Summary Statistics

- summary of a data frame

```
summary(mtcars)
```

- or the logarithmic summary

```
summary(log(mtcars))
```

- you can also extract specific stats for a column in a df

```
mean(mtcars$mpg)
median(mtcars$mpg)
max(mtcars$mpg)
min(mtcars$mpg)
```

- we can check for correlations in the data frame

```
cor(mtcars)

cor(log(mtcars))
```

- scatterplot matrix for every combination possible

```
pairs(iris)
```

```
library(lattice)
splom(iris)
```

*(More on lattice plotting in the course "**Graphs in R**")*

- we can extract a subset of a dataset and store it as object

```
head(iris)
mysubset = subset(iris, Species =="setosa")
mysubset
```

- exclude a certain column

```
mysubset[-2]
```

- we can even get an ordered (increasing) vector with the index positions

```
order(iris$Sepal.Length)
```

Repetition of data types

```
numeric =c(4.5,7.9,3.6)

integer =c(3,5,9)

character =c("versicolor", "setosa", "virginica")

ordinal =c("good", "intermediate", "bad")


frame =data.frame(numeric,integer,ordinal,character)

frame
```

Statistical Packages in R

- **stats** contains all the basic statistical functions of the Base package
- **nortest** for tests of normality
- **multcomp** for inference in parametric models
- **mutoss** for multiple testing procedures
- **agricolae** with Duncan test and Scheffe adjustment
- **coin** for inference in permutations
- **DTK** for Dunnett Tukey Kramer

Inferential Statistics with R

- Z Test - Proportions Test for big parametric populations (n > 30)

```
? prop.test
```

```
set.seed(2489)
ourdata = rnorm(1000, mean=600, sd=80)
hist(ourdata)
```

- we want to test if 50% of "ourdata" is greater than 700
- H0: equal proportion 50/50 of data greater and smaller 700

- at first we need to create an object that sorts for counts greater 700

```
 sortedobject<- ifelse(ourdata >700, "yes", "no")
table(sortedobject)
```

```
totalnumber = 1000
yes = 102
```

```
prop.test(x=yes, n=totalnumber, alternative="two.sided")
```

- this tells us that there is no equal distribution (yes vs no) of >700
- there is also an estimated probability of yes of approx. 0.1

1 sided test is also possible

- here we are stating H0 that the prob >700 is greater than 0.5

```
prop.test(x=yes, n=totalnumber, alternative="less")
```

- here we are stating H0 that the prob >700 is smaller than 0.5

```
prop.test(x=yes, n=totalnumber, alternative="greater")
```

- lets create an object of our Test

```
mytest = prop.test(x=yes, n=totalnumber, alternative="two.sided")
```

- data we can extract from our object

```
names(mytest)
mytest$estimate
```

- this tells us that the probability >700 is approx 10%
- if we would have a small sample we could also use the prop.test
- but correct=T should be used
- alternatively for small n t.test is an option

Tests for normal distribution

- let's get the dataset
- we are again using the ourdata normal vector

```
set.seed(2489)

ourdata = rnorm(1000, mean=600, sd=80)

hist(ourdata)

qqnorm(ourdata)
```

- and we can also use a uniform data vector

```
set.seed(2489)

ouruniformdata = runif(1000)

hist(ouruniformdata)

qqnorm(ouruniformdata)
```

- H0 of the two tests = normality of the dataset

1. Shapiro Wilk Test

? shapiro.test

shapiro.test(ourdata)

shapiro.test(ouruniformdata) # dataset max 5K observations

- you can also extract a column from a dataset

shapiro.test(iris$Sepal.Length)

2. Anderson Darling Test

library(nortest) # get the nortest library

?ad.test

```
ad.test(ourdata)


ad.test(ouruniformdata)
```

Exercise: Test for normality

- Dataset = diamonds, library = ggplot2
- Get familiar with the diamonds dataset. What does the column depth tell us?
- Perform at least 2 graphical tests for normality
- Get familiar with the package nortest and perform at least 3 different tests for normality.

