



**University of
Zurich**^{UZH}



MAKERERE UNIVERSITY

Data Analysis with R:

Lecture Slides (all)

Sonja Hartnack, Terence Odoch & Muriel Buri

July 2019

Goals of the course

To be able to...

- import data sets to R
- describe data with R
- apply basic statistical tests in R
- some ideas for more advanced statistical tools ...
- simulate a data set similar to own research

General remarks

Course schedule:

- Starting at 9:00am / 9:30am (?)
- Tea breaks in between
- Lunch break
- Teaching until 4.30pm (~ 5pm)

Obtaining a certificate is conditional on:

- active participation in class
- attending at least 75 % of the course (lecture & exercises)
- assignments during now and October
- short final exam in October (format to be defined)

Getting to know each other

- My name is ...
- I am doing a Master / a PhD in ...
- I hope to learn in this course how to
- My personal goal for this course is ...

How do we reach these goals

- hands on exercises with R:
 - chickwts
 - ToothGrowth
 - bacteria
 - perulung
 - ... and others.
- interactive discussions & student's present their own solutions
- ask us a lot of questions but also ask google for help!
- group work
- short motivational lectures



Do you all have RStudio and R installed on your computers?

Get started with data set: chickwts



An experiment was conducted to measure and compare the effectiveness of various feed supplements on the growth rate of chickens.

```
# load data set "chickwts"  
data("chickwts", package = "datasets")  
# the head(...) function shows the first 6 observations  
head(chickwts)
```

```
##      weight      feed  
## 1      179 horsebean  
## 2      160 horsebean  
## 3      136 horsebean  
## 4      227 horsebean  
## 5      217 horsebean  
## 6      168 horsebean
```

```
# FUNCTION - open bracket - DATA SET / VARIABLE - close bracket
```

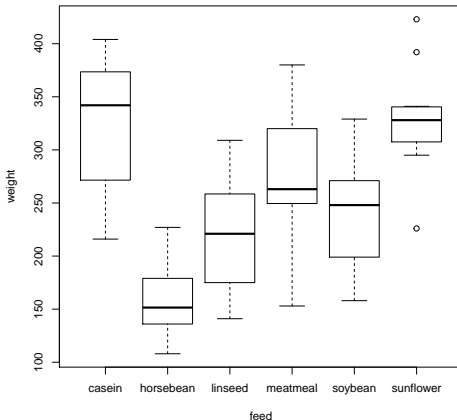
Ideas for plotting the data



Ideas for plotting the data



```
# use x axis to show the categorical variable (feed),  
# y axis to represent the continuous variable (weight)  
# boxplot (y.cont.variable ~ x.cat.variable, data = dataset)  
# ?boxplot  
boxplot(weight ~ feed, data = chickwts)
```



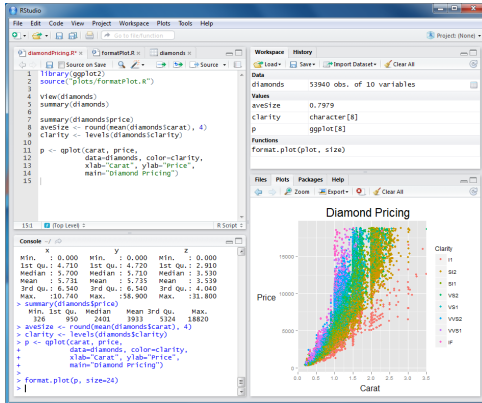
Exercise: Statistical Terminologies



Functionalities in RStudio



- Source
- Console
- Environment, History, Files
- Files, Plots, Packages, Help





- Define manually a new folder called **rcourse** in your personal documents on your personal computer
- Know in which directory you are

```
getwd()
```

```
## [1] "/home/mburi/Documents/git_svn/DataAnalysisWithR/Lectures"
```

- Set directory path

```
# back- and forslash is dependent on the system  
setwd("C:/Users/muriel/Documents/rcourse/")  
setwd("C:\\Users\\muriel\\Documents\\rcourse\\")
```

- Always clean up before starting with new R-Script

```
rm(list=ls()) # empty workspace, delete previously saved variables
```



```
?chickwts  
?boxplot
```

Also, have a look at the examples at the end of the help pages.

Exercise: Getting to know R and `chickwts`



A data frame in R: chickwts



chickwts[ROWS , COLUMNS]

	weight	feed
1	179	horsebean
2	160	horsebean
3	136	horsebean
4	227	horsebean
5	217	horsebean
6	168	horsebean
7	108	horsebean
8	124	horsebean
9	143	horsebean
10	140	horsebean
11	309	linseed
12	229	linseed
13	181	linseed

chickwts[6, 1]

	weight	feed
1	179	horsebean
2	160	horsebean
3	136	horsebean
4	227	horsebean
5	217	horsebean
6	168	horsebean
7	108	horsebean
8	124	horsebean
9	143	horsebean
10	140	horsebean
11	309	linseed
12	229	linseed
13	181	linseed

chickwts[11, 2]

Rows and columns of a data frame: chickwts



Values of ...

```
# Load (internal) data set from R
data("chickwts")

# ... all columns of sixth observation:
chickwts[6, ]

# ... all columns of sixth to eleventh observation:
chickwts[c(6:11), ]

# ... all columns of sixth, eleventh and twentieth observation:
chickwts[c(6, 11, 20), ]

# ... all rows of first column (weight):
chickwts[, 1]

# ... all rows of second column (feed):
chickwts[, 2]

# or use the "$" sign as a reference to column "feed":
chickwts$feed
```


Exercise: Summary statistics for the `chickwts` data set



Rules for importing data into R



- First row of excel sheet contains **variable names**:
y, ap, hilo, week, ID, trt.
- Columns of excel sheet represent **variables**.
- Rows of excel sheet represent **observations per individual** (except for the first row).

	A	B	C	D	E	F	G	H	I	J
1	id	fev1	age	height	sex	resp	symptoms			
2	1	1.56	9.593	124.8	0	0				
3	2	1.18	7.491	111	1	0				
4	3	1.87	9.864	135.7	0	0				
5	4	1.49	8.588	119.1	0	0				
6	5	1.62	8.967	120.9	1	0				
7	6	2.11	9.293	134.3	0	1				
8	7	1.73	9.574	122.1	1	0				
9	8	1.47	8.493	122.6	0	1				
10	9	1.83	8.468	126.8	1	0				
11	10	1.41	9.029	126	0	0				
12	11	1.27	8.274	128	0	0				
13	12	1.34	8.416	127	0	0				
14	13	1.64	9.629	133.7	0	0				
15	14	1.57	8.622	125.5	1	0				
16	15	1.51	9.033	125.9	1	0				
17	16	1.25	8.643	122.3	0	0				



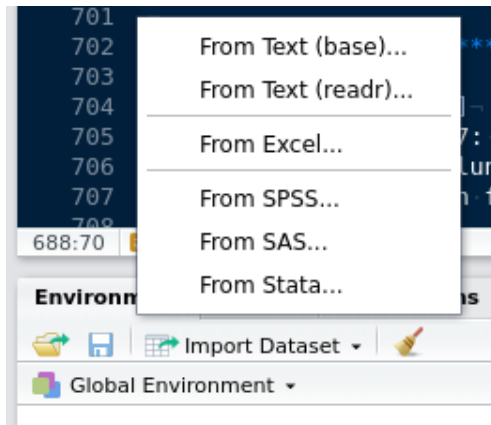
Variable names should ...

- start with a letter (not a number):
y, ap, hilo, week, ID, trt
- longer variables names should be separated with dots:
time.in.weeks
- do not use special characters, such as /, #, @, &, *, ...

How to import external data files into R?



- > Import Dataset > **From Text (base)...** > CSV Files (.csv)
or





- Environment (upper right corner)
- > Import Dataset > **From Text (base)...** > CSV Files (.csv)

```
perulung_ems <- read.csv("perulung_ems.csv", row.names = 1,  
                        sep = ";")  
lung <- data.frame(perulung_ems)
```

- > Import Dataset > **From Text (base)...** > Text Files (.txt)

How to import .txt and .csv files into R? (2/3)

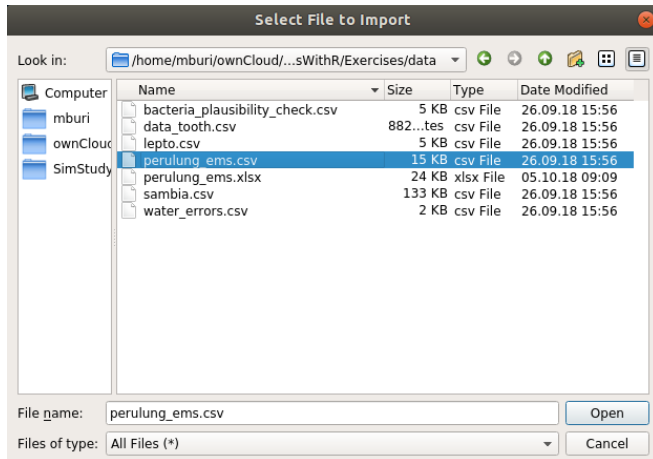


- Environment (upper right corner)
- > Import Dataset > From Text (base)... > CSV Files (.csv)

How to import .txt and .csv files into R? (2/3)



- Environment (upper right corner)
- > Import Dataset > From Text (base)... > CSV Files (.csv)



How to import .txt and .csv files into R? (3/3)



Import Dataset

Name:

Encoding:

Heading: ☒ Yes ☐ No

Row names:

Separator:

Decimal:

Quote:

Comment:

na.strings:

☒ Strings as factors

Input File

```
id;fev1;age;height;sex;respsymptoms
1;1.56;9.593;124.8;0;0
2;1.18;7.491;111.1;0
3;1.87;9.864;135.7;0;0
4;1.49;8.588;119.1;0;0
5;1.62;8.967;120.9;1;0
6;2.11;9.293;134.3;0;1
7;1.73;9.574;122.1;1;0
8;1.47;8.493;122.6;0;1
9;1.83;8.468;126.8;1;0
10;1.41;9.029;126.0;0;0
11;1.27;8.274;128.0;0;0
12;1.34;8.416;127.0;0;0
13;1.64;9.629;133.7;0;0
14;1.57;8.622;125.5;1;0
```

Data Frame

id	fev1	age	height	sex	respsymptoms
1	1.56	9.593	124.8	0	0
2	1.18	7.491	111.0	1	0
3	1.87	9.864	135.7	0	0
4	1.49	8.588	119.1	0	0
5	1.62	8.967	120.9	1	0
6	2.11	9.293	134.3	0	1
7	1.73	9.574	122.1	1	0
8	1.47	8.493	122.6	0	1
9	1.83	8.468	126.8	1	0
10	1.41	9.029	126.0	0	0
11	1.27	8.274	128.0	0	0
12	1.34	8.416	127.0	0	0
13	1.64	9.629	133.7	0	0
14	1.57	8.622	125.5	1	0

```
perulung_ems <- read.csv("perulung_ems.csv", row.names = 1,
                        sep = ";")
lung <- data.frame(perulung_ems)
```




Data from a study of lung function among children living in a deprived suburb of Lima, Peru. Data taken from Kirkwood and Sterne, 2nd edition.

