i]][2] + 0.02, y = i,
       labels = paste(labels[i], "\n[", round(ci_list[[i]][1], 3), ",", round(ci_list[[i]][2], 3), "]")
       col = i, cex = 0.8, pos = 4)
}
```



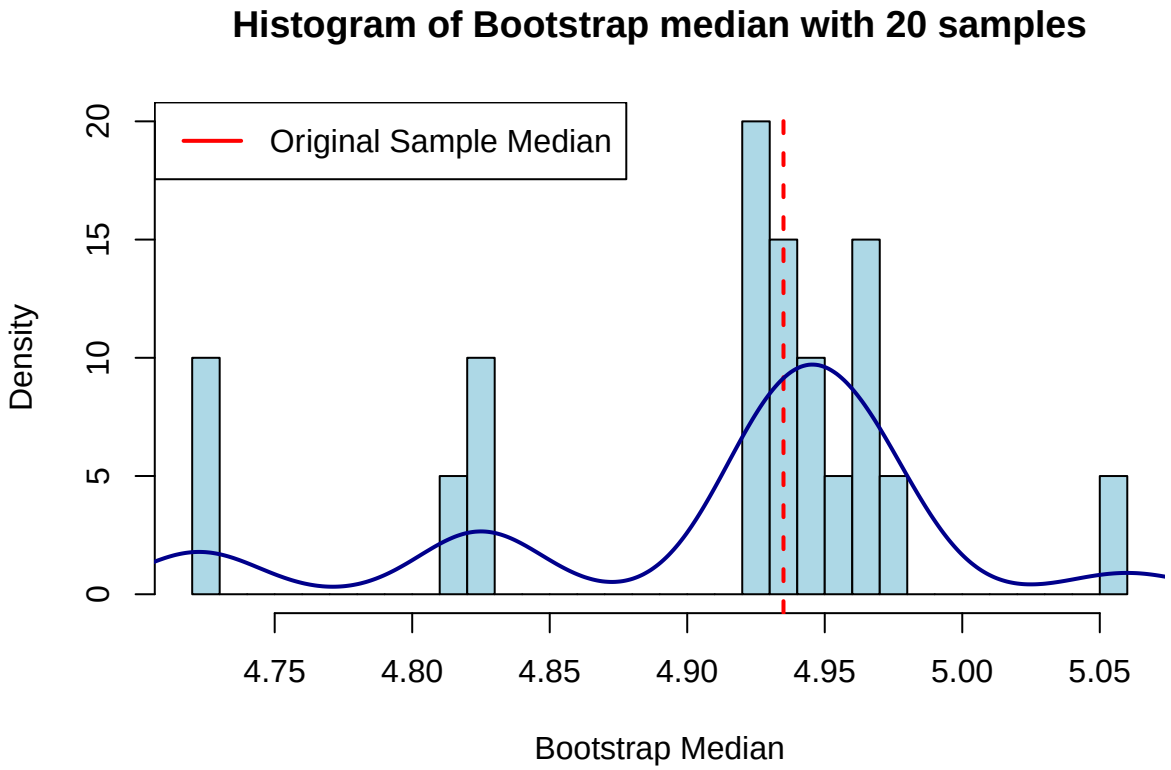
4)

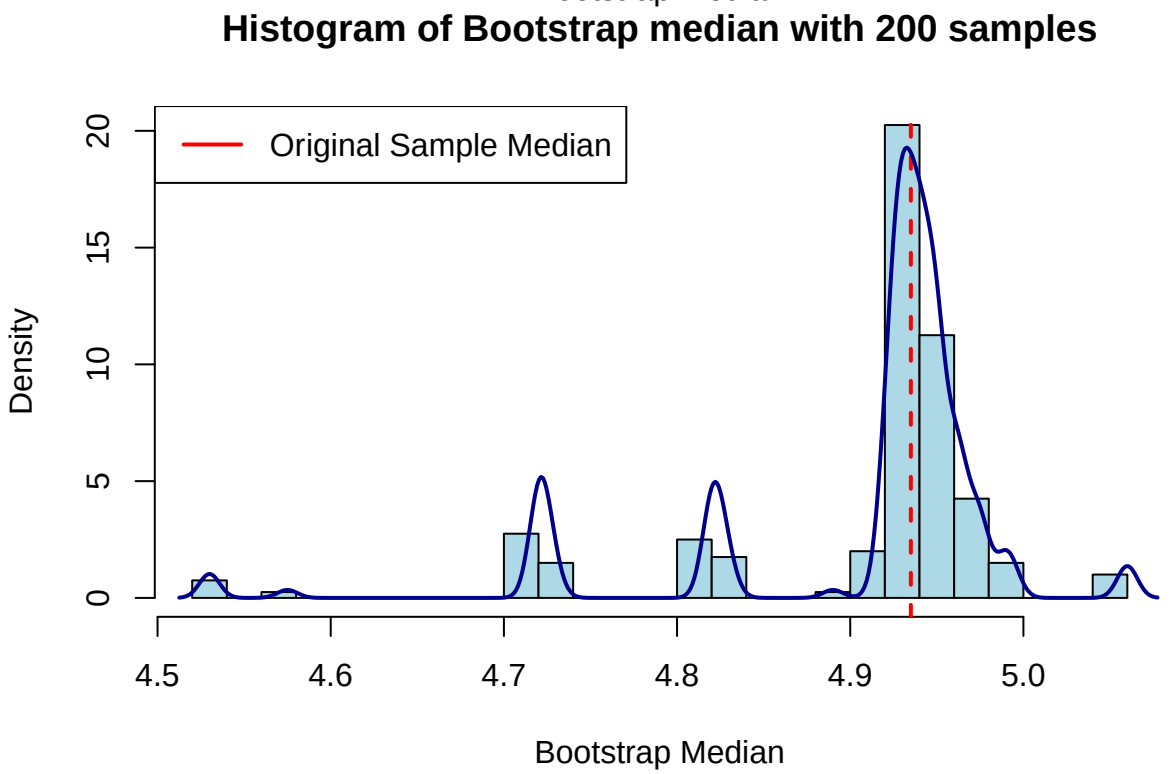
```r
bootstrap_medians <- numeric(n_bootstrap)
for (i in 1:n_bootstrap) {
  bootstrap_sample <- sample(sample_data, size = length(sample_data), replace = TRUE)
  bootstrap_samples[[i]] <- bootstrap_sample
  bootstrap_medians[i] <- median(bootstrap_sample)
}

