Statistical Simulation and Computerintensive Methods

Exercise 5: Sampling Intervals for Models

Markus Kiesel | 1228952

24.11.2021

Contents

Task 1

Consider a two sample problem and the hypothesis H0 μ 1= μ 2 vs H1: μ 1 $\neq$ μ 2 , where μ 1 and μ 2 are the corresponding sample locations. The two samples are:

```
x1 <- c(-0.673, -0.584, 0.572, -0.341, -0.218, 0.603, -0.415, -0.013, 0.763, 0.804,
    0.054, 1.746, -0.472, 1.638, -0.578, 0.947, -0.329, -0.188, 0.794, 0.894, -1.227,
    1.059)

x2 <- c(0.913, -0.639, 2.99, -5.004, 3.118, 0.1, 1.128, 0.579, 0.32, -0.488, -0.994,
    -0.212, 0.413, 1.401, 0.007, 0.568, -0.005, 0.696)
```

First we have a look at some properties of our two samples.

```
# length of sample x1
n1 <- length(x1)
n1
```

```
## [1] 22
```

```
# length of sample x2
n2 <- length(x2)
n2
```

```
## [1] 18
```

```
# standard diviation of original samples
sd(x1)
```

```
## [1] 0.7980614
```

```
sd(x2)
```

```
## [1] 1.707192
```

```
# absolute difference in mean
abs(mean(x1) - mean(x2))
```

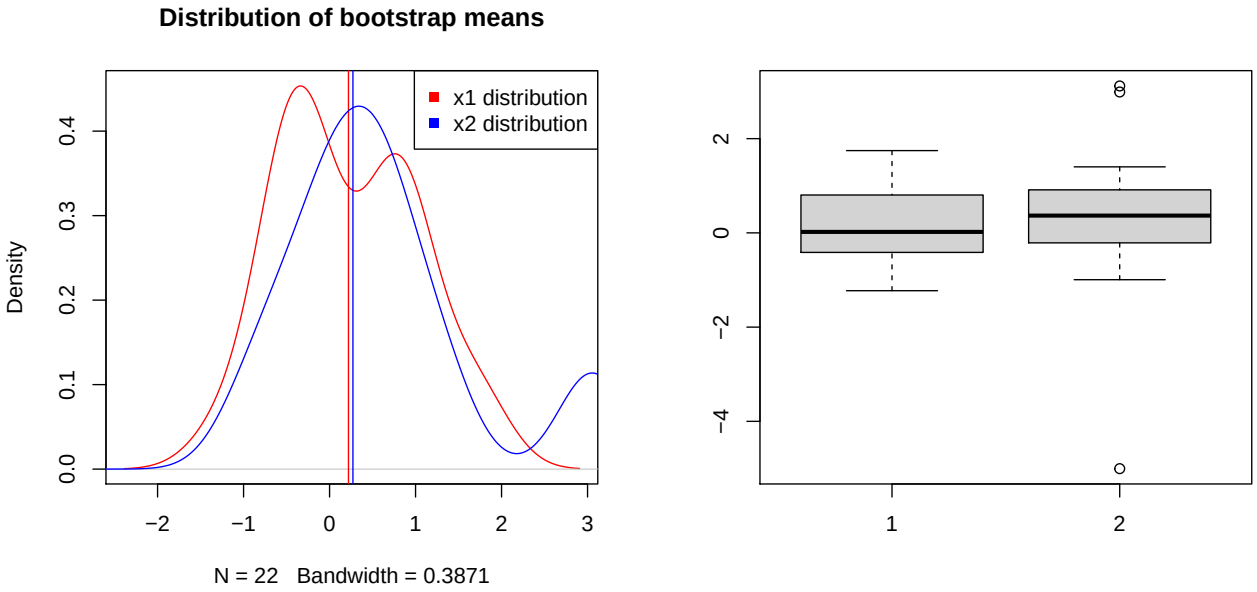
```
## [1] 0.05190404
```

We notice that the two samples have different sizes, and different standard deviations. The difference in mean (our test statistic) is 0.0519.