Variables:

- `fev1`: in liter, "forced expiratory volume in 1 second" measured by a spirometer. This is the maximum volume of air which the children could breath out in 1 second
- `age`: in years
- `height`: in cm
- `sex`: 0 = girl, 1 = boy
- `respsymp`: respiratory symptoms experienced by the child over the previous 12 months

What is a data frame in R?



A data frame is used for storing a list of vectors of equal length. For example, the following variable `df` is a data frame containing three vectors `n`, `s`, `b`.

```
n <- c(2, 3, 5)
s <- c("aa", "bb", "cc")
b <- c(TRUE, FALSE, TRUE)
df <- data.frame(n, s, b) # df is a data frame
```

The characteristics of a data frame are:

- The column names should be non-empty.
- The row names should be unique.
- Each column should contain same number of data items.



```
a <- c(1, 2, 3, 4)
a

## [1] 1 2 3 4

data.frame(a)

##      a
## 1 1
## 2 2
## 3 3
## 4 4

b <- c("d", "h", "h", "d")
mydat <- data.frame(a, b)
mydat

##      a b
## 1 1 d
## 2 2 h
## 3 3 h
## 4 4 d
```

Data frame in R: How to add a variable



```
vartoadd <- c(1.3, 1.5, 1.8, 2.4)
# use "$" to refer to the additional vector variable
mydat$myvar1 <- vartoadd
mydat$myvar2 <- vartoadd
mydat

##    a b myvar1 myvar2
## 1 1 d    1.3    1.3
## 2 2 h    1.5    1.5
## 3 3 h    1.8    1.8
## 4 4 d    2.4    2.4

# What is the dimension (number of rows and columns) of our data frame?
dim(mydat) # 4 rows and 4 columns

## [1] 4 4
```

Exercise: Defining a new data frame





Objects are assigned values using $<-$, an arrow formed out of $<$ and $-$. For example, the following command assigns the value 1 to the object a.

```
a <- 1 # ALWAYS use "gets" assignment operator!  
# a = 1 # DO NOT USE the equal sign as the assignment operator!
```

After this assignment, the object a contains the value 1. Another assignment to the same object will change its value.

```
a <- 5
```

Examples of assigned objects: single number



```
a <- 1
b <- 2
c <- a + b # c = 3
c

## [1] 3
```

Examples of assigned objects: vector



```
a <- c(1, 2, 3, 4, 5)
```

```
b <- 1
```

```
c <- a + b
```

```
c
```

```
## [1] 2 3 4 5 6
```


Examples of assigned objects: model



```
anova_model <- aov(weight ~ feed, data = chickwts)
summary(anova_model)
```

```
##              Df Sum Sq Mean Sq F value    Pr(>F)
## feed           5 231129   46226    15.37 5.94e-10 ***
## Residuals     65 195556     3009
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Examples of assigned objects: data frame



```
bac <- bacteria
str(bac) # $ week: int  0 2 4 11 0 2 6 11 0 2 ...

## 'data.frame': 220 obs. of  6 variables:
## $ y   : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 1 2 2 2 ...
## $ ap  : Factor w/ 2 levels "a","p": 2 2 2 2 1 1 1 1 1 1 ...
## $ hilo: Factor w/ 2 levels "hi","lo": 1 1 1 1 1 1 1 1 2 2 ...
## $ week: int  0 2 4 11 0 2 6 11 0 2 ...
## $ ID  : Factor w/ 50 levels "X01","X02","X03",...: 1 1 1 1 2 2 2 2 3 3 ...
## $ trt : Factor w/ 3 levels "placebo","drug",...: 1 1 1 1 3 3 3 3 2 2 ...

bac_sub <- subset(bac, week == 2)
str(bac_sub) # $ week: int  2 2 2 2 2 2 2 2 2 2 ...

## 'data.frame': 44 obs. of  6 variables:
## $ y   : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 1 2 2 2 ...
## $ ap  : Factor w/ 2 levels "a","p": 2 1 1 2 2 1 1 2 2 2 ...
## $ hilo: Factor w/ 2 levels "hi","lo": 1 1 2 2 2 2 1 1 2 1 ...
## $ week: int  2 2 2 2 2 2 2 2 2 2 ...
## $ ID  : Factor w/ 50 levels "X01","X02","X03",...: 1 2 3 4 5 6 7 8 9 11 ...
## $ trt : Factor w/ 3 levels "placebo","drug",...: 1 3 2 1 1 2 3 1 1 1 ...
```



The `str` function displays the structure of an R object. One line for each "basic" structure is displayed.

```
## 'data.frame': 44 obs. of 6 variables:
## $ y : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 1 2 2 2 ...
## $ ap : Factor w/ 2 levels "a","p": 2 1 1 2 2 1 1 2 2 2 ...
## $ hilo: Factor w/ 2 levels "hi","lo": 1 1 2 2 2 2 1 1 2 1 ...
## $ week: int 2 2 2 2 2 2 2 2 2 2 ...
## $ ID : Factor w/ 50 levels "X01","X02","X03",...: 1 2 3 4 5 6 7 8 9 11 ...
## $ trt : Factor w/ 3 levels "placebo","drug",...: 1 3 2 1 1 2 3 1 1 1 ...
```

Exercise: Different bracket types within R



Data types in R



- numeric

```
data(ToothGrowth)
head(ToothGrowth$len)

## [1]  4.2 11.5  7.3  5.8  6.4 10.0

class(ToothGrowth$len)

## [1] "numeric"
```

- integers

```
head(bacteria$week)

## [1]  0  2  4 11  0  2

bacteria$week[1:6]

## [1]  0  2  4 11  0  2

class(bacteria$week)

## [1] "integer"
```

- (un/ordered) factor

```
chickwts$feed[1:6]

## [1] horsebean horsebean horsebean horsebean horsebean horsebean
## Levels: casein horsebean linseed meatmeal soybean sunflower

levels(chickwts$feed)[1:3]
```



Ordinal variables are represented as ordered factors:

```
bac_growth <- c("none", "+", "++", "+", "+++", "+", "none") # vector
bac_growth <- factor(bac_growth, levels = c("none", "+", "++", "+++"),
                     order = TRUE)

bac_growth

## [1] none +      ++      +      +++     +      none
## Levels: none < + < ++ < +++

#
mood <- c("OK", "Well", "Super", "Super", "Don't ask", "OK") # vector
mood <- factor(mood, levels = c("Don't ask", "Well", "OK", "Super"),
               order = TRUE)

mood

## [1] OK          Well          Super          Super          Don't ask OK
## Levels: Don't ask < Well < OK < Super
```



- numeric variable
- integer variable
- variable with two levels (binary factor)
- ordered variable with **more than** two levels (ordinal)
- unordered variable with **more than** two levels (nominal)

Exercise: Data type of `perulung_ems` data set





<https://www.datacamp.com/community/tutorials/r-packages-guide>

COMMUNITY

News BETA

Tutorials

Cheat Sheets

Open Courses

Podcast - DataFramed

Chat NEW

DATA CAMP

Official Blog

Tech Thoughts

Search

Log In

Adolfo Alvarez
March 23th, 2019

R PROGRAMMING +1

R Packages: A Beginner's Guide

An introduction to R packages based on 11 of the most frequently asked user questions.

4

4/7

f

t

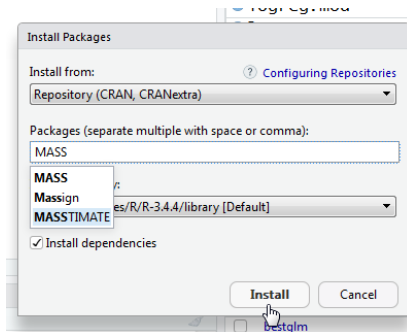
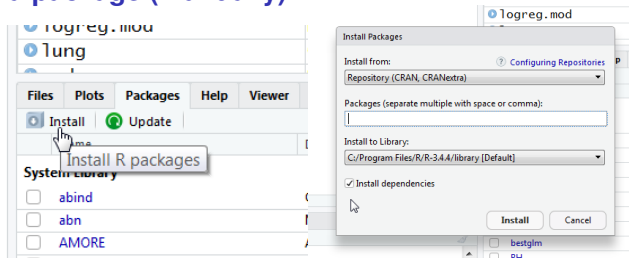
in

R packages are collections of functions and data sets developed by the community. They increase the power of R by improving existing base R functionalities, or by adding new ones. For example, if you are usually working with data frames, probably you will have heard about `dplyr` or `data.table`, two of the most popular R packages.

But imagine that you'd like to do some natural language processing of Korean texts, extract weather data from the web, or even estimate actual evapotranspiration using land surface energy balance models, R packages got you covered! Recently, the official repository (CRAN) reached 10,000 packages published, and many more are publicly available through the internet.

If you are starting with R, today's post will cover the basics of R packages and how to use them. You'll cover the following topics, and 11 frequently asked user questions:

How to install a package (manually) in R



Using R is like cooking ...

Get into the kitchen	Change working directory
Get specialist electric tools into your kitchen (e.g. blender, ice-cream maker, etc.)	Install packages
Switch on your specialist electric tools	Load packages using the "library" function
Bring in your ingredients	Import data and save to R data frames
Check your ingredients	Use the function "summary" and basic tables to check your data for missing or implausible values (e.g. a number in a variable where "yes" or "no" are expected)
Chop things up (if required)	Split or filter data
Cook, using general and specialist tools	Carry out further descriptive and test statistics

How to install a package in R



```
# INSTALL package (only done ONCE!)  
install.packages("MASS")  
# LOAD package (whenever you use something from it!)  
library("MASS")  
data(bacteria)  
?bacteria
```

Exercise: Get to know bacteria data set



How to google for getting help in R

- Google for **select observations in R**.

Why do we need Statistics?

Repeatability of results:

Statistical science allows us to estimate what might happen if an experiment was repeated - but without having to actually repeat it!

Why do we need Statistics?

- Study results must be shown to be robust, i.e. real and not due to random chance
- Best way to demonstrate this is to repeat the same experiment/study many times each with **different** subjects (animals) drawn from the **same study population** and show that the result is truly repeatable
- It is generally totally impractical, in terms of both time and resources, to repeat an experiment many times!

Why do we need Statistics?

- Instead of repeating the experiment many times **probability theory i.e. statistics** is used to **estimate** what might have happened if the experiment had been repeated
- A mathematical model is used to fill this “data gap”
- Generally the most difficult task in statistics is to decide what “model” is most appropriate for a given experiment

What is Statistics? - A definition

A set of analytical tools designed to quantify uncertainty

- If an experiment or procedure is repeated, how likely is it that the new results will be similar to those already observed?
- What is the likely variation in results if the experiment was repeated?

What is Statistics? - A definition

The key scientific purpose of statistics

- to provide **evidence** of the existence of some “effect” of scientific interest
- i.e. evidence based medicine

As a reminder: The importance of study design

Even the most sophisticated statistical analyses cannot rescue a poorly designed study

→ unreliable results

→ inability to answer the main research question

Putting Statistics in Context

- Use common sense as a guide - be skeptical!
- Terminology can also differ greatly between textbooks...
- Wikipedia as good a resource

Exploratory Data Analysis

- get first impression and feeling of the data set
- detect outliers / mistake of data collection
- possibly recode variables

Summary Statistics

Continuous (Integers / Numeric)

- Mean - a measure of location. Always examine the average value of the response variable(s) for the different “treatment” effects in your data
- Median - a robust single value summary of a set of data (50% quantile point) - most useful in highly skewed data or data with outliers
- Standard deviation (sd) - a measure of spread, how variable the data are
- Standard error of the mean (se) - an estimate of how far the sample mean is likely to be from the population mean
- and others: min, max, range, IQR, ...



