Sample distribution and Central Limit Theorem

20-11-2024

Task 1

The subsection below contains the following code:
- Calculating the number of possible bootstrap samples.
- Computing the mean and the median of the original sample.
- Creating 2000 bootstrap samples and computing their means.
- Creating 2000 bootstrap samples and computing their medians.

1. Calculating the number of possible bootstrap samples

```
set.seed(1234)

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)

## Number of possible bootstrap samples is n^n

n <- length(sample_data)
possible_bootstraps <- n^n

cat("The number of possible bootstraps (n = 12) :", possible_bootstraps)
```

```
## The number of possible bootstraps (n = 12) : 8.9161e+12
```

2. Computing the mean and the median of the original sample

```
actual_mean <- mean(sample_data)
actual_median <- median(sample_data)

cat("Actual mean of original sample :", actual_mean, "\n")
```

```
## Actual mean of original sample : 4.840833
```

```
cat("Actual median of original sample :", actual_median)
```

```
## Actual median of original sample : 4.935
```

3.1 Creating 2000 bootstrap samples and computing their means

```
bootstrap_means <- replicate(2000, mean(sample(sample_data, n, replace = TRUE)))

bm_20 <- mean(bootstrap_means[1:20])
bm_200 <- mean(bootstrap_means[1:200])
bm_2000 <- mean(bootstrap_means)

cat("Mean on first 20 bootstrap means :", bm_20, "\n")
```

```
## Mean on first 20 bootstrap means : 4.841792
```

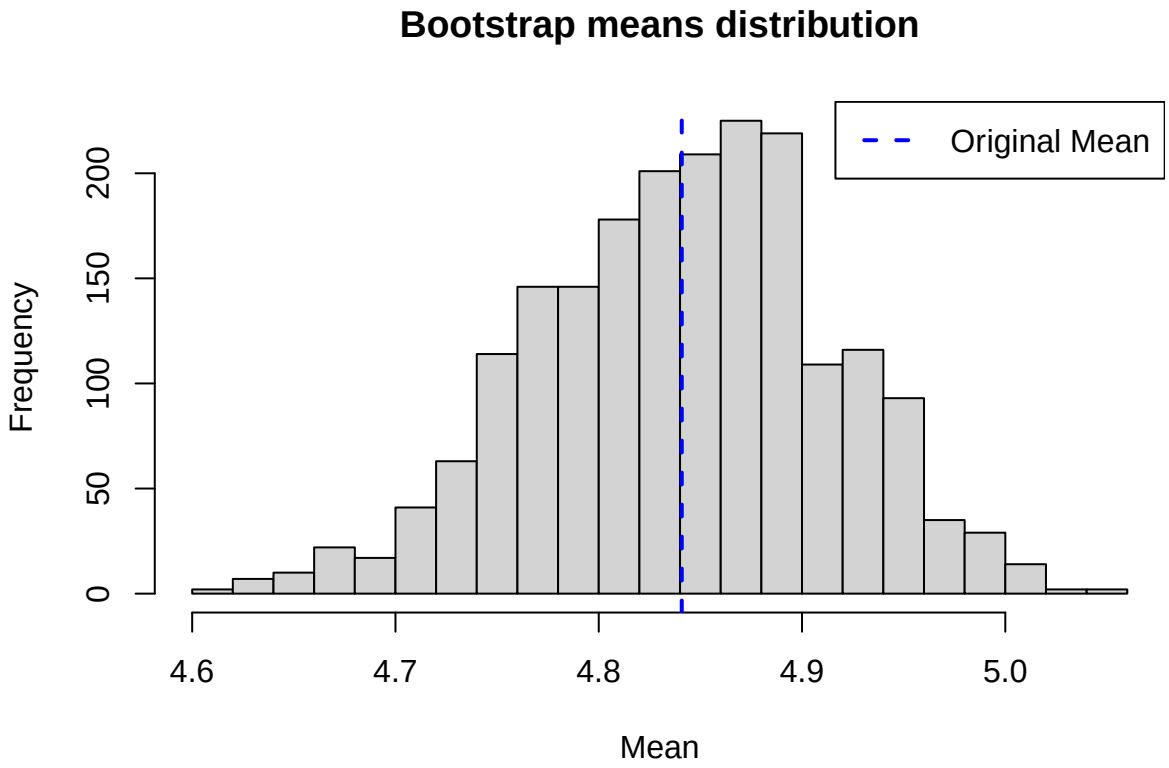
```r
cat("Mean of first 200 bootstrap means :", bm_200, "\n")
```

```
## Mean of first 200 bootstrap means : 4.84505
```

```r
cat("Mean based on all 2000 bootstrap means :", bm_2000)
```

```
## Mean based on all 2000 bootstrap means : 4.841523
```

```r
hist(bootstrap_means, breaks = 30, main = "Bootstrap means distribution", xlab = "Mean")
abline(v = actual_mean, col = "blue", lwd = 2, lty = 2)
legend("topright", legend = c("Original Mean"), col = "blue", lty = 2, lwd = 2)
```



Observation: The mean of the first 20 bootstraps is slightly higher than the first 200 and all 2000 bootstrap samples. When the number of bootstrap samples increases, the estimate mean converges towards the true central value. The histogram is symmetric and centred around the original sample mean. This supports the **Central Limit Theorem**, that suggests that the mean distribution usually follows a normal distribution with increasing sample sizes.

3.2 Computing the quantiles

```r
quant_20 <- quantile(bootstrap_means[1:20], probs = c(0.025, 0.975))
quant_200 <- quantile(bootstrap_means[1:200], probs = c(0.025, 0.975))
quant_2000 <- quantile(bootstrap_means, probs = c(0.025, 0.975))

cat("First 20 means :", quant_20, "\n")
```

```
## First 20 means : 4.729271 4.976104
```

```r
cat("First 20 means :", quant_200, "\n")
```

```
## First 20 means : 4.707458 4.988354
```

```r
cat("First 20 means :", quant_2000, "\n")
```

```
## First 20 means : 4.689167 4.976687
```

```r
## True confidence interval of the mean using t.test

t_test <- t.test(sample_data)$conf.int
cat("True confidence interval of the mean :", t_test)
```

```
## True confidence interval of the mean : 4.674344 5.007323
```

Observation: The CIs become narrower as the number of bootstrap samples increase, highlighting the improved precision of the bootstrap methods with higher sample sizes.

