Principles of Statistical Data Analysis

PC lab on Permutation and Rank Tests

1 Objectives

This lab session is about permutation and rank tests. You are expected to be able to:

1. Construct a permutation null distribution
   (a) Exactly: by enumerating all permutations.
   (b) Approximately: by sampling e.g. 1000 random permutations.
2. Find the appropriate critical value(s), p-value, and correctly formulate the conclusion to the research question.
3. Use an asymptotic approximation of the null distribution.
4. Learn to work with the R `coin` package. For permutation testing, the `coin` package can do exact permutation testing and approximate permutation testing (based on random sample of permutations). The functions in the `coin` package are more reliable than those in R BASE (because the latter does not always allow for permutation testing).
5. Interpret the effect size associated with the Wilcoxon rank sum test.
6. Understand the meaning and the implications of the location-shift assumption.
7. Implement the non-parametric (permutation) Tukey's test for multiple testing.

Note that this document in written with `knitr`. Like R Markdown it is a way to combine LaTeX and R in one document (as compared to R Markdown, `knitr` allows you to include more advanced TeX syntax). This improves the reproducibility of your report. The code in this document is available on Ufora as the file `Lab1Code.R`.

2 Demo

As a warm-up, we consider the shrimps case study from the course notes (Section 3) and we perform several analysis on these data in R. These analyses are merely intended to demonstrate the R-code, e.g. we will use several tests on the same data and go from one- to two-sided tests (something you can never do when doing a 'real' analysis - you know why?) and we will not focus on answering research questions.

Context. It is well known that PCBs easily accumulate in the fat tissue of shrimps. In this experiment two groups of 18 shrimps (actually 18 samples of shrimps, because the PCB concentration is measured by extracting the fat of 100g shrimps) were grown under different conditions. The shrimps were randomised over the two groups. The goal is to assess whether the PCB concentration is affected by the growing conditions. PCB concentrations are measured in pg TEQ/g fat.

First the data are read into R (note that the data file is a .RData file). Before loading the data, make sure you have downloaded it and stored it into your working directory (or change your working directory via 'Session - Set Working Directory'). The dataset "shrimps2" contains the same information as the dataset "shrimps" as used in the course notes, but is differently organised.

```
load("shrimps2.RData")
head(shrimps2)
```

```
##   PCB.conc group
## 1     29.7     1
## 2     24.5     1
## 3     97.7     1
## 4     39.1     1
## 5     22.6     1
## 6     32.4     1
```

Before we start analysing the data, we look at the structure of the data set.

```
str(shrimps2)
```

```
## 'data.frame': 36 obs. of  2 variables:
##  $ PCB.conc: num  29.7 24.5 97.7 39.1 22.6 ...
##  $ group   : num  1 1 1 1 1 1 1 1 1 ...
```

From this output we see that the group variable is a numeric variable, whereas it should be defined as a factor. We make it a factor variable and check the

structure again.

```
shrimps2$group <- as.factor(shrimps2$group)
str(shrimps2)
```

```
## 'data.frame': 36 obs. of  2 variables:
##  $ PCB.conc: num  29.7 24.5 97.7 39.1 22.6 ...
##  $ group   : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
```

In this exercise we want to show how to obtain the exact permutation null distribution. Because this is computationally intensive when $n_1 = n_2 = 18$, we will continue to work with a subset of the original data. We consider the first and last 10 observations so that $n_1 = n_2 = 10$ and rename this dataset to `shrimps`. Evidently, the output below will now be different from the output of the course notes (that uses the entire dataset also named `shrimps`).

```
shrimps <- shrimps2[c(1:10,27:36),]
row.names(shrimps) <- 1:20
str(shrimps)
```

```
## 'data.frame': 20 obs. of  2 variables:
##  $ PCB.conc: num  29.7 24.5 97.7 39.1 22.6 ...
##  $ group   : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 1 ...
```

```
dim(shrimps) # dimensions
```

```
## [1] 20  2
```

```
table(shrimps$group) # sample sizes
```

```
##
##  1  2
## 10 10
```

We are now ready to illustrate how to perform the Wilcoxon rank sum test (a.k.a. the Mann–Whitney test or the Wilcoxon–Mann–Whitney test) on the `shrimps` dataset.

2.1 The exact permutation null distribution

We first use the R base function `wilcox.test` to perform the Wilcoxon rank sum test with an exact null distribution by considering all permutations.

