ESCI upf. International Business

P1 | Basic tools for data visualization

Introduction

Marta Coronado Zamora and Adrià Auladell
20 September 2024

Keep in touch

Adrià Auladell
adria.auladell@ibe.upf-csic.es
Institut de Biologia Evolutiva (UPF-CSIC)

Practical session dynamics

Content (P1-P5)

- Introduction
- Exercises - complete and submit to aul@-ESCI
- Project (divided in to 2 assignments)

Interactive R documents

`R code` can be executed within RStudio!

```
value ← 2
value + 3
```

```
## [1] 5
```

Get started!

Tools for data visualization

Type of tools

Two main types:

- Graphical user interface (GUI)

  Many examples: Perseus computational platform, Cytoscape, Blast2GO, Gephi, ...

- Code-based

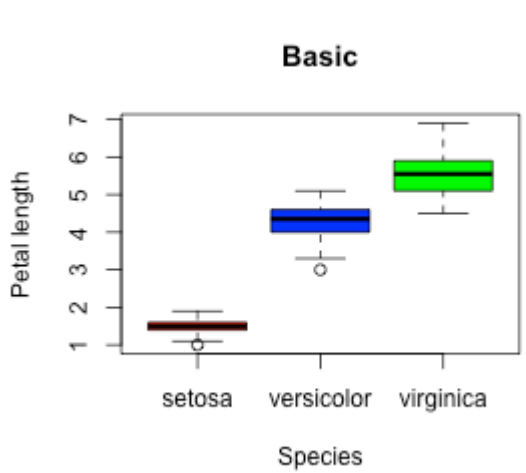
  R (and other computer languages)

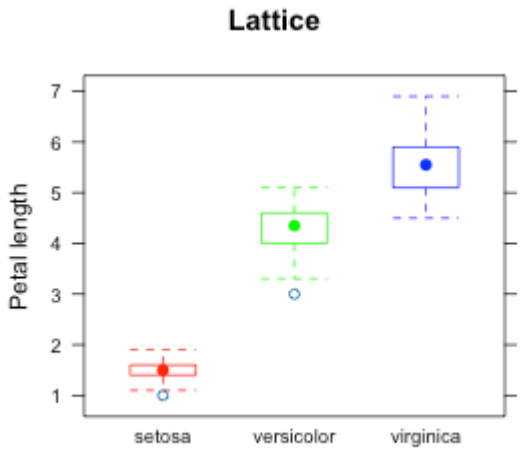
Wide range...

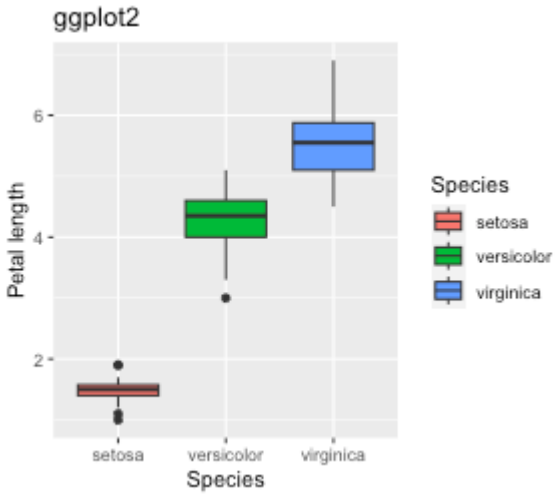
❓ Question
What prons and cons do you think GUI tools have in comparison to code-based?