4.1 Creating 2000 bootstrap samples and computing their medians

```r
## Calculating the bootstrap median for 2000 samples
bootstrap_medians <- replicate(2000, median(sample(sample_data, n, replace = TRUE)))

## Taking the means of the bootstrap medians
median_20 <- mean(bootstrap_medians[1:20])
median_200 <- mean(bootstrap_medians[1:200])
median_2000 <- mean(bootstrap_medians)

cat("Mean of first 20 bootstrap medians:", median_20, "\n")
```

```
## Mean of first 20 bootstrap medians: 4.874
```

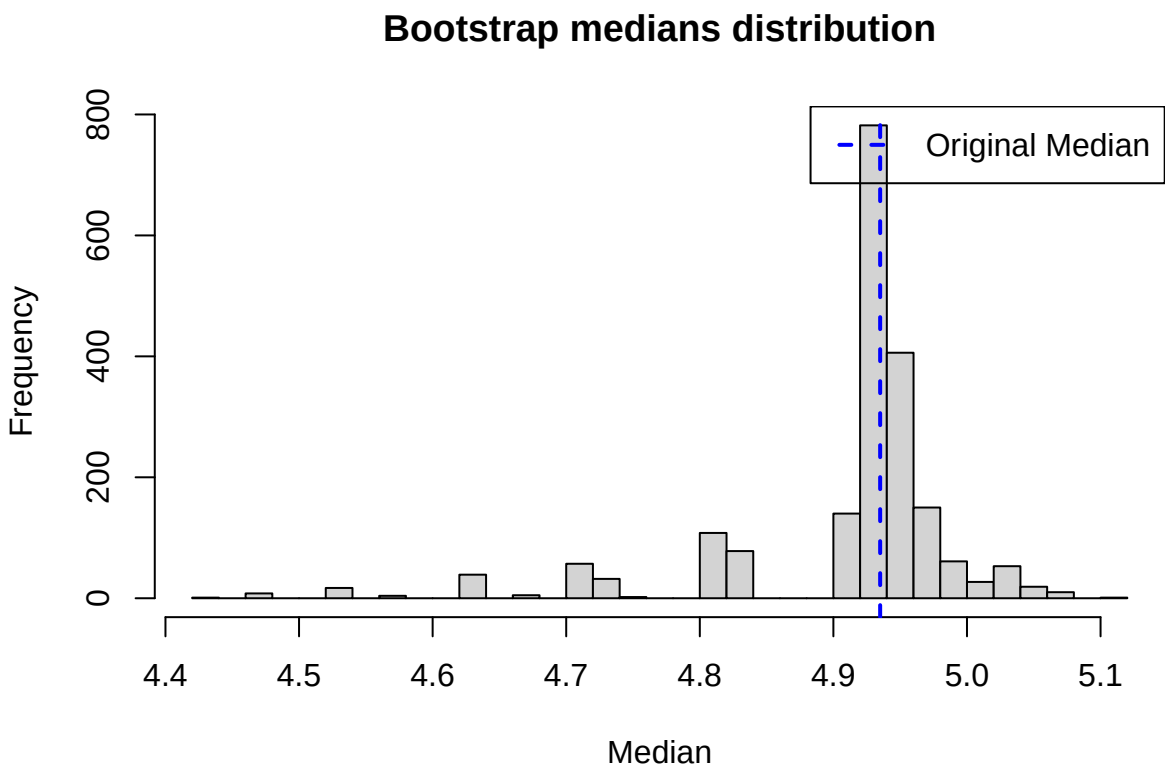
```r
cat("Mean of first 200 bootstrap medians:", median_200, "\n")
```

```
## Mean of first 200 bootstrap medians: 4.9042
```

```r
cat("Mean based on all 2000 bootstrap medians:", median_2000, "\n")
```

```
## Mean based on all 2000 bootstrap medians: 4.913675
```

```r
hist(bootstrap_medians, breaks = 30, main = "Bootstrap medians distribution", xlab = "Median")
abline(v = actual_median, col = "blue", lwd = 2, lty = 2)
legend("topright", legend = c("Original Median"), col = "blue", lty = 2, lwd = 2)
```



Observation: The mean of the first 20 medians is lower than the first 200 and 2000 medians. When the number of bootstrap samples increases, the estimate mean of the bootstrap medians converges towards the true central value. The histogram provides a skewed distribution but shows a strong peak close to the original median.

4.2 Computing the quantiles

```
quant_20_median <- quantile(bootstrap_medians[1:20], probs = c(0.025, 0.975))
quant_200_median <- quantile(bootstrap_medians[1:200], probs = c(0.025, 0.975))
quant_2000_median <- quantile(bootstrap_medians, probs = c(0.025, 0.975))

cat("Quantiles for first 20 bootstrap medians:", quant_20_median, "\n")
```

```
## Quantiles for first 20 bootstrap medians: 4.50375 4.967875
```

```
cat("Quantiles for first 200 bootstrap medians:", quant_200_median, "\n")
```

```
## Quantiles for first 200 bootstrap medians: 4.625 5.025
```

```
cat("Quantiles for all 2000 bootstrap medians:", quant_2000_median, "\n")
```

```
## Quantiles for all 2000 bootstrap medians: 4.625 5.025
```

Observation: The CIs become narrower as the number of bootstrap samples increase, highlighting the improved precision of the bootstrap methods with higher sample sizes.

Task 2

The subsection below contains the following code:
- Creating x and x.clean datasets.
- Estimating the median, mean, and trimmed mean for x and x.clean.
- Non-Parametric bootstrap to calculate all 3 estimators.

- Parametric bootstrap for mean and trimmed mean
- Comparison and summary of findings.