Solution

```
library(ggplot2)

head(diamonds)

attach(diamonds)

qqnorm(depth)  # looks too curvy for normal distr

hist(depth)  # hist looks also not normal

depthsmall = sample(depth, 5000) # sampling to get a vector fitted for shapiro


Base package

shapiro.test(depthsmall)  # Shapiro Wilk test from base pack


Nortest = Tests for normality

library(nortest)  # Pack contains several useful normality tests

cvm.test(depth)  # Cramer von Mises Test, since AD gives NAs for that high significance

lillie.test(depth)  # Kolmogorov Smirnov

sf.test(depthsmall)  # Shapiro Francia

pearson.test(depth)  # Pearson normality test
```

1 sample T test (for population means)

- normally distributed data, 1 variable

? t.test

hist(ourdata)

- we need to specify, x, mu, alternative and if needed the confidence level
- here we are stating H0: mean is 300 or higher

t.test(x=ourdata, mu=300, alternative="less", conf.level=0.95)

- H0: mean is 300 or smaller

t.test(x=ourdata, mu=300, alternative="greater", conf.level=0.95)

- 2 sided test - default

- here we are stating H0: mean is equal to 300

t.test(x=ourdata, mu=300, alternative="two.sided", conf.level=0.95)

2 sample independent t test (Welch Test)

- parametric, 2 sample or population means can be compared

- numeric outcome variable Y vs categorical explanatory variable X (2 levels - e.g. yes vs no)

```
head(mtcars)
```

- am is our 2 level categorical variable (although it is factorized)

```
attach(mtcars)
```

- visual orientation

```
boxplot(data=mtcars, wt~am)
```

- H0: mean wt am1 = mean wt am0
- two sided
- independent : paired=F

```
t.test(mtcars$wt~mtcars$am, alt="two.sided", conf=0.95,
       mu=0, paired=F, var.equal=F)
```

- most of this arguments are not mandatory

t.test(mtcars$wt~mtcars$am)

- an alternative way to write it without the tilde

t.test(mtcars$wt[mtcars$am==0], mtcars$wt[mtcars$am==1])

- how to find out if to assume equal variance:
- you can check the boxplot, do the levene Test or check that variance

var(mtcars$wt[mtcars$am==0]); var(mtcars$wt[mtcars$am==1])

- in this case the var is not equal

- Paired T Test for means of 2 dependent or paired populations (same length)
- the same command can be used but: paired=T

Exercises: 2 sample T Test

- dataset = ships, library = MASS
- get familiar with the dataset ships
- we are interested in the relationship of period and incidents
- use an appropriate graphical tool to show that relationship
- use a T test to compare the incidents in the 2 periods (code it in 2 different ways)
- are there significant differences in period?

Solution

```
library(MASS)
? ships
attach(ships)
head(ships)

boxplot(incidents ~ period)

t.test(incidents ~ period)

t.test(incidents[period==60], incidents[period==75])
```

Mann Whitney U test - Wilcoxon Rank Sum test

- non-parametric (non normal distribution)
- to examine the difference in median of 2 independent populations
- like the 2 sample independent T test
- numeric outcome variable Y vs categorical explanatory variable X (2 levels - e.g. yes vs no)

?wilcox.test

- for out example lets again use the same dataset: mtcars

```
wilcox.test(mtcars$wt~mtcars$am, mu=0, alt="two.sided",
            conf.int=T, conf.level=0.95,
            paired=F, correct=T, exact=F)
```

- correct for continuity correction of p value, exact=exact p value

Wilcoxon Signed rank test does the exact same thing as a paired T Test but

- it is non parametric,
- Wilcox.test can be used like before, just change paired=T

ANOVA

- for normally distributed and independent data sets
- we can use it to compare the means of different groups

```
head(iris)
attach(iris)
```

- let's check if our grouping variable is a factor

```
is.factor(Species)
```

- let's check the levels of it

```
levels(Species)
```

- for a first visual impression

```
boxplot(Sepal.Length~Species)
```

- let's get the means for all groups

```
by(Sepal.Length, Species, mean)
```

F test - one way test (for normally distributed and independent data, nID)

- this is a simpler test with shorter output

```
oneway.test(Sepal.Length~Species)
```

- the command for ANOVA is aov

```
myANOVA <- aov(Sepal.Length~Species, data=iris)
myANOVA

summary(myANOVA)

attributes(myANOVA)
```

- post hoc tests to adjust p value for T1 error

- Tukey test for pairwise comparison and same group size

```
TukeyHSD(myANOVA)
```

- to plot the 95% CI level

```
plot(TukeyHSD(myANOVA)) # as we can see all groups differ (> 0)
```

