



Swiss Institute of
Bioinformatics

First Steps with R in Life Sciences: Statistics

With slides from: Diana Marek, Geoffrey Fucile, Alex
Smith, Linda Dib, Leonore Wigger, Wandrille Duchemin

Statistical hypothesis testing

Two hypotheses in competition:

- H_0 : the **NULL hypothesis** (usually the most conservative – e.g., “no difference”)
- H_1 : the **alternative hypothesis** (usually the one we are actually interested in)

Example:

H_0 : « There is no difference in weight between two given strains of mice »

H_1 : « The average weight in KO mice is different from that in WT mice »

Statistical test:

- Calculate test statistic,
- Calculate associated p-value,
- Check if p-value is small enough to reject H_0 , according to pre-defined significance level.

t-test

Goal:

- Compare a continuous measure between two groups
- Is the difference between the two group means statistically significant?

Assumptions:

- Observations are independent
- The two groups follow a normal distribution
- ~~(Same variance in each group)~~

R uses Welch's t-test, which does not assume equal variance

Example data set: iris

Let's compare *Iris virginica* and *Iris versicolor* petal lengths.

```
data(iris)
# we limit the data to 2 species
iris_f = iris[ iris$Species %in% c('versicolor','virginica') , ]
iris_f$Species = factor(iris_f$Species)

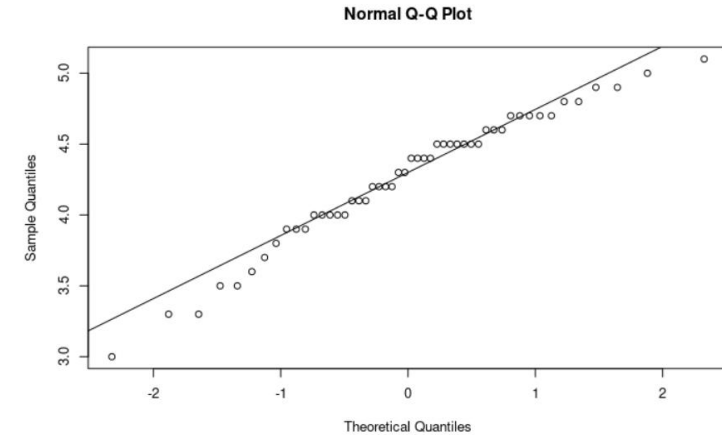
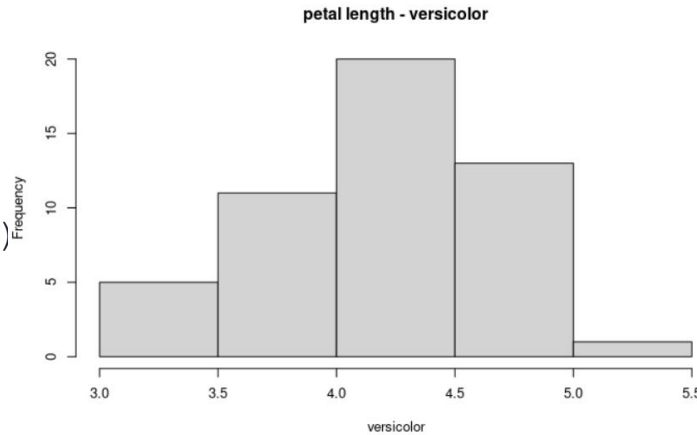
tapply( iris_f$Petal.Length, iris_f$Species, mean )
versicolor  virginica
      4.260      5.552

plen_versicolor = iris_f$Petal.Length[iris_f$Species=='versicolor']
plen_virginica   = iris_f$Petal.Length[iris_f$Species=='virginica']
```

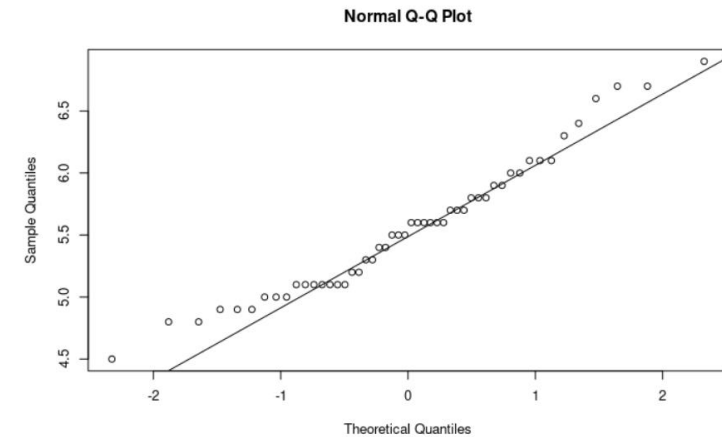
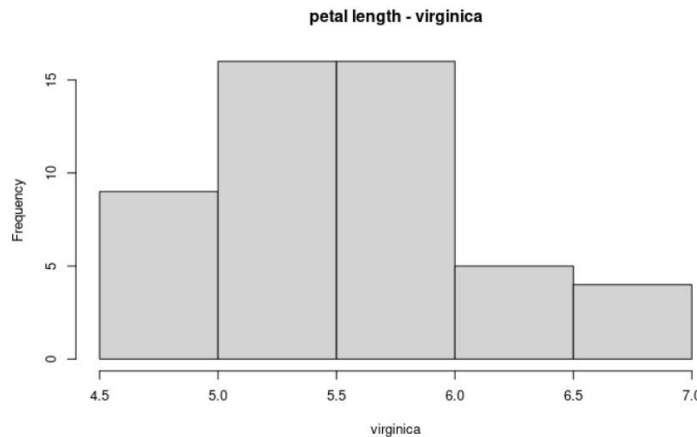
Check normality of the data with plots

We use histograms and QQplot to visually assess normality

```
par(mfrow=c(2,2))
hist(plen_versicolor,
     xlab="versicolor",
     main="petal length - versicolor")
qqnorm(plen_versicolor)
qqline(plen_versicolor)
```



```
hist(plen_virginica,
     xlab="virginica",
     main="petal length - virginica")
qqnorm(plen_virginica)
qqline(plen_virginica)
dev.off()
```

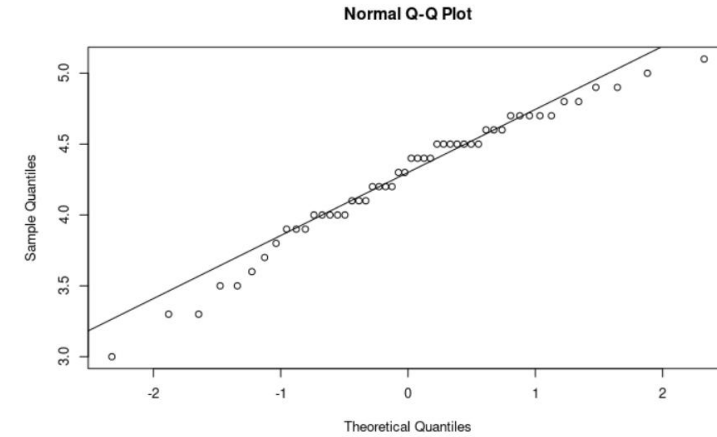
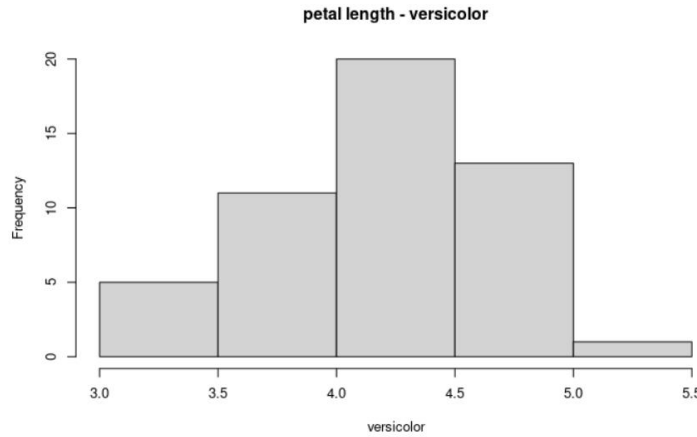


Check normality of the data with tests

We use the Shapiro-Wilks test, it is not ideal as the null hypothesis is normality (what we would like).
So, accepting normality has no type II error control...

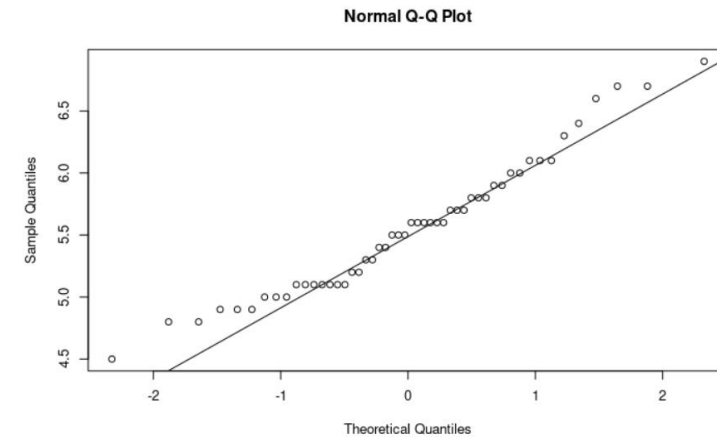
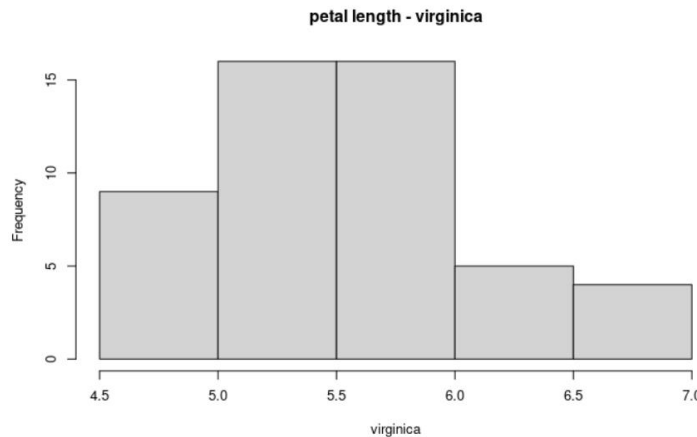
```
shapiro.test(plen_versicolor)  
Shapiro-Wilk normality test
```

```
data:  plen_versicolor  
W = 0.966, p-value = 0.1585
```



```
shapiro.test(plen_virginica)  
Shapiro-Wilk normality test
```

```
data:  plen_virginica  
W = 0.96219, p-value = 0.1098
```