```
mean(x) # mean
```

```
median(x) # median
```

```
sd(x) # standard deviation
```

```
min(x) # minimum
```

```
max(x) # maximum
```

```
range(x) # range
```

```
IQR(x) # interquartile range
```


Continuous Data Summaries

standard deviation

$$s = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}$$

standard error

$$se = \frac{s}{\sqrt{n}}$$

Correlation coefficient

Combination of continuous and continuous

Correlation coefficient a measure association between two continuous variables (common but somewhat limited)

Pearson's correlation coefficient r

$$r = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2} \sqrt{\sum_{i=1}^n (Y_i - \bar{Y})^2}}$$

\bar{X} : mean of variable x

\bar{Y} : mean of variable y



```
# Test for Association/Correlation Between  
# Paired Samples  
cor.test(data$x, data$y, method = "pearson")  
cor.test(data$x, data$y, method = "spearman")  
  
# Scatterplot(s)  
pairs(data$x ~ data$y)  
pairs(data)
```

Summary Statistics

Continuous and factor variables



```
tapply(data$x.cont, data$y.fac, mean)
```

```
tapply(data$x.cont, data$y.fac, median)
```

```
tapply(data$x.cont, data$y.fac, sd)
```



- Median - a robust single value summary of a set of data (50% quantile point) - most useful in highly skewed data or data with outliers
- e.g. 10th and 90th percentile - a measure of spread, how variable the data are

```
quantile(x, probs = c(0.1, 0.9))
```



- proportions - e.g. percentage per grade

```
prop.table(table(data$x.fac))  
prop.table(table(data$x.fac, data$y.fac))
```

- contingency tables e. g. 2 x 2

```
table(data$x.fac)  
table(data$x.fac, data$y.fac)  
prop.table(table(data$x.fac))  
prop.table(table(data$x.fac, data$y.fac))
```

Exercise: Get to know ToothGrowth data set



How to deal with missing values in R? (1/3)

- In R, missing values are represented by the symbol **NA** (not available).
- Impossible values (e. g., dividing by zero) are represented by the symbol **NaN** (not a number).
- Ask yourself why a **NA** and / or **NaN** occurs!

How to deal with missing values in R? (2/3)

- Testing for Missing Values

```
vec1 <- c(1, 2, 3, NA)
is.na(vec1) # returns a vector (FALSE, FALSE, FALSE, TRUE)
# The TRUE indicates the position of the NA in vec1.
```

- Recoding Values to Missing

```
# recode specific values (e. g. 0.001) to missing for variable x
# select rows where x is 0.001 and recode value in column x with NA
tmp.row <- which(dat$x == 0.001)
dat$x[tmp.row] <- NA
```

How to deal with missing values in R? (3/3)

- Excluding Missing Values from specific function calls

```
a <- c(1, 2, NA, 3)
mean(a) # returns NA
mean(a, na.rm=TRUE) # returns 2
```

- Check for complete cases with function `complete.cases(...)`

```
# list rows of data that have missing values
dat[!complete.cases(dat),]
subdat <- dat[complete.cases(dat),]
```

- Create new dataset without missing data with function `na.omit(...)`

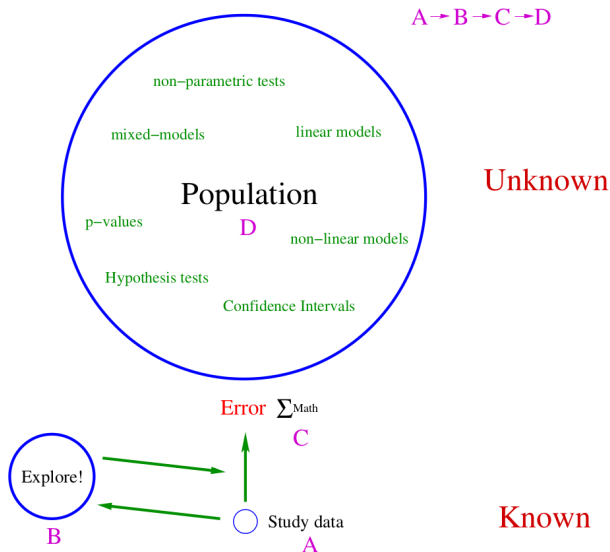
```
new.dat <- na.omit(dat)
```

How to check your data for plausibility?

- Ask yourself what can go wrong?
- Implausible values?
- Impossible values?
- Logical errors?

Data plausibility checks & Missing values

Overview



Basic Statistical Tests

Study data is collected for a purpose - to answer one or more specific scientific questions. The classical way to perform a formal statistical analyses of these data is to formulate these research questions into statistical **hypothesis tests**.

In this section we will go through a simple example in detail to highlight some of the important concepts - the general approach for more complex analyses is exactly same. *Note: the precise technical details are much less important than the concepts!*

Simple Example - One Population

After six weeks will the mean weight of a chicken be more than 250 grams?

There are 71 observations in `chickwts` from which to answer this question. This can be formulated into a statistical hypothesis test. A hypothesis test has two parts, the null hypothesis and the alternative hypothesis. This is typically written as follows:

$$H_0 : \mu \leq 250$$

$$H_A : \mu > 250$$

where μ is the mean weight in the **population** of chickens from which the sample of 71 chickens was drawn.

Simple Example - One Population

After six weeks will the mean weight of a chicken be at least 250 grams?

$$H_0 : \mu \leq 250$$

$$H_A : \mu > 250$$

The null hypothesis (H_0) is the default situation, sometimes called the “state of nature”. In a treatment-control trial, H_0 is typically that the effect of the treatment is not different from the control. In this example our default position is that the mean weight of chickens is ≤ 250 . This is called a single-sided hypothesis test.



We now analyse the 71 observations to see whether there is evidence to **REJECT** the null hypothesis H_0 , and if the null hypothesis is rejected then we can conclude that the available evidence supports the alternative hypothesis.

