Class 3 - Statistics part I

1. Getting basic statistics

This tutorial is based on the iris dataset, that is available by default in R.

```
?iris
```

This famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are Iris setosa, versicolor, and virginica.

```
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Iris dataframe has 4 numeric columns, one for each phenotypic parameter and one factor "Species" with 3 levels

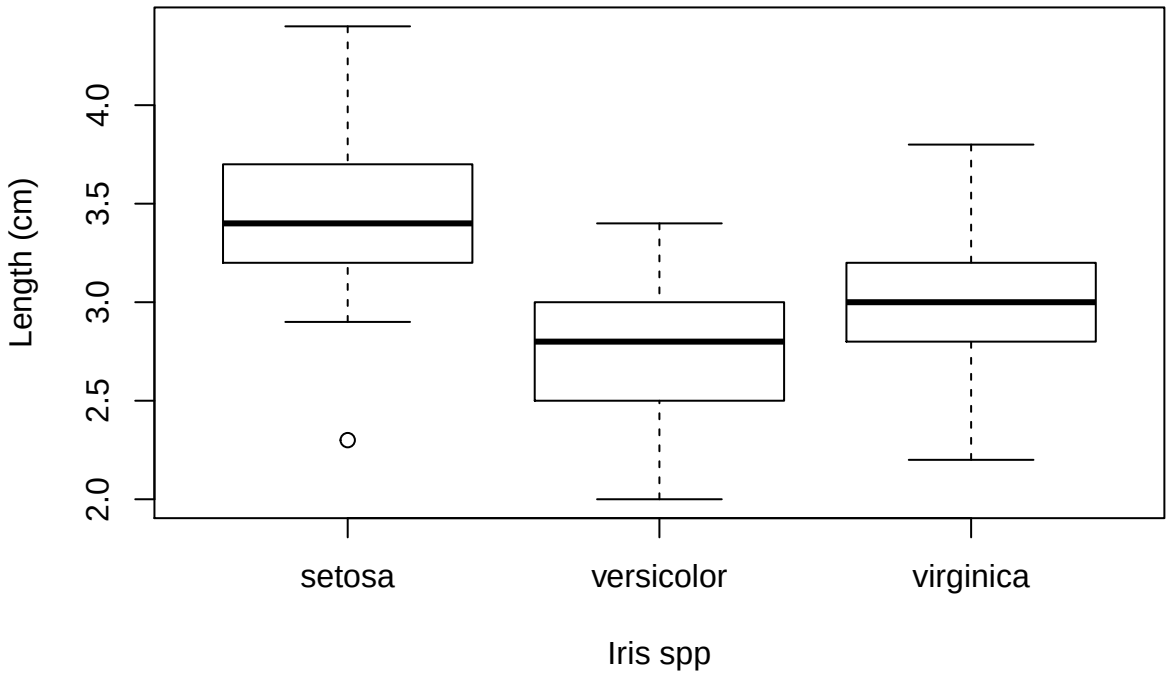
```
# check the global content of a dataframe, look for missing data
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

Everyhting looks ok

```
# to avoid messing up iris, let's create a copy
iris_df <- iris
# check distribution of data
# because "Species" is a factor we can use it top group obervations for plots

# exploring boxplot() arguments to edit the plot)
boxplot(formula = Sepal.Width ~ Species, data = iris_df,
        main = "Sepal Width",
        xlab = "Iris spp",
        ylab = "Length (cm)")
```



Now let's calculate some basic statistics

```
# Considering the structure of iris data frame, it will not be possible to apply colMean()
# or rowMean(), beause different lines belong to different spp.


mean(iris_df[iris_df$Species == "setosa", "Sepal.Width"])
```

```
## [1] 3.428
```

```
mean(iris_df[iris_df$Species == "versicolor", "Sepal.Width"])
```

```
## [1] 2.77
```

```
mean(iris_df[iris_df$Species == "virginica", "Sepal.Width"])
```

```
## [1] 2.974
```

```
sd(iris_df[iris_df$Species == "setosa", "Sepal.Width"])
```

```
## [1] 0.3790644
```

```
sd(iris_df[iris_df$Species == "versicolor", "Sepal.Width"])
```

```
## [1] 0.3137983
```

```
sd(iris_df[iris_df$Species == "virginica", "Sepal.Width"])
```

```
## [1] 0.3224966
```

```
# run these commands one by one can be time comsuming, so there are other options...
# by() funtion applies a given function, on iris$Sepal.Width according to group in iris$Species
by(iris$Sepal.Width, iris$Species, mean)
```

```
## iris$Species: setosa
```

```
## [1] 3.428
## ------------------------------------------------------------
## iris$Species: versicolor
## [1] 2.77
## ------------------------------------------------------------
## iris$Species: virginica
## [1] 2.974
```