We focus on the hypotheses

$$H_0 : F_1 = F_2 \qquad \text{against} \qquad H_A : \Pr(X_1 < X_2) > 1/2.$$

The one-sided alternative is chosen for the purpose of illustration.

```
wilcox.test(formula = PCB.conc~group, data = shrimps,
  alternative = "less", exact = TRUE)


##
##  Wilcoxon rank sum test
##
## data:  PCB.conc by group
## W = 24, p-value = 0.02621
## alternative hypothesis: true location shift is less than 0
```

Note that we are dealing with a one-sided test where the alternative states $H_A : \Pr(X_1 < X_2) > 1/2$. From the R help file of `wilcox.test(x,y)` (use command `help(wilcox.test)`) we read that "...and the one-sided alternative "greater" is that x is shifted to the right of y". So our option `alternative = "less"` implies that x (here this will be `group1` as it is the first level of `levels(shrimps$group)`) is shifted to the *left* of y (here `group2`), which implies that X_1 is expected to take on smaller values than X_2 or equivalently that $\Pr(X_1 < X_2) > 1/2$ (i.e. it is more likely that X_1 is smaller than X_2 or equivalently that X_2 is greater than X_1).

The help files further says "By default (if `exact` is not specified), an exact p-value is computed if the samples contain less than 50 finite values and there are no ties. Otherwise, a normal approximation is used.". This means that the p-value for this example is calculated based on the exact permutation null distribution.

We will now construct the null distribution ourselves.

The total number of permutations is given by

```
choose(20, 10)


## [1] 184756
```

Note that if we would have used the entire dataset (and not the subset of 20 observations), this number explodes to

```
choose(36, 18)
```

```
## [1] 9075135300
```

We can list the permutations in R and store them in a matrix.

```
A <- combn(20, 10)
```

The first five permutations as an illustration.

```
A[,1:5]
```

```
##       [,1] [,2] [,3] [,4] [,5]
##  [1,]    1    1    1    1    1
##  [2,]    2    2    2    2    2
##  [3,]    3    3    3    3    3
##  [4,]    4    4    4    4    4
##  [5,]    5    5    5    5    5
##  [6,]    6    6    6    6    6
##  [7,]    7    7    7    7    7
##  [8,]    8    8    8    8    8
##  [9,]    9    9    9    9    9
## [10,]   10   11   12   13   14
```

The matrix `A` gives the indices of the observations in group 1, so group 2 will contain the remaining observations. Hence, the permutation in the first column corresponds to the original data, while in the second column the observations 10 and 11 have switched groups. Similar for the other columns.

In R the Wilcoxon rank sum test statistic can be obtained as follows:

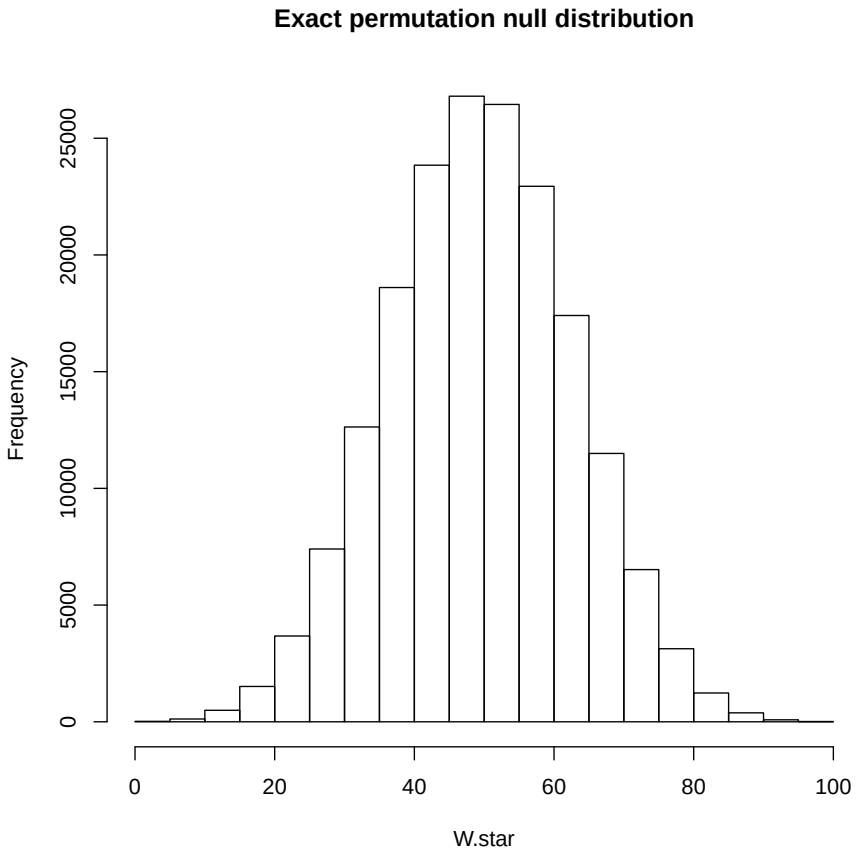
```
test <- wilcox.test(formula = PCB.conc~group, data = shrimps,
        alternative = "less", exact = TRUE)
W.obs <- test$statistic
W.obs
```

```
##  W
## 24
```

