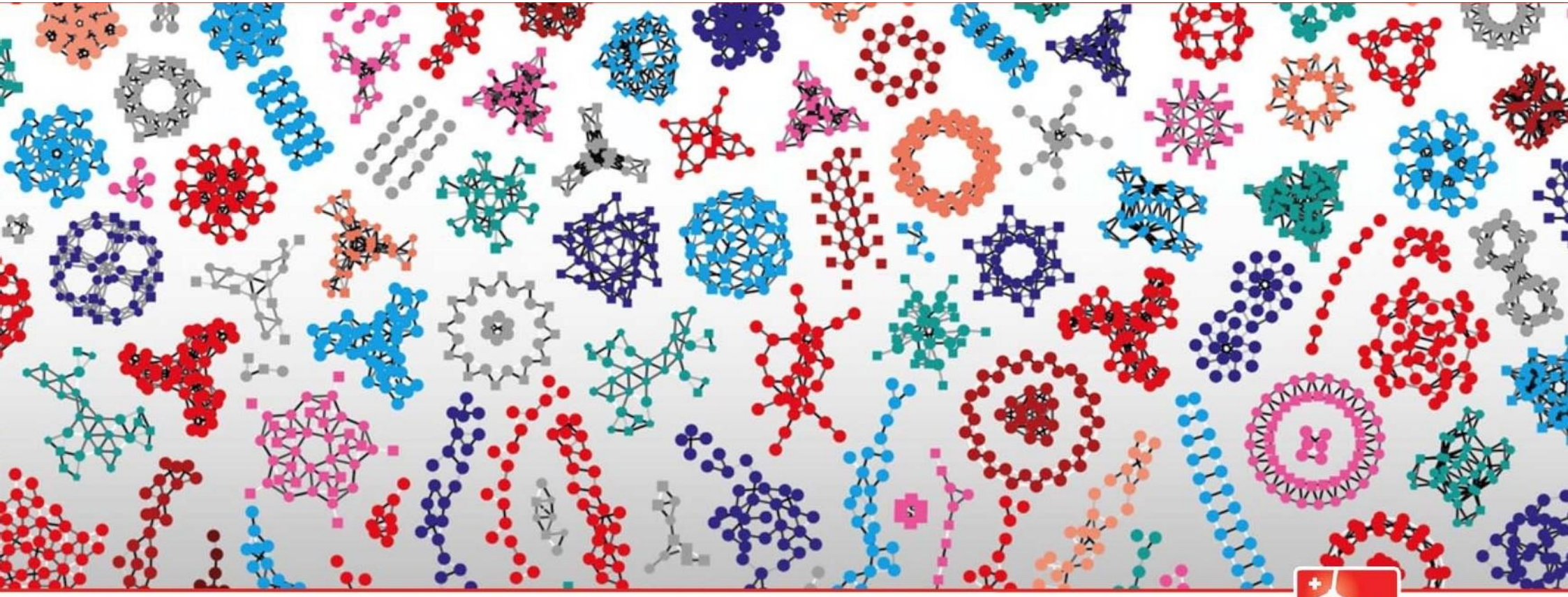


Introduction to Statistics

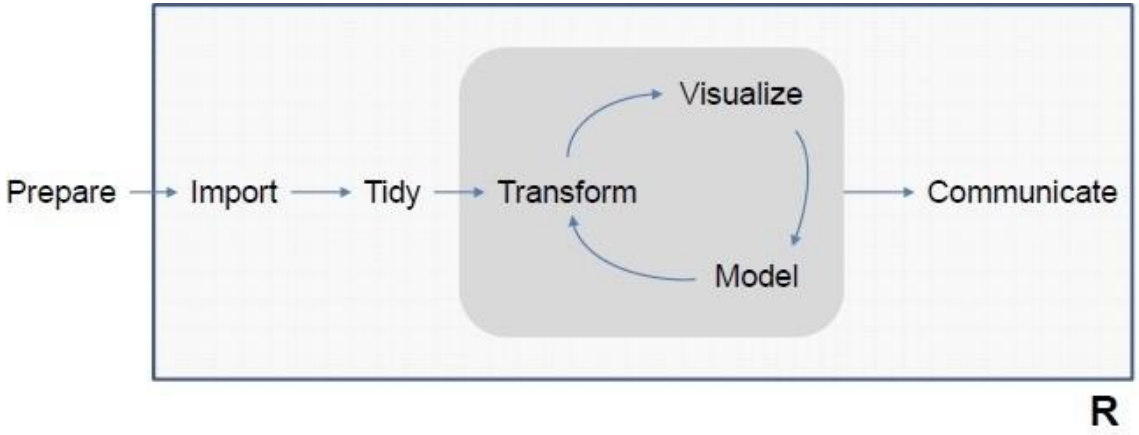
Joao Lourenço (joao.lourenco@sib.swiss) and Rachel Marcone (rachel.marcone@sib.swiss)