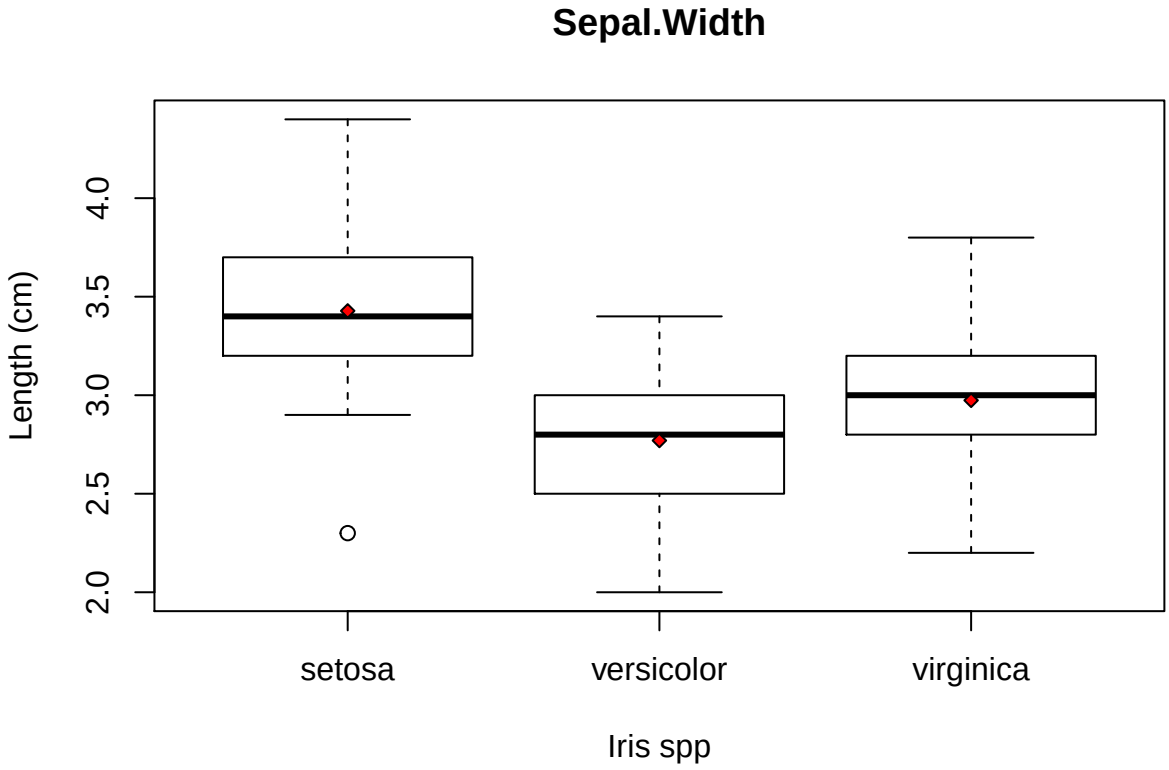
```
by(iris$Sepal.Width, iris$Species, sd)
```

```
## iris$Species: setosa
## [1] 0.3790644
## ------------------------------------------------------------
## iris$Species: versicolor
## [1] 0.3137983
## ------------------------------------------------------------
## iris$Species: virginica
## [1] 0.3224966
```

```
by(iris$Sepal.Width, iris$Species, median)
```

```
## iris$Species: setosa
## [1] 3.4
## ------------------------------------------------------------
## iris$Species: versicolor
## [1] 2.8
## ------------------------------------------------------------
## iris$Species: virginica
## [1] 3
```

```
# we can save the result of by() function in a vector and plot it
mean_sepalw <- as.vector(by(iris$Sepal.Width, iris$Species, mean))
boxplot(formula = Sepal.Width ~ Species, data = iris,
        main = "Sepal.Width",
        xlab = "Iris spp",
        ylab = "Length (cm)")
points(1:3, mean_sepalw, pch = 23, cex = 0.75,
       bg = "red")
```




```
# Since the porpose of this tutorial is the comparison between two populations,
# and to decrease complexity of the commands to use, let's forget that I. setosa
# exists and remove all rows related to this

iris_df <- iris_df[iris_df$Species %in% c("versicolor", "virginica") , ]

# and adjust the factors
iris_df$Species <- factor(iris_df$Species)
levels(iris_df$Species)
```

```
## [1] "versicolor" "virginica"
```

1.1. Goodness of fit tests for normality

If sampled data follows a normal distribution, we can take and compare mean values of two (or more populations)... if not we may need to test the median