Visualization libraries in R

- `base`
- `grid`: `lattice` and `ggplot2`



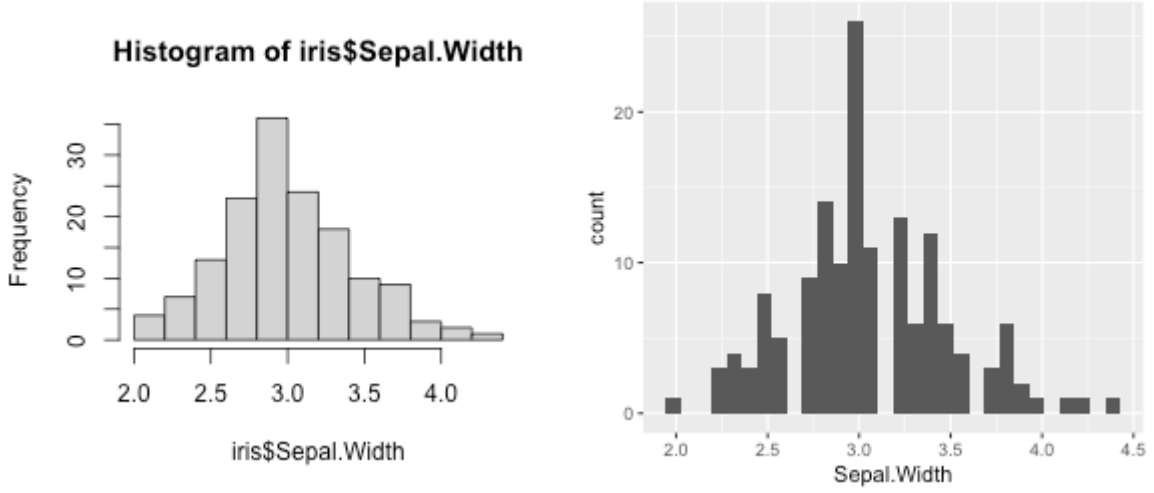




❓ Question
Describe the graphics. In your opinion, which do you think is the simplest? and the most complex? do you think the code to generate the figures reflect the complexity?

Visualization libraries in R

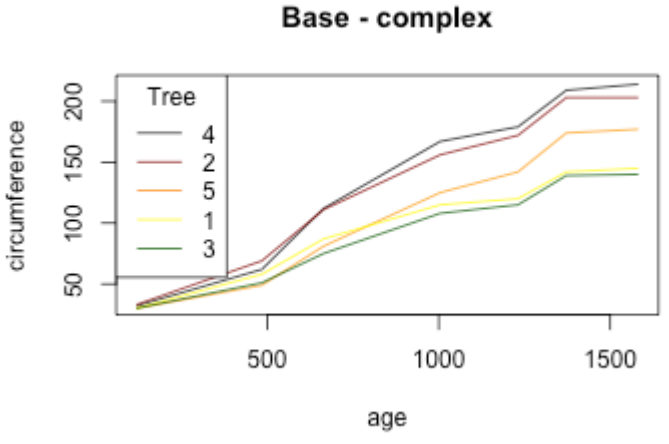
```
# base
hist(iris$Sepal.Width)

# ggplot2
ggplot(iris, aes(Sepal.Width)) +
  geom_histogram()
```

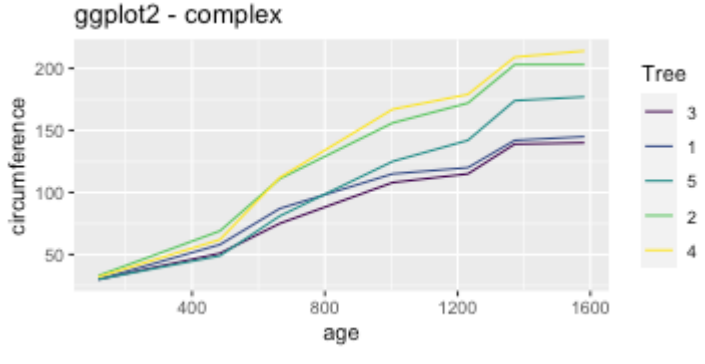


Visualization libraries in R

```
# base
plot(circumference ~ age,
     data=Orange[Orange$Tree %in% "4", ], type =
     main = "Base - complex")
points(circumference ~ age, col="darkred",
       data=Orange[Orange$Tree %in% "2", ], type
points(circumference ~ age, col="orange",
       data=Orange[Orange$Tree %in% "5", ], type
points(circumference ~ age, col="yellow",
       data=Orange[Orange$Tree %in% "1", ], type
points(circumference ~ age, col="darkgreen",
       data=Orange[Orange$Tree %in% "3", ], type
legend("topleft",
       c("4", "2", "5", "1", "3"), title="Tree",
       col=c("black", "darkred", "darkorange", "
       lty=c(1, 1, 1, 1, 1))
```

```
# ggplot2
ggplot(Orange, aes(age, circumference,
      colour = Tree)) + geom_line() +
      labs(title = "ggplot2 - complex")
```