bootstrap_samples_list2 <- list(
  subset_20 = bootstrap_medians[1:20],
  subset_200 = bootstrap_medians[1:200],
  subset_2000 = bootstrap_medians[1:2000]
)

for (i in 1:length(bootstrap_samples_list2)) {
```

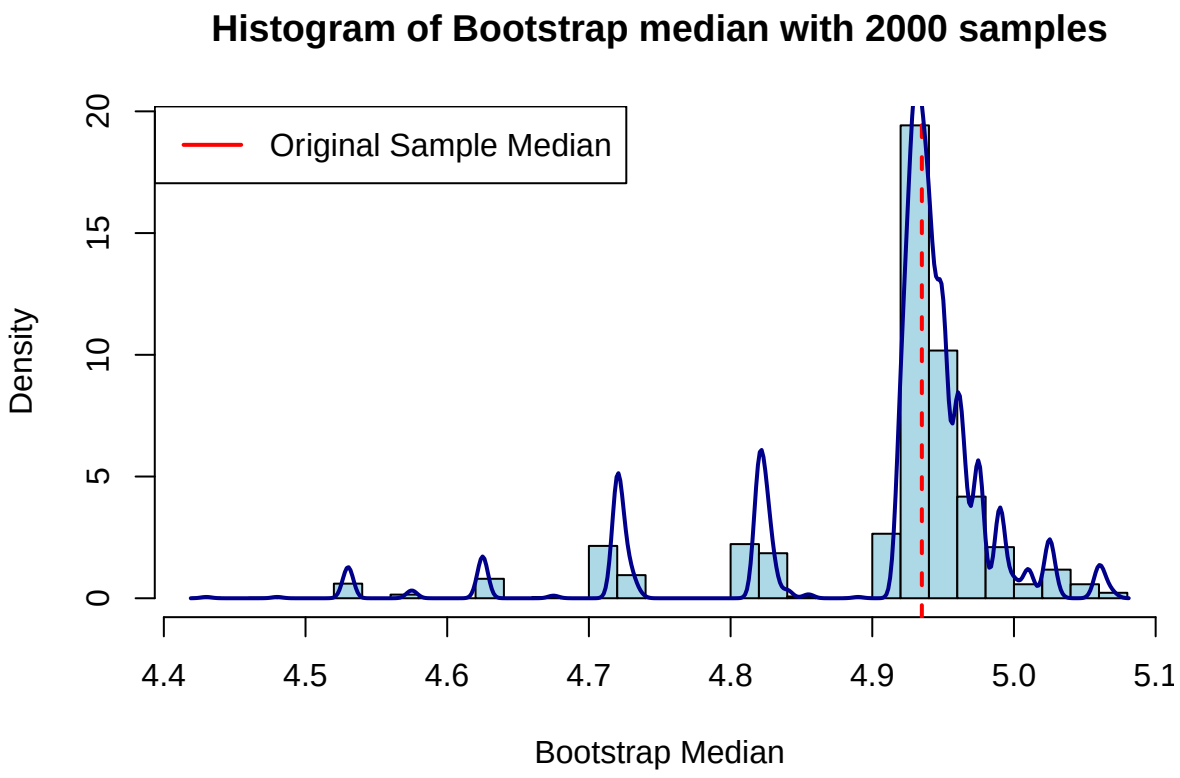
```
  current_sample <- bootstrap_samples_list2[[i]]