1.1.1 Shapiro-Wilk Normality Test

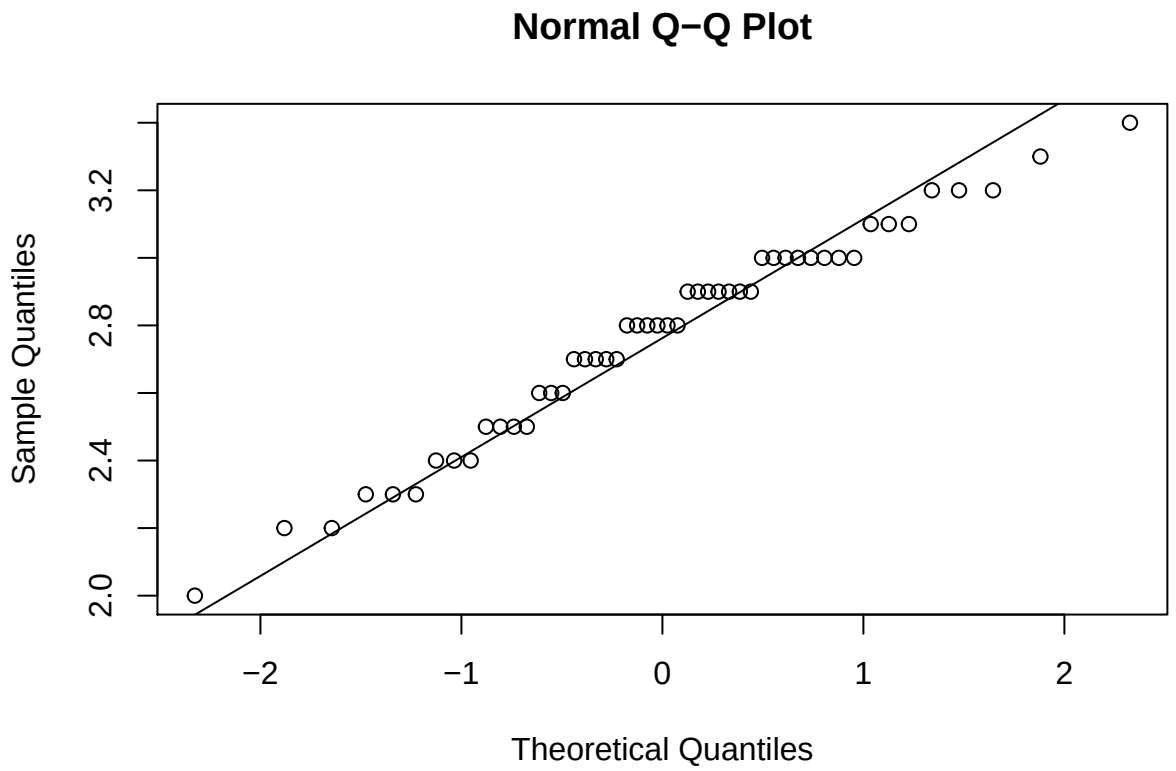
- visual tools link

The Q-Q plot, or quantile-quantile plot, is a graphical tool to help us assess if a set of data plausibly came from some theoretical distribution such as a Normal distribution