```
t.test(chickwts$weight, mu = 250, alternative = "greater")  
t.test(chickwts$weight, mu = 250, alternative = "less")
```

Note that hypothesis testing is concerned with finding evidence in support of the null hypothesis H_0 - the default situation - rather than evidence in favour of the alternative hypothesis.

One Sample t-test

For the chicken weights data an appropriate formal analyses is to use a **one-sample t-test**. This analysis involves calculating a simple summary statistic - called a *t*-statistic - which we do entirely from the observed data.

$$T_{obs} = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

where \bar{x} is the sample mean, s the sample standard deviation and μ is the population mean in the null hypothesis which we wish to test for. We then look up the value of T_{obs} in a set of statistical tables/computer to see what the answer is to our research question.

Important concept - sampling

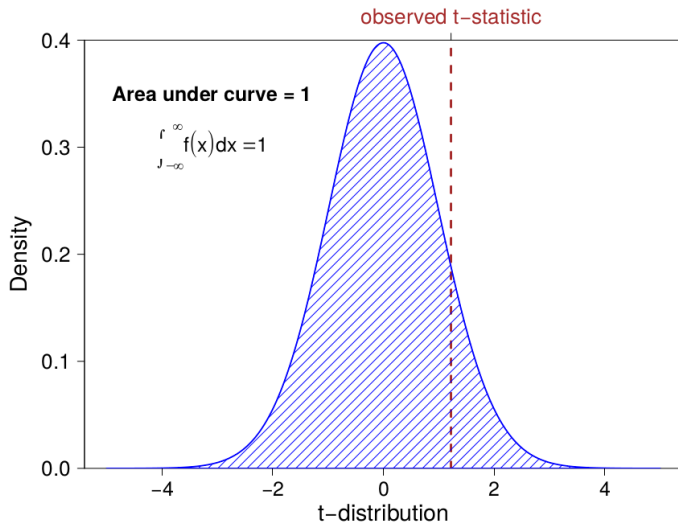
Why is $T_{obs} = \frac{\bar{x} - \mu}{s / \sqrt{(n)}}$ called a t -statistic?

If another sample of 71 chickens from the same population were weighed then the values for \bar{x} and s would be different, and hence the value for T_{obs} . If this was repeated many times and a histogram/Q-Q/P-P plot produced of the values for T_{obs} then this would follow the shape of a known distribution - **student-t probability distribution**. It is this piece of mathematics - knowing what the sampling distribution of T_{obs} is - which allows us to infer information about the population of chickens from which our original 71 chickens were sampled - without actually having to collect lots and lots of other samples of chickens! Mathematical theory is used to fill this data gap.

$$T_{obs} = \frac{261.31 - 250}{78.07 / \sqrt{71}} = 1.22$$

Put the values for the sample mean and standard deviation into the t-statistic formula along with the $\mu = 250$. We now look up the value of this in a t-distribution reference table. All this calculation will be done for you in R but it is important to understand the general process as this is the same for hypothesis testing in other more complex analyses.

One Sample, one-sided, t-test



Important concept - p -values

- The result of a hypothesis test is usually communicated in the form of a **p -value**
- The interpretation of a p -value is of crucial importance - it is the *probability that the test statistic takes values at least as extreme as that observed **assuming that H_0 is true***
- Exactly what **at least as extreme as** refers to depends on the alternative hypothesis H_A .
- This may sound rather abstract but it is usually obvious in practice

Simple Example - One Population

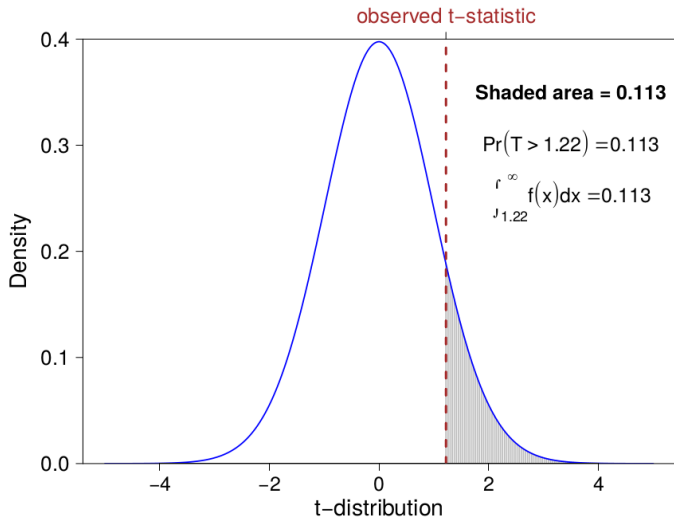
After six weeks will the mean weight of a chicken be at least 250 grams?

$$H_0 : \mu \leq 250$$

$$H_A : \mu > 250$$

The alternative hypothesis is $\mu > 250$ so in this test **at least as extreme as** in the definition of a p -value is the probability of observing a t-statistic which is > 1.22 **assuming that H_0 is true** - this is why 250 was used for μ when calculating T_{obs} .

One Sample, one-sided, t-test



Research Question

The purpose of this hypothesis test analysis is to answer a very specific scientific question:

After six weeks will the mean weight of a chicken be more than 250 grams?

So what is our answer?

The p -value for this hypothesis test is 0.113. Based on this value we can either **reject** H_0 and conclude that the mean weight of chickens in the population is likely to be greater than 250 grams or else we can **accept** H_0 where the mean chicken weight is less than 250 grams.

Research Question - be pragmatic with p -values

By convention a p -value of less than 0.05 is considered to provide **reasonable evidence** for rejecting H_0 . A p -value of between 0.05 and 0.1 might be considered as **weak evidence** against H_0 . Values of less than 0.01 are generally considered as **very strong evidence** for rejecting H_0 . It is **always** best to provide a p -value in any analyses to let the reviewer/client see the strength of evidence rather than simply claiming statistically significant findings!

Communicating Results of Hypothesis Tests

Transparency is essential - the devil can be in the detail - which at the very least should comprise:

- i. what hypothesis was being tested - be clear and precise
- ii. what statistical test was used
- iii. what the p -value is
- iv. what the treatment effect is (more later).

This is particularly crucial if the analyses are to be given to someone *e/se* to then make a judgment on the scientific significance.

Two-sided Tests: One Population

After six weeks will the mean weight of a chicken be equal to 250 grams?

This is now a two sided hypothesis test:

$$H_0 : \mu = 250$$

$$H_A : \mu \neq 250$$

This time the hypothesis test is asking how much evidence is there in our sample data to conclude that in the population of all chickens the mean weight is not equal to 250 grams.

Two-sided Tests: One Population



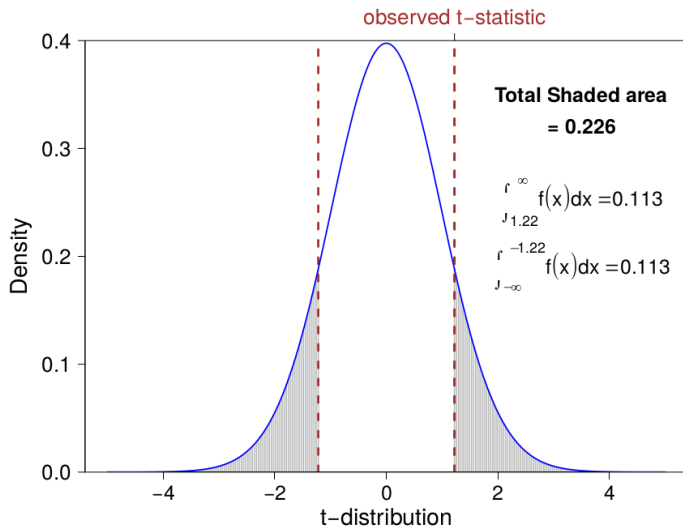
```
# t.test(chickwts$weight, mu = 250, alternative = "two.sided")
t.test(chickwts$weight, mu = 250)

##
## One Sample t-test
##
## data: chickwts$weight
## t = 1.2206, df = 70, p-value = 0.2263
## alternative hypothesis: true mean is not equal to 250
## 95 percent confidence interval:
## 242.8301 279.7896
## sample estimates:
## mean of x
## 261.3099
```

Two-sided Tests

A two-sided test is similar to a one-sided test - the key difference is in what is now defined as **at least as extreme** in the definition of the p -value. This time the alternative hypothesis refers to observing a value of **either** $\bar{x} > 250$ or $\bar{x} < -250$ **assuming that H_0 is true**, which using the t-test approach is equivalent to the probability of observing $T_{obs} > 1.22$ or $T_{obs} < -1.22$ which we can again look up in reference tables.

One Sample, two-sided, t-test



Two-sided Tests

- The two-sided t-test has a p -value which is exactly double the single sided test.
- Think! - Intuitively the p -value should be less for a single sided test as the research question you are asking is much narrower e.g. greater than 250 grams, as opposed to whether the mean chicken weight might be **either** less than 250 grams **or greater** than 250 grams.

→ You are using the same amount of information (71 observations) to answer a narrower research question and so all else being equal you should expect a “more powerful” analyses (e.g. a lower p -value all else being equal)

Exercise: t -test in R



Chi-square Test

There are two very commonly used statistical tests for testing dependence between two categorical variables: Chi-squared test & Fisher's exact test.


To test independence of rows and columns

Risk Factor	Disease		Total
	+	-	
+	a	b	a+b
-	c	d	c+d
Total	a+c	b+d	n = a+b+c+d

- Assumptions: a, b, c, d must have at least 5 observations!

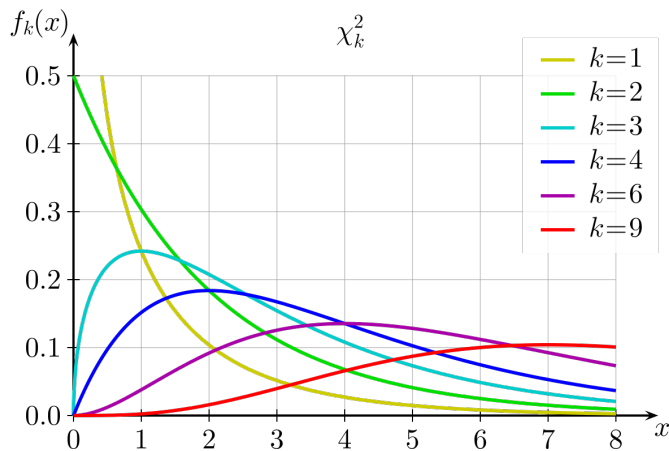
$$\chi^2 = \frac{n * (a * d - b * c)^2}{(a + b) * (c + d) * (a + c) * (b + d)}$$

Test
statistic chi-
square



$$\chi^2 = \sum \frac{(O - E)^2}{E^2}$$

The Chi-square Distribution



Exact Fisher Test: Permutation Test

	<u>Success</u>	<u>Failure</u>	Total
<u>Therapy</u>	7	2	9
<u>New Therapy</u>	2	8	10
Total	9	10	19

7	2
2	8
8	1
1	9
9	0
0	10

$$P = 9! * 10! * 9! * 10! / 19! * 7! * 2! * 2! * 8! = 0.01754$$

$$P = 9! * 10! * 9! * 10! / 19! * 8! * 1! * 1! * 9! = 0.00097$$

$$P = 9! * 10! * 9! * 10! / 19! * 9! * 0! * 0! * 10! = 0.00001$$

For a one-sided test: $p = 0.01754 + 0.00097 + 0.00001 = 0.01852$

Exercise: Chi-square test in R







- Continuous data
 - Histogram
 - Boxplot
- Nominal / Ordinal data
 - Barplot
 - Mosaicplot
 - Scatterplots

Overview: ANOVA and linear models

- Introduction to ANOVA
- How to run an ANOVA in R
- Checking model assumptions in R
- Multiple comparisons options in R
- ANOVA as a special case of a linear model
- The simple linear regression model
- The multiple linear regression model
- Model selection: R^2 and AIC
- Two-way Interactions in R
- Confounding

Hypothesis Testing - One Way ANOVA

We have seen how to perform hypothesis tests when comparing two populations using the two sample t-test. In many analyses we may have multiple populations - for example suppose we have a treatment which has a number of different levels high/medium/low/placebo, or equivalently a number of different treatments. What then is the hypothesis we wish to test?

Is there a difference in the effect of the treatment?

$$H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$$

$$H_A : \text{at least one pair of } \mu_1, \dots, \mu_k \text{ are different}$$

where μ_1, \dots, μ_k denote the mean effect of treatment levels 1 through k .

One Way ANOVA

Analysis of variance, **ANOVA**, to analyze differences between group means. The observed variance in the outcome variable is partitioned into components attributable to different sources of variation.

ANOVA estimates three sample variances (sum of squares)

- a total variance based on all observation deviations from the grand mean
- a variance based on the group mean deviations from the grand mean
- an (error) variance based on all the observation deviations from their group mean

$$\sum_i \sum_j (x_{ij} - \bar{x})^2 = \sum_i n_i (\bar{x}_i - \bar{x})^2 + \sum_i \sum_j (x_{ij} - \bar{x}_i)^2$$



Variance
between groups



Variance
within groups

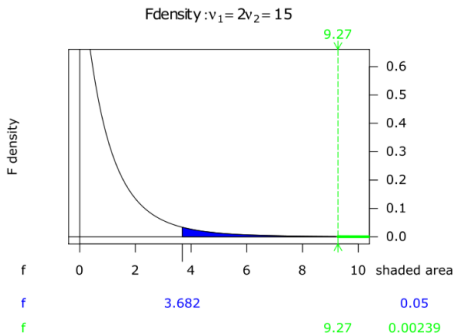
\bar{x} = grand mean
 \bar{x}_i = group mean

F-test / F-distribution

An **F-test**, a statistical test, in which the test statistic has an F-distribution under the null hypothesis is used to assess statistical significance in an ANOVA.

- degrees of freedom

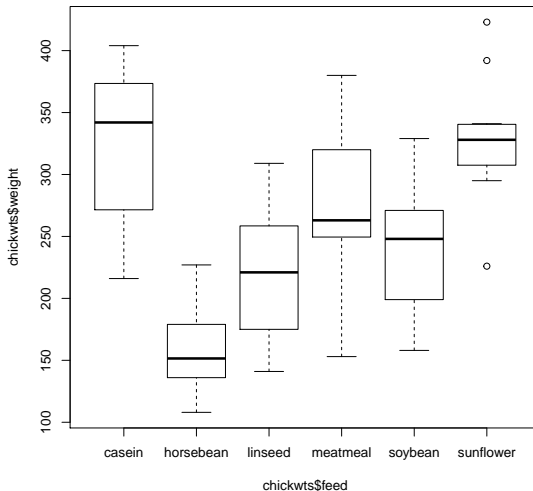
$$F = \frac{\text{variance between groups}}{\text{variance within groups}}$$



ANOVA in R with chickwts



```
data(chickwts)
boxplot(chickwts$weight ~ chickwts$feed)
```



```
# aov.mod <- aov(chickwts$weight ~ chickwts$feed)
aov.mod <- aov(weight ~ feed, data = chickwts)
# What objects can we extract from a anova model?
objects(aov.mod)

## [1] "assign"          "call"            "coefficients"    "contrasts"
## [5] "df.residual"     "effects"         "fitted.values"   "model"
## [9] "qr"              "rank"            "residuals"       "terms"
## [13] "xlevels"

#
summary(aov.mod)

##              Df Sum Sq Mean Sq F value    Pr(>F)
## feed          5 231129   46226    15.37 5.94e-10 ***
## Residuals    65 195556    3009
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Overview: Checking model assumptions

- $\text{mean}(\text{residuals}) = 0$
- Residuals are normally distributed (qqnorm, qqplot)
- Variance homoscedasticity (Bartlett & Levene's Test)
- Cook's distance: Influential data points
- Any pattern(s)?