  cat(paste("Mean of the ", length(current_sample) , " bootstrap medians:", mean(current_sample), "\n"))
}
```

```
## Mean of the  20  bootstrap medians: 4.9115
## Mean of the  200  bootstrap medians: 4.90855
## Mean of the  2000  bootstrap medians: 4.9131225
```

```
for (i in 1:length(bootstrap_samples_list2)) {
  hist_title <- paste("Histogram of Bootstrap median with", length(bootstrap_samples_list2[[i]]), "sampl
  hist(
    bootstrap_samples_list2[[i]],
    breaks = 30,
    probability = TRUE,
    main = hist_title,
    xlab = "Bootstrap Median",
    col = "lightblue",
    border = "black"
  )

  # original sample median
  abline(
    v = median(sample_data),
    col = "red",
    lwd = 2,
    lty = 2
  )

  # density curve
  lines(density(bootstrap_samples_list2[[i]]),
        col = "darkblue",
        lwd = 2)

  legend(
    "topleft",
    legend = c("Original Sample Median"),
    col = c("red"),
    lwd = 2
  )
}
```

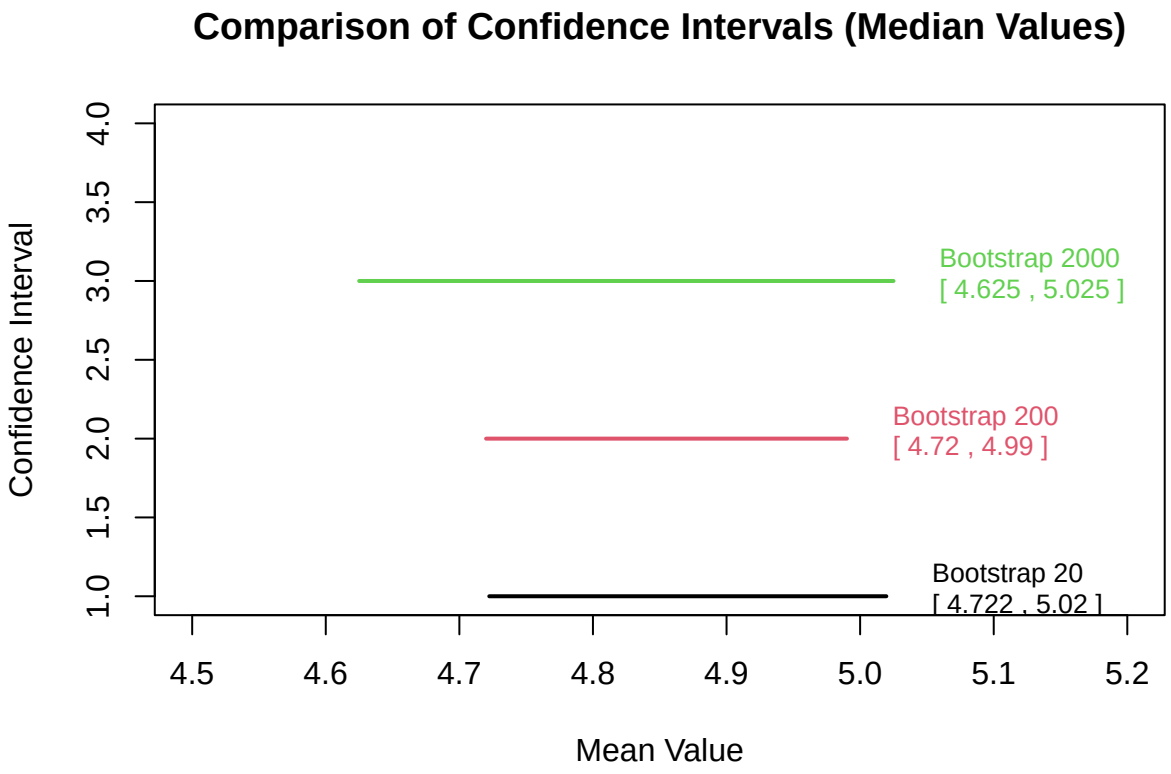

Histogram of Bootstrap median with 20 samples
Original Sample Median
Density
0
5
10
15
20
4.75
4.80
4.85
4.90
4.95
5.00
5.05
Bootstrap Median


Histogram of Bootstrap median with 200 samples
Original Sample Median
Density
0
5
10
15
20
4.5
4.6
4.7
4.8
4.9
5.0
Bootstrap Median



Unlike the mean, which forms a smooth, bell-shaped curve (thanks to the Central Limit Theorem), the median looks irregular and "spiky." This happens because the median is calculated differently, it's more robust and doesn't change as smoothly as the mean when we resample the data.