We use the letter `W` for the test statistic to be consistent with the R output. Recall, however, that `W` actually is the Mann–Whitney statistic U (see the course notes for more details and further in this document).

The `combn` function can be used with a function as the third argument to compute a statistic on each of the combinations (see `help(combn)`). The code below is compact code that will compute the test statistic `W` for each permuted dataset.

```
W.star <- combn(20, 10, function(x) {
                group1 <- shrimps$PCB.conc[x]
                # obs in group 2 = all obs not in group 1
                group2 <- shrimps$PCB.conc[! (1:20) %in% x]
                test <- wilcox.test(group1, group2,
                    alternative = "less")
                return(test$statistic)
        })
# A histogram of the exact permutation null distribution
hist(W.star, breaks = 30,
  main = "Exact permutation null distribution")
```



The critical value (at the 5% level of significance) of this exact permutation null distribution can computed as:

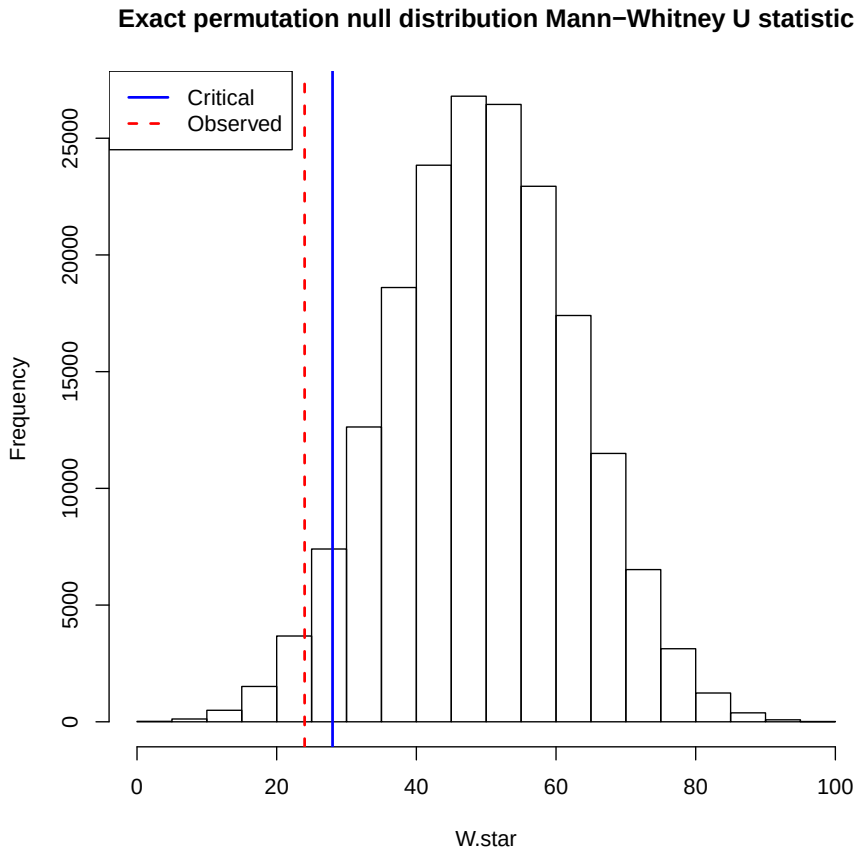
```
criticalValue <- quantile(W.star, probs = 0.05)
criticalValue


## 5%
## 28
```