Checking model residuals: $\text{mean}(\text{residuals}) = 0$



"Unexplained rest of the model"

```
# What are residuals?
chickwts$residuals <- residuals(aov.mod)
tapply(chickwts$weight, chickwts$feed, mean)

##      casein horsebean  linseed  meatmeal   soybean sunflower
## 323.5833  160.2000  218.7500  276.9091  246.4286  328.9167

chickwts[c(1:3),]

##   weight      feed residuals
## 1   179 horsebean    18.8
## 2   160 horsebean    -0.2
## 3   136 horsebean   -24.2

# Save residuals to an objects and check mean of residuals
aov.mod.resid <- residuals(aov.mod)
mean(aov.mod.resid)

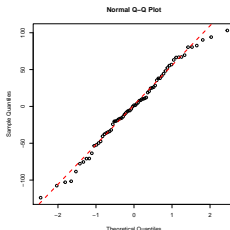
## [1] 7.573045e-16
```


Checking model residuals: Residuals normally distributed

"Unexplained rest of the model"



```
par(mfrow=c(1,1))
qqnorm(aov.mod.resid)
qqline(aov.mod.resid, col = "red", lwd = 3, lty = 2)
```



Shapiro-Wilk test (dependent on sample size --> limited use)

```
shapiro.test(aov.mod.resid)
```

```
##
```

```
## Shapiro-Wilk normality test
```

```
##
```

```
## data: aov.mod.resid
```

```
## W = 0.98616, p-value = 0.6272
```

Checking model residuals: Variance homoscedasticity (1/3)

Hypothesis tests



```
# Bartlett Test
bartlett.test(chickwts$weight ~ chickwts$feed)

##
## Bartlett test of homogeneity of variances
##
## data:  chickwts$weight by chickwts$feed
## Bartlett's K-squared = 3.2597, df = 5, p-value = 0.66

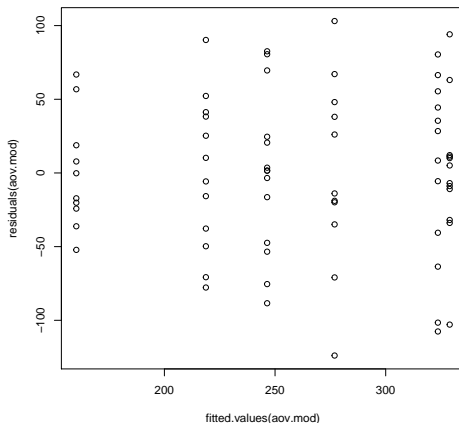
# Levene's Test
# library("Rcmdr")
# levene.test(chickwts$weight ~ chickwts$feed)
```

Checking model residuals: Variance homoscedasticity (2/3)

Graphical interpretation is better!



```
# Plot fitted against residual values  
plot(fitted.values(aov.mod), residuals(aov.mod))
```

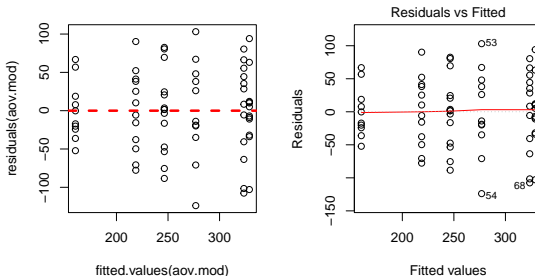


Checking model residuals: Variance homoscedasticity (3/3)

Graphical interpretation is better!



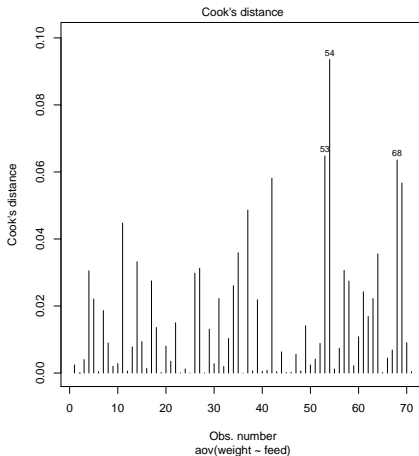
```
# Plot fitted against residual values
par(mfrow=c(1,2), pty="s", mar = c(10, 4, 1, 2))
plot(fitted.values(aov.mod), residuals(aov.mod))
abline(h = 0, col = "red", lwd = 3, lty = 2)
plot(aov.mod, which=1)
```



Checking model residuals: Cook's distance

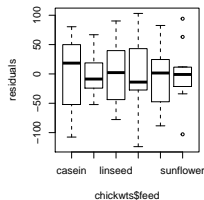
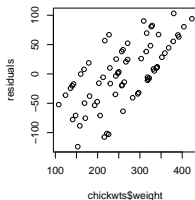
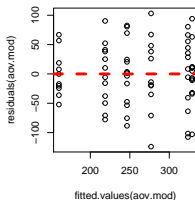


```
# Plot fitted against residual values  
# Cut-off at 3  
par(mfrow=c(1,1), pty="s", mar = c(5, 4, 4, 2))  
plot(aov.mod, which=4)
```



Checking for potential patterns

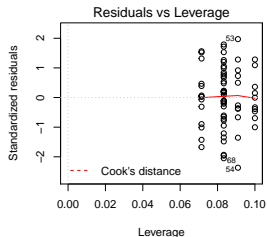
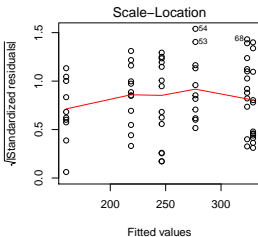
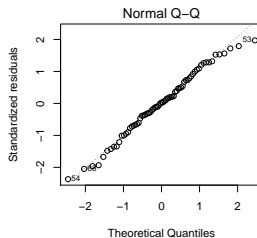
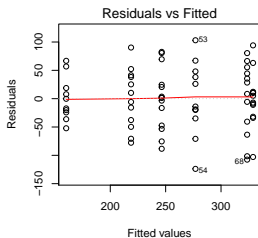
```
par(mfrow=c(1,3), pty="s", mar = c(10, 4, 1, 2))
plot(fitted.values(aov.mod), residuals(aov.mod))
abline(h = 0, col = "red", lwd = 3, lty = 2)
# Plot residuals against variables from the model
plot(chickwts$weight, residuals(aov.mod), ylab = "residuals")
plot(chickwts$feed, residuals(aov.mod),
     xlab = "chickwts$feed", ylab = "residuals")
```



Plot anova objects



```
par(mfrow=c(2, 2))  
plot(aov.mod)
```





So far, we know there is difference between the feed types.

However, we do not yet know which feed type differ.

In principal, we could do multiple t-tests, BUT ... we would use our data several times. Classically, we choose an α -level of 5 %, in cases of multiple comparisons we are facing the familywise error rate:

$$FWE \leq 1 - (1 - \alpha)^n,$$

where $\alpha = 5 \%$ and $n = \text{number of comparisons}$.

```
alpha = 0.05  
1 - (1 - alpha)^1    # n = 1      # [1] 0.05  
1 - (1 - alpha)^5    # n = 5      # [1] 0.2262191  
1 - (1 - alpha)^10   # n = 10     # [1] 0.4012631
```




Hence, we are better off using one of the following procedures to adjust for multiple comparisons:

- **Bonferroni:** p -value correction by testing each individual hypothesis at a significance level of $\frac{\alpha}{m}$, where α is the desired overall α level and m is the number of hypotheses.
- **Dunnett:** Multiple comparisons of each group to a reference.
- **Tukey Honest Significant Differences** (Homogeneous subgroups): Multiple comparisons of all possible combinations.
- ...

Multiple comparisons options in R: Bonferroni



```
aov.mod <- aov(weight ~ feed, data = chickwts)
pairwise.t.test(chickwts$weight, chickwts$feed, p.adj = "none")

##
## Pairwise comparisons using t tests with pooled SD
##
## data:  chickwts$weight and chickwts$feed
##
##           casein  horsebean  linseed  meatmeal  soybean
## horsebean 2.1e-09 -          -          -          -
## linseed   1.5e-05 0.01522    -          -          -
## meatmeal   0.04557 7.5e-06    0.01348 -          -
## soybean    0.00067 0.00032    0.20414 0.17255  -
## sunflower  0.81249 8.2e-10    6.2e-06 0.02644 0.00030
##
## P value adjustment method: none

pairwise.t.test(chickwts$weight, chickwts$feed, p.adj = "bonferroni")

##
## Pairwise comparisons using t tests with pooled SD
##
## data:  chickwts$weight and chickwts$feed
##
##           casein  horsebean  linseed  meatmeal  soybean
## horsebean 3.1e-08 -          -          -          -
## linseed   0.00022 0.22833    -          -          -
## meatmeal   0.68350 0.00011    0.20218 -          -
## soybean    0.00998 0.00487    1.00000 1.00000  -
## sunflower  1.00000 1.2e-08    9.3e-05 0.39653 0.00447
##
## P value adjustment method: bonferroni
```

Multiple comparisons options in R: Dunnett



```
library("multcomp")
# compares always to baseline levels (here: casein) --> saves degrees of freedom
dunnett <- glht(aov.mod, linfct = mcp(feed = "Dunnett"))
summary(dunnett)

##
## Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Dunnett Contrasts
##
##
## Fit: aov(formula = weight ~ feed, data = chickwts)
##
## Linear Hypotheses:
##
##           Estimate Std. Error t value Pr(>|t|)
## horsebean - casein == 0 -163.383    23.485  -6.957 < 0.001 ***
## linseed - casein == 0  -104.833    22.393  -4.682 < 0.001 ***
## meatmeal - casein == 0   -46.674    22.896  -2.039 0.16712
## soybean - casein == 0    -77.155    21.578  -3.576 0.00299 **
## sunflower - casein == 0    5.333    22.393   0.238 0.99945
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

Multiple comparisons options in R: Tukey

Tukey Honest Significant Differences