1.1

Plot the data in a way which visualizes the comparison of means appropriately.

```
par(mfrow = c(1, 2))
plot(density(x1), type = "l", col = "red", main = "Distribution of bootstrap means")
abline(v = mean(x1), col = "red")
lines(density(x2), col = "blue")
abline(v = mean(x2), col = "blue")
legend("topright", legend = c("x1 distribution", "x2 distribution"), pch = c(15,
    15), col = c("red", "blue"))
boxplot(x1, x2)
```



The plots show us that the mean between the two samples are very similar but the distribution in the values is different. We also observe that x2 has several outliers.

1.2

Consider different sampling schemes, such as

1. Sampling with replacement from each group
2. Centering both samples and then resample from the combined samples n1 and n2 times.

Argue for choice what is more natural and which advantages or disadvantages may apply.

Answer:

The key in bootstrapping hypothesis tests is that the sampling should reflect the distribution under H0. Even if the samples do not follow it.

If the null hypothesis is correct, any of the values could have come equally well from x1 or x2. If we canter and combine all samples from both groups together we only assume a common mean (null Hypothesis). Thus, the second approach is more natural.

1.3

Bootstrap using both strategies mentioned above using the t-test statistic. Calculate the bootstrap p-value based on 10000 bootstrap samples and 0.95 as well as 0.99 confidence intervals. Make your decision at the significance level 0.05 or 0.01, respectively.

We have small sample sizes (< 30) and want to use the t-distribution in our tests statistic. As test statistic we use Welch's t-test, which is used only when the two population variances are not assumed to be equal and the two sample sizes may not be equal and hence must be estimated separately.

```
# number of bootstrap samples
m <- 10000
# t statistic to compare 2 groups
Tstatistic <- function(x1, x2) {
    (mean(x1) - mean(x2))/(sqrt(sd(x1)^2 * 1/n1 + sd(x2)^2 * 1/n2))
}
# t statistic for original samples
TX <- Tstatistic(x1, x2)
```

```r
# function to calculate p values and confidence intervals
calc_p_ci <- function(test_stat_boot, print = TRUE) {
    # calculate bootstrap p-value
    p <- (sum(abs(test_stat_boot) > abs(TX)) + 1)/(m + 1)
    # 0.95 confidence interval
    ci95 <- quantile(test_stat_boot, c(0.025, 0.975))
    # 0.99 confidence interval
    ci99 <- quantile(test_stat_boot, c(0.005, 0.995))

    # print values
    if (print) {
        print(paste("P value:", round(p, 4)))
        print(paste("95% CI:", round(ci95[1], 4), round(ci95[2], 4)))
        print(paste("99% CI:", round(ci99[1], 4), round(ci99[2], 4)))
    }

    return(list(p = p, ci95 = ci95, ci99 = ci99))
}
```

First we follow the approach for sampling with replacement from each group.

```r
# create t statistics for all m bootstrap samples by sampling from each group
TX_bygroup <- replicate(m, Tstatistic(x1 = sample(x1, replace = TRUE), x2 = sample(x2,
    replace = TRUE)))
resuts_bygroup <- calc_p_ci(TX_bygroup)
```

```
## [1] "P value: 0.9057"
## [1] "95% CI: -2.5945 1.6528"
## [1] "99% CI: -3.2607 2.1686"
```

Next, we center and combine the data and sample form the combined data.

```r
# center data so they share a common mean (new mean 0)
x1_cent <- x1 - mean(x1)
x2_cent <- x2 - mean(x2)
x_cent <- c(x1_cent, x2_cent)
# create t statistics for all m bootstrap samples by sampling from the combined
# group
TX_cent <- replicate(m, Tstatistic(x1 = sample(x_cent, n1, replace = TRUE), x2 = sample(x_cent,
    n2, replace = TRUE)))
result_cent <- calc_p_ci(TX_cent)
```

```
## [1] "P value: 0.9155"
## [1] "95% CI: -2.0167 1.9441"
## [1] "99% CI: -2.5734 2.5442"
```

