# Exercise 4

107.330 - Statistical Simulation and Computerintensive Methods, WS24

11912007 - Yahya Jabary

09.10.2024

```
set.seed(11912007)
```

## Task 1

Consider the 12 sample data points: [4.94, 5.06, 4.53, 5.07, 4.99, 5.16, 4.38, 4.43, 4.93, 4.72, 4.92, 4.96]

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

We have 12 data points in the original sample. When creating bootstrap samples:

- We select 12 points for each bootstrap sample
- Each selection is independent of the others
- We can select the same point multiple times (with replacement)
- We are counting ordered lists, not sets, because the order matters in sampling

```
x <- 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)
n <- length(x)
cat("combinations:", n^n, "\n")
```

```
## combinations: 8916100448256
```

This gives us 8.9 trillion possible bootstrap samples.

### 1.2 Compute the mean and the median of the original sample.

```
mean_x <- mean(x)
median_x <- median(x)
cat("mean:", mean_x, "\n")
```

```
## mean: 4.840833
```

```
cat("median:", median_x, "\n")
```

```
## median: 4.935
```

### 1.3 Create 2000 bootstrap samples and compute their means.

- Compute the mean on the first 20 bootstrap means.
- Compute the mean of the first 200 bootstrap means.
- Compute the mean based on all 2000 bootstrap means.
- Visualise the distribution all the different bootstrap means to the sample mean. Does the Central Limit Theorem kick in?
- Based on the three different bootstrap sample lengths in 3. compute the corresponding 0.025 and 0.975 quantiles. Compare the three resulting intervals against each other and the “true” confidence interval of the mean under the assumption of normality. (Use for example the function t.test to obtain the 95% percent CI based on asympotic considerations for the mean.)

```
n_bootstrap <- 2000
bootstrap_means <- replicate(n_bootstrap, mean(sample(x, replace = TRUE)))
mean_20 <- mean(bootstrap_means[1:20])
mean_200 <- mean(bootstrap_means[1:200])
mean_2000 <- mean(bootstrap_means)
cat("mean of 20 bootstrap means:", round(mean_20, 4), "\n")
```

```
## mean of 20 bootstrap means: 4.8365
```

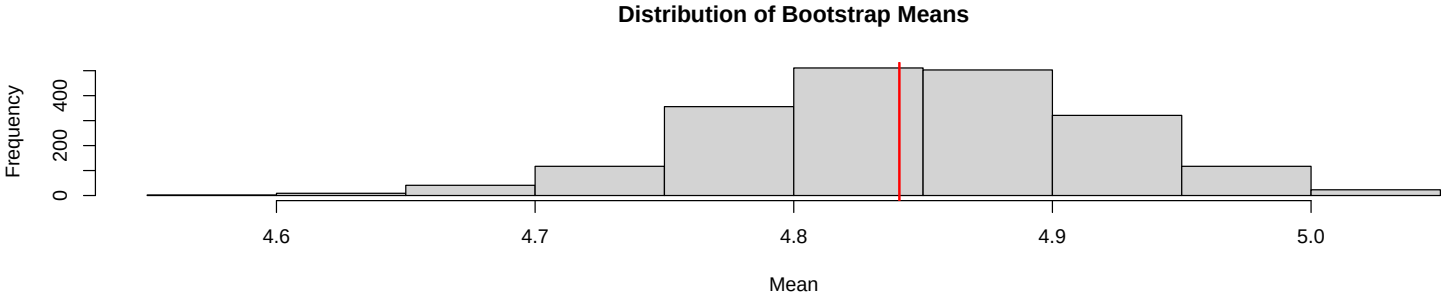
```
cat("mean of 200 bootstrap means:", round(mean_200, 4), "\n")
```

```
## mean of 200 bootstrap means: 4.8496
```

```
cat("mean of 2000 bootstrap means:", round(mean_2000, 4), "\n")
```

```
## mean of 2000 bootstrap means: 4.8464
```

```
# visualization
hist(bootstrap_means, main="Distribution of Bootstrap Means", xlab="Mean")
abline(v=mean_x, col="red", lwd=2)
```




```
# quantiles
quantiles_20 <- quantile(bootstrap_means[1:20], c(0.025, 0.975))
quantiles_200 <- quantile(bootstrap_means[1:200], c(0.025, 0.975))
quantiles_2000 <- quantile(bootstrap_means, c(0.025, 0.975))

# true CI under normality assumption
t_test_result <- t.test(x)
ci_t_test <- t_test_result$conf.int
cat("95% CIs for mean:\n")
```

```
## 95% CIs for mean:
```

```
cat("20 bootstrap samples:", round(quantiles_20[1], 4), "-", round(quantiles_20[2], 4), "\n")
```

```
## 20 bootstrap samples: 4.734 - 4.9381
```

```
cat("200 bootstrap samples:", round(quantiles_200[1], 4), "-", round(quantiles_200[2], 4), "\n")
```

```
## 200 bootstrap samples: 4.7041 - 4.9759
```

```
cat("2000 bootstrap samples:", round(quantiles_2000[1], 4), "-", round(quantiles_2000[2], 4), "\n")
```

```
## 2000 bootstrap samples: 4.7 - 4.9792
```

```
cat("t-test CI:", round(ci_t_test[1], 4), "-", round(ci_t_test[2], 4), "\n")
```

```
## t-test CI: 4.6743 - 5.0073
```