```
ci_list2 <- list()

for (i in 1:length(bootstrap_samples_list2)) {
  current_sample <- bootstrap_samples_list2[[i]]
  quantiles <- quantile(current_sample, probs = c(0.025, 0.975))
  ci_list2[[i]] <- c(quantiles[1], quantiles[2])
}

labels2 <- c("Bootstrap 20", "Bootstrap 200", "Bootstrap 2000")

plot(1, type = "n", xlim = c(4.5, 5.2), ylim = c(1, length(ci_list2) + 1),
     xlab = "Mean Value", ylab = "Confidence Interval",
     main = "Comparison of Confidence Intervals (Median Values)")

for (i in 1:length(ci_list2)) {
  segments(x0 = ci_list2[[i]][1], x1 = ci_list2[[i]][2], y0 = i, y1 = i, lwd = 2, col = i)

  text(x = ci_list2[[i]][2] + 0.02, y = i,
       labels = paste(labels2[i], "\n[", round(ci_list2[[i]][1], 3), ",", round(ci_list2[[i]][2], 3), "]
       col = i, cex = 0.8, pos = 4)
}
```



Task 2

1)

```r
set.seed(1234)

x.clean <- rnorm(1960, mean = 0, sd = 1)

x.cont <- runif(40, min = 4, max = 5)

x <- c(x.clean, x.cont)
```

2) Estimate the median, the mean and the trimmed mean with alpha = 0.05 for x and x.clean.

```r
median_x <- median(x)

mean_x <- mean(x)

trimmed_mean_x <- mean(x, trim = 0.05)

median_x_clean <- median(x.clean)

mean_x_clean <- mean(x.clean)
```

```r
trimmed_mean_x_clean <- mean(x.clean, trim = 0.05)

cat("Results for x (including outliers):\n")
```

```
## Results for x (including outliers):
```

```r
cat("Median:", median_x, "\n")
```

```
## Median: 0.0113797
```

```r
cat("Mean:", mean_x, "\n")
```

```
## Mean: 0.08395508
```

```r
cat("Trimmed Mean (alpha = 0.05):", trimmed_mean_x, "\n\n")
```

```
## Trimmed Mean (alpha = 0.05): 0.03683294
```

```r
cat("Results for x.clean (normal values only):\n")
```

```
## Results for x.clean (normal values only):
```

```r
cat("Median:", median_x_clean, "\n")
```

```
## Median: -0.0172536
```

```r
cat("Mean:", mean_x_clean, "\n")
```

```
## Mean: -0.005968976
```

```r
cat("Trimmed Mean (alpha = 0.05):", trimmed_mean_x_clean, "\n")
```

```
## Trimmed Mean (alpha = 0.05): -0.001462623
```

3) Nonparametric bootstrap

```r
bootstrap_analysis <- function(data, n_bootstrap) {
  median_values <- numeric(n_bootstrap)
  mean_values <- numeric(n_bootstrap)
  trimmed_mean_values <- numeric(n_bootstrap)

  for (i in 1:n_bootstrap) {
    bootstrap_sample <- sample(data, size = length(data), replace = TRUE)