1. Creating x and x.clean datasets and estimating the median, mean, and trimmed mean for x and x.clean

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


set.seed(1234)

mean_x <- mean(x)
median_x <- median(x)
trimmed_mean_x <- mean(x, trim = 0.05)

mean_cleanx <- mean(x.clean)
median_cleanx <- median(x.clean)
trimmed_mean_cleanx <- mean(x.clean, trim = 0.05)

cat("Mean (x): ", mean_x,
    "\nMedian (x): ", median_x,
    "\nTrimmed Mean (x):", trimmed_mean_x, "\n \n")
```

```
## Mean (x):  0.08395508
## Median (x):  0.0113797
## Trimmed Mean (x): 0.03683294
##
```

```
cat("Mean (clean_x): ", mean_cleanx,
    "\nMedian (clean_x): ", median_cleanx,
    "\nTrimmed Mean (clean_x):", trimmed_mean_cleanx, "\n \n")
```

```
## Mean (clean_x):  -0.005968976
## Median (clean_x):  -0.0172536
## Trimmed Mean (clean_x): -0.001462623
##
```

Observation: The contamination in the dataset has caused the mean to increase compared to the median and the trimmed mean, which are robust to outliers. However, for the clean dataset, all the three measures are near zero as expected, highlighting the effectiveness of median and trimmed mean in handling a dataset that is contaminated, when compared to the mean.

2. Nonparametric bootstrap for x and x.clean for all 3 estimators to calculate - standard error - 95 percentile CI

```
## Function to print the results without the data
print_bootstrap_results <- function(bootstrap_list) {
  lapply(bootstrap_list, function(entry) {
    list(
      standard_error = entry$standard_error,
      CI = entry$CI
    )
  })
}
```

```r
## Non-Parametric bootstrap
non_parametric_bootstrap <- function(x, func) {
  l <- replicate(2000, func(sample(x, length(x), replace = TRUE)))
  s_err <- sd(l)
  ci_95 <- quantile(l, probs = c(0.025, 0.975))
  list(standard_error = s_err, CI = ci_95, samples = l)
}

bootstrap_x <- list(mean = non_parametric_bootstrap(x, mean),
                    median = non_parametric_bootstrap(x, median),
                    trimmed_mean = non_parametric_bootstrap(x, function(x) mean(x, trim = 0.05)))


bootstrap_cleanx <- list(mean = non_parametric_bootstrap(x.clean, mean),
                         median = non_parametric_bootstrap(x.clean, median),
                         trimmed_mean = non_parametric_bootstrap(x.clean, function(x) mean(x, trim = 0.

print("___Non-parametric Bootstrap___ ")
```

```
## [1] "___Non-parametric Bootstrap___ "
```

```r
cat("\n___x : \n")
```

```
##
## ___x :
```

```r
print(print_bootstrap_results(bootstrap_x))
```

```
## $mean
## $mean$standard_error
## [1] 0.02627569
##
## $mean$CI
##       2.5%      97.5%
## 0.03420387 0.13656595
##
##
## $median
## $median$standard_error
## [1] 0.02687952
##
## $median$CI
##       2.5%      97.5%
## -0.0447374  0.0588598
##
##
## $trimmed_mean
## $trimmed_mean$standard_error
## [1] 0.02271149
##
## $trimmed_mean$CI
##         2.5%        97.5%
## -0.005937803  0.080148030
```

```r
cat("\n___x.clean : \n")
```

```
##
## ___x.clean :
```

```
print(print_bootstrap_results(bootstrap_cleanx))
```

```
## $mean
## $mean$standard_error
## [1] 0.02226939
##
## $mean$CI
##        2.5%       97.5%
## -0.04842436  0.03881282
##
##
## $median
## $median$standard_error
## [1] 0.02697179
##
## $median$CI
##        2.5%       97.5%
## -0.06777026  0.03808645
##
##
## $trimmed_mean
## $trimmed_mean$standard_error
## [1] 0.02227685
##
## $trimmed_mean$CI
##        2.5%       97.5%
## -0.04700681  0.04031197
```

Observation:

i. Contaminated Dataset (x):
The CI range of the mean is the widest, showing that it is more sensitive to the contamination.

ii. Clean Dataset (x.clean):
All the three measures have tighter CI ranges compared to the contaminated dataset.

The contaminated datasets results show wider CI ranges and higher standard errors, especially for the mean, highlighting its vulnerability to outliers.

3. Parametric bootstrap to calculate:
- bias - standard error - 95 percentile CI - bias corrected estimate

```
## Function to print result without samples
print_parametric_results <- function(bootstrap_list) {
  lapply(bootstrap_list, function(entry) {
    list(
      bias = entry$bias,
      standard_error = entry$standard_error,
      CI = entry$CI,
      bias_corrected_estimate = entry$bias_corrected_estimate
    )
  })
}
```

```r
## Parametric bootstrap
parametric_bootstrap <- function(x, mean_f, scale_f, func) {
  mu <- mean_f(x)
  sigma <- scale_f(x)
  l <- replicate(2000, {
    samples <- rnorm(length(x), mean = mu, sd = sigma)
    func(samples)
  })

  bias <- mean(l) - func(x)
  s_err <- sd(l)
  CI <- quantile(l, probs = c(0.025, 0.975))
  bias_corrected_estimate <- func(x) - bias
  list(bias = bias, standard_error = s_err,
       CI = CI,
       bias_corrected_estimate = bias_corrected_estimate,
       samples = l)
}



p_b_x <- list(
  mean = parametric_bootstrap(x, mean, mad, mean),
  trimmed_mean = parametric_bootstrap(x, mean, mad,
                                      function(x) mean(x, trim = 0.05))
)

p_b_cleanx <- list(
  mean = parametric_bootstrap(x.clean, mean, mad, mean),
  trimmed_mean = parametric_bootstrap(x.clean, mean, sd,
                                      function(x) mean(x, trim = 0.05))
)


print("___Parametric Bootstrap___ ")
```

```
## [1] "___Parametric Bootstrap___ "
```

```r
cat("\n___x : \n")
```

```
##
## ___x :
```

```r
print(print_parametric_results(p_b_x))
```

```
## $mean
## $mean$bias
## [1] 5.963669e-05
##
## $mean$standard_error
## [1] 0.02185546
##
## $mean$CI
##       2.5%      97.5%
## 0.04122087 0.12754851
```

```
##
## $mean$bias_corrected_estimate
## [1] 0.08389544
##
##
## $trimmed_mean
## $trimmed_mean$bias
## [1] 0.04696043
##
## $trimmed_mean$standard_error
## [1] 0.02234332
##
## $trimmed_mean$CI
##       2.5%      97.5%
## 0.03962204 0.12647572
##
## $trimmed_mean$bias_corrected_estimate
## [1] -0.01012749
```

