Statistical Simulation and Computerintensive Methods

Exercise 4: Sample distribution and Central Limit Theorem

Markus Kiesel | 1228952

04.11.2020

Task 1

We consider the following 12 sample data points.

```r
P.samples <- 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)
```

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

We can consider bootstrap samples as permutations with repetition because the same data point can be chosen multiple times. So we can have k^n possible bootstrap samples.

```r
# number of possible bootstrap samples
12^12
```

```
## [1] 8.9161e+12
```

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

```r
# mean of original sample
mean(P.samples)
```

```
## [1] 4.840833
```

```r
# median of original sample
median(P.samples)
```

```
## [1] 4.935
```

1.3 Create 2000 bootstrap samples and compute their means.

```r
# means of 2000 bootstrap samples
B.means<- replicate(2000, mean(sample(P.samples, replace=TRUE)))
```

1.3.1 Compute the mean on the first 20 bootstrap means.

```r
# mean of 20 bootstrap samples
mean(B.means[1:20])
```

```
## [1] 4.843208
```

1.3.2 Compute the mean of the 200 bootstrap means.

```r
# mean of 200 bootstrap samples
mean(B.means[1:200])
```

```
## [1] 4.841087
```

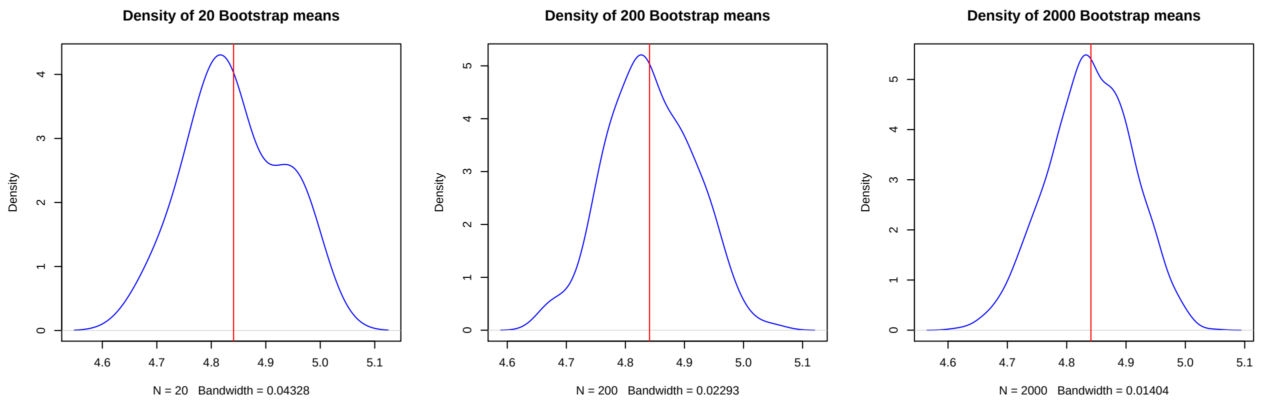
1.3.3 Compute the mean based on all 2000 bootstrap means.

```r
# mean of 2000 bootstrap samples
mean(B.means)
```

```
## [1] 4.83928
```

1.3.4 Visualise the distribution all the different bootstrap means to the sample mean. Does the Central Limit Theorem kick in?

```r
par(mfrow = c(1,3))
# plot density line 20 samples
plot(density(B.means[1:20]), col = "blue", main = "Density of 20 Bootstrap means")
# add vertical line for true mean
abline(v = mean(P.samples), col = "red")
# plot density line 200 samples
plot(density(B.means[1:200]), col = "blue", main = "Density of 200 Bootstrap means")
# add vertical line for true mean
abline(v = mean(P.samples), col = "red")
# plot density line 2000 samples
plot(density(B.means), col = "blue", main = "Density of 2000 Bootstrap means")
# add vertical line for true mean
abline(v = mean(P.samples), col = "red")
```



We can see in the above graph when we plot the density of our means and the true mean that the means of our bootstrap samples are normally distributed and centered around the true mean of our samples. As we increase the sample size the means are more normally distributed.

