R-Bootcamp: Series 2

Dr. Matteo Tanadini

February 2022

1 Exercise: Graphics I

1.1 Question:

Using the `plot()` function and built-in dataset `swiss` create a graph with the following properties

- do not use the formula interface
- plot ”Fertility” on the y-axis
- plot ”Catholic” on the x-axis
- set the colour of the observations to green
- add a title to the graph

1.2 Question:

Using the `plot()` function and built-in dataset `swiss` create a graph with the following properties

- use the formula interface
- plot ”Education” on the y-axis
- plot ”Agriculture” on the x-axis
- set the colour of the points to any colour of your preference[1]
- plot the observations as filled triangles
- set the axes labels as ”edu” and ”agri”
- set the limits of the x-axis to (-5, 100)
- add a title

Going further (*)

Using the `plot()` function and built-in dataset `swiss` create a graph with the following properties

- use the formula interface
- plot ”Education” on the y-axis
- plot ”Agriculture” on the x-axis
- plot the observations with ”Fertility” equal or greater than 76 with a different colour to all the others
- use the letter ”z” as plotting character
- set the expansion factor to be proportional to the variable ”Infant.Mortality”
- add a title written on two lines

[1] You may want to type `colours()` or google ”R + colours” to find all the available options.

- add a subtitle that contains the expression $\pi * 10 \neq \sqrt(10)$

1.3 Question:

We now want you to familiarise with other basic graphical functions. Create the following graphs.

1. an histogram of the ”Fertility” variable found in `swiss` (hint: use `hist()`)
2. a boxplot of where the ”Sepal.Length” variable found in `iris` is plotted against ”Species” (hint: use `boxplot()`)
3. a graph where all variable of the `swiss` dataset are plotted against each other (hint: use `pairs()`)

---

2 Exercise: Graphics II

2.1 Question:

Consider the built-in dataset `Loblolly` (type `?Loblolly` to get more information). Using the `plot()`, `abline()` and `par()` functions create a graph with the following properties.

- the device region is divided in two part such that a graph can be accommodated on the left and one on the right-hand side
- the first plot on the left is a scatterplot of ”height” against ”age” (i.e. ”height” is on the y-axis)
- the plot on the rights is a boxplot where the ”height” of the trees is plotted against the variable ”Seed”
- dots in the scatterplot must be rendered with the colour orange
- the inner area of the boxplots must be rendered with the colour purple
- the border of the boxplots and the whiskers must be rendered with the colour ”cyan”
- both graphs must have a title
- on the scatterplot there is an horizontal dashed line flat on 40
- on the boxplot graph write some text above the very first boxplot on the left
- all graph titles must be written with the colour magenta. Use the function `par()` function to do that

2.2 Question:

Use the Rstudio interface to export this graph as a pdf file. Save the pdf on your Desktop.

3 Exercise: Graphics III

3.1 Question:

The dataset used in this exercise is `Orthodont`, which is about the growth of 29 children. Load the add-on package {`nlme`} such that the dataset is available (Then type `?Orthodont` if you wish more information about the dataset).

*To better understand the way multipanel conditioning and grouping work, run the following **R**-commands. You may try to guess what the resulting plot will be before running the command.*

```
library(lattice)
library(nlme) ## for the dataset
# ? Orthodont

## 1. No multipanel conditioning, points only
xyplot(distance ~ age, data = Orthodont)
##
## 2. No conditioning, points and a regression line
xyplot(distance ~ age, data = Orthodont,
       type = c("p", "r"))
##
## 3. No conditioning, grouping (Sex), points and a regression line
xyplot(distance ~ age, data = Orthodont, groups = Sex,
       type = c("p", "r"))
##
## 4. multipanel conditining for Sex, points and a regression line
xyplot(distance ~ age | Sex, data = Orthodont,
       type = c("p", "r"))
```

3.2 Question:

We now do the same with the {`ggplot2`} package.

```
library(ggplot2)

## 1. No multipanel conditioning, points only
ggplot(data = Orthodont,
       mapping = aes(y = distance,
                     x = age)) +
  geom_point()
##
## 2. No conditioning, points and a regression line
ggplot(data = Orthodont,
       mapping = aes(y = distance,
                     x = age)) +
  geom_point() +
  geom_smooth(method = "lm")

‘geom_smooth()‘ using formula ’y ~ x’

##
## 3. No conditioning, grouping (Sex), points and a regression line
ggplot(data = Orthodont,
       mapping = aes(y = distance,
                     x = age,
                     group = Sex,
                     colour = Sex)) +
  geom_point() +
  geom_smooth(method = "lm")

‘geom_smooth()‘ using formula ’y ~ x’

##
## 4. multipanel conditining for Sex, points and a regression line
ggplot(data = Orthodont,
       mapping = aes(y = distance,
                     x = age)) +
  geom_point() +
  geom_smooth(method = "lm") +
  facet_wrap(. ~ Sex)

‘geom_smooth()‘ using formula ’y ~ x’
```

3.3 Question:

For the next exercise you can choose whether to work with {`lattice`}, {`ggplot2`} or even try with both.

1. Now try to produce a plot where each person is plotted in a different panel, use points and a regression line in each panel. If you use `ggplot()` make sure that the confidence interval around the regression line is turned off (makes little sense with 4 observations to estimate a CI).
2. than replace points and regression lines with lines connecting points for each Subject. In addition you should add a grid in the background. This comes as a default with `ggplot()` but not with `xyplot()` (you may want to type `?xyplot` for help).
3. finally, modify the previous plot and use different points and lines colours for the two genders.

3.4 Question:

NB: before starting to work on this exercise, read the whole assignment here. Indeed, this exercise involves some collaborative work.

*Using any of the built-in datasets (type `data()` for the whole list) produce a graph with the base **R** functions (i.e. `plot()`, `boxplot()`,...). Then save it as a jpeg file.*

Using the same dataset produce a graph with either `xyplot()` or `ggplot()`.

Exchange the two graphs with another participant in the room and without looking at his/her code try to reproduce the exact same graph.

Finally, discuss with the other participant your solution. Again, you may want to involve in this discussion the course instructors.