To double check that we have to look to the critical value on the left hand side (to create the rejection region of this one-sided test), recall that `W` (our test statistic) actually equals $\sum_{i,j} \mathrm{I}\left[X_{1i} > X_{2j}\right]$ (see the Note section in the help file of `wilcox.test`). So for our one sided alternative $H_A : \Pr(X_1 < X_2) > 1/2$ we expect the X_1 values to be smaller than X_2, implying that there will be less pairs for which $X_{1i} > X_{2j}$ and hence under H_A we expect the test statistic

`W` to become smaller. So looking to the left to find the critical value was the right choice. This demonstrates that you always have to carefully examine the behavior of your test statistic of choice (here $\sum_{i,j} \mathrm{I}[X_{1i} > X_{2j}]$) before you can compute the critical value (and the p-value, see further). The fact that `wilcox.test` takes on the convention to look at $X_{1i} > X_{2j}$ while the alternative is typically expressed as $X_{1i} < X_{2j}$ makes life a bit difficult for us. But from the help-file you can always deduce what is done.

We can then add the critical value and the observed statistic to the exact permutation null distribution for visualization. The dotted red line is the observed statistic, the blue line is the critical value.

```
hist(W.star, breaks = 30, main =
  "Exact permutation null distribution Mann-Whitney U statistic")
abline(v = c(criticalValue,W.obs), col = c("blue","red"),
  lty = c(1,2), lwd = 2)
legend("topleft", legend = c("Critical", "Observed"),
  col = c("blue","red"), lty = c(1,2), lwd = 2)
```



We now compute the p-value. It is the area under the permutation null distribution to the left of the observed statistic (as we expect smaller values of the test statistic under the alternative).

```
mean(W.star <= W.obs)
```

```
## [1] 0.02621295
```

That's the same p-value as from the `wilcox.test` output, so we have properly constructed the permutation null distribution ourselves.

Next, we look at the exact null distribution (all enumerations) of the permutation null distribution of the 2-sample t-test statistic. We want to test

$$H_0 : F_1 = F_2 \qquad \text{against} \qquad H_A : \mu_1 \neq \mu_2.$$

We are now considering a two-sided alternative to illustrate how this affects the code (i.e. how to compute the critical values and the p-value). We first look at the output of the t-test in R.

```
t.test(formula = PCB.conc~group, data = shrimps)


##
##  Welch Two Sample t-test
##
## data:  PCB.conc by group
## t = -0.27946, df = 10.738, p-value = 0.7852
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -24.47299  18.97299
## sample estimates:
## mean in group 1 mean in group 2
##           43.72           46.47
```

Next we extract the observed test statistic:

```
test <- t.test(formula = PCB.conc~group, data = shrimps)
t.obs <- test$statistic
t.obs


##          t
## -0.2794627
```

We now construct the exact permutation null distribution of the t-test:

```
t.star <- combn(20, 10, function(x) {
                group1 <- shrimps$PCB.conc[x]
                group2 <- shrimps$PCB.conc[!(1:20) %in% x]
                test <- t.test(group1, group2)
                return(test$statistic)
        })
```

The critical values of this exact permutation null distribution are given by:

```
# Note that we are dealing with a 2-sided test
criticalValues <- quantile(t.star, probs = c(0.025,0.975))
criticalValues


##      2.5%     97.5%
## -2.056563  2.056563
```

The p-value can be computed as follows:

```
mean(abs(t.star) >= abs(t.obs))
```

```
## [1] 0.7725216
```

Other ways to compute the p-value (where we exploit the fact that the permutation null distribution is symmetric about zero):

```
2*(1 - mean(t.star < abs(t.obs)))
```

```
## [1] 0.7725216
```