Check normality of the data with tests

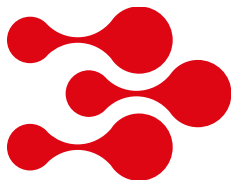
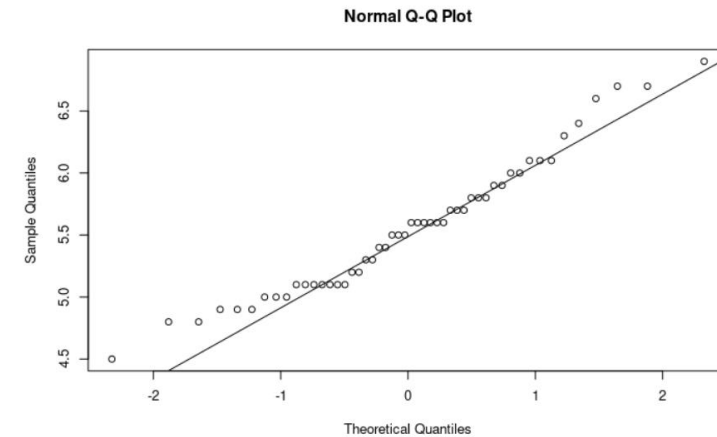
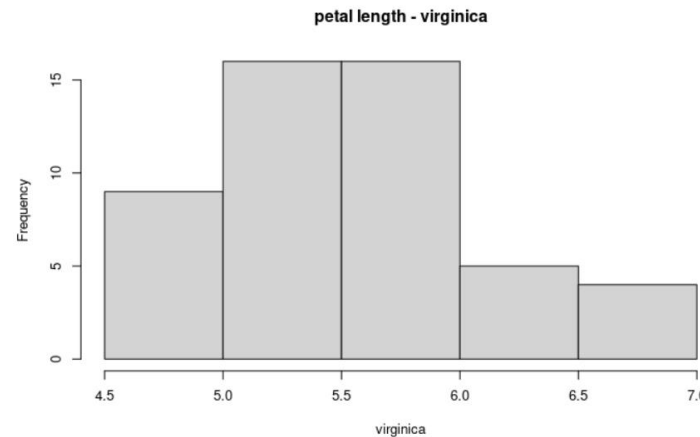
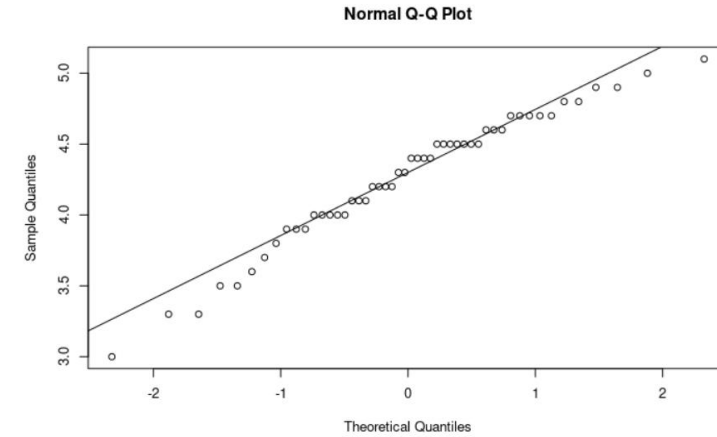
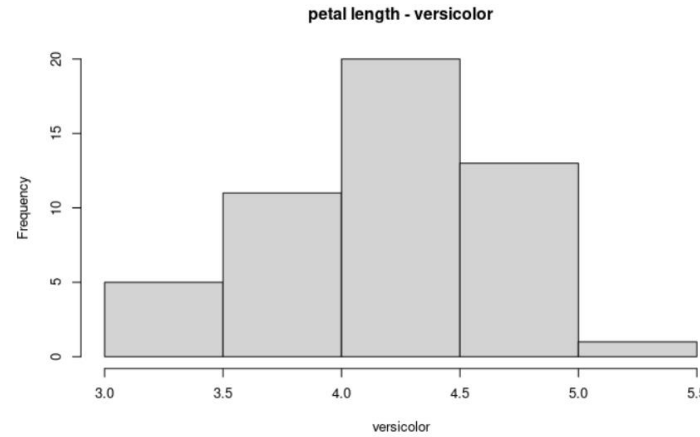
We use the Shapiro-Wilks test, it is not ideal as the null hypothesis is normality (what we would like).
So, accepting normality has no type II error control...

```
shapiro.test(plen_versicolor)  
Shapiro-Wilk normality test
```

```
data:  plen_versicolor  
W = 0.966, p-value = 0.1585
```

```
shapiro.test(plen_virginica)  
Shapiro-Wilk normality test
```

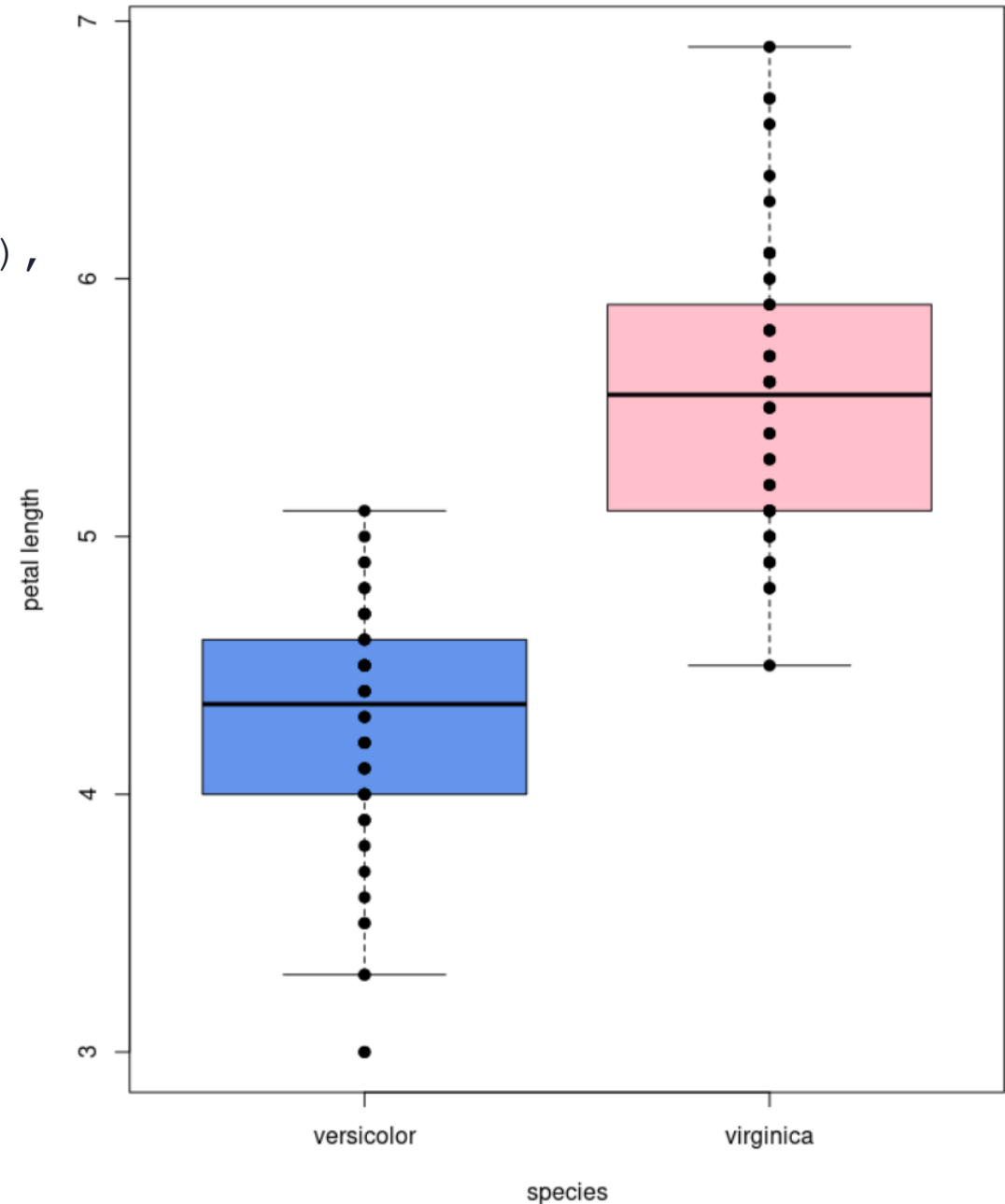
```
data:  plen_virginica  
W = 0.96219, p-value = 0.1098
```



t-test is somewhat robust to non-normal data. No need to be too strict about normality requirement.

Visualize group differences

```
boxplot(Petal.Length ~ Species,  
        data=iris_f,  
        col=c("cornflowerblue", "pink"),  
        ylab="petal length",  
        xlab="species")  
  
points(Petal.Length ~ Species,  
       data=iris_f,  
       col="black",  
       pch = 19)
```



Function t.test()

```
t.test(plen_versicolor, plen_virginica)
```

```
t.test(Petal.Length ~ Species, data=iris_f) #equivalent to the above
```

Welch Two Sample t-test

```
data:  Petal.Length by Species
```

```
t = -12.604, df = 95.57, p-value < 2.2e-16
```

```
alternative hypothesis: true difference in means between group versicolor and group  
virginica is not equal to 0
```

```
95 percent confidence interval:
```

```
-1.49549 -1.08851
```

```
sample estimates:
```

mean in group versicolor	mean in group virginica
4.260	5.552

t.test object

- `t.test()` and other tests return a **list** that can be assigned to a variable.
- View the names of the list's slots using `names()`.
- Access the elements of a list using the `$` or the `[[]]` operators.

```
test_res = t.test(Petal.Length ~ Species, data=iris_f)
```

```
names(test_res)
```

```
[1] "statistic"    "parameter"    "p.value"      "conf.int"     "estimate"  
[6] "null.value"   "stderr"       "alternative"   "method"       "data.name"
```

```
test_res[['p.value']]
```

```
[1] 4.900288e-22
```

Paired data

When the measurements correspond to 2 observations on the same individuals, we need to use the **paired t-test**.

This tests focuses on the difference between the two observations and asks if the mean of this difference is different from 0.

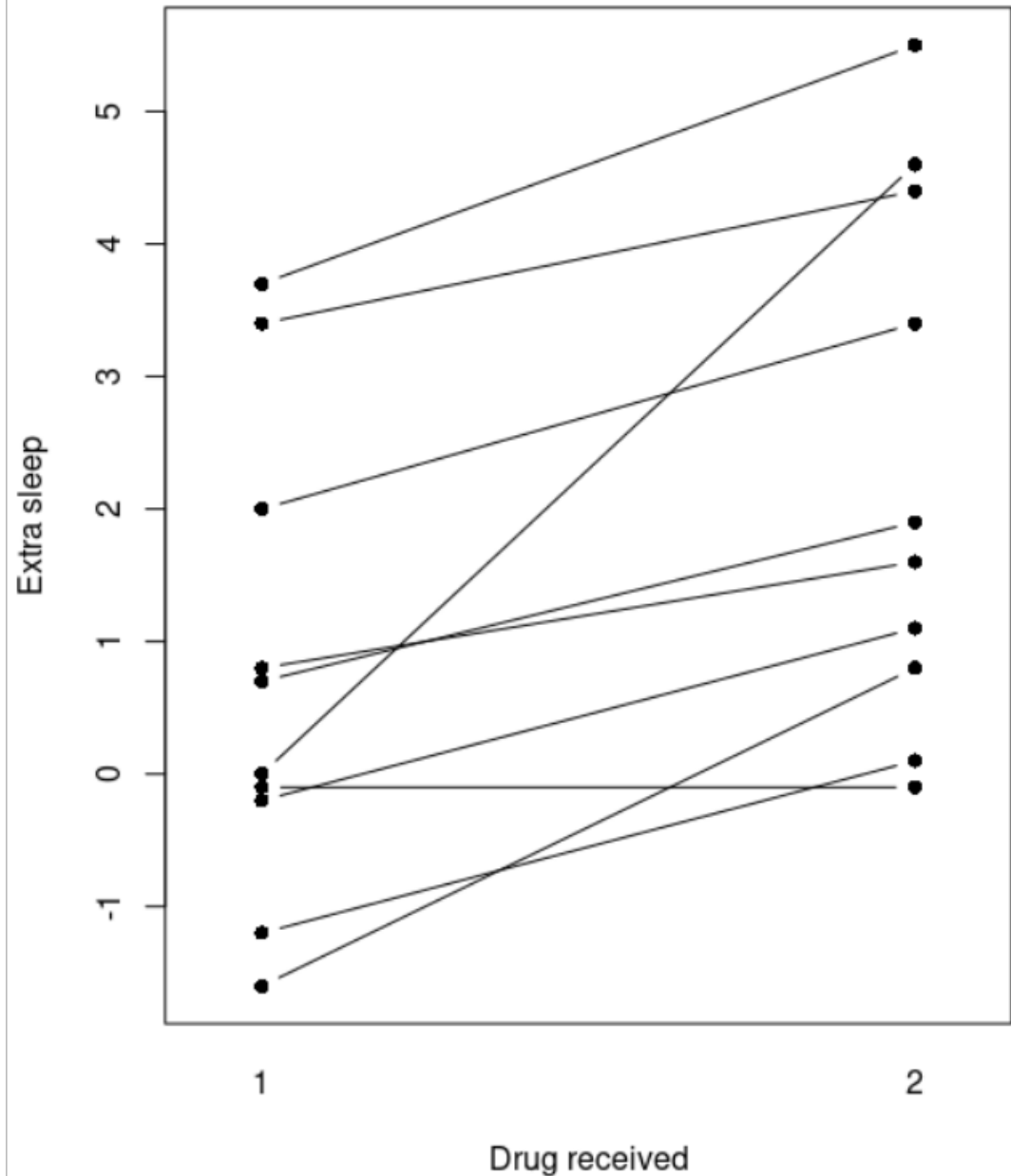
Assumption: the sample mean of the differences are normally distributed