```
library("multcomp")
# compares all factor levels
tukey <- glht(aov.mod, linfct = mcp(feed = "Tukey"))
summary(tukey)

##
## Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = weight ~ feed, data = chickwts)
##
## Linear Hypotheses:
##
##      Estimate Std. Error t value Pr(>|t|)
## horsebean - casein == 0    -163.383    23.485   -6.957 < 0.001 ***
## linseed - casein == 0     -104.833    22.393   -4.682 < 0.001 ***
## meatmeal - casein == 0     -46.674    22.896   -2.039 0.03193
## soybean - casein == 0      -77.155    21.578   -3.576 0.00843 **
## sunflower - casein == 0      5.333    22.393    0.238 0.99989
## linseed - horsebean == 0     58.550    23.485    2.493 0.14101
## meatmeal - horsebean == 0    116.709    23.966    4.870 < 0.001 ***
## soybean - horsebean == 0     86.229    22.710    3.797 0.00425 **
## sunflower - horsebean == 0   168.717    23.485    7.184 < 0.001 ***
## meatmeal - linseed == 0      58.159    22.896    2.540 0.12748
## soybean - linseed == 0       27.679    21.578    1.283 0.79296
## sunflower - linseed == 0     110.167    22.393    4.920 < 0.001 ***
## soybean - meatmeal == 0     -30.481    22.100   -1.379 0.73880
## sunflower - meatmeal == 0     52.008    22.896    2.271 0.22038
## sunflower - soybean == 0     82.488    21.578    3.823 0.00387 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

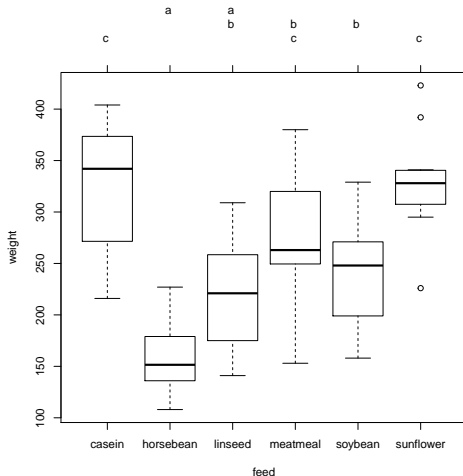
Scanned with CamScanner reported -- single-step method) Lecture Slides (all)

Multiple comparisons options in R: Tukey

Tukey Honest Significant Differences with homogeneous subgroups



```
# summary(tukey)           # standard display
tukey.cld <- cld(tukey)     # letter-based display
# the cld(...) function sets up a compact letter display of all pair-wise comparisons
par(mfrow=c(1,1), mar=c(5, 4, 5, 2))
plot(tukey.cld)
```



Exercise: Anova in R



Introduction: ANOVA as a special case of a linear model

Linear Modelling

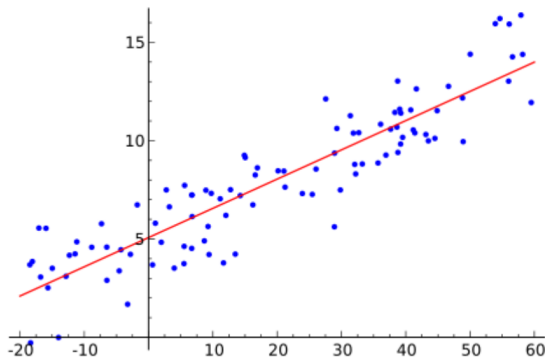
So far we have considered how to determine whether statistically significant differences exist between different "feed" groups (factors).

The explanatory variable has been categorical. We have not yet considered continuous explanatory variables.

ANOVA and linear regression are both a special cases of **linear models**.

Simple linear regression (1/3)

A **simple linear regression** fits a straight line through a set of data points. This straight line is fitted in a way that makes the sum of the squared residuals (the vertical distances between each data point and the fitted line) as small as possible.



Simple linear regression (2/3)

In a simple linear regression model for n data points (x_i, y_i) , $i = 1, \dots, n$ the following equation is used:

$$y_i = \alpha + \beta x_i + \epsilon,$$

where

- α : intercept or constant
- β : slope, regression coefficient (effect size)
- ϵ : error, residuals

Simple linear regression (3/3)

The goal is to find the equation of the straight line

$$f(x) = y = \alpha + \beta x,$$

which would provide a "best" fit for the data points (least square approach).

Assumptions of a linear regression

A **simple linear regression** is based on several assumptions which should be checked carefully.

- linearity of the relationship between the explanatory (independent) and the outcome (dependent) variable
- normality of the residuals
- independence
- constant variance (homoscedasticity)

Linear regression model in R



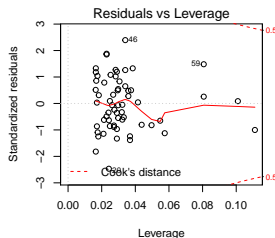
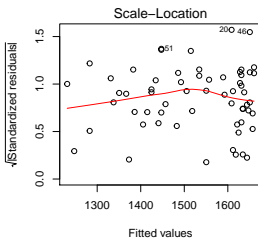
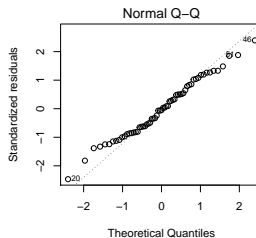
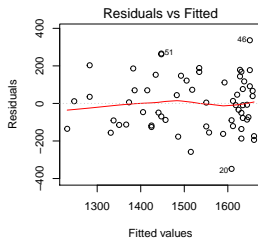
```
data(water)
# mod.hard <- lm(mortality ~ hardness, data = water)
mod.hard <- lm(water$mortality ~ water$hardness)
summary(mod.hard)

##
## Call:
## lm(formula = water$mortality ~ water$hardness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -348.61 -114.52   -7.09   111.52   336.45
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1676.3556    29.2981   57.217  < 2e-16 ***
## water$hardness    -3.2261     0.4847   -6.656 1.03e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 143 on 59 degrees of freedom
## Multiple R-squared:  0.4288, Adjusted R-squared:  0.4191
## F-statistic: 44.3 on 1 and 59 DF, p-value: 1.033e-08
```

Checking linear model assumptions



```
par(mfrow=c(2,2))  
plot(mod.hard)
```



Linear model vs. t.test(...) in R



```
mod.loc <- lm(mortality ~ location, data = water)
coef(mod.loc)

##      (Intercept) locationSouth
##      1633.6000      -256.7923

t.test(water$mortality ~ water$location)

##
## Welch Two Sample t-test
##
## data:  water$mortality by water$location
## t = 7.1427, df = 53.29, p-value = 2.584e-09
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  184.6919 328.8928
## sample estimates:
## mean in group North mean in group South
##      1633.600      1376.808
```

The multiple linear model

An extension of the simple linear model (1/2)

If several explanatory variables are of interest, instead of performing multiple simple regressions, a **multiple linear regression** or **multivariable** approach is appropriate.

- the same assumptions as for simple linear regressions should be checked
- collinearity might be an issue (see `vif(...)` function from package `usdm`)
- model comparison (AIC) (...discussed later)

The multiple linear model

An extension of the simple linear model (2/2)

Simple linear regression model:

$$y_i = \alpha + \beta x_i + \epsilon$$

Multiple linear regression model:

$$y_i = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i} + \dots + \epsilon, \quad (1)$$

$$y_i = \alpha + \beta_1 x_{1i} * \beta_2 x_{2i} + \dots + \epsilon,$$

where

- α : intercept or constant
- β_1 : slope1, regression coefficient (effect size)
- β_2 : slope2, regression coefficient (effect size)
- $\beta_1 * \beta_2$: interaction effect between x_1 and x_2
- ϵ : error, residuals

The multiple linear model

Interpretation of model coefficients

- α : intercept or constant
 - β_1 : slope1, regression coefficient (effect size)
 - β_2 : slope2, regression coefficient (effect size)
 - $\beta_1 * \beta_2$: interaction effect between x_1 and x_2
 - ϵ : error, residuals
-
- $\rightarrow \beta_1$ describes the number of units of a change in the outcome variable y as x_1 changes by one unit, x_2 being held constant.
 - $\rightarrow \beta_2$ describes the number of units of a change in the outcome variable y as x_2 changes by one unit, x_1 being held constant.

The multiple linear model in R



```
mod.hard.loc <- lm(mortality ~ hardness + location, data = water)
summary(mod.hard.loc)

##
## Call:
## lm(formula = mortality ~ hardness + location, data = water)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -222.959  -77.281    7.143   90.751  307.836
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1695.4371    25.3285   66.938 < 2e-16 ***
## hardness      -2.0341     0.4829   -4.212 8.93e-05 ***
## locationSouth -176.7108    36.8913   -4.790 1.19e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 122.1 on 58 degrees of freedom
## Multiple R-squared:  0.5907, Adjusted R-squared:  0.5766
## F-statistic: 41.86 on 2 and 58 DF,  p-value: 5.601e-12
```



- Multiple R^2 : the larger, the better
- Akaike criterion (AIC): the smaller, the better



Two models are nested if one of them is a particular case of the other one: the simpler model can be obtained by setting some coefficients of the more complex model to particular values.

```
mod.hard <- lm(mortality ~ hardness, data = water) # mod 1
mod.loc <- lm(mortality ~ location, data = water) # mod 2
mod.hard.loc <- lm(mortality ~ hardness + location, data = water) # mod 3
```

Among the 3 above models ...

- which ones are nested?
- which ones are not nested?



R^2 is a measure of fit quality:

$$R^2 = \frac{SS_{\text{Regression}}}{SS_{\text{Total}}}$$
$$R^2 = 1 - \frac{SS_{\text{Error}}}{SS_{\text{Total}}}$$

CAREFUL:

- The SS_{Error} always decreases as more predictors are added to the model.
- R^2 always increases and can be artificially large.

Akaike criterion (AIC)

- Model fit (R^2) always improves with model complexity. We would like to strike a good balance between **model fit** and **model simplicity**.
- AIC combines a measure of model fit with a measure of model complexity: The smaller, the better.

For a given data set and a given model:

$$AIC = -2 \cdot \log(L) + 2p$$

L stands for the likelihood. p stands for the number of parameters in the models (penalizes complex models).

Model selection strategy for AIC

- Consider a number of candidate models.
(They need not be nested.)
- Calculate their AIC.
- Choose the model(s) with the smallest AIC.

→ **CAREFUL**: The absolute value of AIC is meaningless. The relative AIC values, between models, is meaningful.