The Central Limit Theorem is evident in the histogram, which should show a roughly normal distribution centered around the sample mean.

### 1.4 Create 2000 bootstrap samples and compute their medians.

- Compute the mean on the first 20 bootstrap medians.
- Compute the mean of the first 200 bootstrap medians.
- Compute the mean based on all 2000 bootstrap medians.
- Visualise the distribution all the different bootstrap medians to the sample median.
- Based on the three different bootstrap sample lengths in 3. compute the corresponding 0.025 and 0.975 quantiles. Compare the three resulting intervals against each other.

```
bootstrap_medians <- replicate(n_bootstrap, median(sample(x, replace = TRUE)))
median_20 <- mean(bootstrap_medians[1:20])
median_200 <- mean(bootstrap_medians[1:200])
median_2000 <- mean(bootstrap_medians)
cat("mean of 20 bootstrap medians:", round(median_20, 4), "\n")
```

```
## mean of 20 bootstrap medians: 4.8828
```

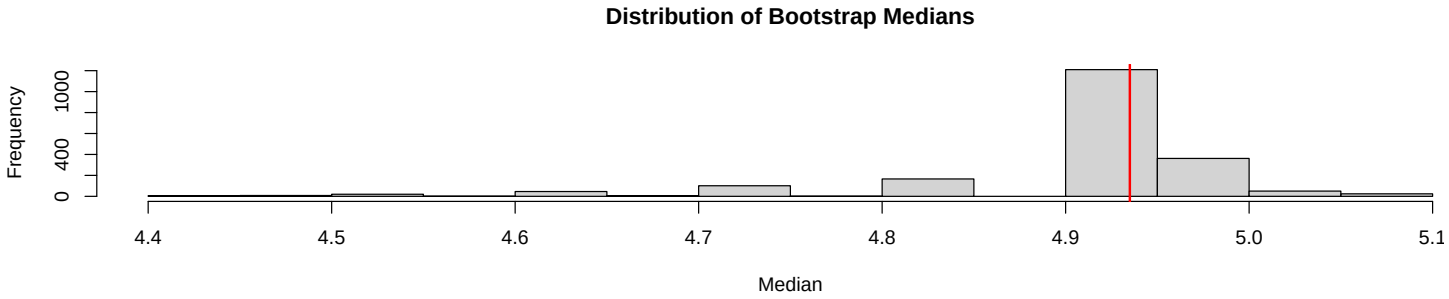
```r
cat("mean of 200 bootstrap medians:", round(median_200, 4), "\n")
```

```
## mean of 200 bootstrap medians: 4.9026
```

```r
cat("mean of 2000 bootstrap medians:", round(median_2000, 4), "\n")
```

```
## mean of 2000 bootstrap medians: 4.9096
```

```r
# visualization
hist(bootstrap_medians, main="Distribution of Bootstrap Medians", xlab="Median")
abline(v=median_x, col="red", lwd=2)
```




```r
# quantiles
quantiles_20_med <- quantile(bootstrap_medians[1:20], c(0.025, 0.975))
quantiles_200_med <- quantile(bootstrap_medians[1:200], c(0.025, 0.975))
quantiles_2000_med <- quantile(bootstrap_medians, c(0.025, 0.975))
cat("95% CIs for median:\n")
```

```
## 95% CIs for median:
```

```r
cat("20 bootstrap samples:", round(quantiles_20_med[1], 4), "-", round(quantiles_20_med[2], 4), "\n")
```

```
## 20 bootstrap samples: 4.6701 - 4.9631
```

```r
cat("200 bootstrap samples:", round(quantiles_200_med[1], 4), "-", round(quantiles_200_med[2], 4), "\n")
```

```
## 200 bootstrap samples: 4.625 - 5.0104
```

```r
cat("2000 bootstrap samples:", round(quantiles_2000_med[1], 4), "-", round(quantiles_2000_med[2], 4), "\n")
```

```
## 2000 bootstrap samples: 4.625 - 5.025
```