Paired data representation

```
data(sleep)
head(sleep, n=3)
```

extra	group	ID
1	0.7	1 1
2	-1.6	1 2
3	-0.2	1 3

```
interaction.plot(response=sleep$extra,
                 x.factor=sleep$group,
                 trace.factor=sleep$ID,
                 legend=FALSE, type="b",
                 lty=1, pch=16,
                 xlab="Drug received",
                 ylab="Extra sleep")
```

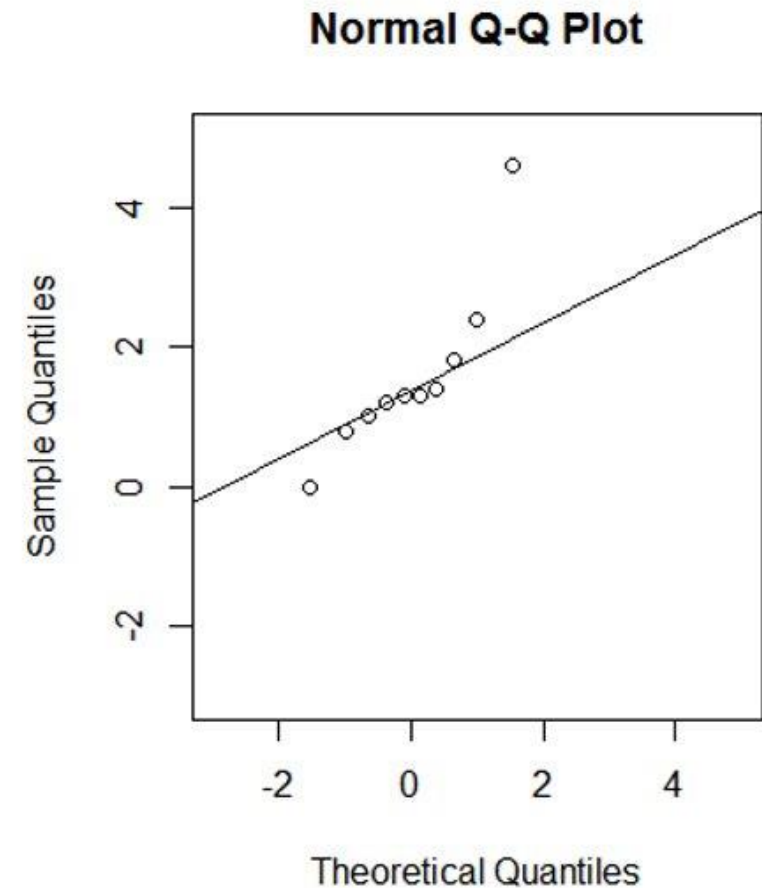
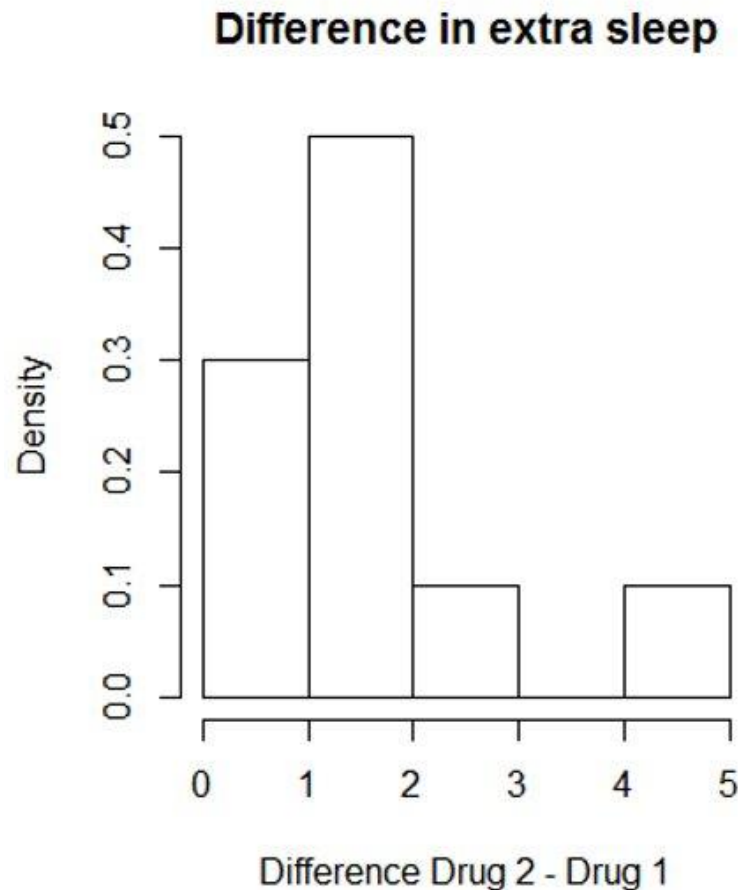


Check normality of the differences between pairs

```
difference = sleep$extra[sleep$group==2]-sleep$extra[sleep$group==1]
```

```
hist(difference, freq=FALSE, xlab="Difference Drug 2 - Drug 1",  
     main="Difference in extra sleep", col="white")
```

```
qqnorm(difference)  
qqline(difference)
```



Paired t-test

```
# using a paired t-test:
```

```
t.test(sleep$extra[sleep$group==1],  
       sleep$extra[sleep$group==2], paired=TRUE)
```

```
# paired values must be at the same position in the two vectors
```

```
# do not use formula notation (extra~sleep) for paired t-test
```

Paired t-test

```
data:  sleep$extra[sleep$group == 1] and sleep$extra[sleep$group == 2]
```

```
t = -4.0621, df = 9, p-value = 0.002833
```

```
alternative hypothesis: true difference in means is not equal to 0
```

```
95 percent confidence interval:
```

```
-2.4598858 -0.7001142
```

```
sample estimates:
```

```
mean of the differences
```

```
-1.58
```

Non-parametric alternatives to the t-test

When the data deviates strongly from normality, a **non-parametric test** can be used in place of a t-test.

Non-parametric tests **do not assume any particular distribution of the data.**

Instead of t-test (without pairing), use Mann-Whitney U test.

```
wilcox.test(sleep$extra[sleep$group==1],  
            sleep$extra[sleep$group==2])  
wilcox.test(extra~group, data=sleep) # equivalent
```

Instead of paired t-test, use Wilcoxon Signed Rank test.

```
wilcox.test(sleep$extra[sleep$group==1],  
            sleep$extra[sleep$group==2], paired=TRUE)
```

These two tests have different names but are **both implemented in the R function wilcox.test.**

wilcox.test()

For the sleep data, a paired test is appropriate.

```
wilcox.test(sleep$extra[sleep$group==1],  
            sleep$extra[sleep$group==2], paired=TRUE)
```

Wilcoxon signed rank test with continuity correction

data: sleep\$extra[sleep\$group==1] and sleep\$extra[sleep\$group==2]

V = 0, p-value = **0.009091**

alternative hypothesis: true location shift is not equal to 0

The conclusion is the same as it was for the paired t-test.

- The p-value is a little higher wilcox.test: 0.009091 (t.test: 0.002833)

wilcox.test() - warning messages

wilcox.test() implements two ways to compute p-values: exact and by approximation

- The method can be selected with parameter **exact=TRUE** or **exact=FALSE**
- The default is "exact" if sample size < 50 *and* there are no ties in the data.
- Otherwise, it is by normal approximation.

Warning messages:

```
1: In wilcox.test.default(sleep$extra[sleep$group == 1], sleep$extra[sleep$group == 2] :  
   cannot compute exact p-value with ties  
2: In wilcox.test.default(sleep$extra[sleep$group == 1], sleep$extra[sleep$group == 2] :  
   cannot compute exact p-value with zeroes
```

These **warnings don't mean that there is an error in the result**. An (approximated) p-value is still provided and can be reported.

Let's practice - 10

Come back to the mice data-set stored in the "mice_data" data frame.

- 1) Considering WT mice weight and KO mice weight separately, check the assumption of normality graphically.
- 2) Make an appropriate plot to visualize the mouse weights grouped by genotype.
- 3) Perform a test to see whether the mouse weight is different between the two genotypes.
- 4) *Repeat steps 1 to 3 for the diet variable.*

Bivariate linear correlation

Goal: Quantify the strength of a linear correlation between two continuous variables

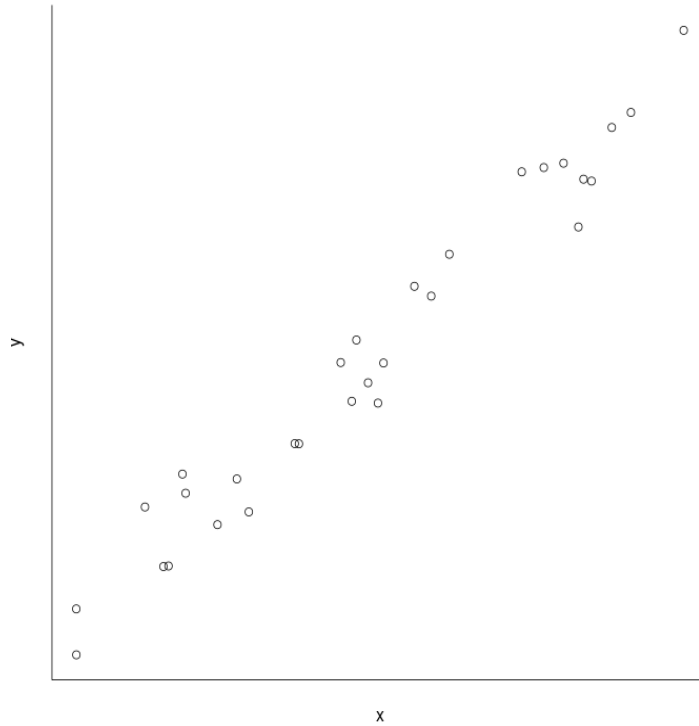
`cor()` computes a correlation between two variables.