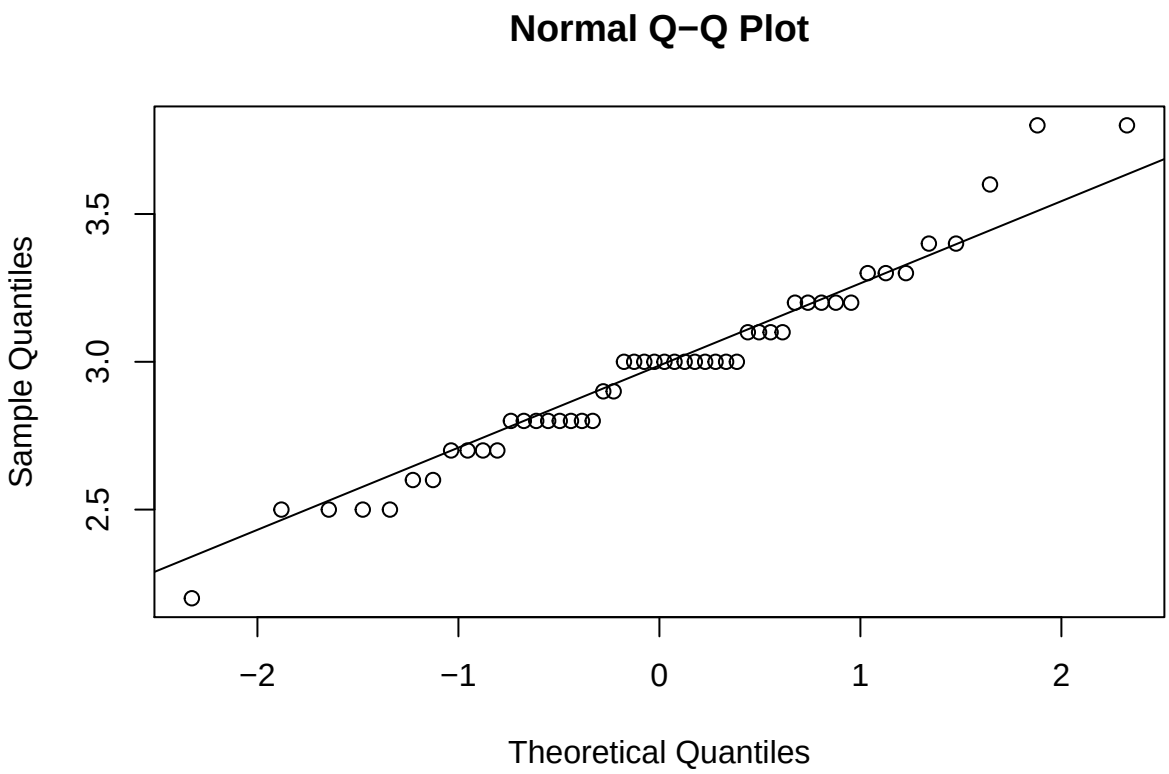
It's just a visual check, not an air-tight proof, so it is somewhat subjective. But it allows us to see at-a-glance if our assumption is plausible, and if not, how the assumption is violated and what data points contribute to the violation.

A Q-Q plot is a scatterplot created by plotting two sets of quantiles against one another. If both sets of quantiles came from the same distribution, we should see the points forming a line that's roughly straight.

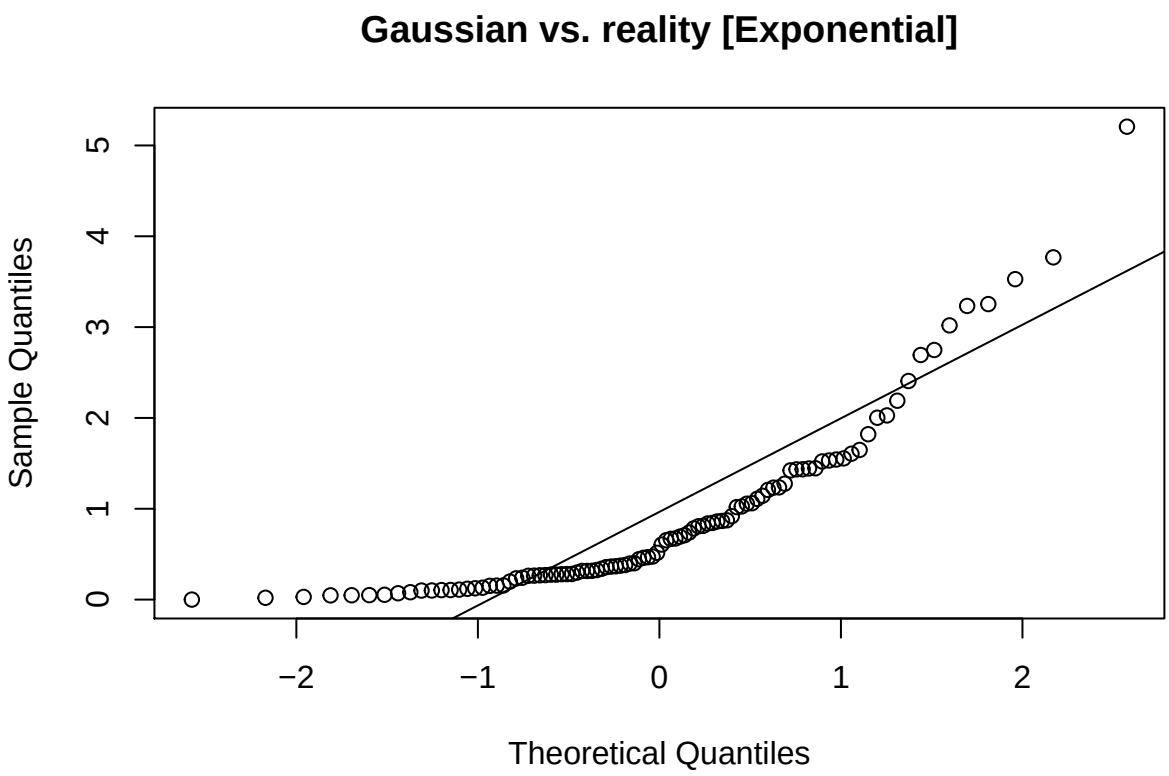
```r
qqnorm(iris_df[iris_df$Species == "versicolor", "Sepal.Width"])
qqline(iris_df[iris_df$Species == "versicolor", "Sepal.Width"])
```