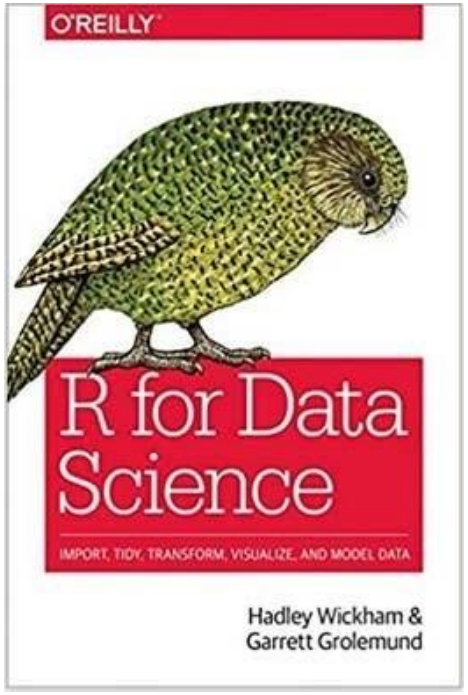
January 2024

SIB Swiss Institute of Bioinformatics

Data analysis with R:
An introduction

Data analysis workflow





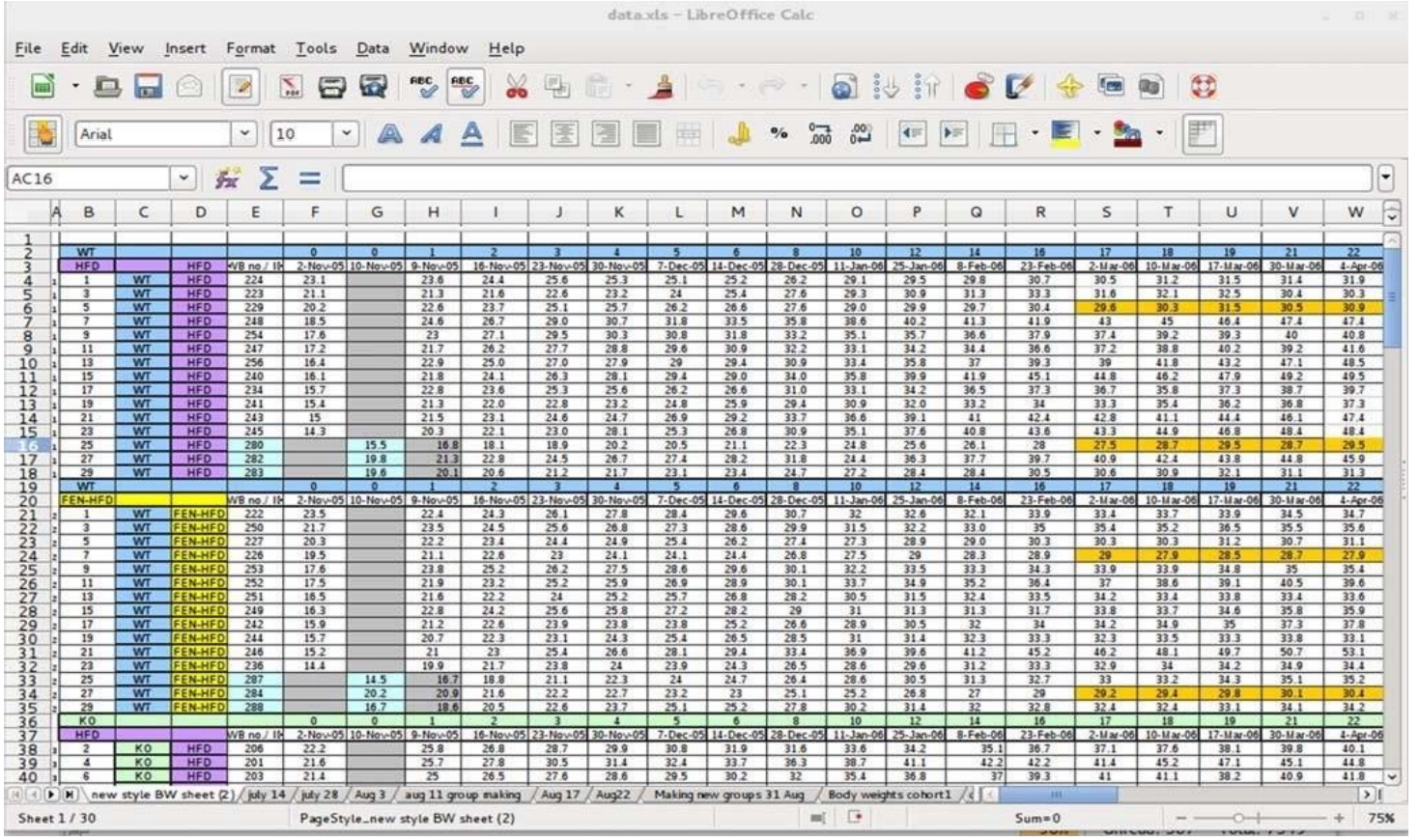
Adapted from Hadley Wickham

Hadley Wickham

Garrett Grolemund

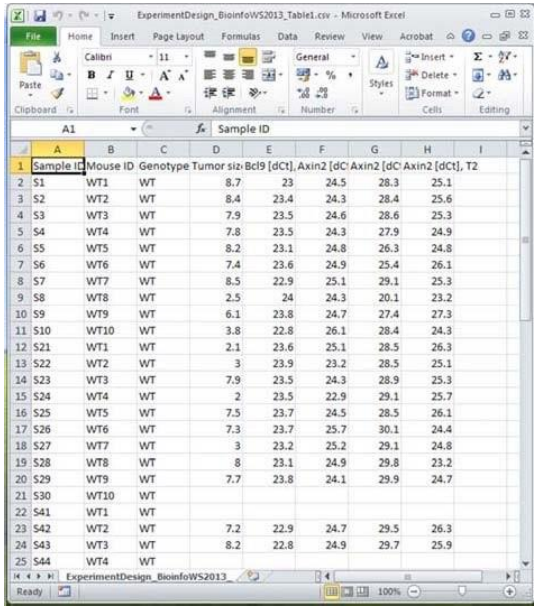
Prepare: make data available in a specific format

- Database
- Flat file
- Proprietary file

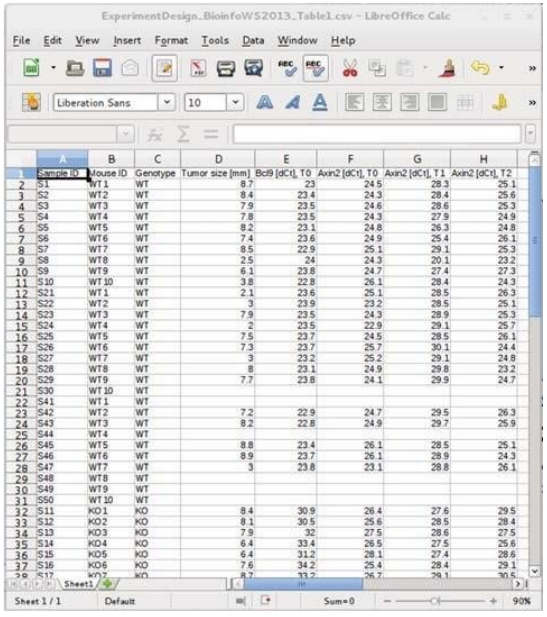


Which tool to use for data analysis ?