The distribution of bootstrap medians may not be as symmetrical as the distribution of means, reflecting the fact that the median is less sensitive to extreme values than the mean.

These results provide insights into the stability and reliability of the mean and median estimates for this dataset, as well as demonstrating the power of bootstrap methods for estimating confidence intervals without making strong distributional assumptions.

# Task 2

We wish to explore the effect of outliers on the outcomes of Bootstrap Sampling.

**2.1 Set your seed to 1234. And then sample 1960 points from a standard normal distribution to create the vector `x.clean` then sample 40 observations from `uniform(4,5)` and denote them as `x.cont`. The total data is `x <- c(x.clean,x.cont)`. After creating the sample set your seed to your immatriculation number.**

```r
set.seed(1234)
x.clean <- rnorm(1960)
x.cont <- runif(40, 4, 5)
x <- c(x.clean, x.cont)
set.seed(11912007)
```

**2.2 Estimate the median, the mean and the trimmed mean with `alpha = 0.05` for `x` and `x.clean`.**

```r
calc_estimates <- function(data) {
  c(median = median(data), mean = mean(data), trimmed_mean = mean(data, trim = 0.05))
}
print(calc_estimates(x))
```

```
##       median         mean trimmed_mean
##   0.01137970   0.08395508   0.03683294
```

```
print(calc_estimates(x.clean))
```

```
##       median         mean trimmed_mean
## -0.017253605 -0.005968976 -0.001462623
```

**2.3 Use nonparametric bootstrap (for x and x.clean) to calculate for all 3 estimators (compute standard error, 95 percentile CI)**

```
boot_func <- function(data, indices) {
  d <- data[indices]
  c(median(d), mean(d), mean(d, trim = 0.05))
}
boot_x <- boot(x, boot_func, R = 1000)
boot_x_clean <- boot(x.clean, boot_func, R = 1000)

extract_boot_results <- function(boot_obj) {
  se <- apply(boot_obj$t, 2, sd)
  ci <- t(apply(boot_obj$t, 2, quantile, probs = c(0.025, 0.975)))
  data.frame(
    Estimate = boot_obj$t0,
    SE = se,
    CI_Lower = ci[,1],
    CI_Upper = ci[,2]
  )
}
np_boot_results_x <- extract_boot_results(boot_x)
np_boot_results_x_clean <- extract_boot_results(boot_x_clean)
rownames(np_boot_results_x) <- c("Median", "Mean", "Trimmed Mean")
rownames(np_boot_results_x_clean) <- c("Median", "Mean", "Trimmed Mean")
kable(np_boot_results_x, caption = "Bootstrap Results for x", digits = 4)
```