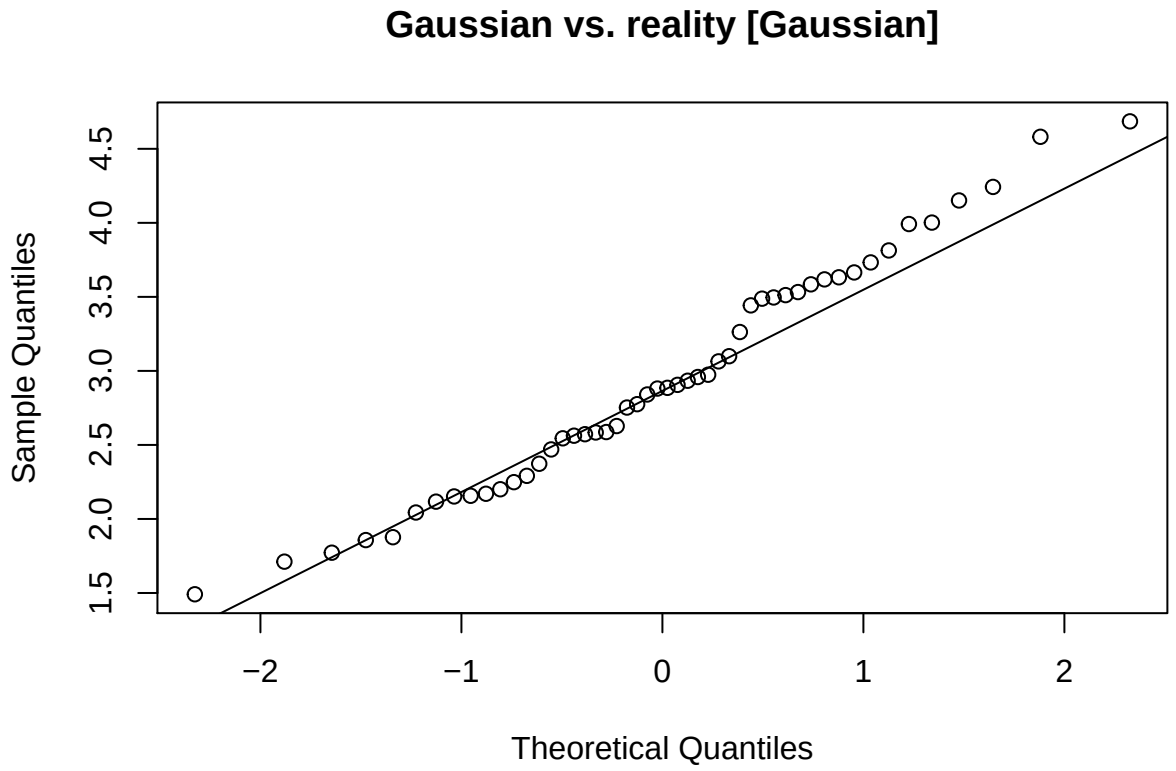

```r
qqnorm(iris_df[iris_df$Species == "virginica", "Sepal.Width"])
qqline(iris_df[iris_df$Species == "virginica", "Sepal.Width"])
```




```r
# for a "bad example" rexp(100) generates random numbers following
# an exponential distribution
qqnorm(rexp(100),main = "Gaussian vs. reality [Exponential]")
qqline(rexp(100))
```



```r
# for a "good example" rnorm(n = 50, mean = 3, sd = 0.8) generates random numbers following
# a normal distribution
qqnorm(rnorm(50, 3, 0.8),main = "Gaussian vs. reality [Gaussian]")
qqline(rnorm(50, 3, 0.8))
```



Variable Sepal Width closely follows a Normal distribution

Yet, this visual approach is usually subjective and does not guarantee that the distribution is normal. Nevertheless, readers of an article can judge the distribution assumption by themselves

1.1.2 Shapiro-Wilk Normality Test

With null hypothesis: - H0: data was drawn from a population with normal distribution and alternative hypothesis - Ha: that it was not.

```r
# applying shapiro.test()

shapiro.test(iris_df[iris_df$Species == "virginica", "Sepal.Width"])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  iris_df[iris_df$Species == "virginica", "Sepal.Width"]
## W = 0.96739, p-value = 0.1809
```

```r
shapiro.test(iris_df[iris_df$Species == "versicolor", "Sepal.Width"])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  iris_df[iris_df$Species == "versicolor", "Sepal.Width"]
## W = 0.97413, p-value = 0.338
```

For sample size 50 and significance level 0.05, W-alpha = 0.947 (reference values), Given that our W-test values are larger than W-alpha we can't reject H0.