Microsoft Excel

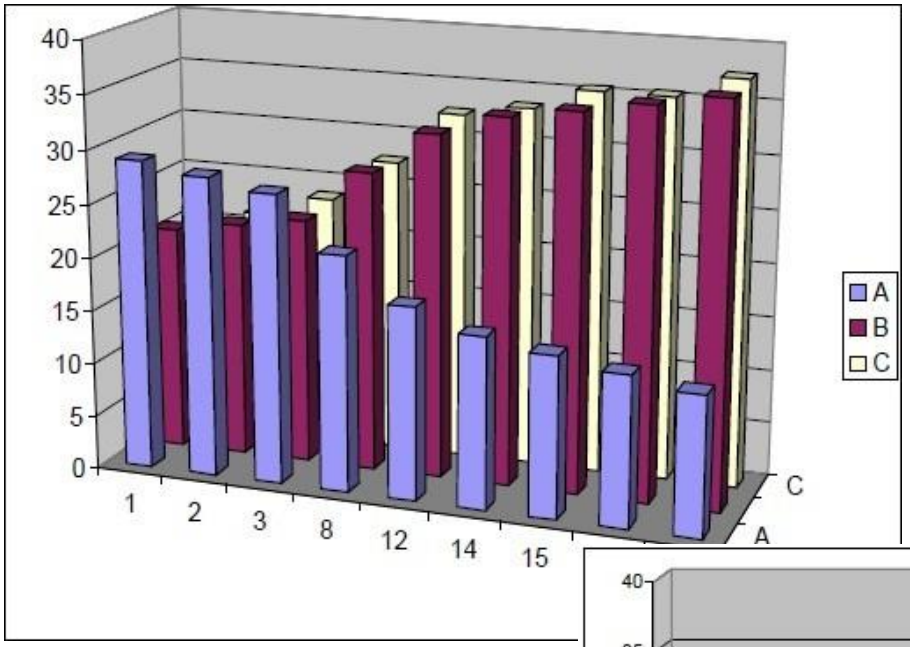


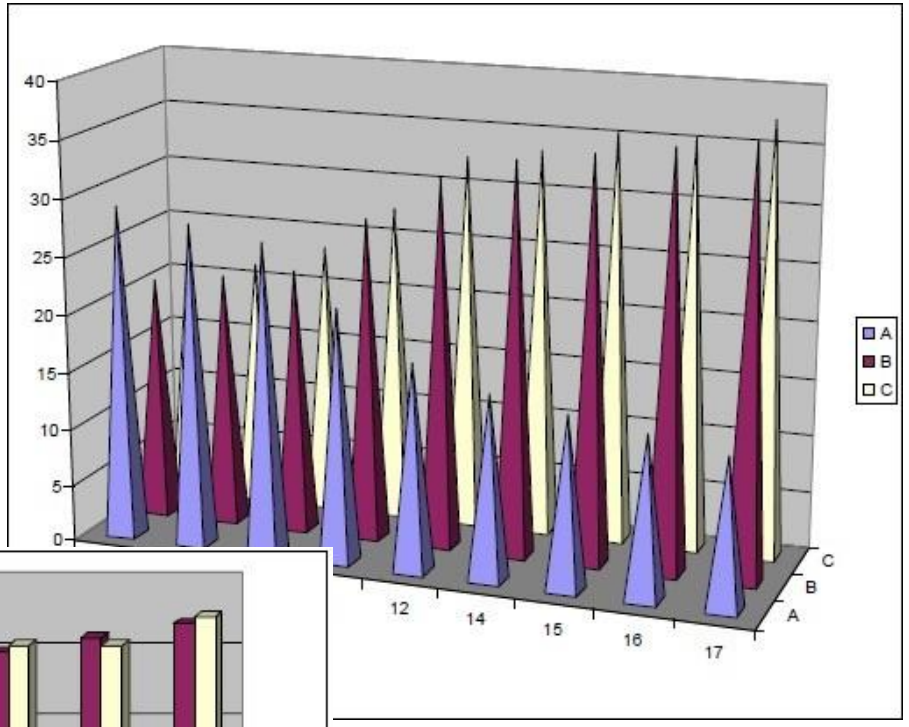
Libreoffice

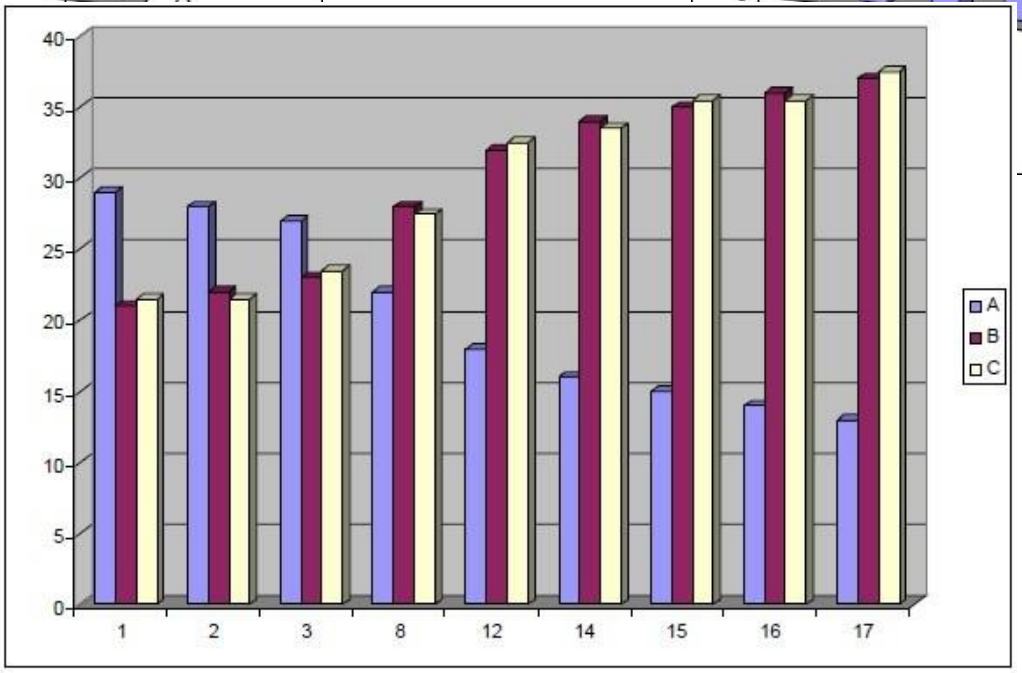


Annoyances with spreadsheets

- Many standard methods in statistics are not available. Other methods only offer basic options (linear regression)
- Different analysis require user to reorganize the data
- Probably ok for simple calculations (basic summary statistics, simple regression)
- Add-ons can be used for missing functions (e.g. StatPlus for Excel)
- Many types of graphics violate standards of good graphics


40
35
30
25
20
15
10
5
0
1
2
3
8
12
14
15
A
B
C
C
A


40
35
30
25
20
15
10
5
0
12
14
15
16
17
A
B
C
C
B
A

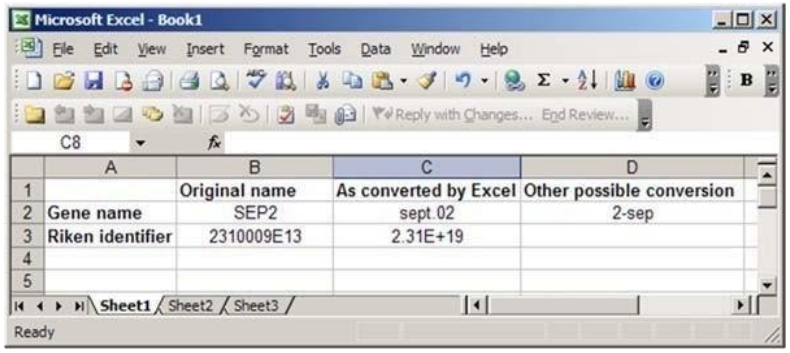

40
35
30
25
20
15
10
5
0
1
2
3
8
12
14
15
16
17
A
B
C

Annoyances with spreadsheets

Mistaken Identifiers: Gene name errors can be introduced inadvertently when using Excel in bioinformatics

Barry R Zeeberg, Joseph Riss, David W Kane, Kimberly J Bussey, Edward Uchio, W Marston Linehan, J Carl Barrett & John N Weinstein

BMC Bioinformatics **5**, Article number: 80 (2004) | Cite this article



"The date conversions affect at least 30 gene names; the floating-point conversions affect at least 2,000 if Riken identifiers are included. These conversions are irreversible; the original gene names cannot be recovered."