- **Default: `method="pearson"`** (linear correlation)
- **Other options: `method="spearman"`, `method="kendall"`** (rank-based correlations)

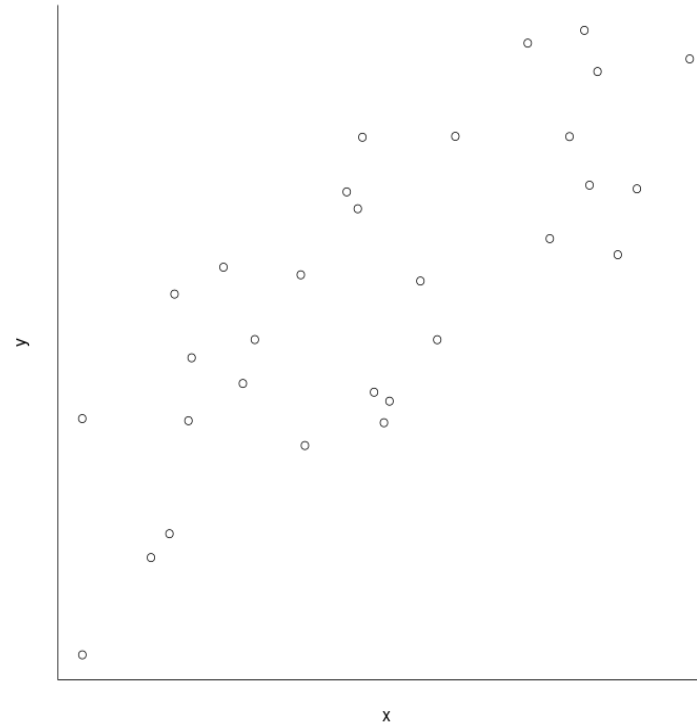
`cor.test()`

- computes a correlation and performs a corresponding statistical test
- for Pearson correlation: p-value from linear regression, same as `lm()`

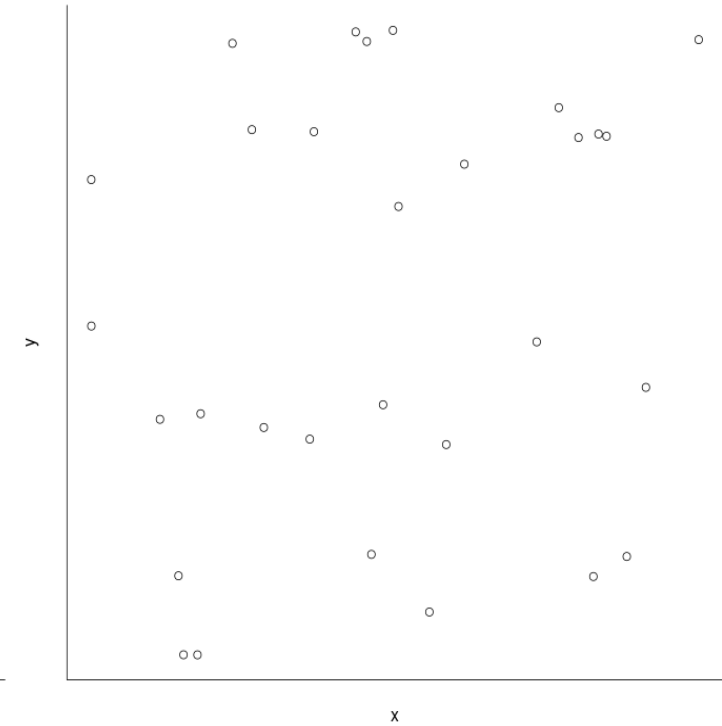
Visualization of linear correlation



Strong linear correlation

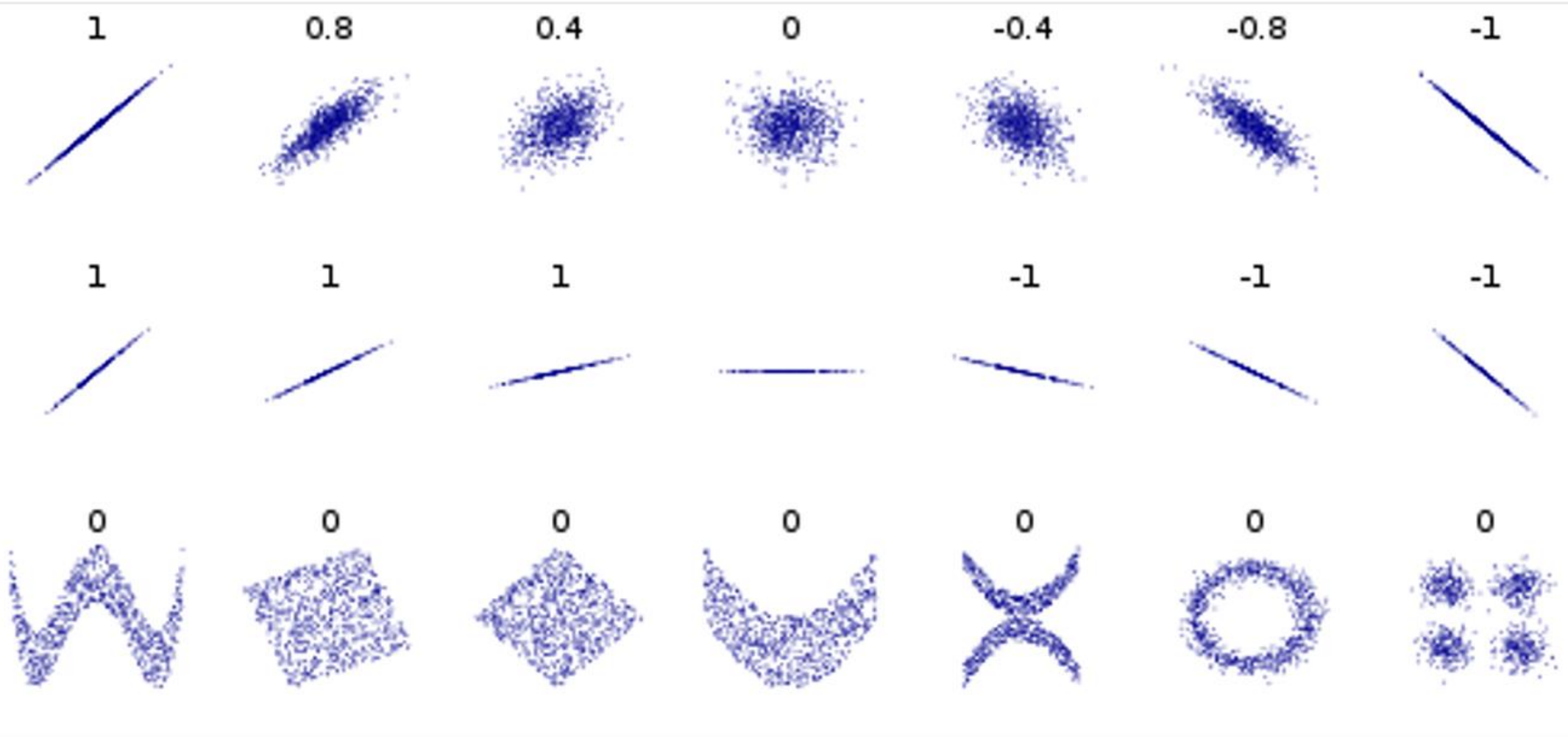


Medium-strong linear correlation



No correlation

More visualization of linear correlation

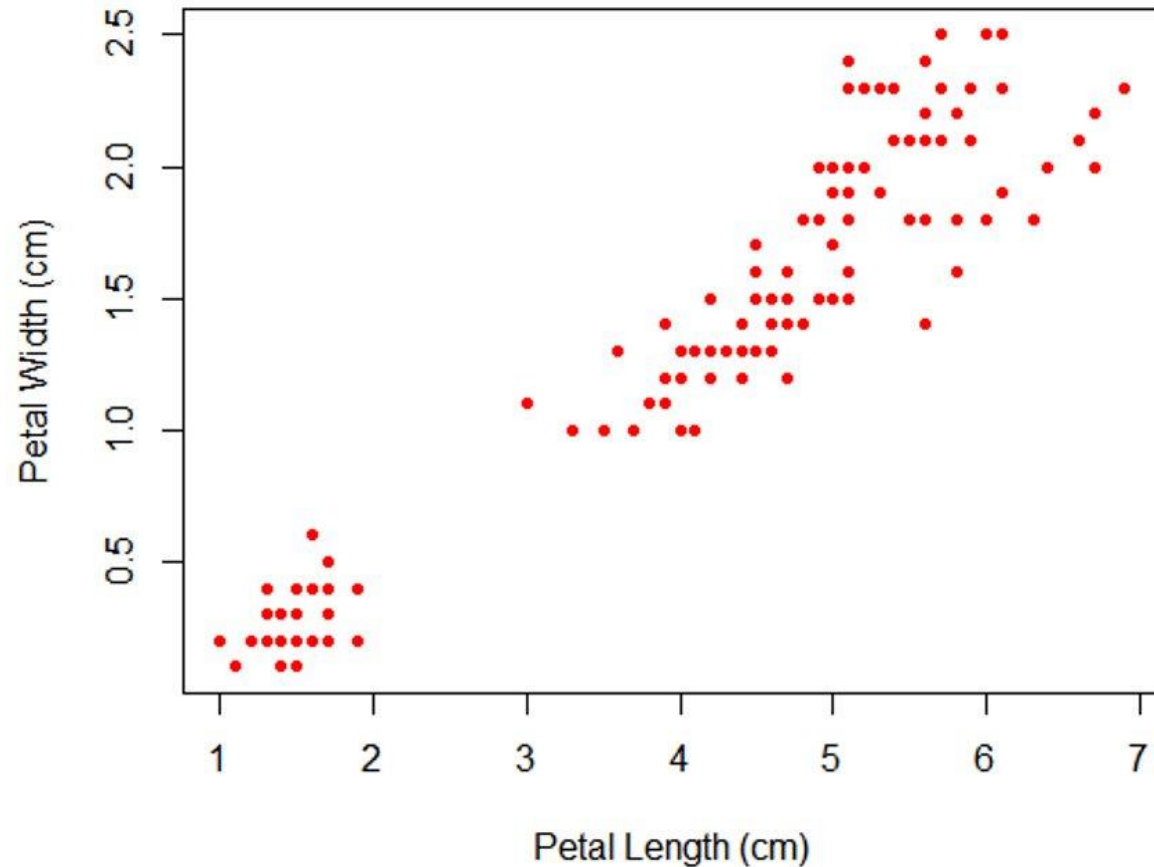


correlation reflects the noisiness and direction of a linear relationship (top row)
but not the slope of that relationship (middle)
nor many aspects of nonlinear relationships (bottom).