```
cat("\n___x.clean : \n")
```

```
##
## ___x.clean :
```

```
print(print_parametric_results(p_b_cleanx))
```

```
## $mean
## $mean$bias
## [1] 0.0003074035
##
## $mean$standard_error
## [1] 0.02155215
##
## $mean$CI
##        2.5%       97.5%
## -0.04654166  0.03606084
##
## $mean$bias_corrected_estimate
## [1] -0.00627638
##
##
## $trimmed_mean
## $trimmed_mean$bias
## [1] -0.004435905
##
## $trimmed_mean$standard_error
## [1] 0.0224231
##
## $trimmed_mean$CI
##        2.5%       97.5%
## -0.05021786  0.03792728
##
## $trimmed_mean$bias_corrected_estimate
## [1] 0.002973281
```

```r
results <- data.frame(
  Statistic = c(
    "Nonparam_SE_x", "Nonparam_CI_x_lower", "Nonparam_CI_x_upper",
    "Param_Bias_x_mad", "Param_SE_x_mad", "Param_CI_x_mad_lower", "Param_CI_x_mad_upper",
    "Nonparam_SE_cleanx", "Nonparam_CI_cleanx_lower", "Nonparam_CI_cleanx_upper",
    "Param_Bias_cleanx_mad", "Param_SE_cleanx_mad", "Param_CI_cleanx_mad_lower", "Param_CI_cleanx_mad_u
  ),

  Mean = c(
    bootstrap_x$mean$standard_error, bootstrap_x$mean$CI[1], bootstrap_x$mean$CI[2],
    p_b_x$mean$bias, p_b_x$mean$standard_error, p_b_x$mean$CI[1], p_b_x$mean$CI[2],
    bootstrap_cleanx$mean$standard_error, bootstrap_cleanx$mean$CI[1], bootstrap_cleanx$mean$CI[2],
    p_b_cleanx$mean$bias, p_b_cleanx$mean$standard_error, p_b_cleanx$mean$CI[1], p_b_cleanx$mean$CI[2]
  ),


  Trimmed_Mean = c(
    bootstrap_x$trimmed_mean$standard_error, bootstrap_x$trimmed_mean$CI[1], bootstrap_x$trimmed_mean$C
    p_b_x$trimmed_mean$bias, p_b_x$trimmed_mean$standard_error, p_b_x$trimmed_mean$CI[1], p_b_x$trimmed_
    bootstrap_cleanx$trimmed_mean$standard_error, bootstrap_cleanx$trimmed_mean$CI[1], bootstrap_cleanx
    p_b_cleanx$trimmed_mean$bias, p_b_cleanx$trimmed_mean$standard_error, p_b_cleanx$trimmed_mean$CI[1]
  )
)