Example of a dataset which is difficult to use with any statistical program

Sample	sample_Init	Study_ID	comments	unique patients	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Age_OP	gender	AFFY
2248	MD_2	BE-03		1	0	1	1	0	20	0	50	M	1
2467	RB_2	BE-04		1	1	1	1	1	12	0	55	M	1
2468	HB_2	BE-05		1	1	1	1	1	13	1	66	M	1
2482	WO_2	ZH-01		1	1	1	1	1	7	1	64	M	1
2484	HW_2	ZH-04		1	1	1	1	1	5	1	50	M	1
2485	BD_2	ZH-05		1	1	1	1	1	6	0	53	F	1
2486	BH_2	ZH-06		1	1	1	1	1	9	1	48	F	1
2487	AW_2	ZH-07		1	1	1	1	1	9	0	53	M	1
2488	AJN_2	ZH-08		1	1	1	1	1	5	0	35	M	1
2489	KO_2	ZH-09		1	0	1	1	1	54	0	59	M	1
2490	BS_2	ZH-11		1	0	1	1	1	150	0	59	M	1
2491	KPR_3	ZH-12		1	1	1	1	1	5	0	32	M	1
2492	CB_3	ZH-13		1	0	1	1	0	6	0	37	F	1
2493	RM_3	ZH-14		1	0	1	1	1	63	0	39	M	1
2496	BR_2	ZH-17		1	1	1	1	1	5	0	61	F	1
2497	SP_2_0	2497		1		0	0			1	58	M	1
2498	NA_2_0	2498		1		0	0			0	54	M	1
2499	GK_2_0	2499		1		0	0			1	68	M	1
2500	HiB_2_0	2500		1		0	0			1	62	M	1
2501	BI_2	2501		1		0	0			0	70	F	1
2502	WJ_2	2502		1		0	0			1	59	M	1
2503	BP_3	2503	autopsy	1		0	0			0	61	M	1
2504	UA_2_0	2504		1		0	0			0	35	F	1
2505	GE_1	2505		0		0	0			1	65	F	1
2506	TS_2	2506		1		0	0			0	50	M	1
2507	HV_2_0	2507		1		0	0			0	65	F	1
2508	TI_3	2508		1		0	0			1	31	F	1
2509	TI_4_0	2509	Rec 2508	0		0	0			1	31	F	1
2510	GE_2_0	2510	Rec 2505	1		0	0			1	67	F	0
2511	SI_2	ZH-18		1	1	1	1	1	5	0	24	F	1
2512	BH_3	ZH-06.1	Rec 2486	0		1	0			1	50	F	1
2513	CG_2	2513		1		0	0			0	63	M	1
1152	NCH1152	NCH1152		Xenograft			0			1		hXenograft	1
1154	NCH1154	NCH1154		Xenograft			0			1		hXenograft	1
1155	NCH1155	NCH1155		Xenograft			0			1		hXenograft	1
1157	NCH1157	NCH1157		Xenograft	1		1		5	1		hXenograft	1
1159	NCH1159	NCH1159		Xenograft	1		1		5	1		hXenograft	1
1161	NCH1161	NCH1161		Xenograft	1		1		5	1		hXenograft	1
BS153 Control	ctrl BS153	ctrl BS153		Cell line						1		hCell line	0

Comparison of statistical packages

From Wikipedia, the free encyclopedia

The following tables compare general and technical information for a number of statistical analysis packages.

General information [edit]

Product	Developer	Latest version	Open source	Software license	Interface	Written in	Scripting languages
ADaMSoft	Marco Scarno	27 April 2015	Yes	GNU GPL	CLI, GUI	Java	
Alteryx	Alteryx Inc.	2019.2 (June 2019)	No	Proprietary	GUI, Python SDK, js SDK	C#, C++, Python, R, js	R, Python
Analyse-it	Analyse-it		No	Proprietary	GUI	C#, C++, Fortran	
ASReml	VSN International	26 March 2014	No	Proprietary	CLI		
BMDP	Statistical Solutions		No	Proprietary			
Dataplot	Alan Heckert	2013	Yes	Public domain	CLI, GUI	Fortran	
ELKI	Ludwig Maximilian University of Munich	0.7.5 (15 February 2019)	Yes	AGPL	CLI, GUI	Java	Shell (computing)

https://en.wikipedia.org/wiki/Comparison_of_statistical_packages

Regression [edit]

Support for various regression methods.

Product	OLS	WLS	2SLS	NLLS	Logistic	GLM	LAD	Stepwise	Quantile	Probit	Cox	Poisson	MLR
ADaMSoft	Yes	Yes	No	Yes	Yes	No	No	Yes					
Alteryx	Yes	Yes			Yes	Yes		Yes		Yes			
Analyse-it	Yes				Yes								Yes
BMDP	Yes				Yes			Yes			Yes		
Epi Info	Yes	No	No	No	Yes	No	No	No			Yes		
EViews	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes
GAUSS	Yes	Yes			Yes	Yes	No		Yes			Yes	Yes
GenStat	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GraphPad Prism	Yes	Yes	No	Yes	Yes	No	No	No	No	No		No	Yes
gretl	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes		Yes	
JMP	Yes	Yes	No	Yes	Yes	Yes	No	Yes	In JMP Pro	Yes	In JMP Pro	Yes	Yes
LIMDEP	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Maple	Yes	Yes	No	Yes[18]	No	No	No	No	No	No	No	No	Yes
Mathematica	Yes	Yes		Yes	Yes[19]	Yes[20]	Yes[21]		Yes	Yes[22]	Yes[23]	Yes	Yes[24]
MATLAB+Statistics Toolbox	Yes	Yes	Yes[25]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
MaxStat Pro	Yes	Yes		Yes	Yes								Yes
MedCalc	Yes	Yes		Yes	Yes			Yes		Yes	Yes		Yes
Minitab	Yes	Yes	No	Yes	Yes	No	No	Yes	No	Yes		Yes	Yes
NCSS	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
NLOGIT	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Orange	Yes	Yes	No	Yes	Yes	No	No	No	No	No	No	No	Yes
Origin	Yes	Yes	No	Yes	No	No	No	No	No		Yes	No	Yes

What is R ?

- R is an open source complete and flexible software environment for statistical computing and graphics.
- It includes :
  - Tools for data import and manipulation
  - Large set of data analysis tools
  - Graphical tools
  - As a programming language, a simple development environment, with a text editor
- R itself is written primarily in C and Fortran, and is an implementation of the statistical language S

Why R ?