Image credit:
wikipedia user DenisBoigelot,
under the CC0 1.0 license

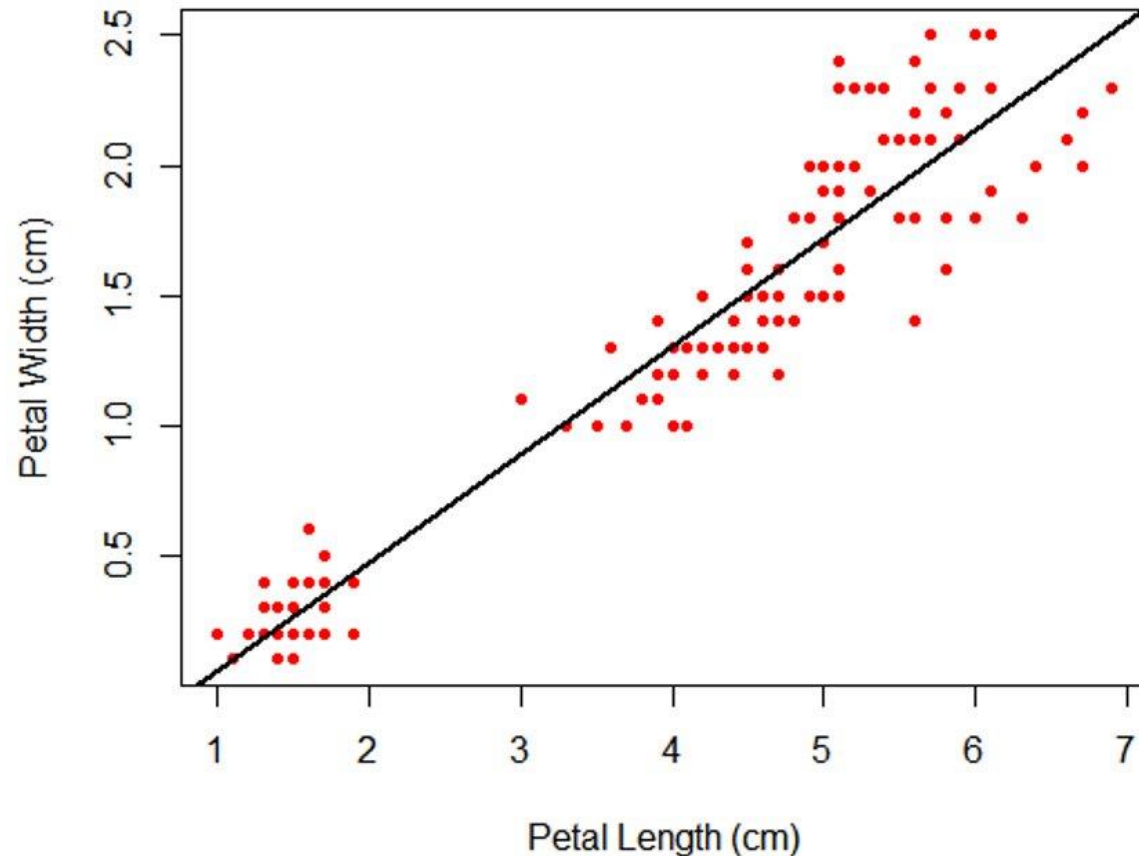
Scatter plot

```
plot(iris$Petal.Length, iris$Petal.Width,  
     col="red", pch=20,  
     xlab="Petal Length", ylab="Petal Width")
```



Scatter plot

```
plot(iris$Petal.Length, iris$Petal.Width,  
     col="red", pch=20,  
     xlab="Petal Length", ylab="Petal Width")  
abline(lm(iris$Petal.Width~iris$Petal.Length),  
       col="black", lwd=2)
```



Linear correlation

```
cor(iris$Petal.Length, iris$Petal.Width, method="pearson")  
[1] 0.9628654
```

```
cor.test(iris$Petal.Length, iris$Petal.Width, method="pearson")
```

Pearson's product-moment correlation

data: iris\$Petal.Length and iris\$Petal.Width

t = **43.387**, df = 148, p-value < **2.2e-16**

alternative hypothesis: true correlation is not equal to 0

95 percent confidence interval:

0.9490525 0.9729853

sample estimates:

cor

0.9628654

Linear regression

Goal: Determine the extent to which there is a linear relationship between an "outcome" variable (dependent variable) and one or more "explanatory" variables (independent variables, predictor variables).

Can a significant part of the variability in the outcome be predicted/explained by the independent variables?

Outcome variable: continuous (e.g. weight, heart rate, blood sugar)

Explanatory variables: continuous or (with adaptations) categorical

In R, the linear regression model is specified by a model formula of the form:

outcome ~ explanatory variables

Simple linear regression

A simple regression model (one explanatory variable) is specified by

$$y = a + b \cdot x + \text{err}$$

a: Intercept

b: coefficient of explanatory var.

x: explanatory var.

err: error term (=residuals)

Assumptions :

- **Homoscedasticity** : independence between residual variance and variables
- Linearity + absence of linear relationship between predictor variables
- independence of the observations.
- **Residuals centered around predicted value** (mean=0)
- + normality of the **residual's mean**
→ only used to assess parameters confidence interval

Otherwise : try log-transform (for heteroskedasticity) or non-parametric methods if the assumptions are not met.

Summary of the data

CLASS dataset, from the program SAS

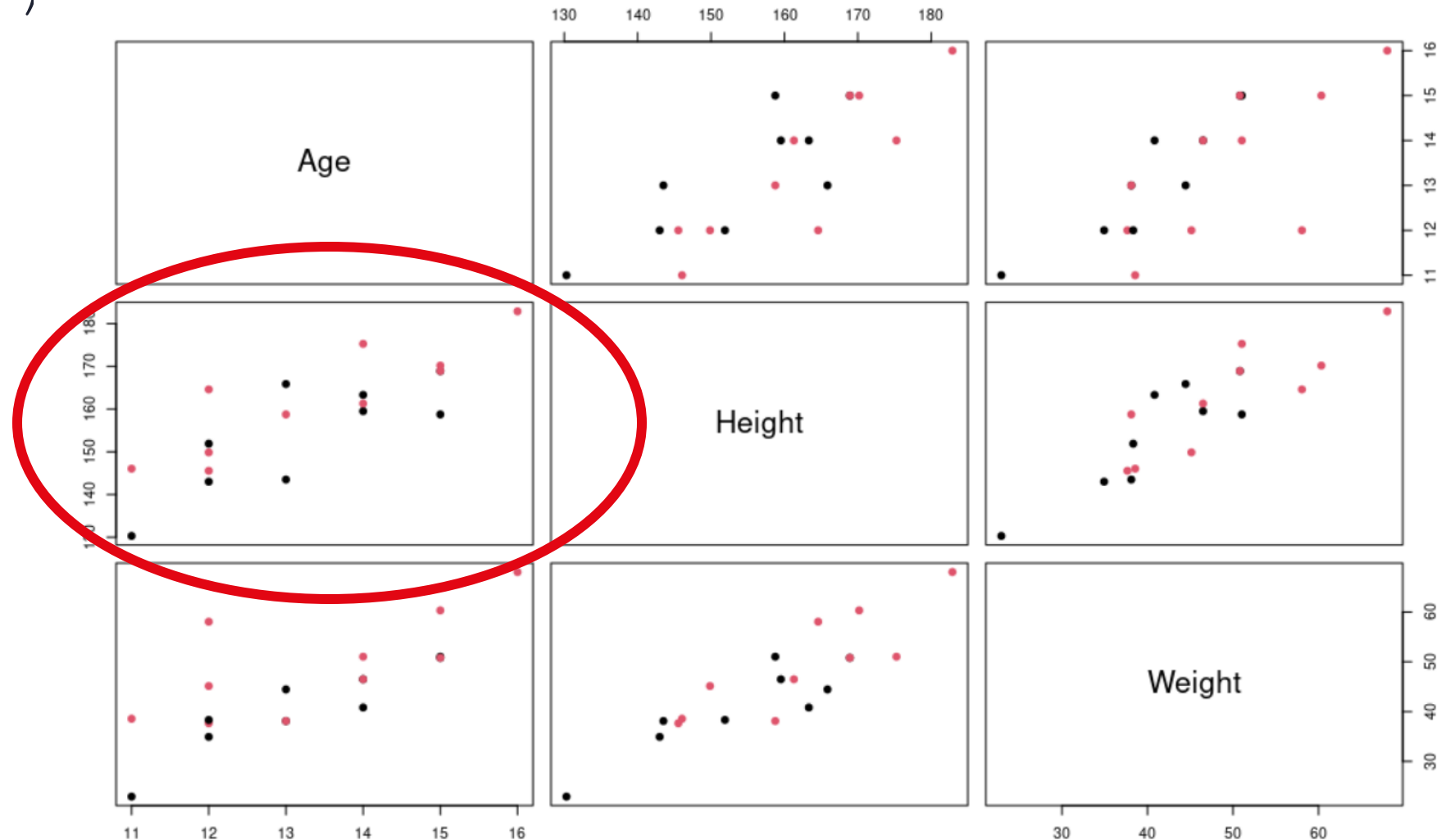
(names removed and units have been modified from imperial to metric)

```
class_data <- read.csv("course_dataset/class.csv")
class_data$Gender=as.factor(class_data$Gender)
summary(class_data)
```

Gender	Age	Height	Weight
F: 9	Min. :11.00	Min. :130.3	Min. :22.91
M:10	1st Qu.:12.00	1st Qu.:148.0	1st Qu.:38.22
	Median :13.00	Median :159.5	Median :45.13
	Mean :13.32	Mean :158.3	Mean :45.37
	3rd Qu.:14.50	3rd Qu.:167.4	3rd Qu.:50.92
	Max. :16.00	Max. :182.9	Max. :68.04

Visual summary of the data

```
pairs(class_data[2:4] , col = class_data$Gender,  
      pch=19 )
```



The `lm()` function

`lm()` : fits a linear model.

- Creates an R object which contains the regression result
- Just printing the result provides only the regression coefficients.
- The **`summary()`** and **`plot()`** functions can be used to provide more information, including diagnostic plots.

Many other functions can be applied to the regression objects:

- **`residuals()`** extracts a vector containing the residuals (error)
- **`coef()`** extracts the regression coefficients
- **`anova()`** produces the corresponding ANOVA table (not covered

Simple linear regression code

```
model_height_age <- lm(Height~Age, data=class_data)
model_height_age
```

Call:

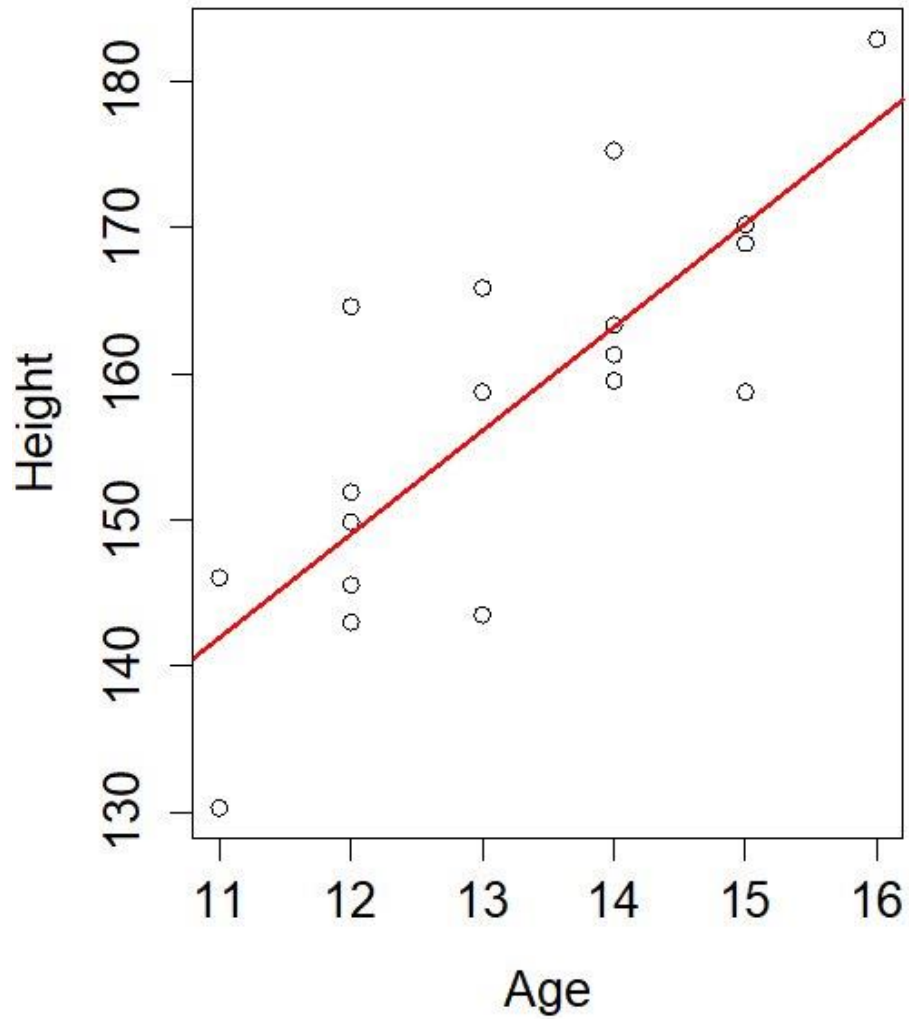
```
lm(formula = Height ~ Age, data = class_data)
```