1.3.5 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.)

```
# 0.025 and 0.975 quantiles for 20 samples
q_20 <- quantile(B.means[1:20], c(0.025, 0.975), type = 1)
# 0.025 and 0.975 quantiles for 200 samples
q_200 <- quantile(B.means[1:200], c(0.025, 0.975), type = 1)
# 0.025 and 0.975 quantiles for 2000 samples
q_2000 <- quantile(B.means, c(0.025, 0.975), type = 1)
# confidence interval of mean under assumption of normality (using t.test)
q_true <- t.test(P.samples)$conf.int[1:2]

# combine CIs to one table
row.names <- c("20 samples",
               "200 samples",
               "2000 samples",
               '"true" estimate')
col.names <- c("2.5% quantile", "97.5% quantile")
comparison_table <- data.frame(
  matrix(c(q_20, q_200, q_2000, q_true), 4, 2, byrow = TRUE),
  row.names = row.names)
colnames(comparison_table) <- col.names
```

As we can see in Table 1 the confidence intervals get better as we increase the sample size. Whereas 20 and 200 samples are to small for creating reasonable estimates with bootstrapping, 2000 samples produces very good estimates. We see again with this CI that the bootstrap means are normally distributed.

Table 1: Confidence Intervals for Bootstrap means

	2.5% quantile	97.5% quantile
20 samples	4.678333	4.995000
200 samples	4.685833	4.976667
2000 samples	4.695833	4.972500
"true" estimate	4.674344	5.007323

1.4 Create 2000 bootstrap samples and compute their medians.

```
# medians of 2000 bootstrap samples
B.medians <- replicate(2000, median(sample(P.samples, replace=TRUE)))
```

1.4.1 Compute the mean on the first 20 bootstrap medians.

```
# mean of first 20 bootstrap medians
mean(B.medians[1:20])
```

```
## [1] 4.905
```

1.4.2 Compute the mean of the first 200 bootstrap medians.

```r
# mean of first 200 bootstrap medians
mean(B.medians[1:200])
```

```
## [1] 4.909225
```

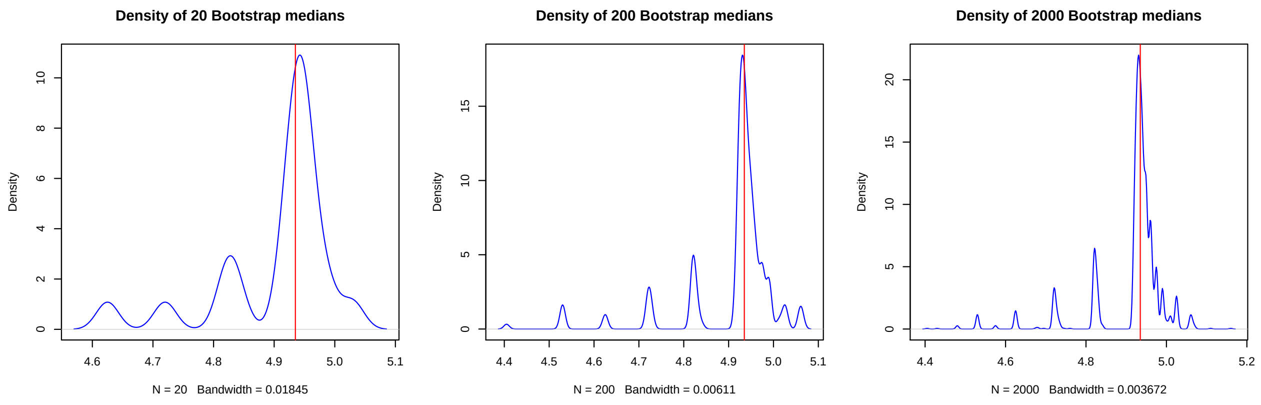
1.4.3 Compute the mean based on all 2000 bootstrap medians.

```r
# mean of all bootstrap medians
mean(B.medians)
```

```
## [1] 4.914428
```