- R has become the tool of choice for statistical analysis in several fields, including life sciences
- Two reasons for this success: it is free and many contributed packages are available (can be installed and run directly from R).
- Well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed.
- Many tools implemented for bioinformatics

Advantages of R

- Advantages of R
  - Availability and compatibility
  - State-of-the-art graphics capabilities
  - Can import files from other (statistical) programs
  - New version every x months
  - Interactive development environments (IDEs) available
  - Large users community

- Advantages of *learning* R
  - Learn to program and do reproducible research
  - Speak the common language

Drawbacks of R

- «Expert friendly»
- Learn by example
- Not very (easily) interactive
- Command-based
- Documentation sometimes cryptic

- (Too) large amount of resources
- Constantly evolving
- Memory intensive and slow at times

Downloading and installing R: the R website

[Home]

Download

CRAN

R Project

About R
Logo
Contributors
What's New?
Reporting Bugs
Conferences
Search
Get Involved: Mailing Lists
Get Involved: Contributing
Developer Pages
R Blog

The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.

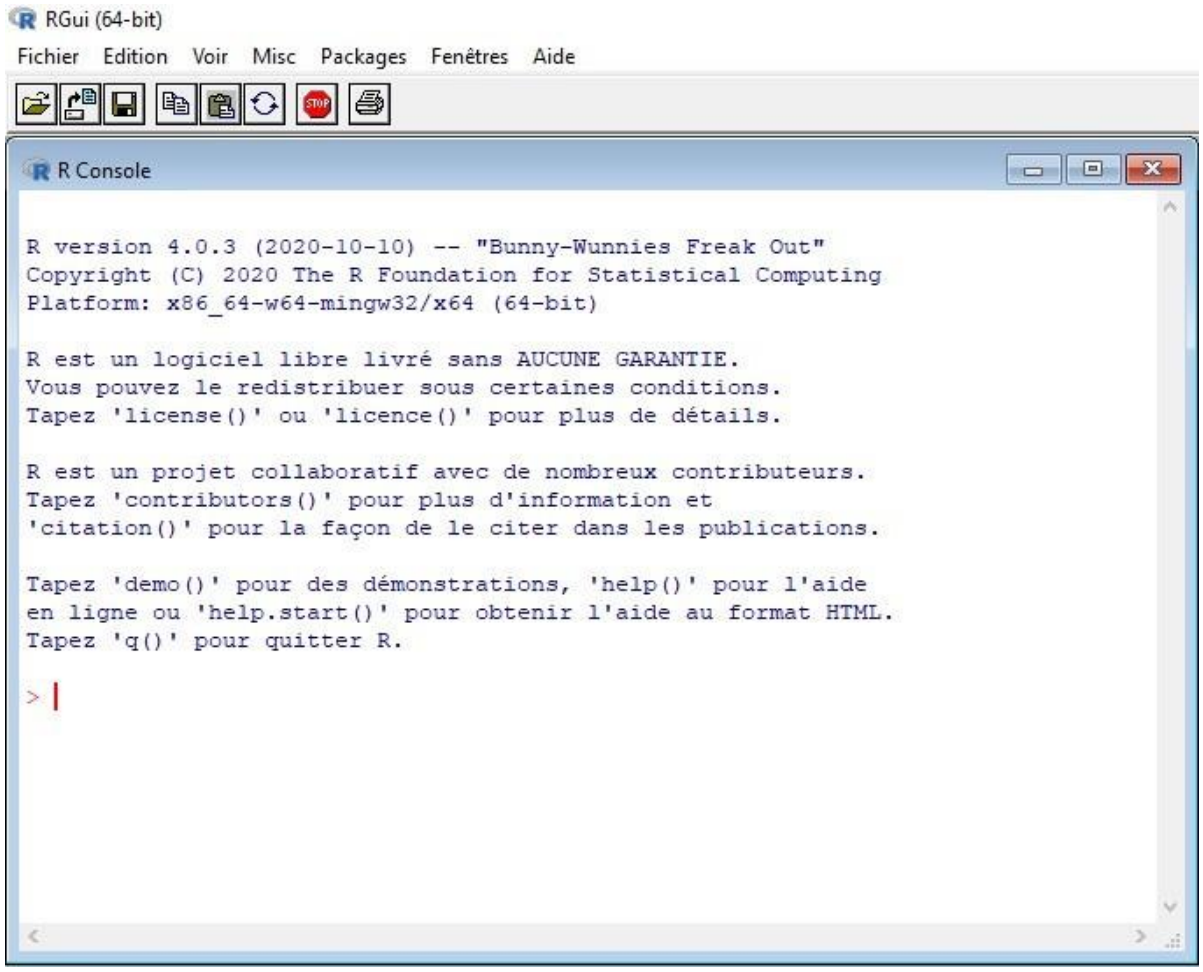
If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

News

- **R version 4.2.2 (Innocent and Trusting)** has been released on 2022-10-31.
- **R version 4.1.3 (One Push-Up)** was released on 2022-03-10.
- Thanks to the organisers of useR! 2020 for a successful online conference. Recorded tutorials and talks from the conference are available on the R Consortium YouTube channel.
- You can support the R Foundation with a renewable subscription as a supporting member