Coefficients:

(Intercept)	Age
64.069	7.079

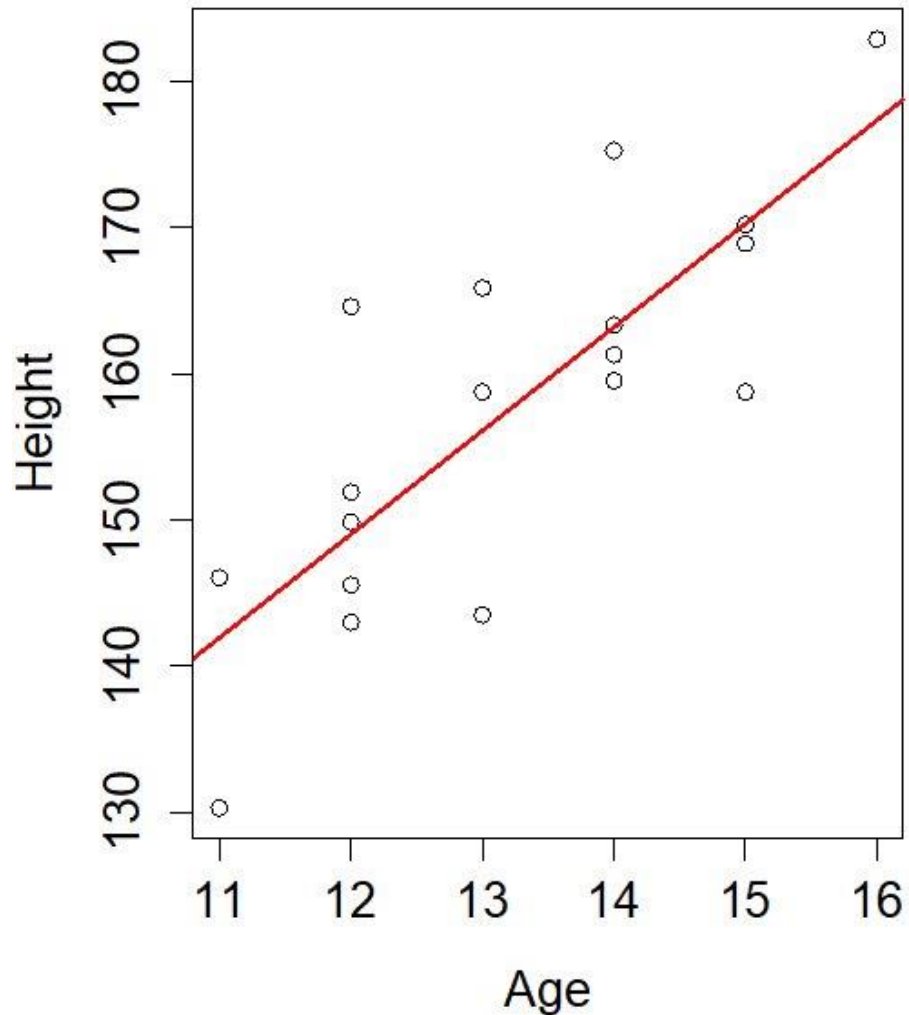
Representation of the fit (I)

```
plot (Height~Age,data=class_data)  
abline(model_height_age, col="red", lwd=2)
```



Representation of the fit (II)

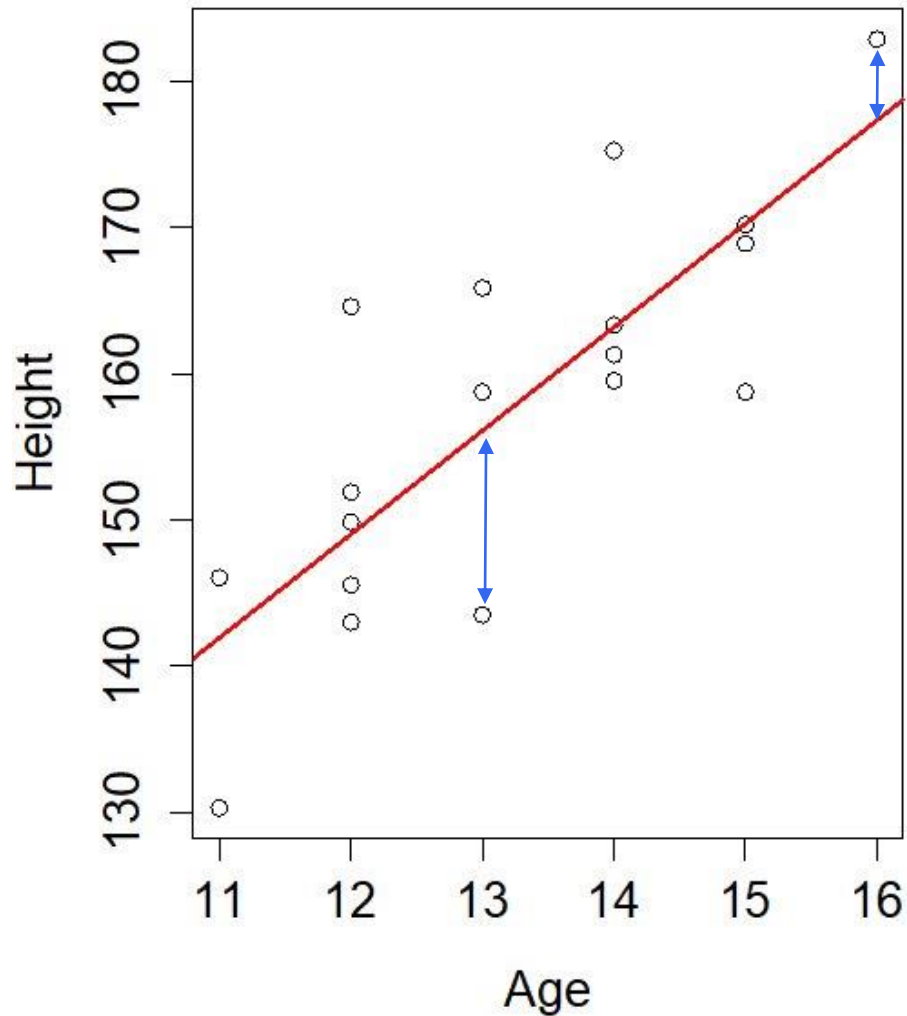
```
plot (Height~Age,data=class_data)  
abline(model_height_age, col="red", lwd=2)
```



Coefficients: y-intercept and slope of the regression line

```
Coef (model_height_age)  
(Intercept)      Age  
64.068667        7.079333
```


Representation of the fit (III)



Residuals: vertical distances of data points from the regression line

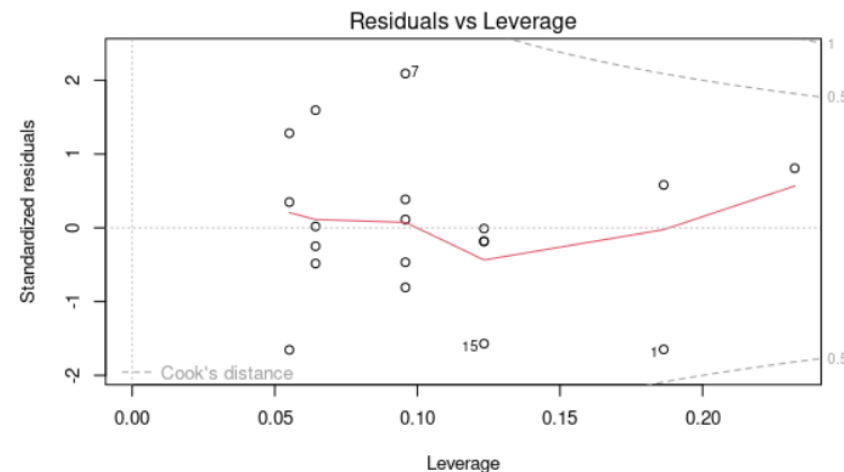
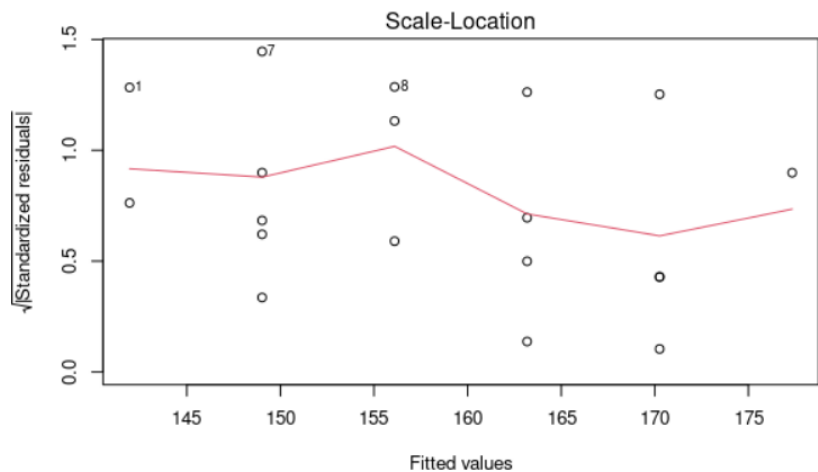
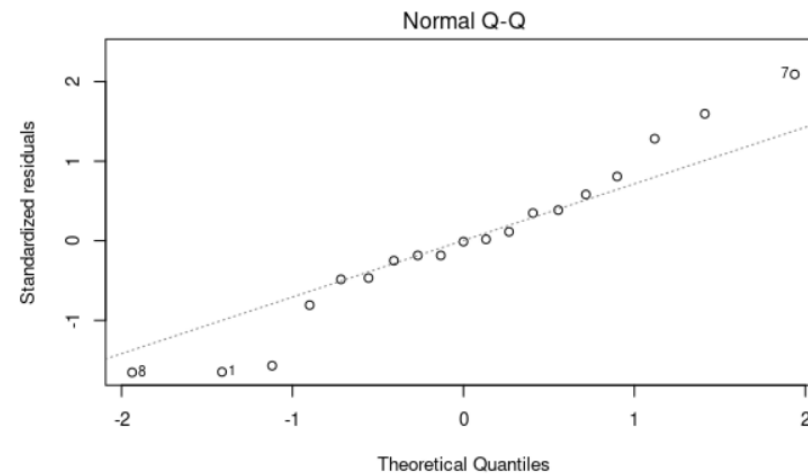
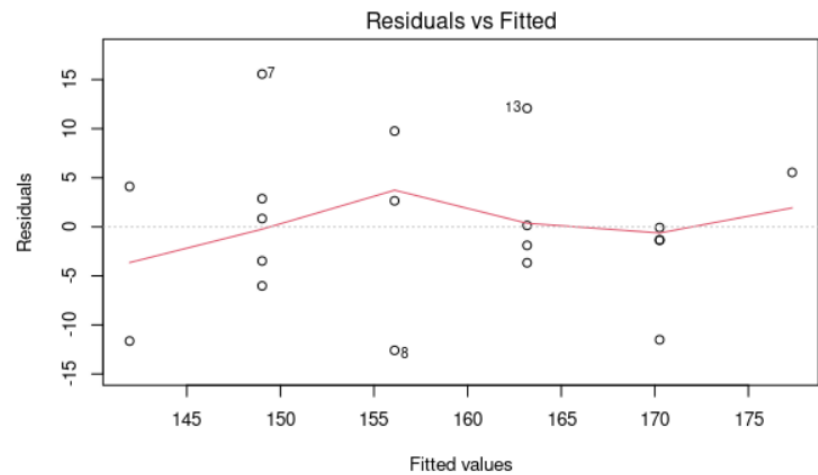
```
residuals(model_height_age)
      1      2      3      4
-11.63933333  4.10866667 -3.47866667  2.87133333
      5      6      7      8
 0.83933333 -6.01866667 15.57133333 -12.59000000
      9     10     11     12
 9.76200000  2.65000000 -3.66733333 -1.88933333
     13     14     15     16
12.08066667  0.14266667 -11.50866667 -1.34866667
     17     18     19
-0.07866667 -1.34866667  5.54200000
```