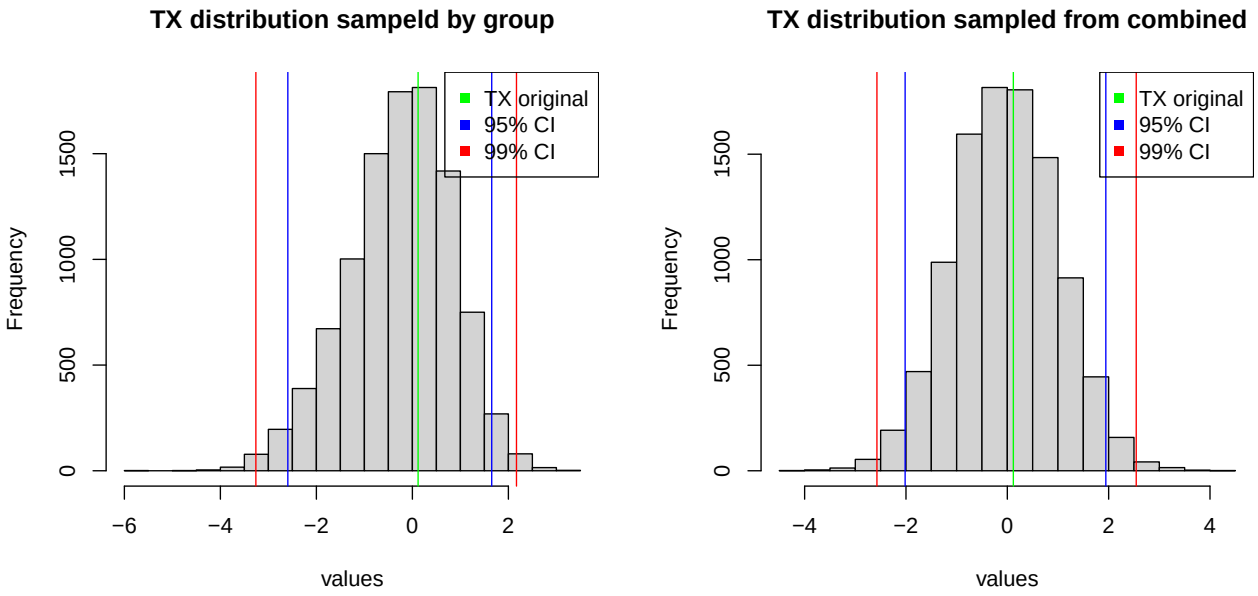
Next, we visualize the results.

```r
par(mfrow = c(1, 2))

values <- TX_bygroup
results <- resuts_bygroup
hist(values, main = "TX distribution sampeld by group")
abline(v = abs(TX), col = "green")
abline(v = c(results$ci95), col = "blue")
abline(v = c(results$ci99), col = "red")
legend("topright", legend = c("TX original", "95% CI", "99% CI"), pch = c(15, 15),
```

```
    col = c("green", "blue", "red"))

values <- TX_cent
results <- result_cent
hist(values, main = "TX distribution sampled from combined")
abline(v = abs(TX), col = "green")
abline(v = c(results$ci95), col = "blue")
abline(v = c(results$ci99), col = "red")
legend("topright", legend = c("TX original", "95% CI", "99% CI"), pch = c(15, 15),
    col = c("green", "blue", "red"))
```



For both sampling approaches the p value is near 0.91 and we can not reject H0 for neither the 5% nor the 1% significance level.

The plots showing the distribution of the difference in means and confidence intervals show that our difference in means of our original sample has to be significantly higher to reject H0.

1.4

What would be a permutation version of the test? Implement the corresponding permutation test and obtain p-value and confidence intervals as in 3. to get a corresponding test decision at the same significance levels.

A permutation test exploits special symmetry that exists under H0 to create a permutation distribution of the test statistic. In our two-sample hypothesis test this means all permutations of the combined sample are equally probable. Instead of sampling with replacement we create permutations of our combined samples and split the permutation into two samples.