https://www.r-project.org/

R console

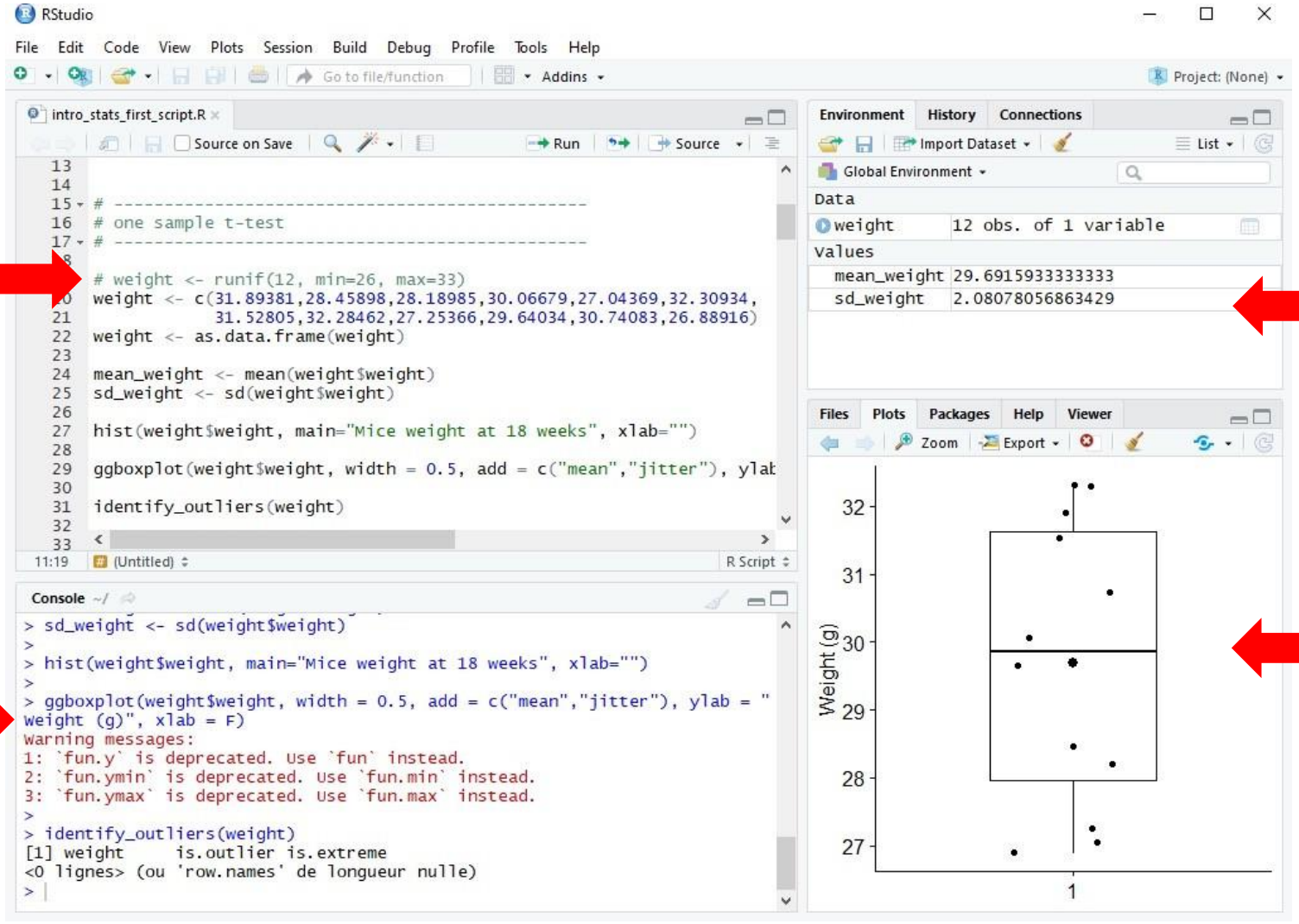


The prompt “>” indicates that R is waiting for you to type a command

RStudio interface

Editor

Console, terminal



Workspace, history

File explorer, plots, packages, help

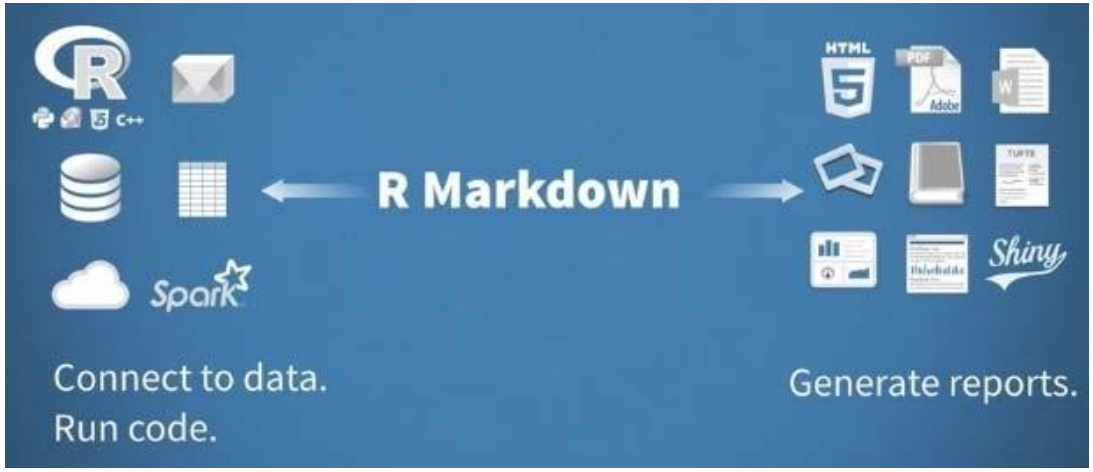
R scripts and workspace

- R script (.R file)
  - Very useful instead of typing commands on the console.
  - Allows you to keep track of what you are doing and make any modification easier
  - To actually execute some commands, you can select the lines and run the execution

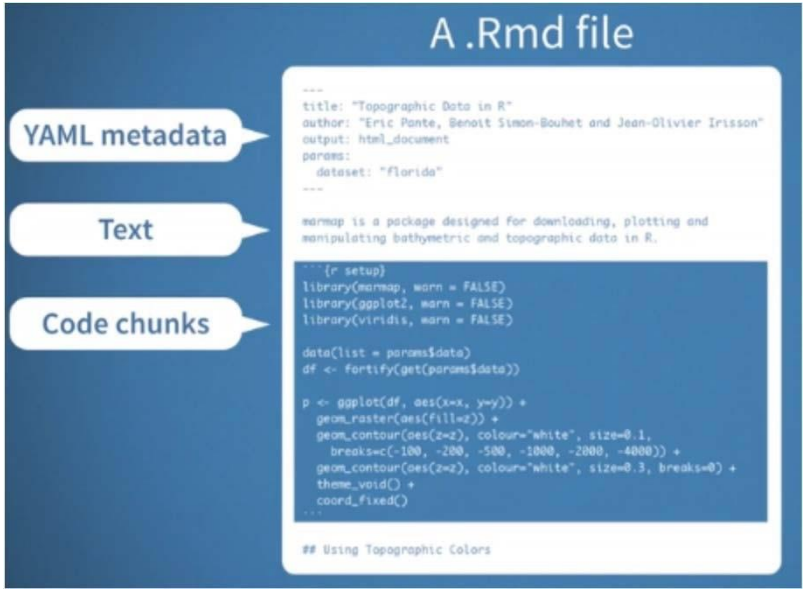
- Workspace (.Rdata file)
  - The internal memory where R will store the objects you created during the session.
  - To list what is in your workspace: `ls()`
  - To empty the workspace from all objects: `rm(list=ls())`
  - To save only specific R objects: `save(object_name(s),"name_of_file.RData")`
  - To save your entire workspace: `save.image("name_of_file.RData")`
  - To load your workspace / specific R objects: `load("name_of_file.RData")`

R Markdown

- R Markdown provides an authoring framework for data science. You can use a single R Markdown file to both:
  - save and execute code
  - generate high quality reports that can be shared with an audience
- R Markdown documents are fully reproducible and support dozens of static and dynamic output formats



https://rmarkdown.rstudio.com/lesson-1.html



Leaving R

- To leave R, use the q()command (or "quit" from the menu in RStudio):

```
> q()
Save workspace image? [y/n/c]:
```

Answers:

y save workspace image

n **don't save workspace image**

c cancel quitting

Functions, operators and variables

```
CIhigh <- mean(x) + 1.96*sd(x)/sqrt(n)
```

Variables: objects stored in memory

Functions: always followed by parenthesis

Operators

R syntax

- Case sensitive: A is not a
- Variable names can include A-Z, a-z, 0-9, …. but can not start with a number
- Commands can be separated by ; or newline

```
> x <- 2; x+2
[1] 4
```

- # indicates comments:

```
> maxvalue <- 2 # Data above two is not relevant
```

R help

```
> ?sum # equivalent to help(sum)
```

sum {base} R Documentation

Sum of Vector Elements

Description

`sum` returns the sum of all the values present in its arguments.