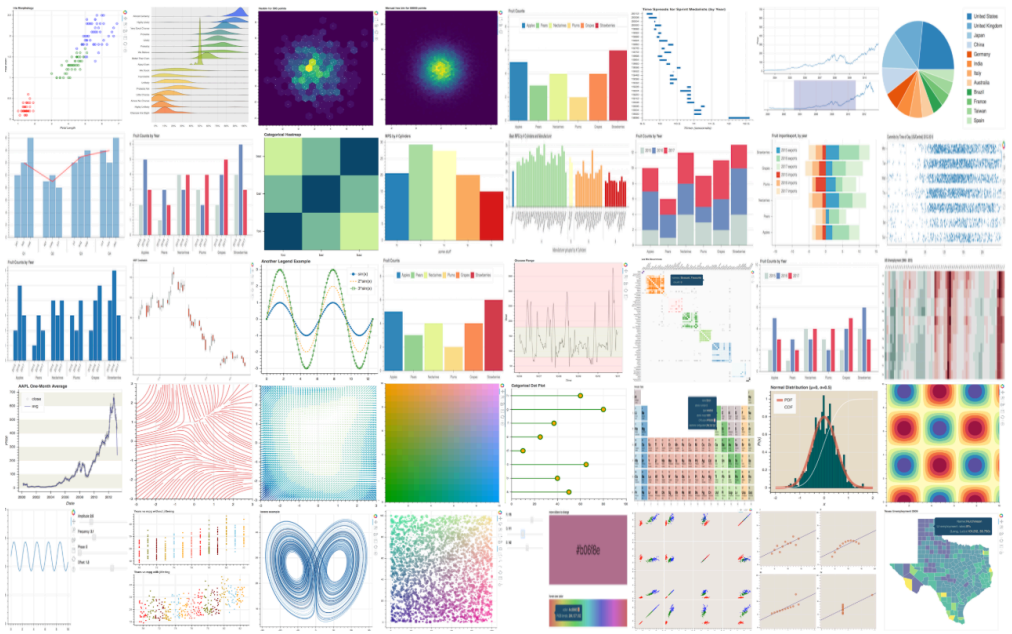




Other visualization libraries

(Outside our scope)

- Python
  - `matplotlib`, `seaborn`
  - `Bokeh`, `pygal`
- Java: `Processing`
- Javascript: `D3.js`



Basic R knowledge

Installing a package

```
# Download and install a package from CRAN
install.packages("ggplot2")

# Download and install a package from GitHub(you need the devtools library installed)
devtools::install_github("yihui/xaringan")
```

Loading a package

```
# Load the library to the current session
library("ggplot2")
library("xaringan")
```

Loading data

```
# Loading a tab-separated file with a header
data ← read.table("data.txt", header = TRUE, sep = "\t")
```

Data types and structures

Main data types (other will not be discussed: *complex* and *raw*):

- **Logical**: can only take on two values: true (`TRUE`, `T`) or false (`FALSE`, `F`)
- **Numeric**: real or decimal (`2`, `15.5`)
- **Integer**: `2L` (the `L` tells R to store this as an integer)
- **Character**: any type of character or number (`"a"`, `"swc"`, `"2"`)

ℹ To know the data type, you can use the `class()` function.

```
type_list ← list(TRUE, 1.2, 10L, "a")
sapply(type_list, class)
```

```
## [1] "logical"   "numeric"   "integer"   "character"
```

Data types and structures

Elements of the previous data types may be combined to form data structures. Main structures:

- **Vector**: collection of elements that holds data of a single data type
- **Matrix**: vector with dimensions (the number of rows and columns)
- **Factor**: to deal with categorical variables
- **List**: a special type of vector where each element can be a different type
- **Data Frame** ★: a special type of list where every element of the list has same length

```r
# A vector x of mode numeric
x ← c(1, 2, 3)

# A vector y of mode logical
y ← c(TRUE, TRUE, FALSE, FALSE)

# A vector z of mode character
z ← c("Sarah", "Tracy", "Jon")
```

Data types and structures

Elements of the previous data types may be combined to form data structures. Main structures:

- **Vector**: collection of elements that holds data of a single data type
- **Matrix**: vector with dimensions (the number of rows and columns)
- **Factor**: to deal with categorical variables
- **List**: a special type of vector where each element can be a different type
- **Data Frame** ★: a special type of list where every element of the list has same length

```
matrix22 ← matrix(
  c(1, 2, 3, 4),
  nrow = 2,
  ncol = 2)
matrix22
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
```

Data types and structures

Elements of the previous data types may be combined to form data structures. Main structures:

- **Vector**: collection of elements that holds data of a single data type
- **Matrix**: vector with dimensions (the number of rows and columns)
- **Factor**: to deal with categorical variables
- **List**: a special type of vector where each element can be a different type
- **Data Frame** ★: a special type of list where every element of the list has same length

```
factor_vector ← as.factor(c("rna", "dna", "dna", "rna"))
factor_vector
```

```
## [1] rna dna dna rna
## Levels: dna rna
```

```
str(factor_vector)
```

```
##  Factor w/ 2 levels "dna","rna": 2 1 1 2
```

Data types and structures

Elements of the previous data types may be combined to form data structures. Main structures:

- **Vector**: collection of elements that holds data of a single data type
- **Matrix**: vector with dimensions (the number of rows and columns)
- **Factor**: to deal with categorical variables
- **List**: a special type of vector where each element can be a different type
- **Data Frame** ⭐: a special type of list where every element of the list has same length

```
x ← list(1, "a", TRUE, 1+4i)
x
```

```
## [[1]]
## [1] 1
##
## [[2]]
## [1] "a"
##
## [[3]]
## [1] TRUE
##
## [[4]]
## [1] 1+4i
```

Data types and structures

Elements of the previous data types may be combined to form data structures. Main structures:

- **Vector**: collection of elements that holds data of a single data type
- **Matrix**: vector with dimensions (the number of rows and columns)
- **Factor**: to deal with categorical variables
- **List**: a special type of vector where each element can be a different type
- **Data Frame** ⭐: a special type of list where every element of the list has same length

```
dat ← data.frame(id = letters[1:10], x = 1:10, y = 11:20)
dat
```

```
##    id  x  y
## 1   a  1 11
## 2   b  2 12
## 3   c  3 13
## 4   d  4 14
## 5   e  5 15
## 6   f  6 16
## 7   g  7 17
## 8   h  8 18
## 9   i  9 19
## 10  j 10 20
```

Tidy data

Data frames with one observation per row and one variable per column.

```
not_tidy
```

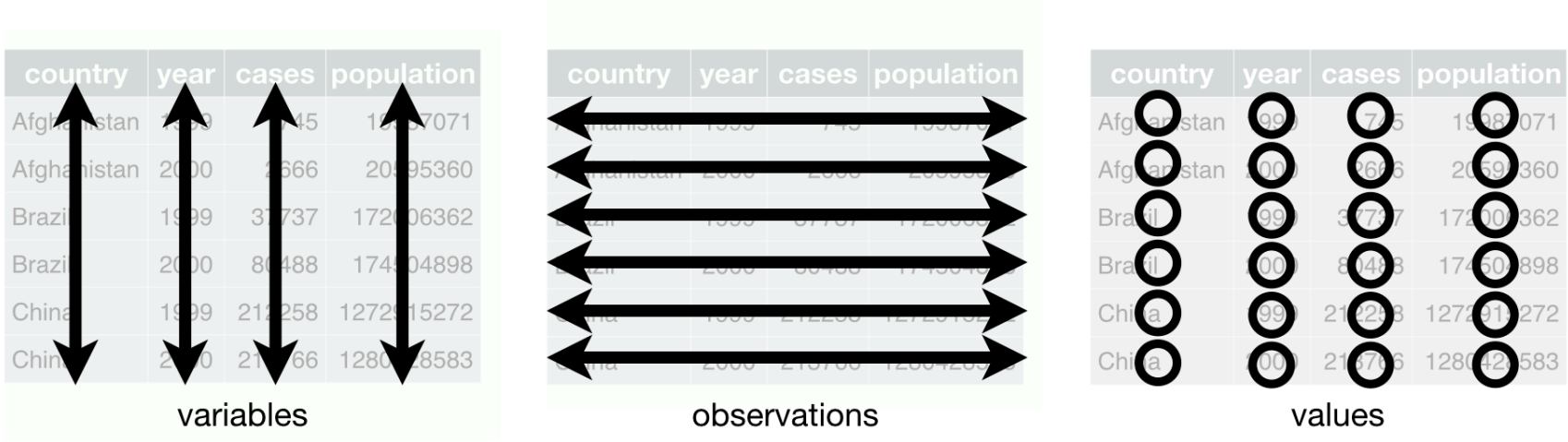
```
##   student course_age First_exam Second_exam
## 1   Marta       1_18          9           6
## 2    Joan       1_19          7           8
```

```
tidy
```

```
##   student course age        exam score
## 1   Marta      1  18  First_exam     9
## 2    Joan      1  19  First_exam     7
## 3   Marta      1  18 Second_exam     6
## 4    Joan      1  19 Second_exam     8
```

Tidy data

Data frames with one observation per row and one variable per column.