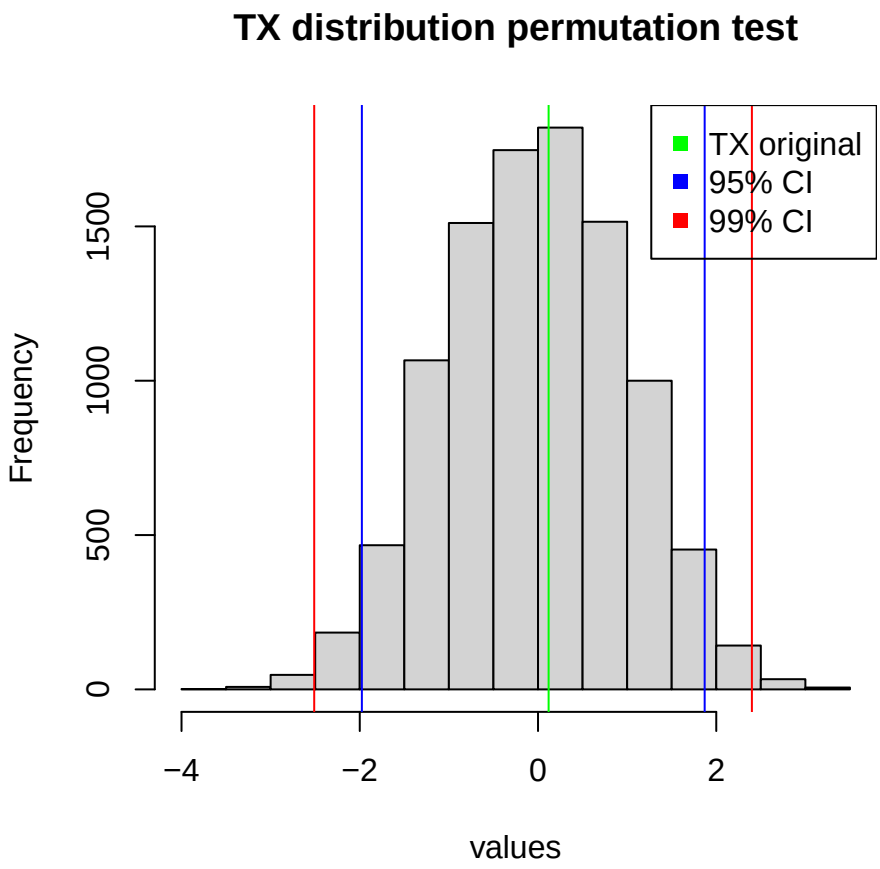
```
perm_test <- function() {
    # sample without replacement from combined observations x1 and x2
    permutation <- sample(c(x1, x2))
    # split into samples z and y
    lower_split = length(x1)
    upper_split = lower_split + 1
    size = length(c(x1, x2))
    z <- permutation[1:lower_split]
    y <- permutation[upper_split:size]
```

```
    # return test statistic for samples
    return(Tstatistic(z, y))
}

# create test statistics m times
TX_perm <- replicate(m, perm_test())
result_perm <- calc_p_ci(TX_perm)
```

```
## [1] "P value: 0.9125"
## [1] "95% CI: -1.9763 1.8697"
## [1] "99% CI: -2.5111 2.3998"
```

```
values <- TX_perm
results <- result_perm
hist(values, main = "TX distribution permutation test")
abline(v = abs(TX), col = "green")
abline(v = c(results$ci95), col = "blue")
abline(v = c(results$ci99), col = "red")
legend("topright", legend = c("TX original", "95% CI", "99% CI"), pch = c(15, 15),
    col = c("green", "blue", "red"))
```



The p value and confidence intervals for the permutation test is very similar to the results we obtained with using bootstrap. We can not reject H0.

1.5

The Wilxocon rank sum test statistic is the sum of ranks of the observations of sample 1 computed in the combined sample. Use bootstrapping with both strategies mentioned above and obtain p-value and confidence

intervals as in 3. to get a corresponding test decision at the same significance levels.

```
# create ranks
wilcox_rank_sum <- function(x1, x2) {
    x_combined <- c(x1, x2)
    ranks <- c()
    # simplified ranking ignoring tied values (all values not equal in x1 and
    # x2)
    ranks[order(x_combined)] <- 1:length(x_combined)
    r1 <- sum(ranks[1:n1])
    u1 <- r1 - (n1 * (n1 + 1)/2)
    return(u1)
}
# calculate test statistic for original sample
TX <- wilcox_rank_sum(x1, x2)

# calculate wilcox rank sum test statistic by group
TX_wil_bygroup <- replicate(m, wilcox_rank_sum(x1 = sample(x1, replace = TRUE), x2 = sample(x2,
    replace = TRUE)))
result_wil_bygroup <- calc_p_ci(TX_wil_bygroup)
```

```
## [1] "P value: 0.4817"
## [1] "95% CI: 108 254"
## [1] "99% CI: 86.995 277"
```