1.4.4 Visualise the distribution all the different bootstrap medians to the sample median.

```r
par(mfrow = c(1,3))
# plot density line 20 samples
plot(density(B.medians[1:20]), col = "blue", main = "Density of 20 Bootstrap medians")
# add vertical line for true median
abline(v = median(P.samples), col = "red")
# plot density line 200 samples
plot(density(B.medians[1:200]), col = "blue", main = "Density of 200 Bootstrap medians")
# add vertical line for true median
abline(v = median(P.samples), col = "red")
# plot density line 2000 samples
plot(density(B.medians), col = "blue", main = "Density of 2000 Bootstrap medians")
# add vertical line for true median
abline(v = median(P.samples), col = "red")
```



We can see in see in the above Figure that the bootstrap medians are not distributed normally and has some weird distribution as we increase the number of bootstrap samples

1.4.5 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.

```r
# 0.025 and 0.975 quantiles for 20 samples
q_20 <- quantile(B.medians[1:20], c(0.025, 0.975), type = 1)
# 0.025 and 0.975 quantiles for 200 samples
q_200 <- quantile(B.medians[1:200], c(0.025, 0.975), type = 1)
# 0.025 and 0.975 quantiles for 2000 samples
q_2000 <- quantile(B.medians, c(0.025, 0.975), type = 1)
```

```r
# combine CIs to one table
row.names <- c("20 samples",
               "200 samples",
               "2000 samples")
col.names <- c("2.5% quantile", "97.5% quantile")
comparison_table <- data.frame(
  matrix(c(q_20, q_200, q_2000), 3, 2, byrow = TRUE),
  row.names = row.names)
colnames(comparison_table) <- col.names
```

We can see in Table 2 that the confidence intervals increases with the number of bootstram medians.

Table 2: Confidence Intervals for Bootstrap medians

	2.5% quantile	97.5% quantile
20 samples	4.625	5.030
200 samples	4.530	5.030
2000 samples	4.625	5.025

Task 2

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
set.seed(1234)
# create normal distributed sample
x.clean <- rnorm(1960)
# create uniform distributed sample
x.cont <- runif(40, 4, 5)
# combine both samples
x <- c(x.clean, x.cont)
# set seed to immatriculation number
set.seed(1228952)
```

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

```r
# median of sample x
median(x)
```

```
## [1] 0.0113797
```

```r
# mean of sample x
mean(x)
```

```
## [1] 0.08395508
```

```r
# trimmed mean of sample x
mean(x, trim = 0.05)
```

```
## [1] 0.03683294
```

```r
# median of sample x.clean
median(x.clean)
```

```
## [1] -0.0172536
```

```r
# mean of sample x.clean
mean(x.clean)
```

```
## [1] -0.005968976
```

```r
# trimmed mean of sample x.clean
mean(x.clean, trim = 0.05)
```

```
## [1] -0.001462623
```

2.3 Use nonparametric bootstrap (for x and x.clean) to calculate

- standard error
- 95 percentile CI of all 3 estimators.

First we use nonparametric bootstrap by sampling from our samples x and x.clean.

```r
# sample size of one bootsrap
n <- 100
# number of bootstrap samples
m <- 2000

# medians of bootstrap samples from x
Bnp.x.medians <- replicate(m, median(sample(x, n, replace=TRUE)))
# means of bootstrap samples from x
Bnp.x.means <- replicate(m, mean(sample(x, n, replace=TRUE)))
# trimmed means of bootstrap samples from x
Bnp.x.tmeans <- replicate(m, mean(sample(x, n, replace=TRUE), trim = 0.05))

# medians of bootstrap samples from x.clean
Bnp.xclean.medians <- replicate(m, median(sample(x.clean, n, replace=TRUE)))
# means of bootstrap samples from x.clean
Bnp.xclean.means <- replicate(m, mean(sample(x.clean, n, replace=TRUE)))
# trimmeds mean of bootstrap samples from x.clean
Bnp.xclean.tmeans <- replicate(m, mean(sample(x.clean, n, replace=TRUE), trim = 0.05))
```

Now we can calculate the standard error of each estimator.