Model selection with AIC in R



```
mod1 <- lm(mortality ~ hardness, data = water)
mod2 <- lm(mortality ~ location, data = water)
mod3 <- lm(mortality ~ hardness + location, data = water)
AIC(mod1, mod2, mod3)
```

```
##      df      AIC
## mod1  3 782.5692
## mod2  3 778.5186
## mod3  4 764.2366
```

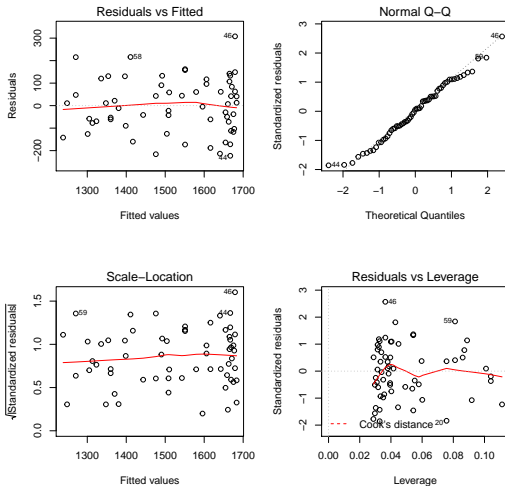
```
round(AIC(mod1, mod2, mod3), 2)
```

```
##      df      AIC
## mod1  3 782.57
## mod2  3 778.52
## mod3  4 764.24
```


mod.hard.loc(mod3) is the best!



```
mod3 <- lm(mortality ~ hardness + location, data = water)
par(mfrow=c(2,2))
plot(mod3)
```



Exercise 16





There are three different interactions:

- **Interaction between two categorical variables**
- Interaction between one continuous and one categorical variables
- Interaction between two continuous variables



Model specification in R: Be aware, an interaction is never tested without its corresponding main effects included in the model.

```
# Interaction between two categorical variables
# mod.dose.supp.int <- lm(len ~ dose.fac + supp + dose.fac:supp,
# data = ToothGrowth)
mod.dose.supp.int <- lm(len ~ dose.fac * supp, data = ToothGrowth)
# summary(mod.dose.supp.int)
```

$$\begin{aligned} y \sim & \beta_{\text{baseline}((\text{dose}==\text{low}) \& (\text{supp}==\text{OJ}))} + \beta_{\text{dose}==\text{med}} + \\ & \beta_{\text{dose}==\text{high}} + \beta_{\text{supp}==\text{VC}} \\ & + \beta_{(\text{dose}==\text{med}) \& (\text{supp}==\text{VC})} \\ & + \beta_{(\text{dose}==\text{high}) \& (\text{supp}==\text{VC})} \end{aligned}$$

Two-way Interactions in R (3/3)



```
# Interaction between two categorical variables
# mod.dose.supp.int <- lm(len ~ dose.fac + supp + dose.fac:supp,
# data = ToothGrowth)
mod.dose.supp.int <- lm(len ~ dose.fac * supp, data = ToothGrowth)
summary(mod.dose.supp.int)

##
## Call:
## lm(formula = len ~ dose.fac * supp, data = ToothGrowth)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -8.20  -2.72  -0.27   2.65   8.27
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      13.230      1.148   11.521 3.60e-16 ***
## dose.faced       9.470       1.624    5.831 3.18e-07 ***
## dose.fachigh     12.830       1.624    7.900 1.43e-10 ***
## suppVC          -5.250       1.624   -3.233 0.00209 **
## dose.faced:suppVC -0.680       2.297   -0.296 0.76831
## dose.fachigh:suppVC 5.330       2.297    2.321 0.02411 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.631 on 54 degrees of freedom
## Multiple R-squared:  0.7937, Adjusted R-squared:  0.7746
## F-statistic: 41.56 on 5 and 54 DF, p-value: < 2.2e-16
```

Two-way Interactions in R: Interpretation of coefficients (1/2)



```
coef(mod.dose.supp.int)
```

```
##          (Intercept)          dose.facmed          dose.fachigh
##             13.23             9.47             12.83
##          suppVC dose.facmed:suppVC dose.fachigh:suppVC
##             -5.25             -0.68             5.33
```

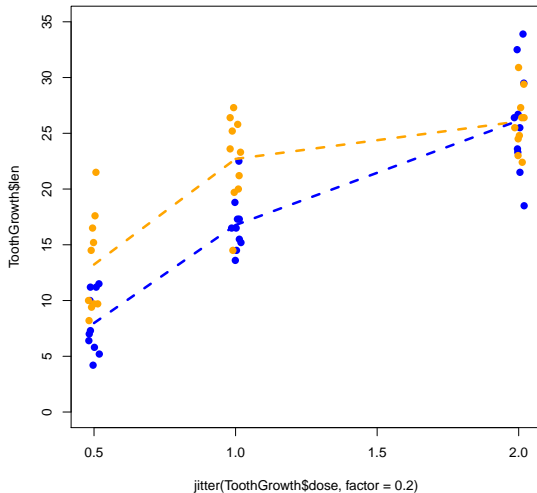
$$\begin{aligned} y \sim & \beta_{\text{baseline}((\text{dose}==\text{low}) \& (\text{supp}==\text{OJ}))} + \beta_{\text{dose}==\text{med}} + \\ & \beta_{\text{dose}==\text{high}} + \beta_{\text{supp}==\text{VC}} \\ & + \beta_{(\text{dose}==\text{med}) \& (\text{supp}==\text{VC})} \\ & + \beta_{(\text{dose}==\text{high}) \& (\text{supp}==\text{VC})} \end{aligned}$$



```
coef(mod.dose.supp.int)
```

```
##          (Intercept)          dose.facmed          dose.fachigh  
##          13.23          9.47          12.83  
##          suppVC dose.facmed:suppVC dose.fachigh:suppVC  
##          -5.25          -0.68          5.33
```

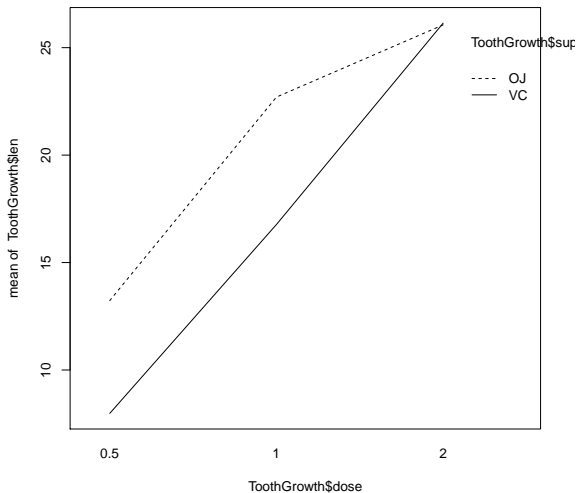
- → `dose.facmed:suppVC`
change in the slope between the low and med dose.fac group under the supplement type (supp) VC in comparison to the intercept. In other words, by changing the dose from low to med within the supplement group VC, the slope **decreases** by approx. -0.68 .
- → `dose.fachigh:suppVC`
change in the slope between the low and high dose.fac group under the supplement type (supp) VC in comparison to the intercept. In other words, by changing the dose from low to high within the supplement group VC, the slope **increases** by approx. $+5.33$.



Interactions



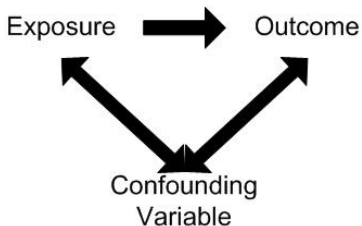
```
interaction.plot(ToothGrowth$dose, ToothGrowth$supp, ToothGrowth$len,  
                fixed = TRUE)
```



Exercise 17



- **Confounding variable** - is associated with the exposure and the outcome
- **Confounding variable** - is not part of the causal path between exposure and the outcome



Some things to remember

- Not adjusting / controlling for a confounding variable may lead to biased results. It is good practice to present as well the crude (not adjusted ORs) and the adjusted ones. Adjustment is typically done if difference $> 10\%$.
- Check also for interaction (= effect modification) e. g. using logistic regression.
- Do not adjust for a variable C if it is a common effect of E and D (collider) or if it is in the causal pathway of E and D.

Exercise 18



Other topics

So far, we have covered some of the basic **tools** necessary for helping you to start to analyse your own study data.

Many things have been mentioned others have not been included - some are worth mentioning as they are particularly common in some areas of veterinary research.

Normally distributed response variables

So far, we have only considered analyses where the response variable is a **continuous** measurement.

We now have a brief look at two other common types of **response** variables.

The methods we use are largely similar to what we have looked at previously.



- numeric & integers

```
ToothGrowth$len[1:6]
## [1]  4.2 11.5  7.3  5.8  6.4 10.0

bacteria$week[1:6]
## [1]  0  2  4 11  0  2
```

- unordered factor with 2 levels

```
lung$sex[1:6]
## [1] female male  female female male  female
## Levels: female male
```

- (un/ordered) factor (with more than 2 levels)

```
chickwts$feed[1:6]
## [1] horsebean horsebean horsebean horsebean horsebean horsebean
## Levels: casein horsebean linseed meatmeal soybean sunflower
```


Generalised Linear Modelling

- random component: $\mathbf{E}(\mathbf{Y}) = \mu$
- systematic component: covariates: $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_p$ produce a linear predictor $\eta = \sum_1^p \mathbf{x}_j \beta_j$
- link between random and systematic components: $\mu = \eta$, e.g.
 - **Logistic Regression:**
 $\eta_i = g(\mu_i)$, $g(\cdot)$ is logit function for binomial data
 - **Poisson Regression:**
 $\eta_i = g(\mu_i)$, $g(\cdot)$ is exponential function for poisson data



E. g. a logistic regression with one covariate (`sex`) fitting a model to $\mathbb{P}(Y = \text{TRUE}) = \pi$, e. g. presence of respiratory symptoms (experienced by the child over the previous 12 months) , gives

$$\log \left(\frac{\pi}{1 - \pi} \right) = \beta_0 + \beta_1 \mathbf{x}_1$$

```
logreg.mod <- glm(respsymptoms ~ sex,  
                  family = binomial, # description of the link function  
                  data = lung)  
summary(logreg.mod)
```

Logistic Regression: unordered factor with 2 levels as response



```
logreg.mod <- glm(respsymptoms ~ sex,  
                  family = binomial, # description of the link function  
                  data = lung)  
summary(logreg.mod)  
  
##  
## Call:  
## glm(formula = respsymptoms ~ sex, family = binomial, data = lung)  
##  
## Deviance Residuals:  
##      Min       1Q   Median       3Q      Max   
## -0.7440  -0.7440  -0.6915  -0.6915   1.7597   
##  
## Coefficients:  
##              Estimate Std. Error z value Pr(>|z|)      
## (Intercept)  -1.1429     0.1276  -8.957  <2e-16 ***  
## sexmale      -0.1663     0.1901  -0.875    0.382      
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## (Dispersion parameter for binomial family taken to be 1)  
##  
##      Null deviance: 682.85  on 635  degrees of freedom  
## Residual deviance: 682.08  on 634  degrees of freedom  
## AIC: 686.08  
##  
## Number of Fisher Scoring iterations: 4
```

Logistic Regression: unordered factor with 2 levels as response

Interpretation of model coefficients



```
summary(logreg.mod)

##
## Call:
## glm(formula = respsymptoms ~ sex, family = binomial, data = lung)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -0.7440  -0.7440  -0.6915  -0.6915   1.7597
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.1429     0.1276  -8.957  <2e-16 ***
## sexmale      -0.1663     0.1901  -0.875   0.382
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 682.85  on 635  degrees of freedom
## Residual deviance: 682.08  on 634  degrees of freedom
## AIC: 686.08
##
## Number of Fisher Scoring iterations: 4

cbind(logodds = coef(logreg.mod), odds = exp(coef(logreg.mod)))

##              logodds      odds
## (Intercept) -1.1428851 0.3188976
## sexmale     -0.1662919 0.8467990
```

Odds ratios versus log odds ratios

- The logistic regression model in R outputs the log odds ratios.
- $\exp(\log \text{ odds ratio}) = \text{odds ratio}$
- An **odds ratio (OR)** is a measure of association between an exposure (here: `sex`) and an outcome (here: `respsymptoms`). The OR represents the odds that an outcome will occur given a particular exposure, compared to the odds of the outcome occurring in the absence of that exposure.