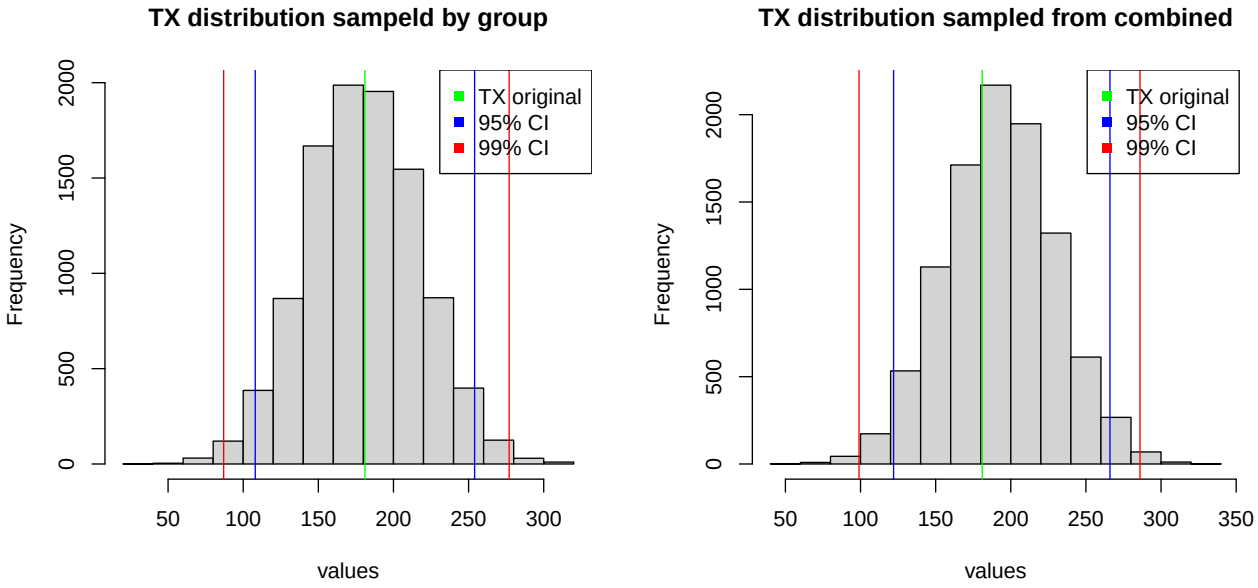
```
# calculate wilcox rank sum test statistic by centering and combining
TX_wil_cent <- replicate(m, wilcox_rank_sum(x1 = sample(x_cent, n1, replace = TRUE),
    x2 = sample(x_cent, n2, replace = TRUE)))
result_wil_cent <- calc_p_ci(TX_wil_cent)
```

```
## [1] "P value: 0.6297"
## [1] "95% CI: 122 266"
## [1] "99% CI: 99 286"
```

```
par(mfrow = c(1, 2))

values <- TX_wil_bygroup
results <- result_wil_bygroup
hist(values, main = "TX distribution sampeld by group")
abline(v = abs(TX), col = "green")
abline(v = c(results$ci95), col = "blue")
abline(v = c(results$ci99), col = "red")
legend("topright", legend = c("TX original", "95% CI", "99% CI"), pch = c(15, 15),
    col = c("green", "blue", "red"))

values <- TX_wil_cent
results <- result_wil_cent
hist(values, main = "TX distribution sampled from combined")
abline(v = abs(TX), col = "green")
abline(v = c(results$ci95), col = "blue")
abline(v = c(results$ci99), col = "red")
legend("topright", legend = c("TX original", "95% CI", "99% CI"), pch = c(15, 15),
    col = c("green", "blue", "red"))
```



The Wilcoxon rank-sum test is a non-parametric test of the null hypothesys that is used to test if one distribution is stochastically greater than the other. This makes it not directly comparable with our previous methods.