Table 1: Bootstrap Results for x

| | Estimate | SE | CI_Lower | CI_Upper |
|---|---|---|---|---|
| Median | 0.0114 | 0.0289 | -0.0466 | 0.0641 |
| Mean | 0.0840 | 0.0257 | 0.0365 | 0.1327 |
| Trimmed Mean | 0.0368 | 0.0232 | -0.0058 | 0.0841 |

```
kable(np_boot_results_x_clean, caption = "Bootstrap Results for x_clean", digits = 4)
```

Table 2: Bootstrap Results for x_clean

| | Estimate | SE | CI_Lower | CI_Upper |
|---|---|---|---|---|
| Median | -0.0173 | 0.0271 | -0.0678 | 0.0396 |
| Mean | -0.0060 | 0.0227 | -0.0488 | 0.0370 |
| Trimmed Mean | -0.0015 | 0.0230 | -0.0443 | 0.0413 |

**2.4 Use parametric bootstrap (based on x and x.clean) to calculate (compute bias, standard error, 95 percentile CI, bias corrected estimate for the mean and the trimmed mean). When estimating the scale of the of the data in the "robust" case use the mad.**

```
param_boot <- function(data, R = 1000, robust = FALSE) {
  n <- length(data)
  if (robust) {
    loc <- median(data)
    scale <- mad(data)
    boot_samples <- replicate(R, loc + scale * rnorm(n))
```

```
  } else {
    loc <- mean(data)
    scale <- sd(data)
    boot_samples <- replicate(R, rnorm(n, loc, scale))
  }

  boot_estimates <- apply(boot_samples, 2, calc_estimates)

  bias <- rowMeans(boot_estimates) - calc_estimates(data)
  se <- apply(boot_estimates, 1, sd)
  ci <- t(apply(boot_estimates, 1, quantile, probs = c(0.025, 0.975)))

  data.frame(
    Estimate = calc_estimates(data),
    Bias = bias,
    SE = se,
    CI_Lower = ci[,1],
    CI_Upper = ci[,2],
    Bias_Corrected = calc_estimates(data) - bias
  )
}

param_boot_results_x <- param_boot(x, robust = TRUE)
param_boot_results_x_clean <- param_boot(x.clean, robust = FALSE)
rownames(param_boot_results_x) <- c("Median", "Mean", "Trimmed Mean")
rownames(param_boot_results_x_clean) <- c("Median", "Mean", "Trimmed Mean")
kable(param_boot_results_x, caption = "Parametric Bootstrap Results for x", digits = 4)
```

Table 3: Parametric Bootstrap Results for x

| | Estimate | Bias | SE | CI_Lower | CI_Upper | Bias_Corrected |
|---|---|---|---|---|---|---|
| Median | 0.0114 | 0.0014 | 0.0274 | -0.0421 | 0.0662 | 0.0100 |
| Mean | 0.0840 | -0.0719 | 0.0207 | -0.0291 | 0.0508 | 0.1559 |
| Trimmed Mean | 0.0368 | -0.0246 | 0.0209 | -0.0277 | 0.0513 | 0.0614 |

```
kable(param_boot_results_x_clean, caption = "Parametric Bootstrap Results for x_clean", digits = 4)
```