Check model assumptions

```
par(mfrow=c(2,2))  
plot(model_height_age)
```

Commentaries on these plots:

<https://library.virginia.edu/data/articles/diagnostic-plots>



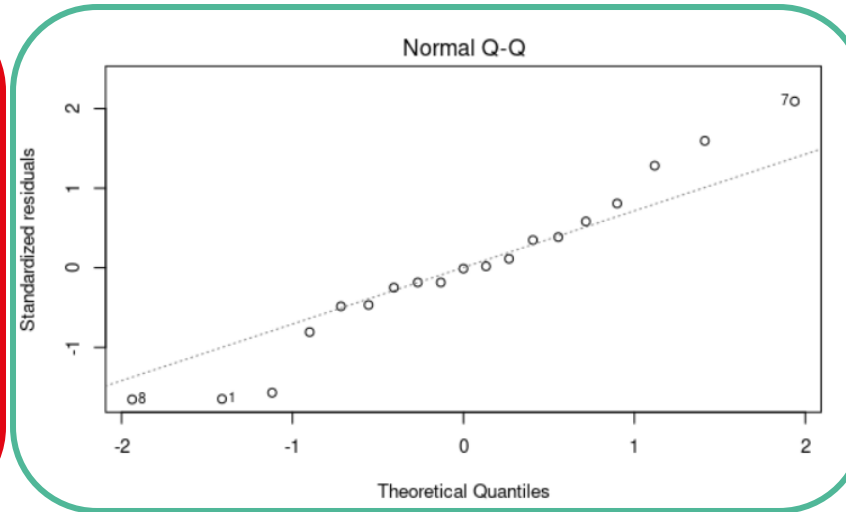
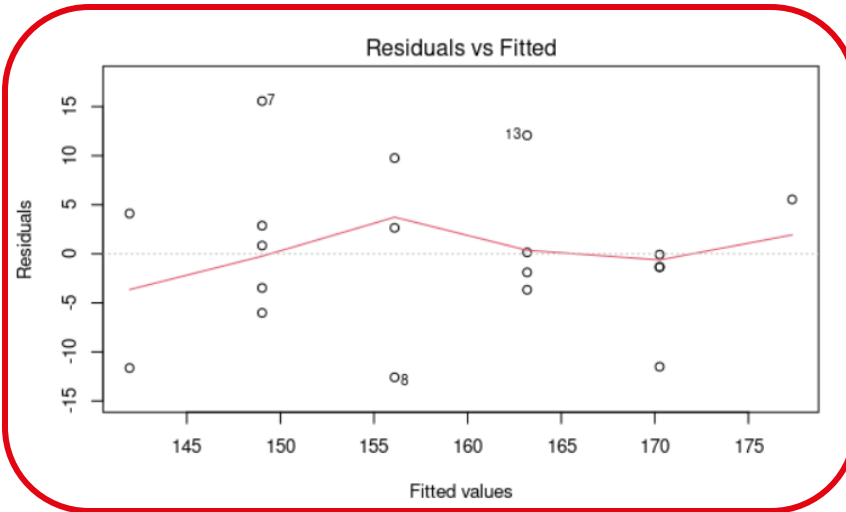
Check model assumptions

```
par(mfrow=c(2,2))  
plot(model_height_age)
```

Commentaries on these plots:

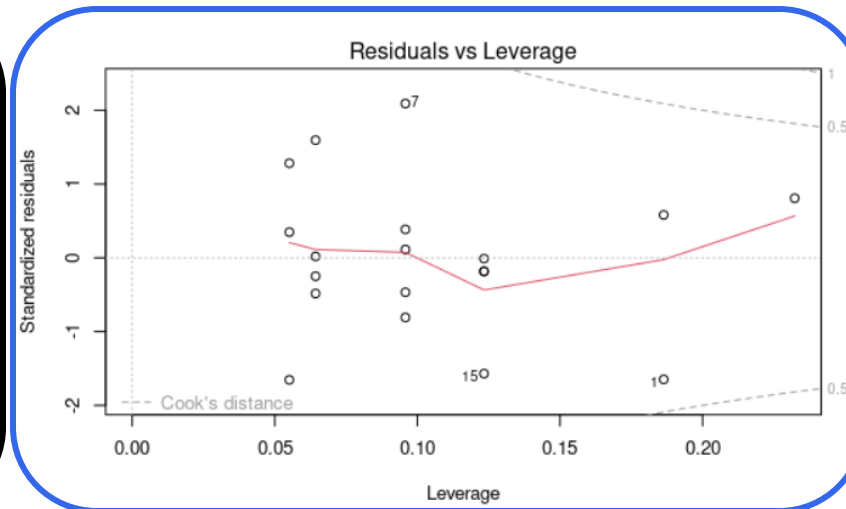
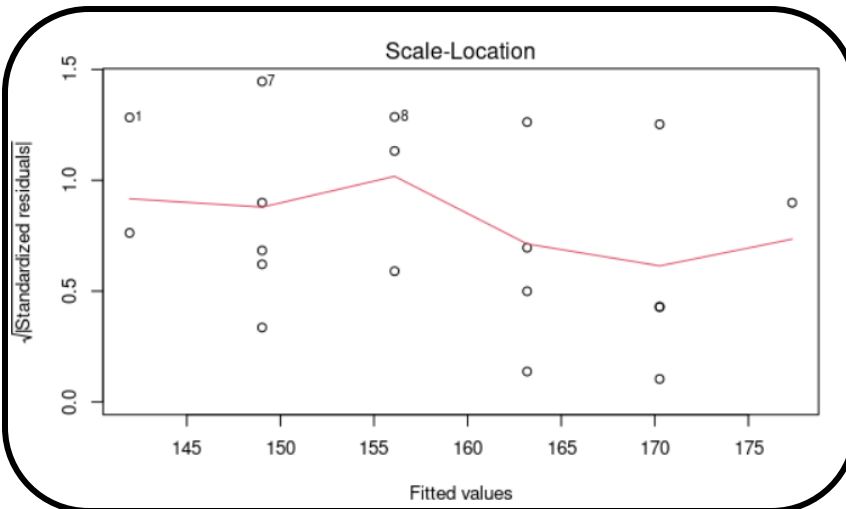
<https://library.virginia.edu/data/articles/diagnostic-plots>

Residuals centered
around 0



Normality of residuals

Homoskedasticity



Influence of points

Model summary

```
summary(model_height_age)
```

```
Call:
```

```
lm(formula = Height ~ Age, data = class_data)
```

```
Residuals:
```

Min	1Q	Median	3Q	Max
-12.5900	-3.5730	-0.0787	3.4900	15.5713

```
Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	64.069	16.565	3.868	0.00124	**
Age	7.079	1.237	5.724	2.48e-05	***

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 7.832 on 17 degrees of freedom
```

```
Multiple R-squared:  0.6584, Adjusted R-squared:  0.6383
```

```
F-statistic: 32.77 on 1 and 17 DF,  p-value: 2.48e-05
```

Model summary

```
summary(model_height_age)
```

Call:

```
lm(formula = Height ~ Age, data = class_data)
```

Residuals:

Min	1Q	Median	3Q	Max
-12.5900	-3.5730	-0.0787	3.4900	15.5713

Residuals: difference between observed and fitted

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	64.069	16.565	3.868	0.00124	**
Age	7.079	1.237	5.724	2.48e-05	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Parameter significance:

- T-test
- Residual normality assumption

Residual standard error: 7.832 on 17 degrees of freedom

Multiple R-squared: 0.6584, Adjusted R-squared: **0.6383**

F-statistic: 32.77 on 1 and 17 DF, p-value: **2.48e-05**

R-squared : fraction of the variance explained by the model

F-test: does the model explain significantly more than a model with just the intercept?

Let's practice - 11

The data set "Pima" comes from a study on diabetes in women of Pima Indian heritage.

We are using a subset (Pima.tr).

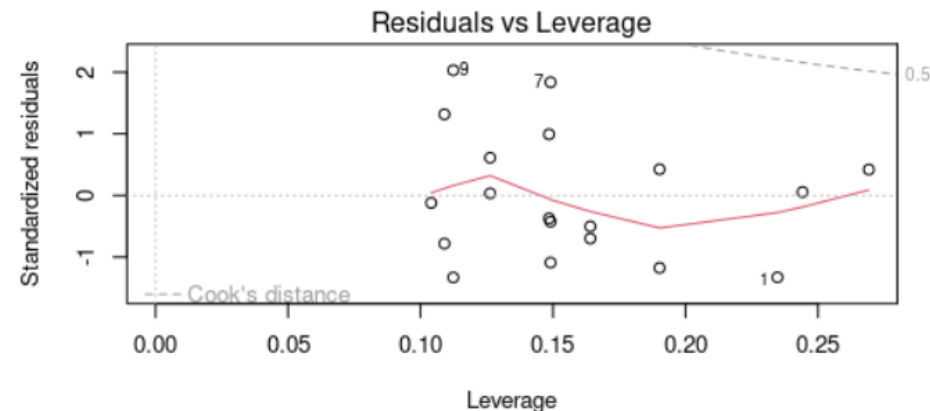
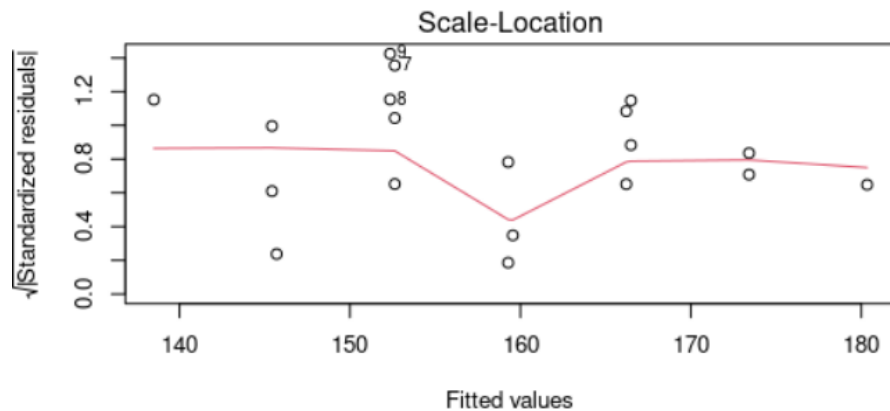
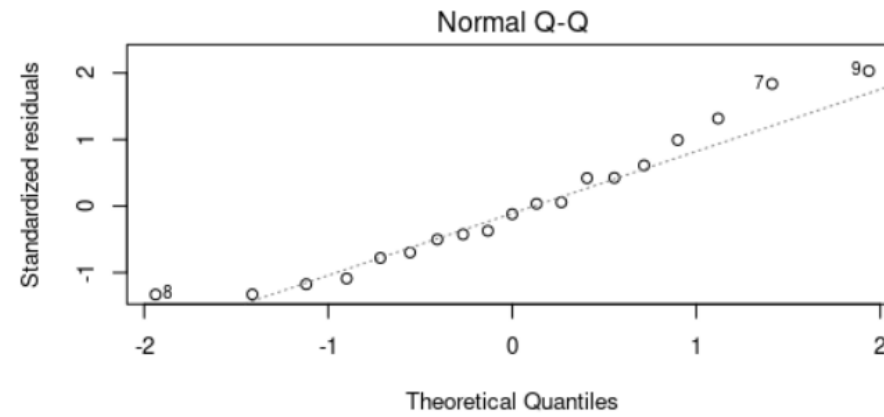
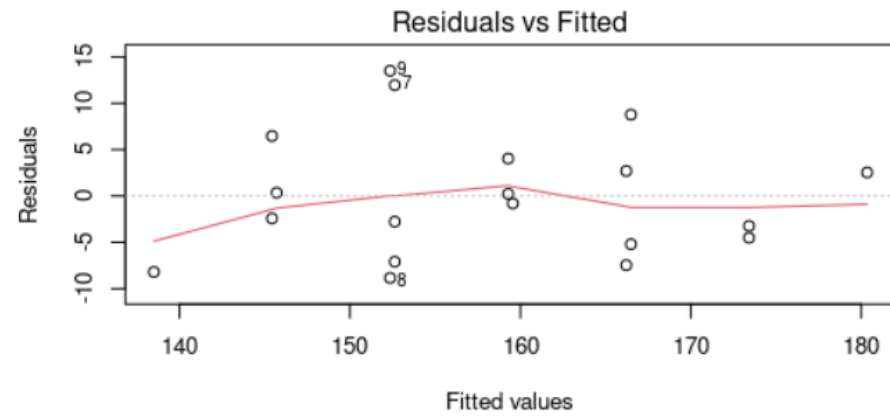
1. Load the package MASS using `library()`. Load the dataset Pima.tr using `data(Pima.tr)`. Use `?Pima.tr` to get an idea which variables it contains.
2. Hypothesis: Blood glucose level (glu) is associated with diastolic blood pressure (bp). Run a linear model to test the hypothesis.
3. Visualize the fit with a scatter plot and a trend line.
4. Check assumptions of the model (homoscedasticity, mean of residual at 0, normality of the residuals) graphically.