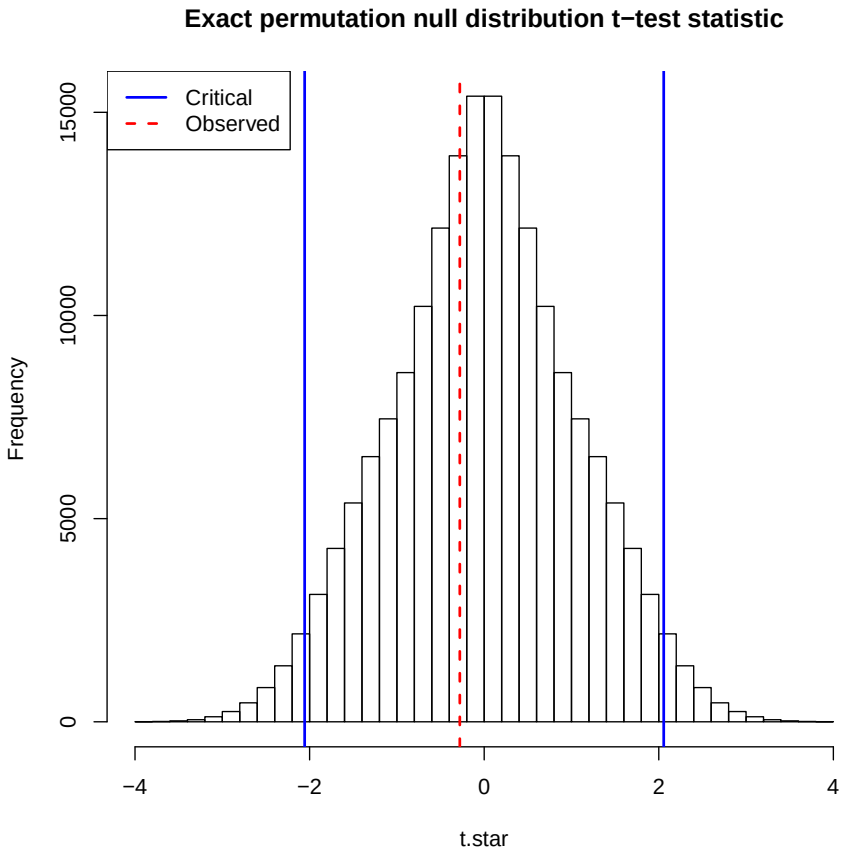
or

```
mean(t.star^2>= t.obs^2)
```

```
## [1] 0.7725216
```

We can add the critical value and the observed statistic to the exact permutation null distribution for visualisation. The dotted red line is the observed statistic and the blue lines are the critical values.

```
hist(t.star, breaks = 30, main =
  "Exact permutation null distribution t-test statistic")
abline(v = c(criticalValues, t.obs),
  col = c("blue","blue","red"), lty = c(1,1,2), lwd = 2)
legend("topleft", legend = c("Critical", "Observed"),
  col = c("blue","red"), lty = c(1,2), lwd = 2)
```



2.2 An approximate permutation null distribution

We have seen the exact permutation null distributions for the t-test statistic and the Wilcoxon rank sum test statistic. Next we look at how to approximate such exact permutation null distributions (even with our powerful computers, an exact null distribution can be computationally too demanding for large samples).

We again consider the following hypotheses:

$$H_0 : F_1 = F_2 \qquad \text{against} \qquad H_A : \mu_1 \neq \mu_2.$$

We choose the 2-sample t-test statistic. The exact permutation null distribution can be approximated by the following R-code.

```
test.obs <- t.test(formula = PCB.conc ~ group,
        data = shrimps)
t.obs <- test.obs$statistic # observed statistic
t.obs


##          t
## -0.2794627


N <- 1000 # 1000  permutations out of choose(20,10) = 184756.
perm.data <- shrimps   # copy data for permutations
t.star.approx <- replicate(N, {
                # shuffle the group labels
                perm.data$group <- sample(perm.data$group)
                test.perm <- t.test(PCB.conc ~ group,
                    data = perm.data)
                return(test.perm$statistic)
        })
```

The `sample()` function samples the `group` labels without replacement each time the `replicate()` function iterates over `N = 1000`. An alternative way of simulating the permutation null distribution is by using a `for` loop (in the course notes, typically for loops are used to make the code less condense).

```
N <- 1000
t.star.approx <- rep(NA,N)
for(i in 1:N)
{
        perm.data$group <- sample(perm.data$group)
        test <- t.test(PCB.conc ~ group,
          data = perm.data)
        t.star.approx[i]<-test$statistic
}
```

We can then move on to calculate the two sided critical values.

```
criticalValues <- quantile(t.star.approx, probs = c(0.025,0.975))
criticalValues


##      2.5%     97.5%
## -2.073933  1.837933

# Because of the random sampling there will be different values
# for each time that you execute this script!
```

Note that the left critical value is now not necessarily the equal opposite of the right critical value. This is a consequence of approximating the null distribution with ramdom sampling.