print(results)
```

```
##                    Statistic          Mean Trimmed_Mean
## 1              Nonparam_SE_x  2.627569e-02  0.022711495
## 2         Nonparam_CI_x_lower  3.420387e-02 -0.005937803
## 3         Nonparam_CI_x_upper  1.365660e-01  0.080148030
## 4            Param_Bias_x_mad  5.963669e-05  0.046960431
## 5              Param_SE_x_mad  2.185546e-02  0.022343316
## 6        Param_CI_x_mad_lower  4.122087e-02  0.039622040
## 7        Param_CI_x_mad_upper  1.275485e-01  0.126475723
## 8          Nonparam_SE_cleanx  2.226939e-02  0.022276852
## 9    Nonparam_CI_cleanx_lower -4.842436e-02 -0.047006809
## 10   Nonparam_CI_cleanx_upper  3.881282e-02  0.040311970
## 11     Param_Bias_cleanx_mad  3.074035e-04 -0.004435905
## 12       Param_SE_cleanx_mad  2.155215e-02  0.022423103
## 13 Param_CI_cleanx_mad_lower -4.654166e-02 -0.050217858
## 14 Param_CI_cleanx_mad_upper  3.606084e-02  0.037927275
```

Observation:

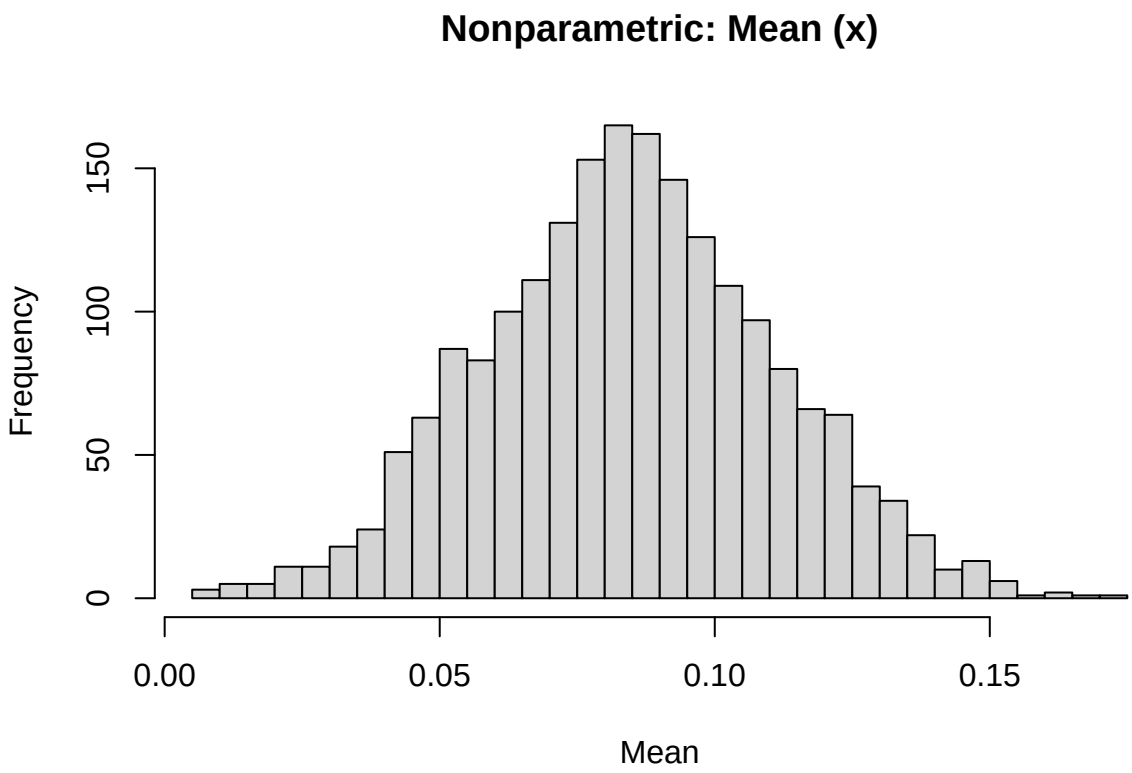
i. Contaminated Dataset (x):
   - The mean shows a negligible bias (~0.00031) with a standard error of ~0.0218, showing stability even thought the dataset is contaminated.
   - The trimmed mean does have a higher bias (~0.0471) but maintains a reasonable standard error (~0.0221) and a similar CI, indicating robustness.
   - The bias corrected estimated for trimmed mean is negative, showing that is adjusts for outliers.

ii. Clean Dataset (x.clean):
   - Both mean and trimmed mean have a negligible bias and small standard errors, showing that they are accurate when the dataset is clean.

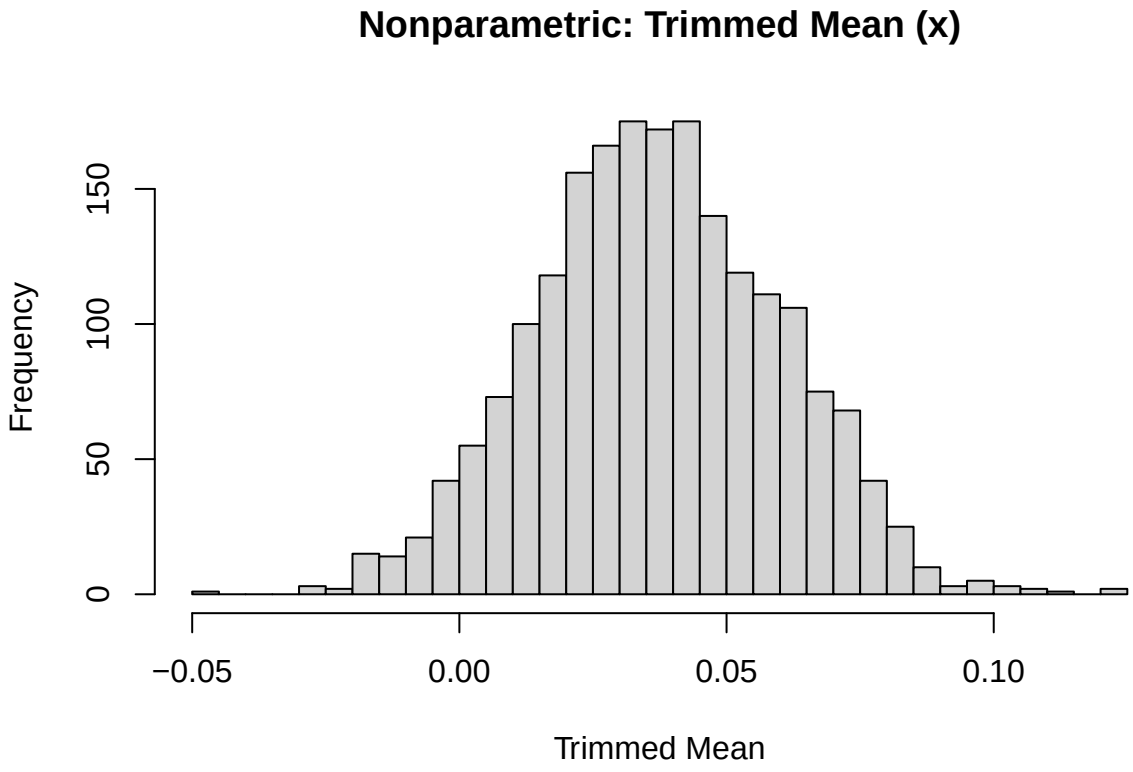
- The CI is narrower than the contaminated dataset, which is expected due to the the absence of extreme values.

The mean is slightly more sensitive to contamination. The trimmed mean is robust for both datasets.

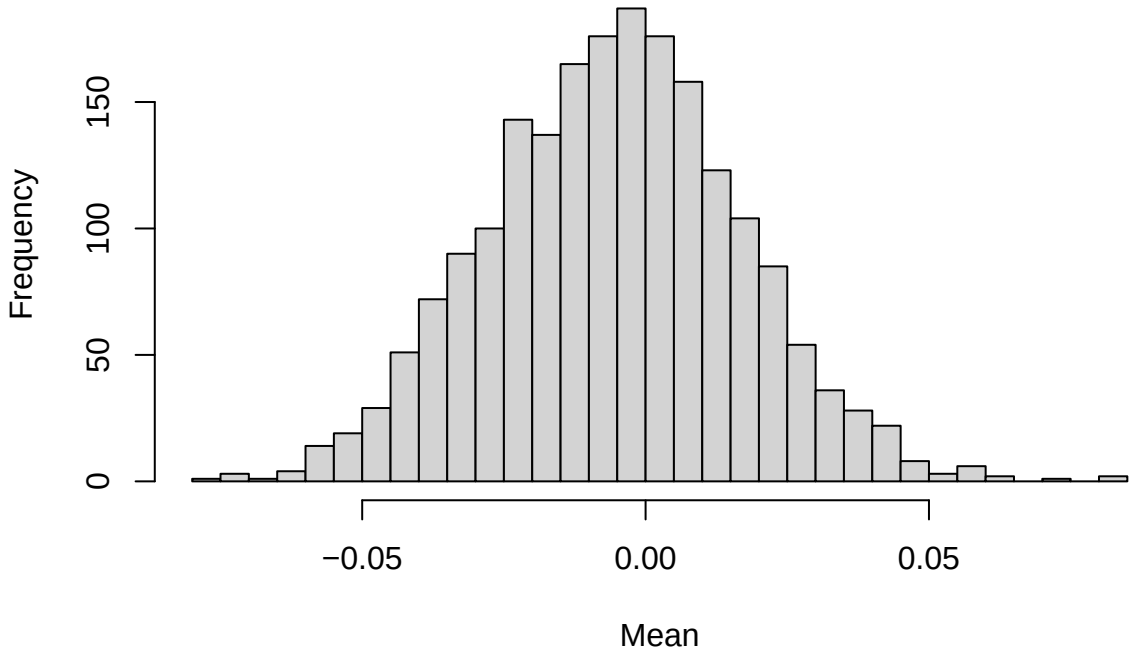
```
## Nonparametric histograms