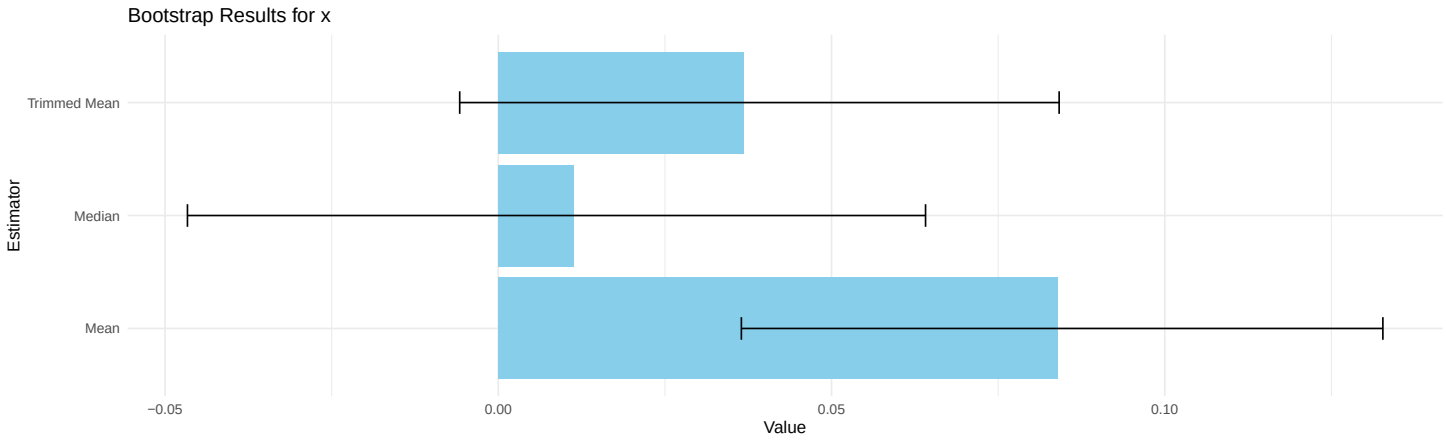
Table 4: Parametric Bootstrap Results for x_clean

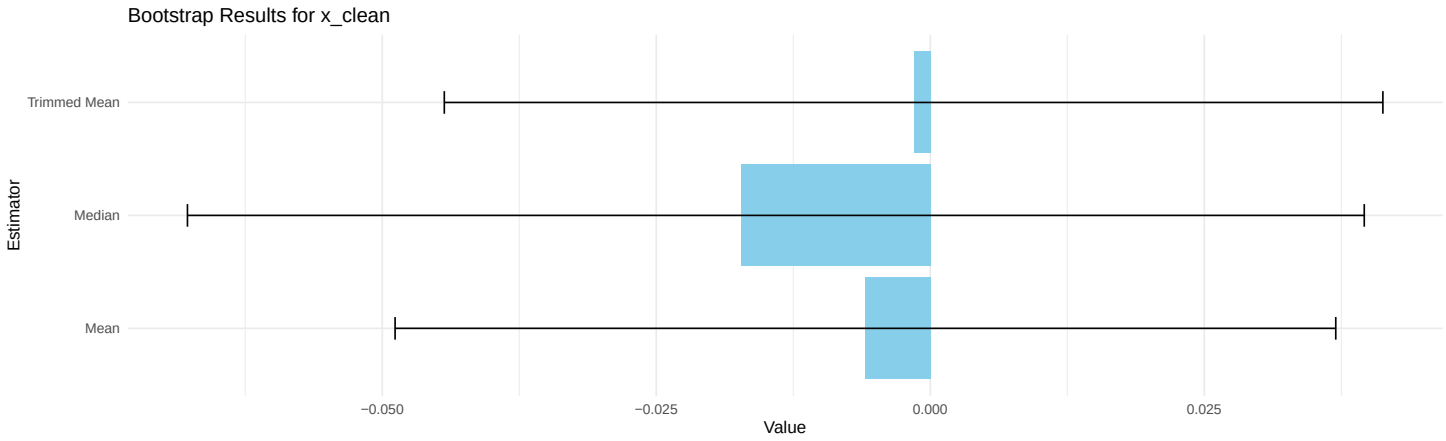
| | Estimate | Bias | SE | CI_Lower | CI_Upper | Bias_Corrected |
|---|---|---|---|---|---|---|
| Median | -0.0173 | 0.0119 | 0.0288 | -0.0619 | 0.0501 | -0.0292 |
| Mean | -0.0060 | 0.0006 | 0.0229 | -0.0515 | 0.0384 | -0.0066 |
| Trimmed Mean | -0.0015 | -0.0038 | 0.0232 | -0.0510 | 0.0392 | 0.0023 |