Furthermore, the **p-value** of the tests is also given Here p-value means the following, in the case of Sepal Width of *I. virginica* with sample size 50 there is a 18.09% probability of observing a W-test = 0.96739 or more extreme while H0 being TRUE, therefore, we can't reject H0

Sepal.Width sampling data for all three species is normally distributed, therefore we can use parametric tests.

1.1.3 Homogeneity of variances With null hypothesis: - H0: variance from both population samples is homogeous
and alternative hypothesis - Ha: variance is different.

```
bartlett.test(Sepal.Width ~ Species, data = iris_df)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  Sepal.Width by Species
## Bartlett's K-squared = 0.036258, df = 1, p-value = 0.849
```

the obtained p-value is higher than the sifnificance level so we can't reject H0. Variances from both sample goups are homogeneous

1.2. Comparing two populations

1.2.1 Unpaired Two sample t-test

For samples following a normal distribution (parametric test).

(tutorial link)

Formulate a question or hypothesis to test:

```
We can use Sepal Witdh to discriminate *Iris versicolor* and *Iris virginica* (hyp.)
or
Is Sepal Width from *Iris versicolor* different from *Iris virginica*?
```

The null hypothesis under test is: - H0 : mean1 = mean2 (or the two observations (samples) are randomly drawn from the same population) - H1: mean1 != mean2 (two-tailed) mean1 < mean2 (one-tailed) mean1 > mean2 (one-tailed)

This test may be applied in R using function:

```
t.test(formula = values ~ group, data = df, ...)
or
t.test(group1, group2, ...)
```

```
# two-tailed
t.test(formula = Sepal.Width~Species, data = iris_df, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  Sepal.Width by Species
## t = -3.2058, df = 98, p-value = 0.001819
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.33028246 -0.07771754
```

```
## sample estimates:
## mean in group versicolor  mean in group virginica
##                    2.770                    2.974
```

```
# when we need to filter a column for more than one argument we may use the %in% operator

# "iris_df$Species %in% c("virginica", "versicolor")" expression means: give all line index of cells th

# different, and simpler way of calling the function
t.test(iris_df[iris_df$Species == "versicolor", 2], iris_df[iris_df$Species == "virginica", 2], var.equa
```

```
##
##  Two Sample t-test
##
## data:  iris_df[iris_df$Species == "versicolor", 2] and iris_df[iris_df$Species == "virginica", 2]
## t = -3.2058, df = 98, p-value = 0.001819
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.33028246 -0.07771754
## sample estimates:
## mean of x mean of y
##     2.770     2.974
```

```
# one-tailed
t.test(Sepal.Width~Species, data = iris_df, alternative = "less", var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  Sepal.Width by Species
## t = -3.2058, df = 98, p-value = 0.0009096
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##         -Inf -0.09833009
## sample estimates:
## mean in group versicolor  mean in group virginica
##                    2.770                    2.974
```

The p-value of the test is 0.001819, which is the probability of observing a value of t = -2.0819 or more extreme (farther than t = 0))

Also, assuming a significance level alpha = 0.05 (which should be established a priori) , we may conclude that mean Sepal.With for virginica is significantly different from versicolor with a p-value = 0.001819 (or for a significance level of $p < 0.05$).

```
# for a one-tailed test it is import the check the order of the levels,
# when using formulas (e.g. formula = Sepal.Width~Species)
# As it is right now the reference is versicolor (alphabetical order)
# and in fact versicolor mean is lower than virginica
# But what if you set the factor level reference for viiginica

iris_df$Species <- relevel(iris_df$Species, ref="virginica")

# in this case if you use alternative = less, it will test if
# virginica is lower than versicolor

t.test(formula = Sepal.Width~Species, data = iris_df, alternative = "less", var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  Sepal.Width by Species
## t = 3.2058, df = 98, p-value = 0.9991
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##       -Inf 0.3096699
## sample estimates:
##  mean in group virginica mean in group versicolor
##                    2.974                    2.770
```

```r
# which we already know that it's not.
# That's why you need to be carefull with factor order and
# with one-tailed tests

# to avoid setting levels of factors you may also choose to run the command without the formula

t.test(iris_df[iris_df$Species == "versicolor", 2], iris_df[iris_df$Species == "virginica", 2], alterna
```

```
##
##  Two Sample t-test
##
## data:  iris_df[iris_df$Species == "versicolor", 2] and iris_df[iris_df$Species == "virginica", 2]
## t = -3.2058, df = 98, p-value = 0.0009096
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##         -Inf -0.09833009
## sample estimates:
## mean of x mean of y
##     2.770     2.974
```

Note 1: Welch Two Sample t-test is used by default by t.test() function. This an adaptation of Student's t-test, and is more robust for cases when two samples have unequal variances and/or unequal sample sizes

Note 2: The t-test is invalid for small samples from non-normal distributions, but it is valid for large samples from non-normal distributions (n>50).