## Dataset x
hist(bootstrap_x$mean$samples, breaks = 30,
     main = "Nonparametric: Mean (x)", xlab = "Mean")
```



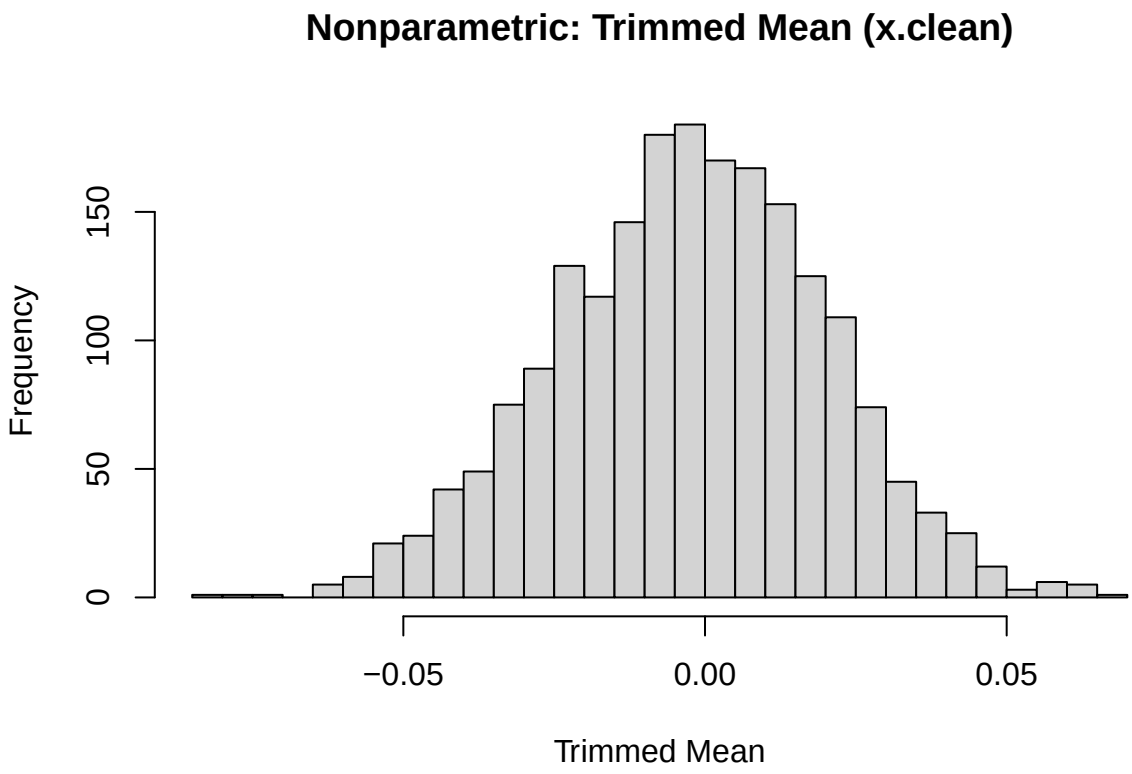

```
hist(bootstrap_x$trimmed_mean$samples, breaks = 30,
     main = "Nonparametric: Trimmed Mean (x)", xlab = "Trimmed Mean")
```




```
## Dataset x.clean
hist(bootstrap_cleanx$mean$samples, breaks = 30,
     main = "Nonparametric: Mean (x.clean)", xlab = "Mean")
```