```r
# standard error of the median of sample x
np.x.median.se <- sd(Bnp.x.medians)
# standard error of the mean of sample x
np.x.mean.se <- sd(Bnp.x.means)
# stadard error of the trimmed mean of sample x
np.x.tmean.se <- sd(Bnp.x.tmeans)

# standard error of the median of sample x.clean
np.xclean.median.se <- sd(Bnp.xclean.medians)
# standard error of the mean of sample x.clean
np.xclean.mean.se <- sd(Bnp.xclean.means)
# stadard error of the trimmed mean of sample x.clean
np.xclean.tmean.se <- sd(Bnp.xclean.tmeans)
```

Last, we can calculate the 95 percentile CI of all 3 estimators.

```r
# 95 percentile CI median x
np.x.median.ci <- quantile(Bnp.x.medians, c(0.025, 0.975), type = 1)
# 95 percentile CI mean x
np.x.mean.ci <- quantile(Bnp.x.means, c(0.025, 0.975), type = 1)
# 95 percentile CI trimmed mean x
np.x.tmean.ci <- quantile(Bnp.x.tmeans, c(0.025, 0.975), type = 1)

# 95 percentile CI median x.clean
np.xclean.median.ci <- quantile(Bnp.xclean.medians, c(0.025, 0.975), type = 1)
# 95 percentile CI mean x.clean
np.xclean.mean.ci <- quantile(Bnp.xclean.means, c(0.025, 0.975), type = 1)
# 95 percentile CI trimmed mean x.clean
np.xclean.tmean.ci <- quantile(Bnp.xclean.tmeans, c(0.025, 0.975), type = 1)
```

2.4 Use parametric bootstrap (based on x and x.clean) to calculate

- 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.

First we use parametric bootstrap by sampling from our original distribution.

```r
# sample size of one bootsrap
n <- 100
# number of bootstrap samples
m <- 2000
# function to create samples from original distributions
para_samples <- function(n) {
  perc_unif <- 40 / 2000
  n_unif <- n * perc_unif
  n_norm <- n * (1 - perc_unif)
  return(c(rnorm(n_norm), runif(n_unif, 4, 5)))
}

# means of bootstrap samples from distribution of x
Bp.x.means <- replicate(m, mean(para_samples(n)))
# trimmed means of bootstrap from distribution of x
Bp.x.tmeans <- replicate(m, mean(para_samples(n), trim = 0.05))

# means of bootstrap samples from distribution of x.clean
Bp.xclean.means <- replicate(m, mean(rnorm(n)))
# trimmeds mean of bootstrap from distribution of x.clean
Bp.xclean.tmeans <- replicate(m, mean(rnorm(n), trim = 0.05))
```

Now we calculate the bias of our estimators.

```r
# bias of mean from distribution of x
p.x.mean.bias <- mean(Bp.x.means) - mean(x)
# bias of trimmed mean from distribution of x
p.x.tmean.bias <- mean(Bp.x.tmeans) - mean(x, trim = 0.05)

# bias of mean from distribution of x.clean
```

```r
p.xclean.mean.bias <- mean(Bp.xclean.means) - mean(x)
# bias of mean from distribution of x.clean
p.xclean.tmean.bias <- mean(Bp.xclean.tmeans) - mean(x, trim = 0.05)
```

Next, we calculate the standard error.

```r
# standard error of the mean of x
p.x.mean.se <- sd(Bp.x.means)
# stadard error of the trimmed mean of x
p.x.tmean.se <- sd(Bp.x.tmeans)

# standard error of the mean of x.clean
p.xclean.mean.se <- sd(Bp.xclean.means)
# stadard error of the trimmed mean of x.clean
p.xclean.tmean.se <- sd(Bp.xclean.tmeans)
```

Next, we can calculate the 95 percentile CI of both estimators.