Note 3: When using a two-tailed test you are testing for the possibility of the relationship in both directions, regardless of the direction of the relationship you hypothesize. Therefore is most cases, or when in doubt use two-tailed. Choosing a one-tailed test for the sole purpose of attaining significance or after running a two-tailed test that failed to reject the null hypothesis is not appropriate. Chose One-tailed after considering the consequences of missing an effect in the other direction.

More information on tailed tests in link.

1.2.1 Paired Two sample t-test

In case you have paired data you can use argument *paired* in t.test()

This test may be applied in R using function:

```
t.test(formula = values ~ group, data = df, paired = TRUE, ...)
or
t.test(group1, group2, paired = TRUE...)
```

Paired data applies to data made under the same conditions, test two treaments on the same individual (or twins) or in a before and after situation.

In this case the order of the observations matter, i.e. observation 1 from before/treatmentA was made in the same/similar individual as obs.1 from after/treatmentB

In an hypothetical situation, image that ChemA and ChemB are two chemicals applied on the righ and left sepal of the same flower. Width in both sepals are measured after 10 days... Results are saved in PairedDataExample.csv

```
iris_pair <- read.csv("data/PairedDataExample.csv", row.names = 1)

# package PairedData creates paired objects
# We'll just use it to create a plot

# install.packages('PairedData')
library(PairedData)
```

```
## Loading required package: MASS
```

```
## Loading required package: gld
```

```
## Warning: package 'gld' was built under R version 3.5.2
```

```
## Loading required package: mvtnorm
```

```
## Warning: package 'mvtnorm' was built under R version 3.5.2
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

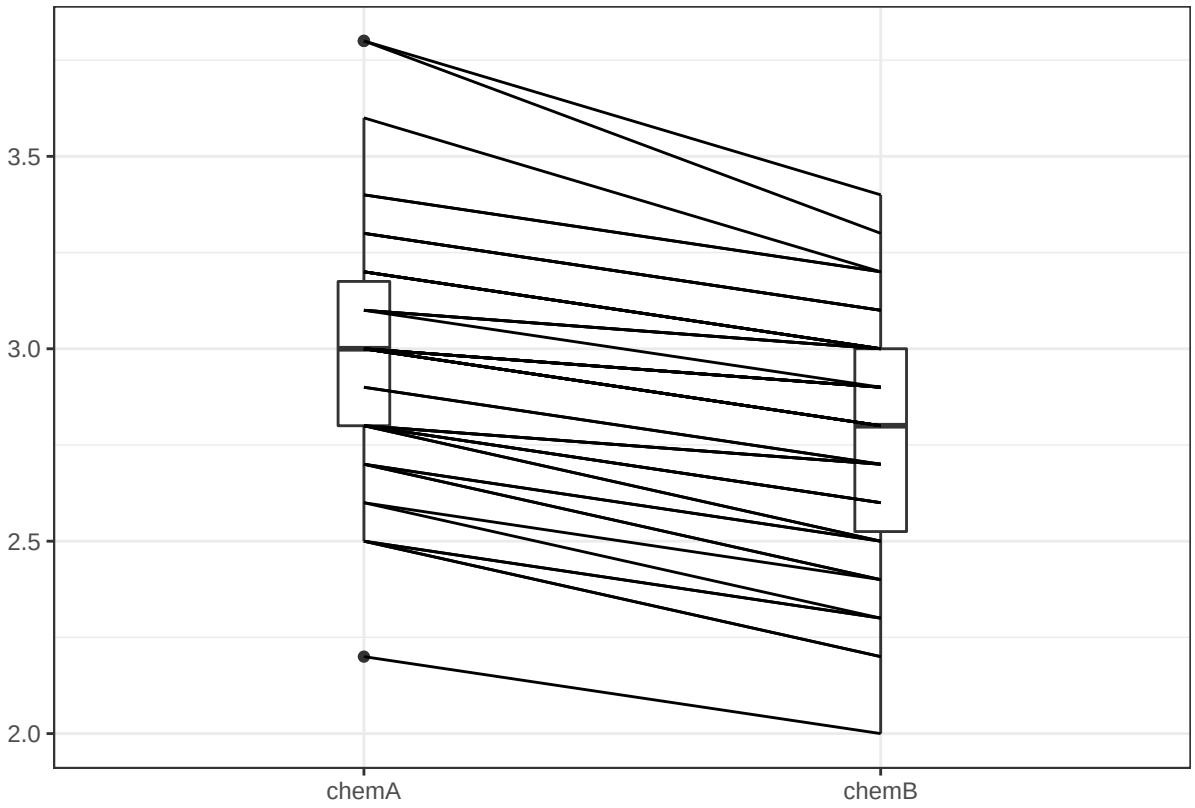
```
##
## Attaching package: 'PairedData'
```