For this test statistic we can observe that there is a difference in p values and CIs depending on how we sample from the data. With both approaches we are not able to reject the H0.

1.6

Compare your results to the results using t.test and wilcox.test.

```
# Two Sample t-test
t.test(x1, x2)
```

```
##
##  Welch Two Sample t-test
##
## data:  x1 and x2
## t = -0.11881, df = 23.027, p-value = 0.9065
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.9556081  0.8518000
## sample estimates:
## mean of x mean of y
## 0.2198182 0.2717222
```

When we compare the results to our approaches using bootstrap or permutation test we get a very similar p-value.

Although the p-value is very similar the confidence interval is calculated very differently.

```
# Wilcoxon rank sum test
wilcox.test(x1, x2)
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  x1 and x2
## W = 181, p-value = 0.6572
```

```
## alternative hypothesis: true location shift is not equal to 0
```

For this test we get a similar p value as our implementation of the test statistic when we sampled from the centered and combined samples which is the more natural approach. The statistic of our original sample was correctly computed with the value 181.

Task 2

Consider the model y=3+2*x1+x2 + ϵ where x1 comes from a normal distribution with mean 2 and variance 3, x2 comes from a uniform distribution between 2 and 4 and ϵ from a student's t distribution with 5 degrees of freedom . In addition, there is a predictor x3 coming from a uniform distribution between -2 and 2.

2.1

Create a sample of size 200 from the model above and for the independent predictor x.

```
# create dummy data
n <- 200
x1 <- rnorm(n, 2, sqrt(3))
x2 <- runif(n, 2, 4)
eps <- rt(n, 5)
y = 3 + 2 * x1 + x2 + eps

x3 <- runif(n, -2, 2)
df <- data.frame(y, x1, x2, x3)
```

2.2

Do residual bootstrap for linear regression and fit the model y~x1+x2+x3 . Get the percentile CI for the coefficients. Can you exclude x3 ?

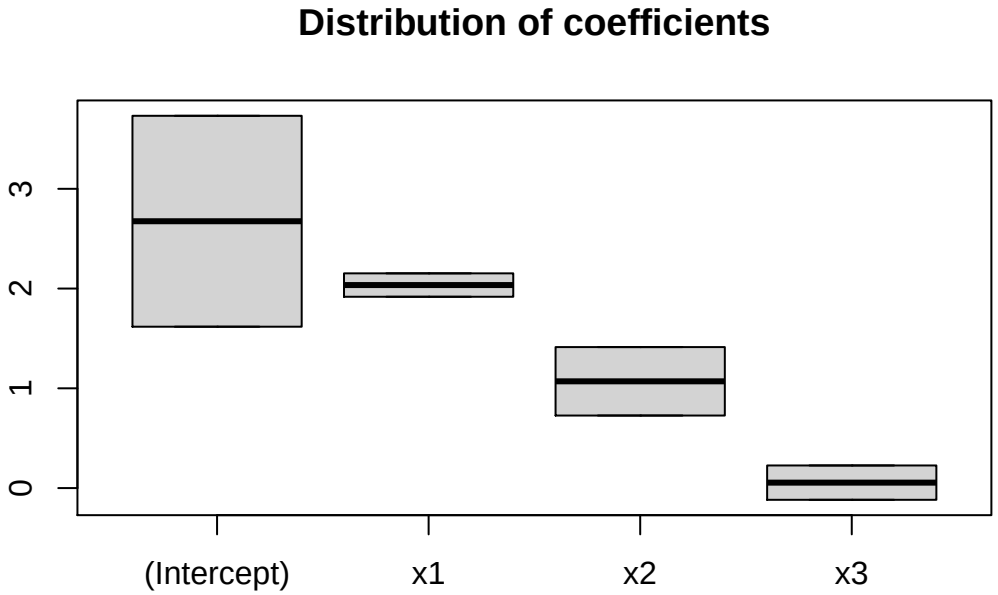
In this approach we use non-parametric bootstrap and sample directly from the residuals.

```
# fit linear model
fit <- lm(y ~ ., data = df)
# extract residuals
res <- resid(fit)
# extract y_hat
y_hat <- fitted(fit)
# non-parametric residual bootstrap
res_boot <- replicate(m, sample(res, replace = TRUE))

fit_boot <- lm(y_hat + res_boot ~ x1 + x2 + x3)
cis <- apply(coef(fit_boot), 1, quantile, probs = c(0.025, 0.975))
t(cis)
```

```
##                   2.5%     97.5%
## (Intercept)  1.6175914 3.7317966
## x1           1.9176585 2.1523987
## x2           0.7272039 1.4132114
## x3          -0.1176196 0.2266456
```

```
boxplot(cis, main = "Distribution of coefficients")
```