    median_values[i] <- median(bootstrap_sample)
```

```
    mean_values[i] <- mean(bootstrap_sample)
    trimmed_mean_values[i] <- mean(bootstrap_sample, trim = 0.05)
  }

  se_median <- sd(median_values)
  se_mean <- sd(mean_values)
  se_trimmed_mean <- sd(trimmed_mean_values)

  ci_median <- quantile(median_values, probs = c(0.025, 0.975))
  ci_mean <- quantile(mean_values, probs = c(0.025, 0.975))
  ci_trimmed_mean <- quantile(trimmed_mean_values, probs = c(0.025, 0.975))

  return(list(
    se = list(median = se_median, mean = se_mean, trimmed_mean = se_trimmed_mean),
    ci = list(
      median = ci_median,
      mean = ci_mean,
      trimmed_mean = ci_trimmed_mean
    )
  ))
}
```

```
nonparamtric_x <- bootstrap_analysis(x, 2000)
nonparametric_x.clean <- bootstrap_analysis(x.clean, 2000)

xdf <- as.data.frame(nonparamtric_x)
print(xdf)
```

```
##        se.median    se.mean se.trimmed_mean   ci.median   ci.mean
## 2.5%  0.02868065 0.02587341      0.02368954 -0.04695338 0.0353476
## 97.5% 0.02868065 0.02587341      0.02368954  0.06599349 0.1344884
##       ci.trimmed_mean
## 2.5%     -0.008240382
## 97.5%     0.082515485
```

```
x.cleandf <- as.data.frame(nonparametric_x.clean)
print(x.cleandf)
```

```
##        se.median    se.mean se.trimmed_mean   ci.median     ci.mean
## 2.5%  0.02615421 0.02219964      0.02231887 -0.06013084 -0.04867766
## 97.5% 0.02615421 0.02219964      0.02231887  0.03615287  0.03830022
##       ci.trimmed_mean
## 2.5%      -0.04360975
## 97.5%      0.04289933
```

4) Parametric bootstrap

```
parametric_bootstrap <- function(data, n_bootstrap, robust) {

  data_mean <- mean(data)
```

```
  data_sd <- if (robust) { mad(data) } else { sd(data) }


  mean_values <- numeric(n_bootstrap)
  trimmed_mean_values <- numeric(n_bootstrap)

  for (i in 1:n_bootstrap) {
    bootstrap_sample <- rnorm(length(data), mean = data_mean, sd = data_sd)

    mean_values[i] <- mean(bootstrap_sample)
    trimmed_mean_values[i] <- mean(bootstrap_sample, trim = 0.05)
  }

  bias <- mean(mean_values) - data_mean
  bias_trimmed <- mean(trimmed_mean_values) - mean(data, trim = 0.05)

  se_mean <- sd(mean_values)
  se_trimmed_mean <- sd(trimmed_mean_values)

  ci_mean <- quantile(mean_values, probs = c(0.025, 0.975))
  ci_trimmed_mean <- quantile(trimmed_mean_values, probs = c(0.025, 0.975))

  bias_corrected_estimate <- data_mean - bias
  bias_corrected_estimate_trimmed <- mean(data, trim = 0.05) - bias_trimmed

  return(list(
    bias = list(mean = bias, trimmed_mean = bias_trimmed),
    se = list(mean = se_mean, trimmed_mean = se_trimmed_mean),
    ci = list(mean = ci_mean, trimmed_mean = ci_trimmed_mean),
    bias_corrected = list(mean = bias_corrected_estimate, trimmed_mean = bias_corrected_estimate_trimme
  ))
}
```

```
paramertic_x <- parametric_bootstrap(x, 2000, robust = FALSE)
paramertic_x.clean <- parametric_bootstrap(x.clean, 2000, robust = FALSE)