		Outcome	
		Yes	No
Predictor	Yes	A	B
	No	C	D

$$OR = \frac{(A * D)}{(B * C)}$$

Logistic Regression: unordered factor with 2 levels as response

Interpretation of model coefficients (cont'd)



```
cbind(logodds = coef(logreg.mod), odds = exp(coef(logreg.mod)))
```

```
##           logodds      odds
## (Intercept) -1.1428851 0.3188976
## sexmale     -0.1662919 0.8467990
```

The coefficient β_{sex} is the estimated amount by which the log odds of respsymptoms would in-/decrease if sex is male. The log odds of respsymptoms when sex is female is just above in the first row ($\beta_{(\text{Intercept})}$, reference level of sex).

Logistic Regression: probability of success as response



- A researcher is examining beetle mortality after 5 hours of exposure to carbon disulphide, at various levels of concentration of the gas.
- Beetles were exposed to gaseous carbon disulphide at various concentrations (in mg/L) for five hours (Bliss, 1935) and the number of beetles killed were noted.

```
# install.packages("investr")
library("investr")
data(beetle)
colnames(beetle) <- c("Dose", "Num.Beetles", "Num.Killed")
str(beetle)

## 'data.frame': 8 obs. of 3 variables:
## $ Dose : num  1.69 1.72 1.76 1.78 1.81 ...
## $ Num.Beetles: num  59 60 62 56 63 59 62 60
## $ Num.Killed : num   6 13 18 28 52 53 61 60

beetle$Num.Surv <- beetle$Num.Beetles - beetle$Num.Killed
```

Logistic Regression: probability of success as response



```
logreg.mod <- glm(cbind(Num.Killed, Num.Surv) ~ Dose,
                  family = binomial, # description of the link function
                  data = beetle)
summary(logreg.mod)

##
## Call:
## glm(formula = cbind(Num.Killed, Num.Surv) ~ Dose, family = binomial,
##      data = beetle)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.5941  -0.3944   0.8329   1.2592   1.5940
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -60.717      5.181  -11.72  <2e-16 ***
## Dose           34.270      2.912   11.77  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 284.202  on 7  degrees of freedom
## Residual deviance:  11.232  on 6  degrees of freedom
## AIC: 41.43
##
## Number of Fisher Scoring iterations: 4
```


Logistic Regression: probability of success as response

Interpretation of model coefficients



```
summary(logreg.mod)

##
## Call:
## glm(formula = cbind(Num.Killed, Num.Surv) ~ Dose, family = binomial,
##      data = beetle)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.5941  -0.3944   0.8329   1.2592   1.5940
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -60.717      5.181  -11.72  <2e-16 ***
## Dose           34.270      2.912   11.77  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 284.202  on 7  degrees of freedom
## Residual deviance:  11.232  on 6  degrees of freedom
## AIC: 41.43
##
## Number of Fisher Scoring iterations: 4

cbind(logodds = coef(logreg.mod), odds = exp(coef(logreg.mod)))

##              logodds          odds
## (Intercept) -60.71745 4.273114e-27
## Dose         34.27033 7.645631e+14
```

Logistic Regression: probability of success as response

Interpretation of model coefficients (cont'd)



```
cbind(logodds = coef(logreg.mod), odds = exp(coef(logreg.mod)))
```

```
##           logodds           odds
## (Intercept) -60.71745 4.273114e-27
## Dose         34.27033 7.645631e+14
```

The coefficient β_{Dose} is the estimated amount by which the log odds of being killed would in-/decrease if Dose would increase by one unit. The log odds of being killed when Dose is 0 is just above in the first row ($\beta_{(\text{Intercept})}$).

Some things to remember

- Using generalised linear models (glm) is very similar to the previous models we have seen for normal data (lm) - things like residuals need to be checked for randomness, although this is more difficult with counts. Also the residuals being normally distributed is not relevant to these models.
- **As always explore the data first visually and with tables before doing any formal analyses.**

Short excursion to quantile regression

- When talking about regression, we almost exclusively model the conditional mean of a response given one or more explanatory variables.
- With the classical linear model we cannot skewed or otherwise non normal distribution as the corresponding quantiles from the linear model will be misleading. Therefore, we shift our attention to a completely distribution free approach that directly addresses conditional quantile modelling.
(Text taken from Hothorn, Torsten, and Brian S. Everitt. A handbook of statistical analyses using R. CRC press, 2014.)

Short excursion to quantile regression

The data set `sambia.csv` (given on the switch drive) contains information on malnutrition of sambian children. The outcome variable is the *z-score*, which gives information on the severity of malnutrition.

The following covariates are considered:

<code>zscore</code>	Z-score of the child
<code>sex</code>	Sex of the child (0 = female, 1 = male)
<code>age.child</code>	Age of the child
<code>work</code>	Is the mother working (0 = no, 1 = yes)
<code>age.mother.birth</code>	Age of the mother at birth of child
<code>bmi</code>	BMI of the mother

A *z-score* indicates how many standard deviations an element is from the mean. A *z-score* can be calculated from the following formula. $z = (X - \mu) / \sigma$ where z is the *z-score*, X is the value of the element, μ is the population mean, and σ is the standard deviation.

```
sambia <- read.csv("sambia.csv", sep = ",")
str(sambia)
sambia$sex <- factor(sambia$sex, levels = c(0,1),
                    labels = c("female", "male"))
sambia$work <- factor(sambia$work, levels = c(0,1),
                    labels = c("no", "yes"))
```

```
str(sambia)
```

```
## 'data.frame': 4421 obs. of 7 variables:
## $ X : int 1 2 3 4 5 6 7 8 9 10 ...
## $ zscore : int -159 -205 -192 -146 69 -80 -34 -17 -225 468 ...
## $ sex : Factor w/ 2 levels "female","male": 1 1 2 2 1 1 1 2 1 2 ...
## $ age.child : int 4 26 56 6 54 1 2 2 29 14 ...
## $ work : Factor w/ 2 levels "no","yes": 1 1 2 1 1 1 1 2 2 2 ...
## $ age.mother.birth: num 24.7 22.8 15.3 21.5 17.5 ...
## $ bmi : num 21.8 21.8 20.4 22.3 22.3 ...
```

Quantile regression in R (cont'd)



```
library("quantreg")

## Loading required package: SparseM
##
## Attaching package: 'SparseM'
## The following object is masked from 'package:base':
##
##      backsolve
##
## Attaching package: 'quantreg'
## The following object is masked from 'package:survival':
##
##      untangle.specials

# tau = 2.5 %
rq.025 <- rq(zscore ~ sex + age.child + work + bmi + age.mother.birth,
             tau = 0.025,
             data = sambia)

# tau = 5 %
rq.05 <- rq(zscore ~ sex + age.child + work + bmi + age.mother.birth,
            tau = 0.05,
            data = sambia)

# tau = 50 %
rq.5 <- rq(zscore ~ sex + age.child + work + bmi + age.mother.birth,
           tau = 0.5,
           data = sambia)
```

Quantile regression in R (cont'd)

Comparison of the three models rq.025, rq.05, rq.5



```
rq_coef <- cbind(coef(rq.025), coef(rq.05), coef(rq.5))
rownames(rq_coef) <- rownames(summary(rq.5)$coefficients)
colnames(rq_coef) <- paste0("tau = ", c("0.025", "0.05", "0.5"))
round(rq_coef, 3)
```

##	tau = 0.025	tau = 0.05	tau = 0.5
## (Intercept)	-523.808	-491.812	-242.188
## sexmale	1.511	-3.816	-13.197
## age.child	-2.235	-2.496	-1.903
## workyes	-13.947	-16.280	-6.489
## bmi	6.947	7.884	4.966
## age.mother.birth	0.244	0.465	0.830

Quantile regression in R (cont'd)

Estimate `rq` for a grid of quantiles



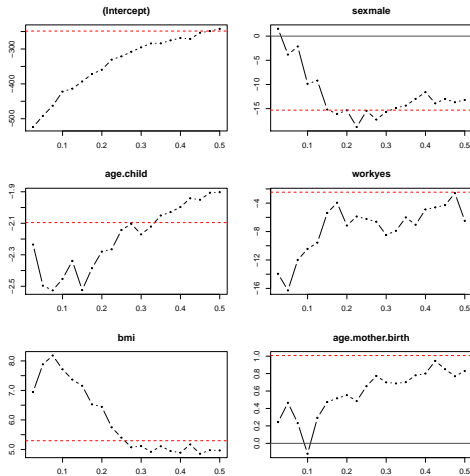
```
# Estimate rq for a single tau
# rq.5 <- rq(zscore ~ sex + age.child + work + bmi + age.mother.birth,
#           tau = 0.5,
#           data = sambia)
# Estimate rq for a grid of tau
rq <- rq(zscore ~ sex + age.child + work + bmi + age.mother.birth,
         tau = seq(from = 0.025, to = 0.5, by = 0.025),
         data = sambia)
```

Quantile regression in R (cont'd)

Plot results



```
plot(rq) # plot including line for the OLS coefficient (as estimated by lm)
```



Quantile regression in R (cont'd)

Plot results in more details



```
# A sequence of coefficient estimates for quantile regressions with  
# varying tau parameters is visualized along with associated confidence bands.  
plot(summary(rq))
```