Usage

```
sum(..., na.rm = FALSE)
```

Arguments

`...`	numeric or complex or logical vectors.
`na.rm`	logical. Should missing values (including `NaN`) be removed?

Using R as a calculator

```
> 2*3
[1] 6
>log(6)/2^2
[1] 0.4479399
>exp(6)-4
[1] 399.4288
> pi-3
[1] 0.1415927
```

Using R as a programming language

```
> x <- 2.0
> x
[1] 2.0
> y = 3.0 # Equivalent to y <- 3.0
> y; x
[1] 3
[1] 2
>1/x
[1] 0.5
```

Creating vectors using the c() command

```
> x <- c(1.3, 0.32  10.5, 5.9, 6.3)
                ,
> x
[1] 1.30 0.32 10.5   5.90 6.30
                0
> y <- c(x, 1.4, x, x); y
[1] 1.30 0.32 10.5   5.90 6.30
                0
[6] 1.40 1.30 0.32 10.50 5.90
[11]  6.30 1.30 0.3   10.50 5.90
                  2
[16]  6.30
```

Vector operations

Vector operations work element by element:

```
> x <- c(1.3, 0.32, 10.5, 5.9, 6.3)
> y <- x*2; y
[1] 2.60 0.64 21.00 11.80 12.60
>z  <- x*y; z
[1] 3.38 0.21 220.50 69.62 79.38
```

Recycling

- If a vector is too short, R recycles it (reuses it) as needed:

```
> x <- c(1.3, 0.32, 10.5, 5.9)
> y <- c(2, 10)
> x*y
[1] 2.6 3.2 21.0 59.0
1.3*2 0.32*10 10.5*2 5.9*10
```

- A warning message is displayed if the shortest vector can not be recycled entirely:

```
> x <- c(1.3, 0.32, 10.5, 5.9, 6.3)
> x*y
[1] 2.6 3.2 21.0 59.0 12.6
Warning message:
In x * y :
longer object length is not a multiple of shorter object length
```

Generating sequences of numbers

```
> 1:10
[1] 1 2 3 4 5 6 7 8 9 10
```

This is equivalent to:

```
>c(1,2,3,4,5,6,7,8,9,10)
[1] 1 2 3 4 5 6 7 8 9 10
> 10:1
[1] 10 9 8 7 6 5 4 3 2 1
```

Beware of operator priority

```
> x <- 2*1:10
# equivalent to x <- 2*(1:10)
> x
[1] 2 4 6 8 10 12 14 16 18 20
> n <- 10
> 1:n-1
# equivalent to (1:n)-1
[1] 0 1 2 3 4 5 6 7 8 9
> 1:(n-1)
[1] 1 2 3 4 5 6 7 8 9
```

The seq() function: the same, but more flexible

```
> seq(from=1, to=10)
[1] 1 2 3 4 5 6 7 8 9 10
> seq(from=1, to=5, by=0.5)
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
> x <- seq(from=1, to=5, length=17)
> x
[1] 1.00 1.25 1.50 1.75 2.00 2.25 2.50 2.75
[9] 3.00 3.25 3.50 3.75 4.00 4.25 4.50 4.75
[17  5.0
]    0
```

Non numeric vectors: boolean (logical) values

```
> x <- seq(from=1, to=5, length=17)
> x
 [1] 1.00 1.25 1.50 1.75 2.00 2.25 2.50 2.75
 [9] 3.00 3.25 3.50 3.75 4.00 4.25 4.50 4.75
[17] 5.00
> y <- x<5  # help("<") shows list of relational operators
> y
 [1]  TRUE TRUE  TRUE TRUE  TRUE TRUE
 [7]  TRUE TRUE  TRUE TRUE  TRUE TRUE
[13] TRUE TRUE FALSE
>sum(x<5)
[1] 16
```

Missing values are designated by NA

```
> z <- c(1:3,NA)
> z
[1] 1 2 3 NA
> is.na(z)
[1] FALSE FALSE FALSE TRUE
> mean(z)
[1] NA
> mean(z, na.rm=TRUE)
[1] 2
```

Character strings

```
> char <- c("hello","world","!"); char
[1] "hello" "world" "!"
```

Vectors can not combine numbers and characters:

```
> char <- c("hello",3:5,"world"); char
[1] "hello" "3" "4" "5" "world"
> char <- c(char, NA); char
[1] "hello" "3" "4" "5" "world" NA
```

Selecting subsets of vectors using [ ]

```
> x <- 10:30
> x[2]
[1] 11
> x[1:5]
[1] 10 11 12 13 14
```

Selecting subsets of vectors using [ ] and boolean vectors

```
> x <- 10:30
> x[x>25]
[1] 26 27 28 29 30
>x <-c(seq(from=5, to=10,by=0.5),NA,
seq(from=11,to=15,by=0.5),NA,
seq(from=16,to=20,by=0.5))
> x[!is.na(x)]
[1] 5.0 5.5 6.0 6.5 7.0 7.5 8.0 8.5
[9] 9.0 9.5 10.0 11.0 11.5 12.0 12.5 13.0
[17] 13.5 14.0 14.5 15.0 16.0 16.5 17.0 17.5
[25] 18.0 18.5 19.0 19.5 20.0
```

Changing parts of vectors using [ ]

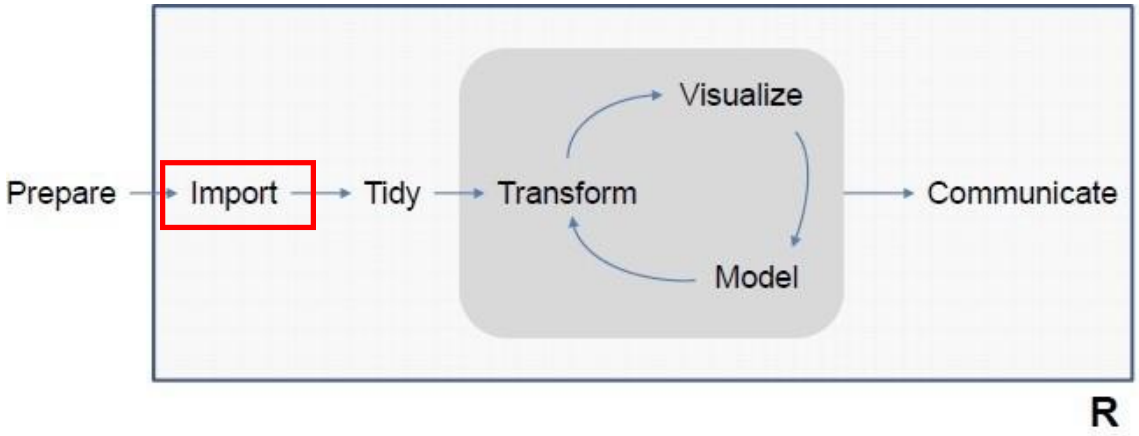
```
> x[32] <- 200
> x[c(10,29)] <- c(1,100)
> x[x>15] <- NA
```