x_parametric_df <- as.data.frame(paramertic_x)
print(x_parametric_df)
```

```
##           bias.mean bias.trimmed_mean    se.mean se.trimmed_mean   ci.mean
## 2.5%  -3.113478e-05        0.04707094 0.02587111      0.02608972 0.0326523
## 97.5% -3.113478e-05        0.04707094 0.02587111      0.02608972 0.1338023
##       ci.trimmed_mean bias_corrected.mean bias_corrected.trimmed_mean
## 2.5%       0.03121574          0.08398621                   -0.010238
## 97.5%      0.13353606          0.08398621                   -0.010238
```

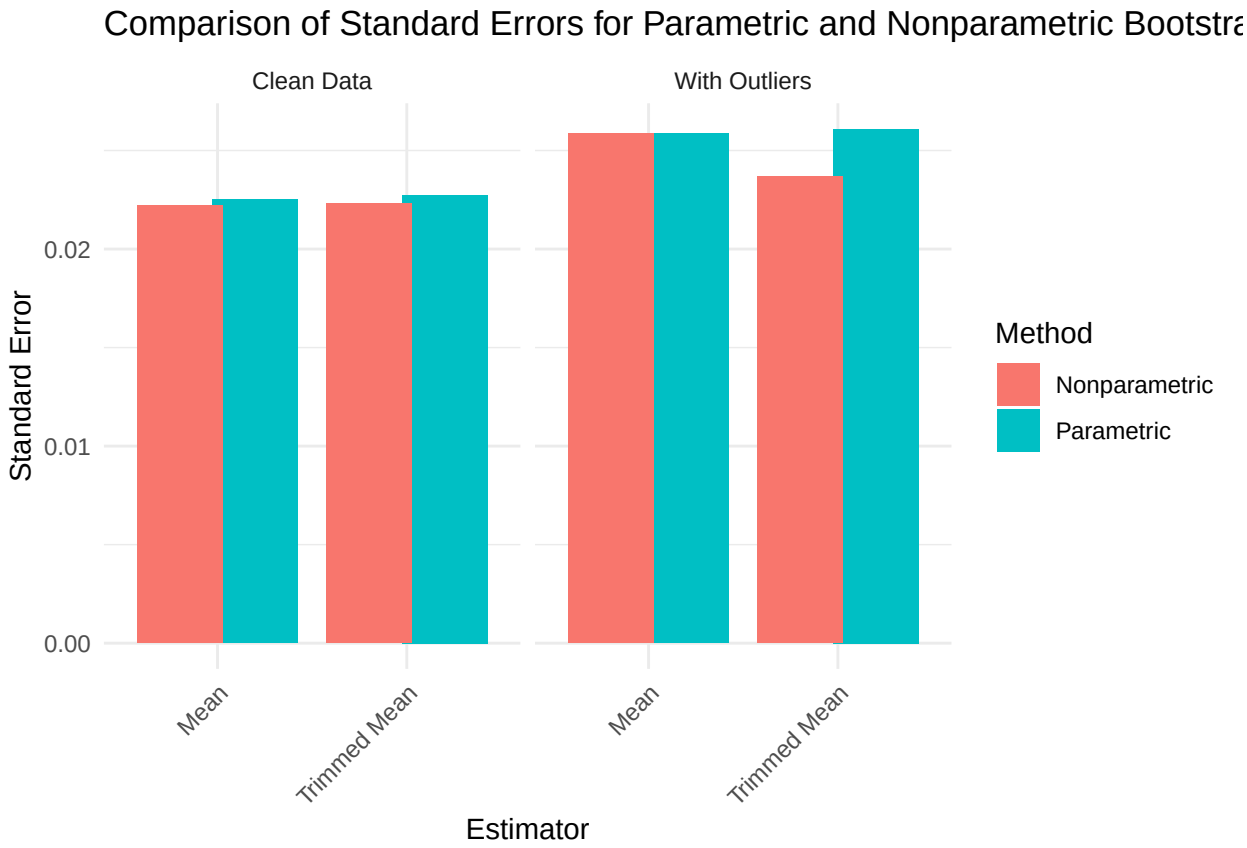
```
x.clean_parametric_df <- as.data.frame(paramertic_x.clean)
print(x.clean_parametric_df)
```

```
##          bias.mean bias.trimmed_mean    se.mean se.trimmed_mean     ci.mean
## 2.5%  0.0004426659      -0.003987303 0.02252173      0.02273837 -0.04797690
## 97.5% 0.0004426659      -0.003987303 0.02252173      0.02273837  0.03876946
```