```
hist(bootstrap_cleanx$trimmed_mean$samples, breaks = 30,
     main = "Nonparametric: Trimmed Mean (x.clean)", xlab = "Trimmed Mean")
```



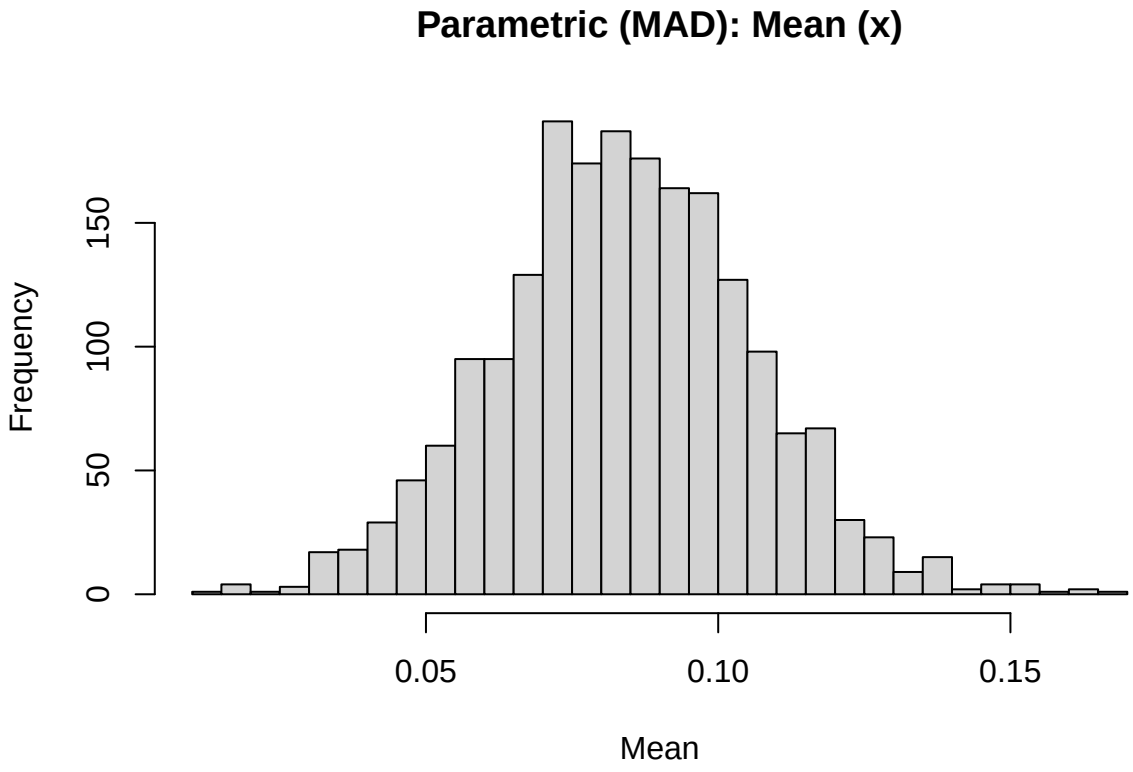
Observation:

i. Contaminated Dataset (x):
   - The distributions for both mean and trimmed mean are more towards the positive side because of contamination.
   - The trimmed mean shows a narrower spread, highlghting its sensitivity to the outliers.

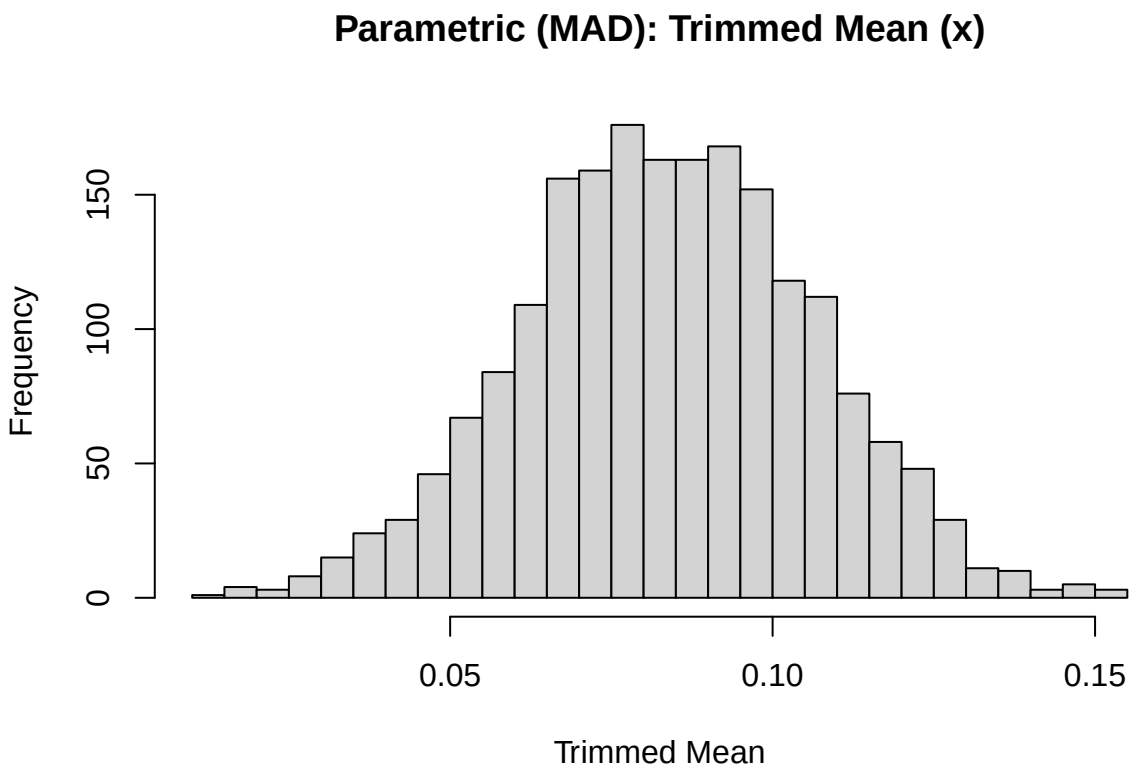
ii. Clean Dataset (x.clean):
   - The distributions for both measures are more symmetric and centred near zero. - Both measure have similar spreads.

The trimmed mean provides a more stable estimate for contaminated dataset while also maintaining comparable accuray to the mean in normal datasets.

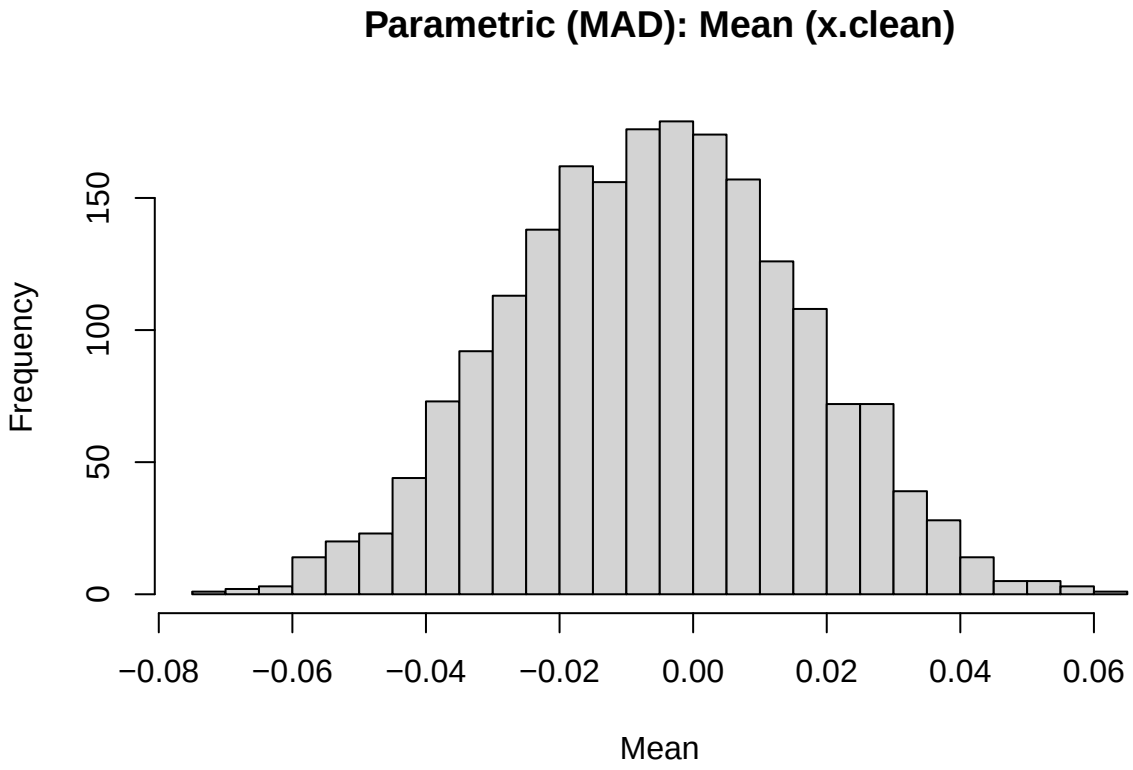
```
## Parametric histograms