We see that the confidence interval of x3 is very near 0 for almost all bootstraped models which stronly suggests that x3 is just noise.

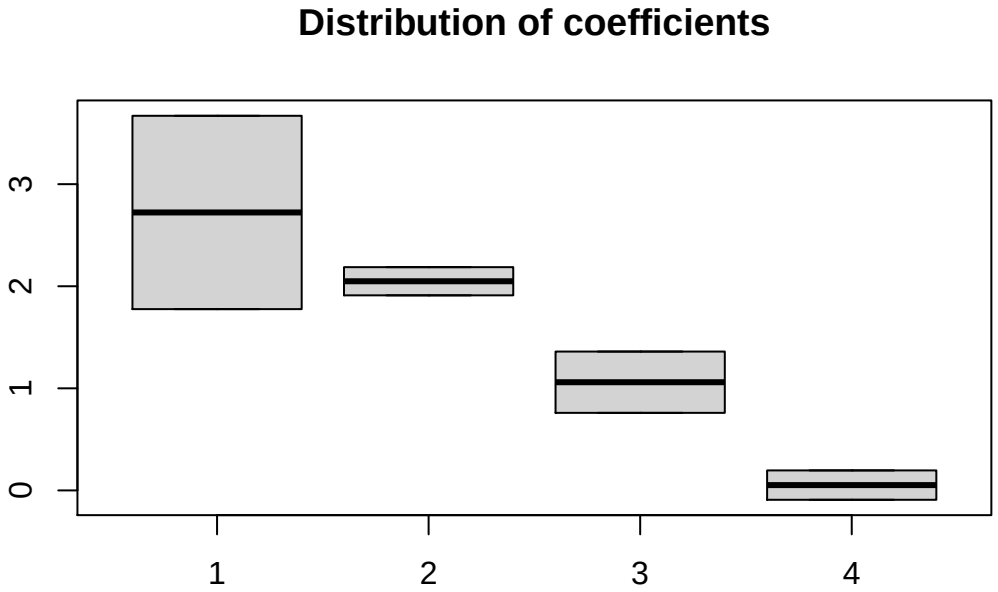
2.3

Do pairs bootstrap for linear regression and fit the model y~x1+x2+x3 . Get the percentile CI for the coefficients. Can you exclude x3 ?

```
# use pairs bootstrap
coeffic <- matrix(0, 4, m)
for (i in 1:m) {
    # sample indexes of samples with replacement
    s <- sample(nrow(df), n, replace = TRUE)
    # train model on selected samples and extract coefficents
    coeffic[, i] <- coef(lm(y ~ ., data = df, subset = s))
}
# calculate confidence intervals for coefficents
cis <- apply(coeffic, 1, quantile, probs = c(0.025, 0.975))
t(cis)
```

```
##             2.5%     97.5%
## [1,]  1.77585941 3.6696760
## [2,]  1.91073616 2.1868440
## [3,]  0.75943580 1.3601864
## [4,] -0.09207563 0.1957527
```

```
boxplot(cis, main = "Distribution of coefficients")
```