```
##       ci.trimmed_mean bias_corrected.mean bias_corrected.trimmed_mean
## 2.5%      -0.04843353        -0.006411642                  0.00252468
## 97.5%      0.03864086        -0.006411642                  0.00252468
```

5) Compare and summarize your findings with tables and graphically.

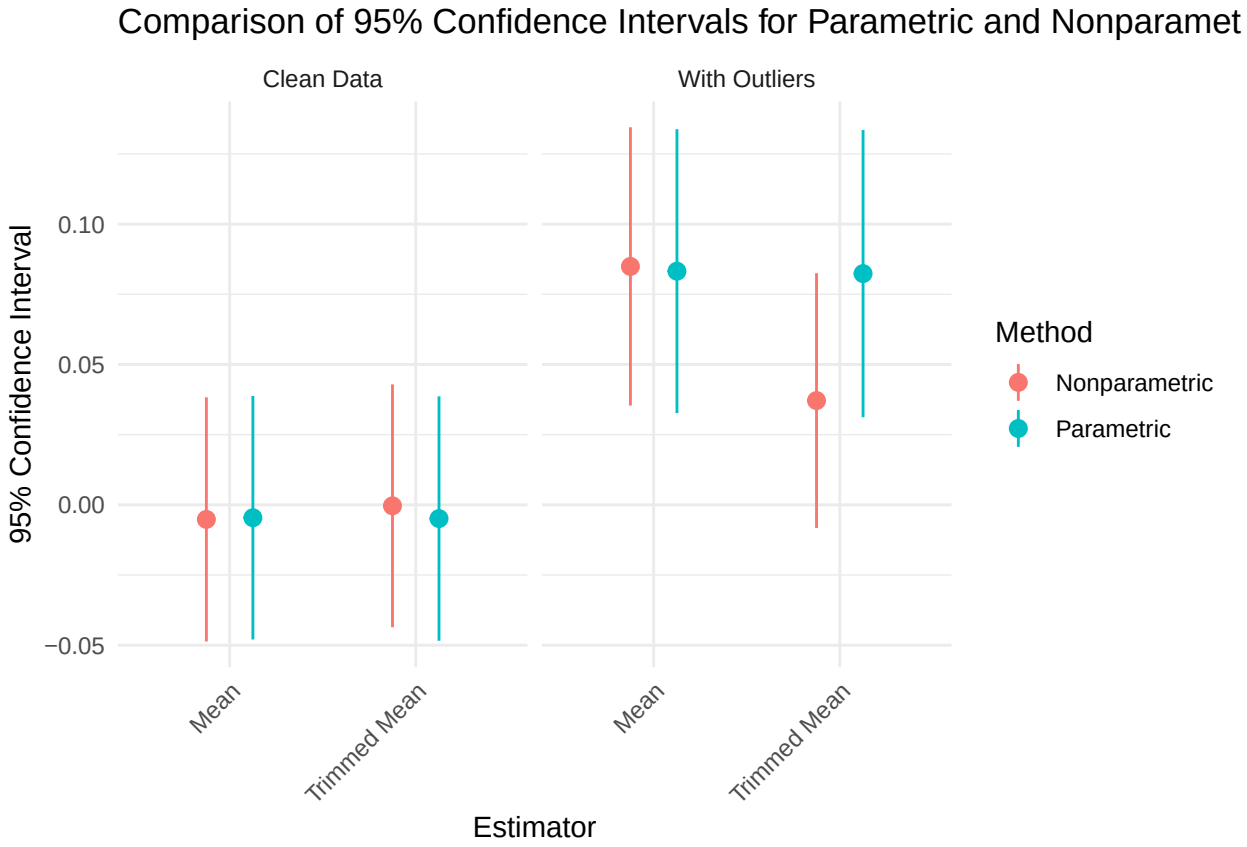
```r
library(ggplot2)

se_data_ggplot <- data.frame(
  Estimator = rep(c("Mean", "Trimmed Mean"), times = 4),
  Method = rep(c("Parametric", "Nonparametric"), each = 4),
  Data_Type = rep(c("With Outliers", "Clean Data"), each = 2, times = 2),
  SE = c(
    # Parametric (1. With Outliers, 2.. Clean Data)
    paramertic_x$se$mean, paramertic_x$se$trimmed_mean,
    paramertic_x.clean$se$mean, paramertic_x.clean$se$trimmed_mean,
    # Nonparametric (1. With Outliers, 2. Clean Data)
    nonparamtric_x$se$mean, nonparamtric_x$se$trimmed_mean,
    nonparametric_x.clean$se$mean, nonparametric_x.clean$se$trimmed_mean
  )
)


ggplot(se_data_ggplot, aes(x = Estimator, y = SE, fill = Method)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.8)) +
  facet_wrap(~ Data_Type) +
  theme_minimal() +
  labs(title = "Comparison of Standard Errors for Parametric and Nonparametric Bootstrap",
       y = "Standard Error", x = "Estimator") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