Tidy data

Two types of tidy data structures:

1. **Wide format** (most common): in a wide form, the multiple measures of a single observation are stored in a single row.

```
##          Pop Murder Assault UrbanPop Rape
## 1    Alabama   13.2     236       58 21.2
## 2     Alaska   10.0     263       48 44.5
## 3    Arizona    8.1     294       80 31.0
## 4   Arkansas    8.8     190       50 19.5
## 5 California    9.0     276       91 40.6
## 6   Colorado    7.9     204       78 38.7
```

1. **Long format**: each row corresponds to one measure on one observation.

```
## # A tibble: 6 × 3
##   Pop     Measure  Value
##   <chr>   <chr>    <dbl>
## 1 Alabama Murder    13.2
## 2 Alabama Assault  236
## 3 Alabama UrbanPop  58
## 4 Alabama Rape      21.2
## 5 Alaska  Murder    10
## 6 Alaska  Assault  263
```

Tidy data

The function to change from wide to long format has evolved to facilitate its usage:

```
#Before (~circa 2015)
reshape2::melt(
    USArrests,
    id.vars= "Pop",
    variable.name = "Measure",
    value.name = "Value"
    )

# Inbetween: tidyr::gather

#After (~circa 2023)
tidyr::pivot_longer(
  USArrests,
  names_to = "Measure",
  values_to = "Value",
  cols = -Pop
  )
```

Getting help ❓

- `?read.table`, `?str`, `?as.factor`
- Press F1 (in RStudio)
- Stack Overflow (`R`, `ggplot2`)
- Ask your classmates or your teacher

Exercise: describe a data set

Read the file in this link, ensure it has a tidy and long format and indicate the data type of each variable.

Practice

Introduction to `ggplot2`

- Open the document `P1_exercises.Rmd` in RStudio and complete the exercises.
- Upload the completed document to Aul@-ESCI at the end of the session.

Project

Group project

The project has 3 different parts (A, B and C) divided in two big assignments.

- You can deliver the parts separately to get feedback before submitting the final version
- Each part must be submitted before next practical session
- The first assignment will contain parts A and B
- The second assignment will contain part C
- ~15 minutes in the end of each class devoted to discuss your problems

Project

Group project

Part A

- **1.** Create groups of ~4 people
- **2.** Choose a data set with the following requirements
  - Tabular format (txt, csv, tsv...)
  - More than 80 observations
  - At least 6 variables
  - At least 2 discrete and 3 continuous variables
  - Data with biological meaning
  - Different from the ones chosen by other groups

Project

Group project

- **3.** Describe your data set:
  - Where and why was the information collected?
  - Which is the meaning of each variable?
  - Do the variables have unit? Which one?
  - Does the data set have a long format?
- **4.** Write the code to:
  - Read it into R
  - Reshape the data if necessary into long format
  - Check the variable classes and update them if necessary

Write 3 and 4 in an `R Markdown` document and **submit it before next practical session** (one per group).

ℹ If you need help formatting the R Markdown, ask me for a guide of an introduction to R Markdown.

Data sets from research articles

- "Zika virus evolution and spread in the Americas" (Table S2)
- "Great ape genetic diversity and population history" (Table S1 or S3)
- "Transcriptome and genome sequencing uncovers functional variation in humans". Table with cis eQTLs in EUR (description)
- "Signatures of archaic adaptive introgression in present-day human Populations" (Table S3)
- "The evolutionary history of dogs in the Americas" (Table S1)
- "Ancient genomes document multiple waves of migration in Southeast Asian prehistory" (Table S1)
- "Population-scale long-read sequencing uncovers transposable elements associated with gene expression variation and adaptive signatures in *Drosophila*" (Table S10)
- "Comprehensive characterization of 536 patient-derived xenograft models prioritizes candidates for targeted treatment" (Table S1)
- "Pan-cancer analysis of whole genomes" (Table S1)
- "The genomic basis of copper tolerance in *Drosophila* is shaped by a complex interplay of regulatory and environmental factors" (Table S3)
- "Transposons contribute to the diversification of the head, gut, and ovary transcriptomes across *Drosophila* natural strains" (Chimeric gene-TE transcripts data)