## Dataset x
hist(p_b_x$mean$samples, breaks = 30,
     main = "Parametric (MAD): Mean (x)", xlab = "Mean")
```



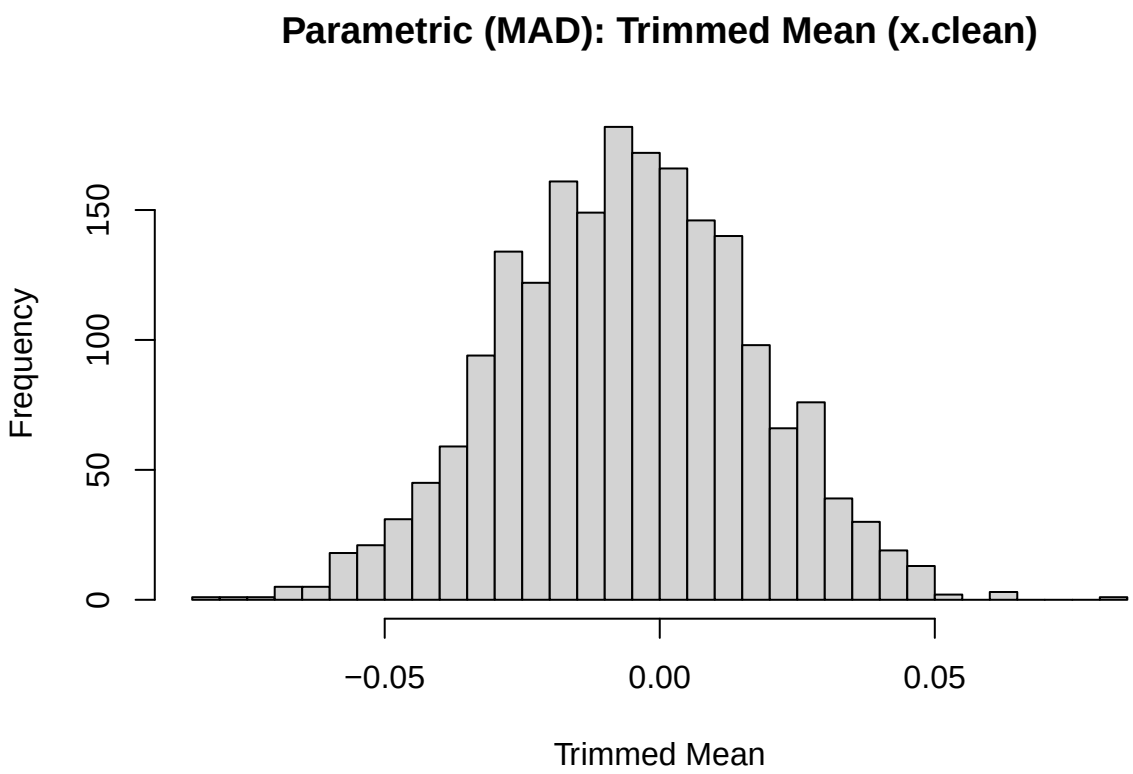

```
hist(p_b_x$trimmed_mean$samples, breaks = 30,
     main = "Parametric (MAD): Trimmed Mean (x)", xlab = "Trimmed Mean")
```




```
## Dataset x.clean
hist(p_b_cleanx$mean$samples, breaks = 30,
     main = "Parametric (MAD): Mean (x.clean)", xlab = "Mean")
```




```
hist(p_b_cleanx$trimmed_mean$samples, breaks = 30,
     main = "Parametric (MAD): Trimmed Mean (x.clean)", xlab = "Trimmed Mean")
```



Observation:

i. Contaminated Dataset (x):

- The distributions for both mean and trimmed mean are more towards the positive side because of contamination, with the mean having a wider spread.

- The trimmed mean exhibits a narrower distribution, showing its effectiveness to outliers.

ii. Clean Dataset (x.clean):

- The distributions for both measures are more symmetric and centred near zero. - Both measure have similar spreads, highlighting its efficiency in clean datasets.

The contaminated dataset results in wider distributions for both measures. The trimmed mean consistently shows narrower distributions, affirming its effectiveness with handling outliers. The trimmed mean provides a more stable estimate for contaminated dataset while also maintaining comparable accuray to the mean in normal datasets.

Task 3: Methodology of bootstrap

Bootstrapping is a statistical resampling methods that helps estimate the distribution of a statistic by repeatedly sampling from the observed dataset with replacement. Bootstrapping can be used when:
- the theoretical distribution of a statistic of interest is unknown or complicated.
- the sample size is too small for a straightforward statistical inference.
- the power calculations have to be performanced with a small pilot sample.

Key Steps in Bootstrapping:
1. Creating multiple samples with the same size as the original dataset by sampling with replacement.
2. Computing the required statistical estimates for each sample (mean, median, and trimmed median).
3. Creating the confidence intervals using percentiles of the distribution. For example, a 95% confidence interval is taken using 2.5th and 97.5th percentiles.

Parametric Bootstrapping: In paramtric bootstrapping, we first assume the data follows a certain distribution, and estimate measures from the data. From the assumed distributions, samples are generated and these improve the efficiency, if the assumptions hold.

Non-parametric bootstrapping:
In non-parametric sampling, resampling is done directly from the observed data without any distributional assumptions. This makes it ideal for robust measures like trimmed means or medians, especially when the dataset contains outliers.

Construction of Confidence Intervals:
The bootstrap distributions generates in both tasks were used to estimate the confidence interval (2.5th and 97.5th percentiles). These intervals give an estimate of the variability of the statistic (mean or median), and the width of the interval reflects the precision.

Key Observations:
- Non-parametric bootstrapping is flexible but since we do not rely on distributional assumptions, it is ideal for robust estimators like the median.
- Parametric bootstrapping, which uses distributional assumptions, is more efficient when the assumptions hold. It can also work better with contaminated datasets using robust estimators like MAD and trimmed mean.