```
## The following object is masked from 'package:base':
##
##     summary
```

```
# sorting will just be done to make a better correspondance between the fake pairs
chemA <- iris_pair$ChemA
chemB <- iris_pair$ChemB

# check how paired observations changes under both conditions
# paired() function will be used to create an object of class "paired"
# we'll use this object just for ploting data, it is nor necessary
# for hypothesis tests

pd = paired(chemA, chemB)
plot(pd, type = "profile") + theme_bw()
```




```
# t.test

t.test(chemA, chemB, paired = TRUE, var.equal = TRUE)
```

```
##
##  Paired t-test
##
## data:  chemA and chemB
## t = 16.848, df = 49, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1796674 0.2283326
## sample estimates:
## mean of the differences
##                   0.204
```

```
#or
t.test(iris_pair$ChemA, iris_pair$ChemB, paired = TRUE, var.equal = TRUE)
```

```
##
##  Paired t-test
##
## data:  iris_pair$ChemA and iris_pair$ChemB
## t = 16.848, df = 49, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1796674 0.2283326
## sample estimates:
## mean of the differences
```

```
##                     0.204
```

p-value < 2.2e-16 which is lower than significance level of 0.05, meaning that we can reject H0 and consider that both treatments have a significant effect

1.2.1 Non-parametric Two sample test

When normallity of the data can not be assumed, the mean is not a good estimator for the central value of a population. In this case, the median can be tested.

The Wilcoxon Mann-Whitney test may be used in this case

The null hypothesis under test is: - H0 : median1 = median2 (or the two observations (samples) are randomly drawn from the same population) - H1: median1 != median2 (two-tailed) median1 < median2 (one-tailed) median1 > median2 (one-tailed)

```
wilcox.test(formula = Sepal.Width~Species, data = iris_df)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Sepal.Width by Species
## W = 1659, p-value = 0.004572
## alternative hypothesis: true location shift is not equal to 0
```

```
# alternative option
wilcox.test(iris_df[iris_df$Species == "virginica", "Sepal.Width"], iris_df[iris_df$Species == "versicol
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  iris_df[iris_df$Species == "virginica", "Sepal.Width"] and iris_df[iris_df$Species == "versic
## W = 1659, p-value = 0.004572
## alternative hypothesis: true location shift is not equal to 0
```

The calculated p-value 0.004572, therefore, we reject H0 considering a significance level of 0.05

Note 1: Non-parametric tests such as Wilcoxon Mann-Whitney don't assume normality, but they require a simetrical population. If the distribution of a population is biased (a lot of similar observations, for example) the accuracy of these tests is also affected. Drawing a boxplot to check sample distribution is always advisable

---

Activity 3

a) Hypothesis tests for two populations:

- a.1) Choose the best way to import file "feeders_select_exercise.csv" and save it to an object named feeders.

This file contains measurements (OD_avg and pH) taken from a sucrose solution present in 10 different bird feeders. Measurements were made at day 1 and at day 7 after changing the solution. The purpose will be to see differences between both days

- a.2) Check the str() to confirm that, al least, both numeric variables (OD_avg and pH) are in fact numeric and not factors. DurGrowth should be a factor.
- a.3) Calculate the mean, sd and median of variables pH and OD_avg for day 1 and day 7.
- a.4) How many measurement (n) were taken in day 1 and in day 7?

- a.5) Is pH data for day 7 and day 1 normally distributed? What about OD_avg? Which type of tests would you use for each variable.
- a.6) Draw a boxplot for pH as function of DurGrowth. Is there something you could to the data to solve the normality issue?
- a.7) measurements in day 1 and day 7 were made always on the same feeder. What kind of strategy should be used for to compare these two days?
- a.8) Perform the Bartlet test and the t.test/wilcoxon (depending on the normality tests results) to investigate if pH and OD_avg are different between the two days. *look at column FEED.N (unique ID of the feeder) to understand the order of the measurements in the table*