You can see that the standard errors are almost the same for both methods when using clean data, but the nonparametric one has a bit higher value.

```
ci_data_ggplot <- data.frame(
  Estimator = rep(c("Mean", "Trimmed Mean"), times = 4),
  Method = rep(c("Parametric", "Nonparametric"), each = 4),
  Data_Type = rep(c("With Outliers", "Clean Data"), each = 2, times = 2),
  Lower = c(
    # Parametric (With Outliers, Clean Data)
    paramertic_x$ci$mean[1], paramertic_x$ci$trimmed_mean[1],
    paramertic_x.clean$ci$mean[1], paramertic_x.clean$ci$trimmed_mean[1],
    # Nonparametric (With Outliers, Clean Data)
    nonparamtric_x$ci$mean[1], nonparamtric_x$ci$trimmed_mean[1],
    nonparametric_x.clean$ci$mean[1], nonparametric_x.clean$ci$trimmed_mean[1]
  ),
  Upper = c(
    # Parametric (1. With Outliers, 2. Clean Data)
    paramertic_x$ci$mean[2], paramertic_x$ci$trimmed_mean[2],
    paramertic_x.clean$ci$mean[2], paramertic_x.clean$ci$trimmed_mean[2],
    # Nonparametric (1. With Outliers, 2.Clean Data)
    nonparamtric_x$ci$mean[2], nonparamtric_x$ci$trimmed_mean[2],
    nonparametric_x.clean$ci$mean[2], nonparametric_x.clean$ci$trimmed_mean[2]
  )
)


ggplot(ci_data_ggplot, aes(x = Estimator, y = (Lower + Upper) / 2, ymin = Lower, ymax = Upper, color = I
  geom_pointrange(position = position_dodge(width = 0.5)) +
```

```
  facet_wrap(~ Data_Type) +
  theme_minimal() +
  labs(title = "Comparison of 95% Confidence Intervals for Parametric and Nonparametric Bootstrap",
       y = "95% Confidence Interval", x = "Estimator") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The chart compares the 95% confidence intervals for both parametric and nonparametric bootstrap methods for clean data versus data with outliers. In the clean data section, the confidence intervals are relatively narrow and similar for both methods, showing that they perform consistently when there are no outliers. However, in the data with outliers section, you can see that the confidence intervals are much wider, particularly for the nonparametric method, which suggests increased uncertainty due to the presence of outliers. This means that outliers have a bigger impact on the accuracy of nonparametric methods compared to parametric methods.

Task 3

Bootstrapping is a statistical method that we use to estimate the sampling distribution of an estimator (like the mean, median, etc.) by repeatedly resampling from the original dataset. In simple terms, it involves creating many new samples from the given data, allowing us to make inferences about the population, especially when we don't have a lot of data or assumptions about its distribution.

To construct confidence intervals using bootstrapping, we draw multiple bootstrap samples from the data, which are created by sampling with replacement. This means that the same data point could be included multiple times in one sample. Each of these bootstrap samples is then used to calculate an estimator (like the mean or median). By doing this repeatedly, we get a distribution of those estimates.

For example, in our task, we drew 2000 bootstrap samples from both the original dataset with outliers (x) and a cleaned dataset (x.clean). Each bootstrap sample was used to calculate statistics like the mean and the trimmed mean. By doing this multiple times, we got an empirical distribution of these means, which allowed us to calculate confidence intervals.

Parametric vs. Nonparametric Bootstrapping:

- Parametric bootstrapping assumes that the data comes from a specific distribution (like in our case normal distribution). It generates bootstrap samples based on the scale (like mean and standard deviation) of that assumed distribution. This makes parametric bootstrapping more stable since it relies on an underlying model to filter out noise, which makes it less affected by extreme values or outliers.
- Nonparametric bootstrapping, on the other hand, makes no assumptions about the data distribution. It directly resamples the original data points. This makes it more flexible but also more sensitive to outliers or irregularities, as it uses whatever values are present in the data without any smoothing assumptions.

From the charts we analyzed:

- The Standard Error is higher for the nonparametric method when we have outliers. This means there's more uncertainty when outliers are included.
- The confidence intervals were also wider for the nonparametric method with outliers, showing greater variability.

The key takeaway is that parametric bootstrapping can be more stable when data is "clean," while nonparametric bootstrapping is more flexible but sensitive to outliers.