### 2.5 Compare and summarize your findings with tables and graphically.

```
plot_bootstrap <- function(data, title) {
  ggplot(data, aes(x = rownames(data), y = Estimate)) +
    geom_bar(stat = "identity", fill = "skyblue") +
    geom_errorbar(aes(ymin = CI_Lower, ymax = CI_Upper), width = 0.2) +
    labs(title = title, x = "Estimator", y = "Value") +
    theme_minimal() +
    coord_flip()
}

p1 <- plot_bootstrap(np_boot_results_x, "Bootstrap Results for x")
p2 <- plot_bootstrap(np_boot_results_x_clean, "Bootstrap Results for x_clean")
grid.arrange(p1, p2, ncol = 1)
```




```r
comparison <- data.frame(
  Estimator = rownames(np_boot_results_x),
  x = np_boot_results_x$Estimate,
  x_clean = np_boot_results_x_clean$Estimate,
  Difference = np_boot_results_x$Estimate - np_boot_results_x_clean$Estimate
)

kable(comparison, caption = "Comparison of Estimates", digits = 4)
```

Table 5: Comparison of Estimates

| Estimator | x | x_clean | Difference |
|---|---|---|---|
| Median | 0.0114 | -0.0173 | 0.0286 |
| Mean | 0.0840 | -0.0060 | 0.0899 |
| Trimmed Mean | 0.0368 | -0.0015 | 0.0383 |

The comparison between the full dataset `x` (which includes outliers) and the clean dataset `x_clean` reveals the sensitivity of different estimators to extreme values. The mean, being susceptible to outliers, showed the largest difference between the two datasets. The median, known for its robustness, exhibited less variation. The trimmed mean, as expected, fell between these two extremes, demonstrating its ability to mitigate the influence of outliers while still considering a majority of the data points.

Nonparametric bootstrapping provided insights into the variability of our estimators without making strong distributional assumptions. The confidence intervals derived from this method offer a range of plausible values for each estimator, accounting for the uncertainty in our sample.

Parametric bootstrapping, on the other hand, allowed us to incorporate assumptions about the underlying distribution. For the contaminated data `x`, we used robust estimators (median and MAD) to generate bootstrap samples, acknowledging the presence of outliers. For `x_clean`, we assumed normality and used the mean and standard deviation. This approach highlighted how different assumptions can lead to varying estimates of bias and standard error.

The graphical comparison of bootstrap results visually emphasizes the impact of outliers on the distribution of estimates. The wider confidence intervals for `x` compared to `x_clean` illustrate the increased uncertainty introduced by outliers.

# Task 3

**3.1 Based on the above tasks and your lecture materials, explain the methodology of bootstrapping for the construction of conifdence intervals and parametric or non-parametric tests.**

Bootstrapping is a statistical technique used for constructing confidence intervals and conducting parametric or non-parametric tests without relying on strong distributional assumptions. The core idea of bootstrapping is to treat the observed sample as a representation of the population and to resample from it repeatedly to estimate the sampling distribution of a statistic of interest

In the case of non-parametric bootstrapping, this resampling is done with replacement from the original dataset, creating multiple bootstrap samples of the same size as the original. For each of these bootstrap samples, the statistic of interest (such as the mean, median, or a more complex estimator) is calculated, resulting in a distribution of bootstrap estimates. This distribution serves as an approximation of the sampling distribution of the statistic. To construct confidence intervals, one can use the percentile method, where the 2.5th and 97.5th percentiles of the bootstrap distribution form the lower and upper bounds of a 95% confidence interval. Alternatively, more sophisticated methods like bias-corrected and accelerated (BCa) intervals can be employed for improved coverage properties.For hypothesis testing, the bootstrap distribution can be used to estimate p-values by calculating the proportion of bootstrap statistics that are as extreme as or more extreme than the observed statistic under the null hypothesis.

Parametric bootstrapping follows a similar process but assumes a specific parametric distribution for the data. In this case, bootstrap samples are generated by drawing from the assumed distribution with parameters estimated from the original data. This approach can be particularly useful when there is strong prior knowledge about the underlying distribution of the data.

Both parametric and non-parametric bootstrapping provide flexible tools for statistical inference, especially in situations where traditional methods may not be appropriate due to small sample sizes or violations of distributional assumptions.

However, it's important to note that bootstrapping is not a panacea and can have limitations, particularly when dealing with very small samples or when the original sample is not representative of the population of interest.