- coefficients

```
coefficients(myANOVA)
```

- that means Virginica is 1.58 bigger than the reference (setosa)

Levene Test

- used to compare the variances of different groups (homoscedasticity)
- similar application as aov

```
library(car)

leveneTest(Sepal.Length, Species, data=iris, center="mean")
```

- in this case since there is a significant p value we can assume var is unequal

Exericse ANOVA

```
set.seed(234)
myobject = data.frame(group=rep(c("a","b","c"),10),
                      numeric=c(rnorm(5,5),6:15, rep(c(1,20,98),5)))
```

- Create the object myobject as stated above. There are 3 groups in it and every group has
- 10 observations in the column numeric
- get 3 different visual impressions of this data (hint! jitter, boxpl, violin could work)
- perform an ANOVA and do at least 2 post hoc tests (tests for multiple comparison!)
- hint: think about ways of adjusting p values (multiplicity!)
- what is the problem if you would not adjust for p values in this post hoc tests?
- which test would you choose if myobject would not be normally distributed?
- perform a non-parametric test instead of ANOVA

Solution

- myobject: 3 groups, balanced, rnorm!

```
set.seed(234)
myobject = data.frame(group=rep(c("a","b","c"),10),
                      numeric=c(rnorm(5,5),6:15, rep(c(1,20,98),5)))

levels(myobject$group)
```

- simple boxplot in base

```
boxplot(data=myobject, numeric~group)
```

- violin plot gets a better view on the distribution of the data

```
library(lattice)
bwplot(data=myobject, numeric~group, panel=panel.violin)
```

- jitter plot displays all the points individually

```
library(ggplot2)
qplot(data=myobject, formula=y~x, x=group, y=numeric, geom="jitter")

myanova=aov(data=myobject, numeric~group)
```

```
summary(myanova)

TukeyHSD(myanova)
plot(TukeyHSD(myanova))
coefficients(myanova)
```

- pairwise t test with adjusted p value as an alternative post hoc test

```
pairwise.t.test(x=myobject$numeric, g=myobject$group, p.adj="BH")

library(DTK)
```

- Tukey Kramer test

```
TK.test(x=myobject$numeric, f=myobject$group)
```

- we need those post hoc tests to adjust the p values - otherwise the T1 error rate
- would be inflated (higher than significance level - o.o5)

```
kruskal.test(data=myobject, numeric~group)
```

Kruskal Wallis test

- for non-normally distributed data - non parametric
- equivalent to ANOVA

```
kruskal.test(data=iris, Sepal.Length~Species)
```

Chi Square of independence test – parametric

- to test the independence of 2 categorical variables

```
attach(mtcars)
```

- we are now going to test if there is a correlation between vs and am

```
?chisq.test
```

- contingency table

```
table(am,vs)
```

- first visual impression

```
barplot((table(am,vs)), beside=T)
```

- the test itself

chisq.test(table(am,vs)) # **test for independence (H0)**

- Fishers exact test - does the same thing for non-parametrics

?fisher.test

fisher.test(table(am,vs))

Exercise Chi Squared Test

- data = bacteria, library = MASS
- get familiar with the dataset and get a contingency table of y and treatment group
- get a visual impression of the data
- check for independence with a suitable test

Solution

library(MASS)

?bacteria

head(bacteria)

attach(bacteria)

table(y, trt)

barplot((table(y,trt)), beside=T)

chisq.test(table(y, trt))

- independence can be rejected

Correlations

Pearson correlation coefficient - 2 numeric variables, linear coefficient, parametric

Spearman rank correlation coefficient - 2 numeric variables, monotonic coefficient, non-parametric

Kendall rank correlation coefficient - x-y concordance of pairs, non-parametric

?cor.test

plot(Sepal.Length, Sepal.Width)

cor(Sepal.Length, Sepal.Width, method="pearson") # pearson is default,
 # x y order does not matter

cor(Sepal.Length, Sepal.Width, method="spearman")

```
cor(Sepal.Length, Sepal.Width, method="kendall")
```

- for a more specific output

```
cor.test(Sepal.Length, Sepal.Width, method="pearson")
```

- to get the covariance

```
cov(Sepal.Length, Sepal.Width)
```

- we can get a correlation matrix

```
cor(iris[,1:4]) # the character column Species must be excluded
```