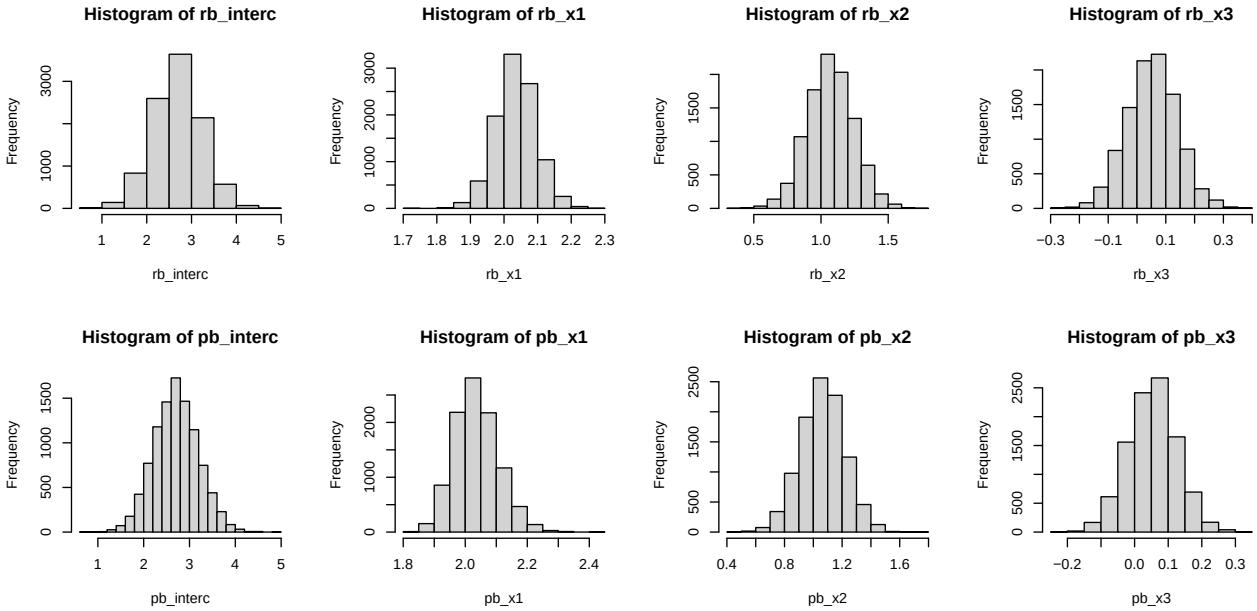


We observe similar confidence intervals for using pairs bootstrap and again the distribution of x3 coefficients stronly suggest that x3 represents noise.

2.4

Compare the two approaches in 2. and 3. and explain the differences in the sampling approach and how this (might) affect(s) the results.

```
par(mfrow = c(2, 4))
# visualize coefficient distributions for residual bootstrap
rb <- coef(fit_boot)
rb_interc <- rb[1, ]
hist(rb_interc)
rb_x1 <- rb[2, ]
hist(rb_x1)
rb_x2 <- rb[3, ]
hist(rb_x2)
rb_x3 <- rb[4, ]
hist(rb_x3)
# visualize coefficient distributions for pairs bootstrap
pb <- coeffic
pb_interc <- pb[1, ]
hist(pb_interc)
pb_x1 <- pb[2, ]
hist(pb_x1)
pb_x2 <- pb[3, ]
hist(pb_x2)
pb_x3 <- pb[4, ]
hist(pb_x3)
```



The above plots show that the coefficients for residual bootstrap and pairs bootstrap are distributed very similar but not the same.

The assumptions made for bootstrapping linear regression are very strong. For example the error distribution my not be normally (errors could be much higher or lower for some observations) which is especially a problem for residual bootstrap. On the other hand bootstrapping pairs is less sensitive to assumptions. The standard error estimate obtained by bootstrapping pairs, gives reasonable answers even if some assumptions are completely wrong. The only assumption is that the original pairs were randomly sampled from some distribution F.

Task 3

Summarise the bootstrapping methodology, its advantages and disadvantages based on your exercises for constructing parametric and non-paramteric confidence bounds for estimators, test statistics or model estimates.

Answer:

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.

Further we can use bootstrapping for hypothesis testing where we sample under the H0 and check if the observed sample is outside of our confidence interval.

Another useful application of bootstrapping is in the context of linear regression where we are able to compute confidence intervals for the coefficients of our linear model which gives us a higher certainty of our liner model.