Finding the length of a vector

```
> x <- 1:5
> length(x)
[1] 5

> y <- 1:16
>len <- length(y) ; len
[1] 16
```

Data analysis workflow



Adapted from Hadley Wickham

Importing data into R

- R can import flat files using e.g. the commands:

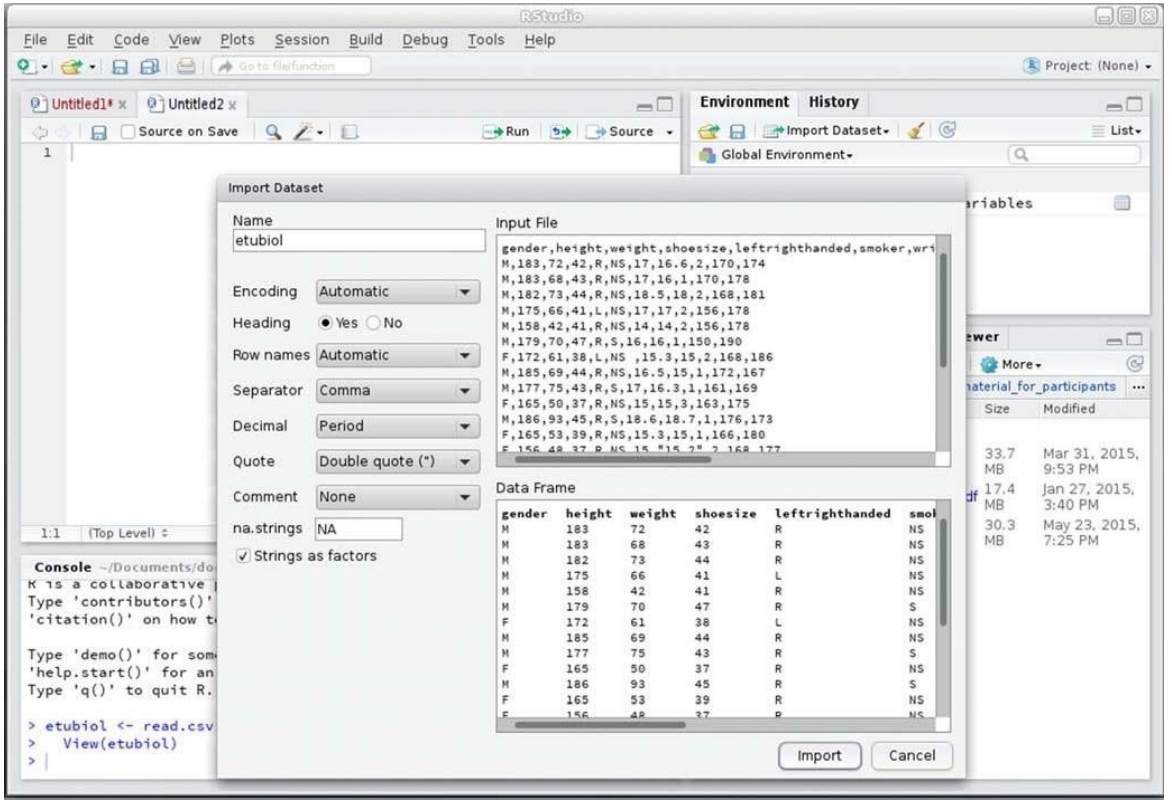
`read.table()`

`read.csv()`

`read.delim()`

(with many options – check the help).

- R can also:
  - Read Excel spreadsheets
  - Read plenty of other formats
  - Directly access databases
  - Access files over the web



Data frames

- Data frames are made of columns having all the same number of elements
- They look like matrices, except that the columns can hold different variables types
- They are typically used to store data, with
  - Each row being an experimental unit
  - Each column being a measurement

```
> data[,1] # access first column
> data[, "data1"] # access column "data1"
> data$data1 # … same
```

Creating data frames

```
> x <- 1:10
> y <- seq(from=5,to=10,length=10)
> z <- c("A","B","B","A","A","A","B","A","B","B")
> df <- data.frame(d1=x, d2=y, fact=z)
> df
 d1         d2 fact
1   1   5.000000    A
2   2   5.555556    B
..
> names(df)
[1] "d1" "d2" "fact"
>dim(df)
[1] 10  3
```

Adding new columns

```
> df$d3 <- 10:1
> df
 d1         d2 fact d3
1   1   5.000000    A 10
2   2   5.555556    B  9
…
>  summary(df)
       d1              d2             fact                d3
 Min.   : 1.00   Min.   : 5.00   Length:10          Min.   : 1.00
 1st Qu.: 3.25   1st Qu.: 6.25   Class :character   1st Qu.: 3.25
 Median : 5.50   Median : 7.50   Mode  :character   Median : 5.50
 Mean   : 5.50   Mean   : 7.50                      Mean   : 5.50
 3rd Qu.: 7.75   3rd Qu.: 8.75                      3rd Qu.: 7.75
 Max.   :10.00   Max.   :10.00                      Max.   :10.00
```

Select data from a data frame

- Select all values of "d2" for which "fact" is "B"

```
> df[ df$fact == "B", "d2" ]
[1]  5.555556  6.111111  8.333333  9.444444 10.000000
```

- Select all values of "d1" for which "fact" is "B " and "d2" > 7

```
> df[ (df$fact == "B" & df$d2 > 7), "d1" ]
[1]  7  9 10
```

- Select all values of "d3" for which "fact" is "A " or "d2" < 6

```
>df[ (df$fact == "B" | df$d2 < 6), "d3" ]
[1] 10  9  8  4  2  1
```

```
> df
   d1       d2 fact d3
1   1 5.000000    A 10
2   2 5.555556    B  9
3   3 6.111111    B  8
4   4 6.666667    A  7
5   5 7.222222    A  6
6   6 7.777778    A  5
7   7 8.333333    B  4
8   8 8.888889    A  3
9   9 9.444444    B  2
10 10 10.000000   B  1
```

Exercise

- **Import students.csv into a variable (call it data)**
- **Extract the weight of women only in a new variable**
- **Extract the weights of the people who weight more than 80 kilos**
- **Extract the entries of men who weight more than 80 kg (you can use the "&" operator to include two conditions)**

If you do not know what to do:

1.Extract the weight of women only in a new variable
2.Extract the weights of the people who weight more than 80 kilos
3.Extract the entries of men who weight more than 80 kg
[you can use the "&" operator to include two conditions]