We finally compute the p-value.

```
mean(abs(t.star.approx) >=  abs(t.obs))


## [1] 0.762


# Again, because of the random sampling the p-value will be
# different for each execution.
```

You can also make histograms of the null distribution and visualize the critical values and observed test statistic as before.

2.3 The asymptotic null distribution

For the two-sided Wilcoxon rank sum test (two-sided for the sake of illustration), an R command of one line can do this:

```
wilcox.test(formula = PCB.conc~group, data = shrimps,
  exact = FALSE, correct  = FALSE)


##
##  Wilcoxon rank sum test
##
## data:  PCB.conc by group
## W = 24, p-value = 0.04937
## alternative hypothesis: true location shift is not equal to 0
```

Setting `exact = FALSE`, forces the `wilcox.test` function to approximate the null distribution with a normal distribution. For comparison purposes (see further), we also set the continuity correction to FALSE.

If you want to calculate this p-value yourself, you will have to standardize the test statistic `W` (see Section 3 of the course notes). Once you have standardized it, the asymptotic null distribution equals the standard normal distribution. Instead of transforming the statistic of `wilcox.test` (the `W`, which is actually the Mann–Whitney statistic U), we will compute $W_1 = \sum_{i=1}^{n_1} R(X_{1i})$ ourselves. This can be easily done with the function `rank`.

```r
# rank transform the outcomes
shrimps$RankPCB <- rank(shrimps$PCB)
# take the sum of the ranks of group 1
W1 <- sum(shrimps$RankPCB[shrimps$group == 1])
```

Next we standardize W_1 (see Section 3.4 in the course notes):

```r
n1 <- table(shrimps$group)[1]
n2 <-  table(shrimps$group)[2]
n <- n1 + n2
Exp.W1 <- n1*(n+1)/2
Var.W1 <- n1*n2*(n+1)/12
W1.stand <- (W1 - Exp.W1)/sqrt(Var.W1)
W1.stand
```

```
##         1
## -1.965415
```

We now compare this with the critical values of a standard normal distribution.

```r
criticalValues <- c(qnorm(0.025), qnorm(0.975))
criticalValues
```

```
## [1] -1.959964  1.959964
```

The two-sided p-value is given by

$$2 \times \mathrm{P}\left[Z \geq |W_1^{\text{stand}}|\right], \quad Z \sim N(0,1).$$

In R this becomes the following (don't forget that R calculates left tails and we want to compute the right tail):

```r
2*(1 - pnorm(abs(W1.stand)))
```

```
##          1
## 0.04936619
```

Or equivalently (as the square of Z has a χ^2_1-distribution)

```r
1-pchisq(W1.stand^2, df = 1)
```

```
##          1
## 0.04936619
```

This is the same p-value as from the `wilcox.test` output.

2.4 The coin package

Finally we illustrate how the Wilcoxon rank sum test can be performed using the coin R-package, using an exact, aproximate exact and an asymptotic null distribution. This package contains many exact tests and its functions are more flexible than those in the R base distribution. First, make sure that the package is installed.

```r
# install.packages("coin")
library("coin")
# An exact permutation null distribution is used
wilcox_test(formula = PCB.conc~group, data = shrimps,
  distribution = "exact")
```

```
##
##  Exact Wilcoxon-Mann-Whitney Test
##
## data:  PCB.conc by group (1, 2)
## Z = -1.9654, p-value = 0.05243
## alternative hypothesis: true mu is not equal to 0
```

```r
# An approximation of the exact null distribution is used
wilcox_test(formula = PCB.conc~group, data = shrimps,
  distribution = approximate(B = 1000))
```

```
##
##  Approximative Wilcoxon-Mann-Whitney Test
##
## data:  PCB.conc by group (1, 2)
## Z = -1.9654, p-value = 0.064
## alternative hypothesis: true mu is not equal to 0
```

```r
# An asymptotic  approximation is used
wilcox_test(formula = PCB.conc~group, data = shrimps,
  distribution = "asymptotic")
```

```
##
##  Asymptotic Wilcoxon-Mann-Whitney Test
##
## data:  PCB.conc by group (1, 2)
## Z = -1.9654, p-value = 0.04937
## alternative hypothesis: true mu is not equal to 0
```

Note that `wilcox_test` directly gives the standardized test statistic. This makes it easy to compare it to a standard normal, but difficult to estimate the probabilistic index $\mathrm{P}[X_1 < X_2]$. The output of the `wilcox.test` function (in base R) can be used for the latter.