```r
# 95 percentile CI mean x
p.x.mean.ci <- quantile(Bp.x.means, c(0.025, 0.975), type = 1)
# 95 percentile CI trimmed mean x
p.x.tmean.ci <- quantile(Bp.x.tmeans, c(0.025, 0.975), type = 1)

# 95 percentile CI mean x.clean
p.xclean.mean.ci <- quantile(Bp.xclean.means, c(0.025, 0.975), type = 1)
# 95 percentile CI trimmed mean x.clean
p.xclean.tmean.ci <- quantile(Bp.xclean.tmeans, c(0.025, 0.975), type = 1)
```

Last we bias corrected estimate for the mean and the trimmed mean.

The sample mean is an unbiased estimator of the population mean.

2.5 Compare and summarize your findings with tables and graphically.

```r
# results of 2.3
row.names <- c("bootstrap medians from x",
               "bootstrap means from x",
               "bootstrap trimmed means from x",
               "bootstrap medians from x.clean",
               "bootstrap means from x.clean",
               "bootstrap trimmed means from x.clean")
# results for standard error
np.se <- c(np.x.median.se, np.x.mean.se, np.x.tmean.se,
          np.xclean.median.se, np.xclean.mean.se, np.xclean.tmean.se)
# results for confidence intervals
np.ci <- matrix(c(np.x.median.ci, np.x.mean.ci, np.x.tmean.ci,
           np.xclean.median.ci, np.xclean.mean.ci, np.xclean.tmean.ci),
           6, 2, byrow = TRUE)
# create comparison table
comparison_table <- data.frame(
  matrix(c(np.se, np.ci), 6, 3),
  row.names = row.names)
colnames(comparison_table) <- c("standard error", "2.5 percentile", "97.5 percentile" )
```

Table 3: Non-parametric bootstrapping estimates

	standard error	2.5 percentile	97.5 percentile
bootstrap medians from x	0.1199975	-0.2158668	0.2596840
bootstrap means from x	0.1134341	-0.1289395	0.3075372
bootstrap trimmed means from x	0.1037057	-0.1630677	0.2478366
bootstrap medians from x.clean	0.1154371	-0.2318341	0.2140804
bootstrap means from x.clean	0.0991916	-0.2076081	0.1816488
bootstrap trimmed means from x.clean	0.1001807	-0.1892193	0.1899574

```r
# results of 2.4
row.names <- c("bootstrap means from x",
               "bootstrap trimmed means from x",
               "bootstrap means from x.clean",
               "bootstrap trimmed means from x.clean")
# results for bias
p.bias <- c(p.x.mean.bias, p.x.tmean.bias,
             p.xclean.mean.bias, p.xclean.tmean.bias)
# results for standard error
p.se <- c(p.x.mean.se, p.x.tmean.se,
          p.xclean.mean.se, p.xclean.tmean.se)
# results for confidence intervals
p.ci <- matrix(c(p.x.mean.ci, p.x.tmean.ci,
           p.xclean.mean.ci, p.xclean.tmean.ci),
          4, 2, byrow = TRUE)

comparison_table <- data.frame(
  matrix(c(p.bias, p.se, p.ci), 4, 4),
  row.names = row.names)
colnames(comparison_table) <- c("bias", "standard error", "2.5 percentile", "97.5 percentile" )
```

Table 4: Parametric bootstrapping estimates

	bias	standard error	2.5 percentile	97.5 percentile
bootstrap means from x	0.0036460	0.1010012	-0.1100905	0.2868637
bootstrap trimmed means from x	0.0041782	0.1010328	-0.1533274	0.2356850
bootstrap means from x.clean	-0.0849995	0.0978142	-0.1985379	0.1853834
bootstrap trimmed means from x.clean	-0.0404674	0.1033455	-0.2058844	0.1965346

Task 3

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

Bootstrapping is especially useful when we have a small sample and need to calculate confidence intervals or standard errors for our estimates. Because we do not need to know the underlying distribution of our samples. One useful use-case is that we can calculate confidence intervals by bootstrapping and visually check if an estimate is outside this CI. For example if an estimate is outside the 95% CI the p-value is < 0.05 and we can say there is a statistically significant difference. We can further make use of the central limit theorem when using bootstrapping. What we have to look for is that outliers do not have to much influence on the creation of our bootstrap samples as seen in our experiments.