Additional content : model with 2 covariables (I)

```
model2 <- lm(Height~Age+Gender, data=class_data)
```

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

```
plot(model2)
```



Additional content : model with 2 covariables (II)

```
summary( model2 )
```

Call:

```
lm(formula = Height ~ Age + Gender, data = class_data)
```

Residuals:

Min	1Q	Median	3Q	Max
-8.8462	-4.8523	-0.8102	3.3677	13.5058

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	62.291	14.957	4.165	0.000731	***
Age	6.928	1.117	6.202	1.27e-05	***
GenderM	7.204	3.251	2.216	0.041517	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

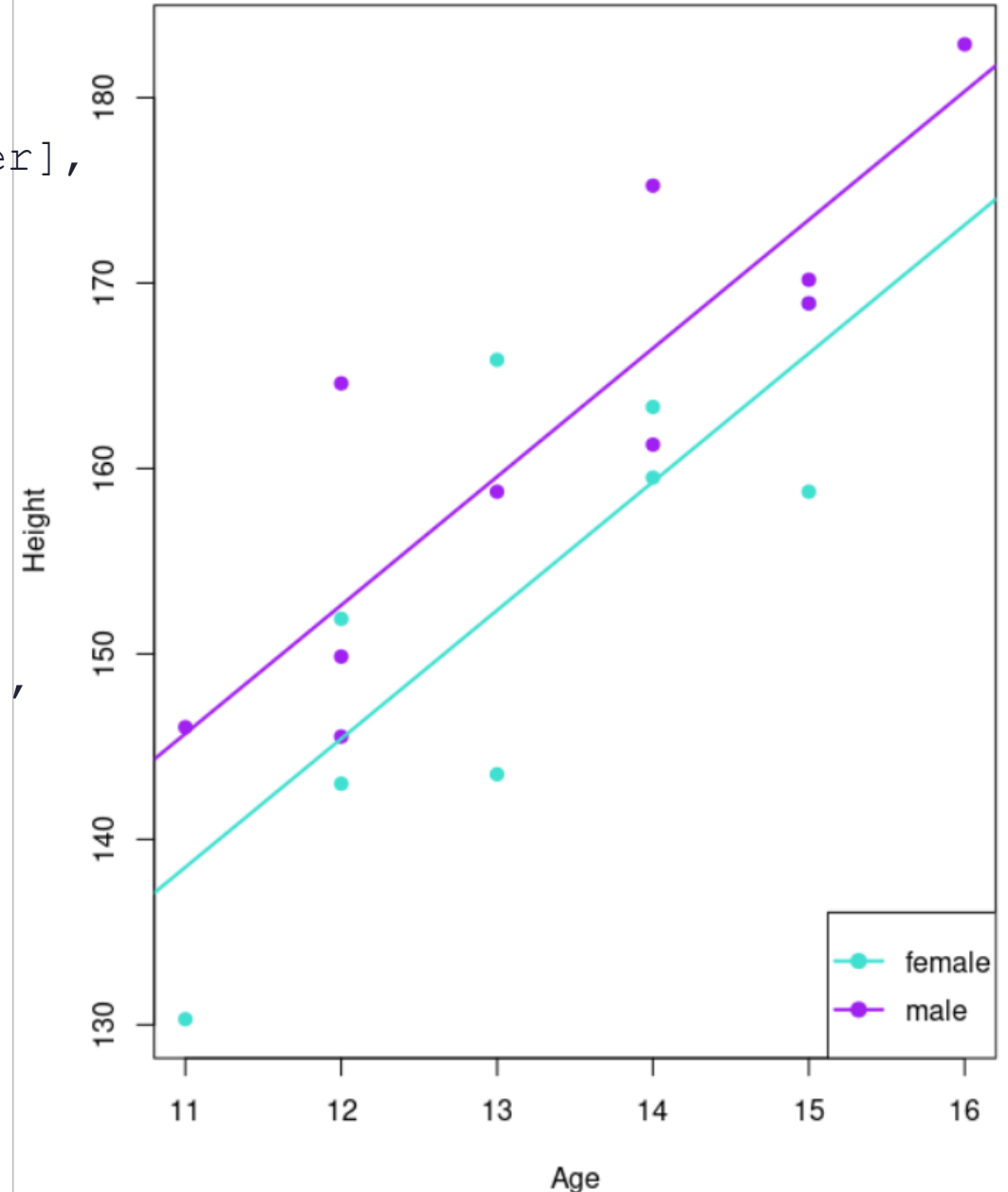
Residual standard error: 7.061 on 16 degrees of freedom

Multiple R-squared: 0.7387, Adjusted R-squared: 0.706

F-statistic: 22.61 on 2 and 16 DF, p-value: 2.176e-05

Additional content : model with 2 covariables (III)

```
plot( Height~Age,  
      col=c('turquoise','purple')[Gender],  
      pch=19, data=class_data )  
  
C = coef( model2 )  
abline(a=C[1],b=C[2] ,lwd=2,  
       col='turquoise')  
abline(a=C[1]+C[3],b=C[2] lwd=2,  
       col='purple')  
  
legend('bottomright',c('female','male'),  
      pch=19,lwd=2,  
      col=c('turquoise','purple'))
```



The next steps

- **R manuals:** <http://cran.r-project.org/manuals.html>
- **Datacamp free tutorials:** <https://www.datacamp.com/courses/free-introduction-to-r>
- **STHDA (Statistical Tools for High Throughput Data Analysis) free tutorials:** <http://www.sthda.com/english/>
- **Stackoverflow** documentation, resources and user forum: <http://stackoverflow.com/tags/r/info>
- **Rseek** - search engine on numerous online R resources: <http://www.rseek.org>

Credits and Acknowledgments

Content and slides developed by:

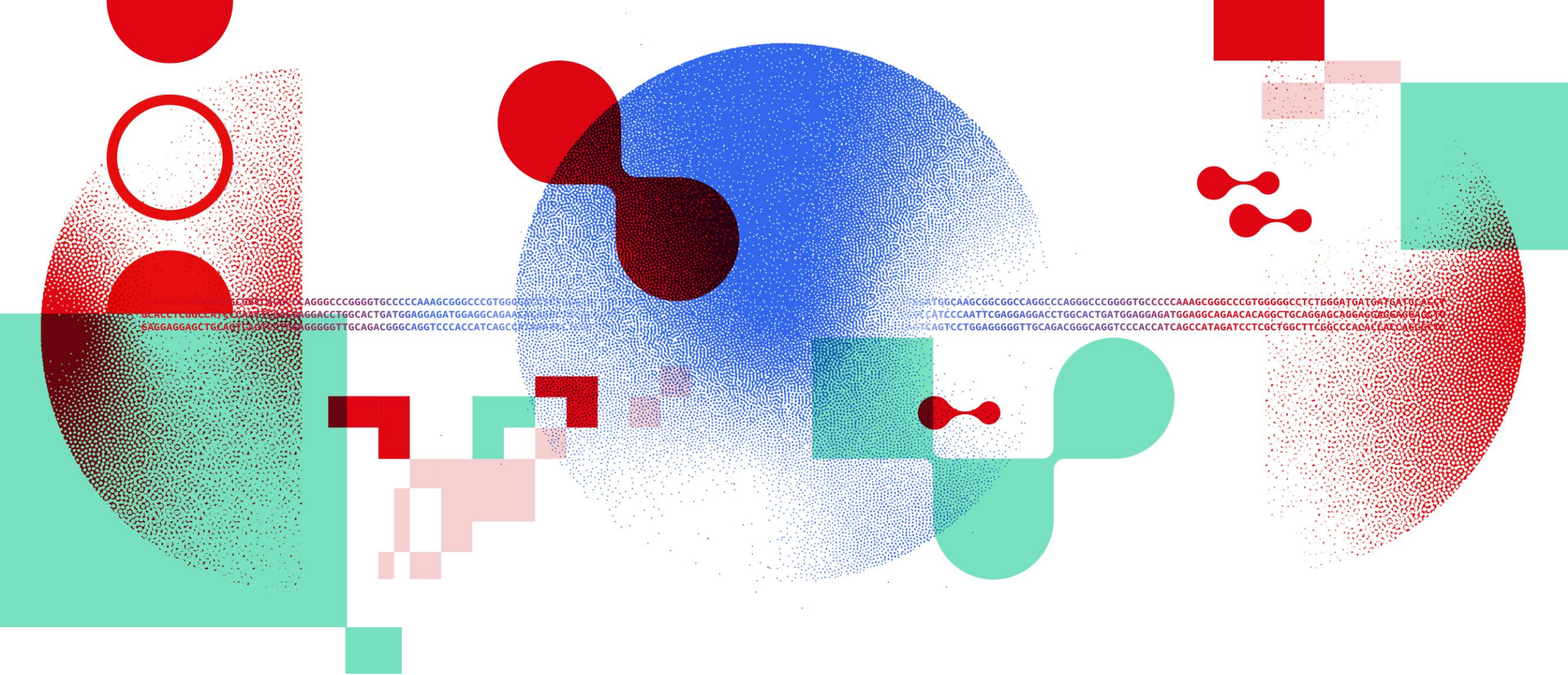
Diana Marek, Geoffrey Fucile, Alex Smith, Linda Dib, Leonore Wigger, Wandrille Duchemin

Content inspired by material from:

- Owen L. PETCHEY and “Getting started with R” book
- Daniel WEGMANN and Frédéric SCHÜTZ
- Robert STOJNIĆ and Ian ROBERTS
- Jenny DRNEVICH

<http://www.sib.swiss/training>

Any questions? Contact training@sib.swiss



Thank you

sib.swiss