3 Exercises

3.1 Exercise 1

Dataset: vitB1.txt

The aim of the study is to assess the effect of vitamin B1 on the growth of mushrooms. The researchers started the experiment with 22 young mushrooms that were randomly assigned to two treatment groups. There were 11 mushrooms assigned to Group 1, which is a control group (no vitamin B1), and the 11 mushrooms in group 2 were grown with a supplement of vitamin B1. The outcome measured is the growth in mg. Although the researchers hope that the supplement of vitamin B1 will have a beneficial effect on the growth, they are not certain, and also wish to detect a reduction in growth if this were the case. For the questions below, use the 5% level of significance. You are asked to perform several tests on the same data - this is intended to demonstrate the differences/similarities between the tests.

Questions.

1. Answer the research question by using a permutation test. You may use the t-test statistic here, but state very clearly the null and alternative hypotheses.
2. As a part of the process towards the solution of part (1) of this exercise, you have enumerated (or approximated) the permutation null distribution of the test statistic. Compare this exact null distribution with the null distribution that you would have used if the observations in both groups were normally distributed.
3. Construct a permutation test to test the null hypothesis that the variances of the growth are equal in the two treatment groups.

4. Answer the original research question by using the Wilcoxon rank sum test.

3.2 Exercise 2

In the first part of the exercise you have to illustrate by means of a simulation study that the 5% level permutation based Wilcoxon rank-sum test has indeed a 5% chance of a type I error in a population model.

1. Find the 5% level critical value of the permutation Wilcoxon rank-sum test. Note that a permutation test guarantees that the type I error rate is exactly controlled conditional on the data, at least when the null distribution is not too discrete (randomisation model).

2. With the critical value found in the previous step, you set up a simulation study in which you compare the Wilcoxon rank-sum test and the 2-sample t-test under the null hypothesis. In a simulation study the sample observations are repeatedly randomly sampled from two populations (population model). The estimated type I error rates have thus a repeated sampling interpretation, not conditional on the data.
   Do not only sample from normal distributions, but try some other distributions as well (e.g. t-distribution `rt(n,df)`, exponential distribution `rexp(n,rate)`, uniform distribution `runif(n,min,max)`, lognormal distribution `rlnorm(n,meanlog,sdlog)`, Weibull distribution `rweibull(n,shape)`, ...). Use a rather small sample size, e.g. $n = 10$ in each sample.

3.3 Exercise 3

Change the R-code you used for the previous exercise, but now the objective is to estimate powers of the Wilcoxon rank-sum test and the 2-sample t-test under various alternatives. All tests have to be performed at the 5% level of significance. If you use the same sample sizes as in Exercise 3, you don't have to re-compute the critical values. Choose the alternatives so that you can demonstrate the following properties:

- In location-shift models, the Wilcoxon rank-sum test does indeed test for location-shifts.
- For many non-normal distributions, the Wilcoxon rank-sum test has larger power than the 2-sample t-test, but in normal distributions the 2-sample t test performs better.

- The Wilcoxon rank-sum test is indeed a test that detects $\Pr(X_1 \leq X_2) \neq 1/2$. Confirm this in a simulation study in which you choose alternatives so that $\Pr(X_1 \leq X_2) \neq 1/2$, but $\mu_1 = \mu_2$.

3.4 Exercise 4

Dataset: carageenan.txt.

An important characteristic of puddings is the elasticity. Elasticity is typically measured by the elasticity modulus G (unit: Pa). In this experiment puddings were produced according to three recipes: high, medium and low carageenan concentration. Each of the groups consists of 3 puddings.
The researchers want to know if the carrageenan concentration has an effect on the elasticity modulus.

The data can be read into R as follows (assuming the data are stored in your working directory):

```
pudding <- read.table("carrageenan.txt", header = T, sep = " ")
```

Questions:

1. Answer the research question in terms of the mean carrageenan concentration by using permutation tests (5% level of significance). Use the Tukey method for the post-hoc tests (i.e. the multiple testing).
2. Answer the research question by using a Kruskal-Wallis rank sum test (5% level of significance). Use the Tukey method for the post-hoc tests.