

Laboratory 1

1 R and RStudio

R is a statistical computer program for performing statistical analysis introduced in 1996; it is an open source software (unlike Minitab, SPSS etc) that is extensively used for academic purposes.

In R you can program from the prompt line or using a GUI; we will use a graphical user interface: RStudio an open-source software too (used on Linux, Windows or Mac).

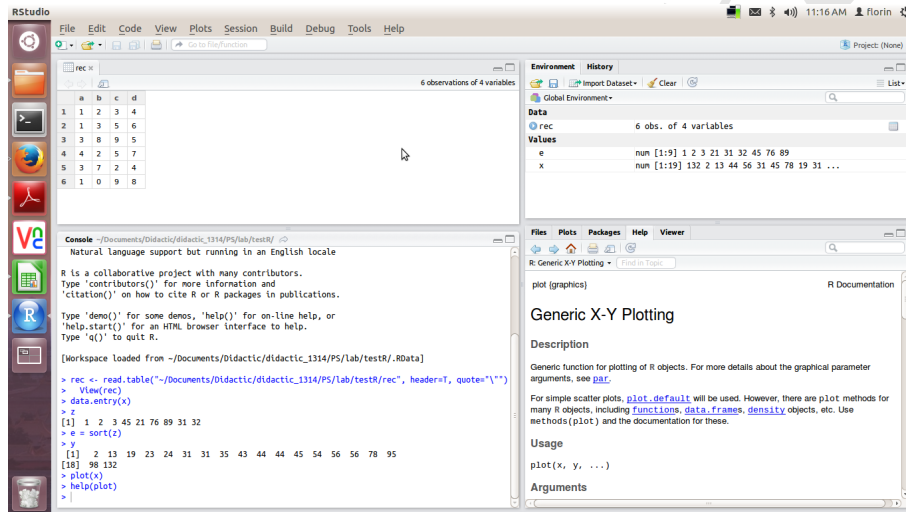


Figure 1: RStudio screenshot

RStudio has (see figure from above) four panels (starting from up left corner clockwise):

- an editing panel in which you can edit and run R scripts (which contain functions and commands) or data files;
- a panel that contains the history and can present the variables;
- a panel containing the help and in which the graphics are displayed;
- one panel that contains the prompt line (here you can execute R commands).

1.1 RStudio session

A session starts with setting of the working directory: [Session](#) → [Set Working Directory](#) → [Choose Directory](#) and will end by saving the workspace (from the dialog window "Save workspace image to `/.RData?`" choose "Save" in [Session](#) → [Save Workspace As](#)).

1.2 Variables and types

In R variables are vectors or matrices. Any variable can be displayed by calling its name. Types are: numerical, character strings (like "a43fdt") and boolean (TRUE or T, and FALSE or F).

Assignment There are two symbols for assignment in R: `=` and `<-` (without any spaces, it is recommend for compatibility with older versions of R).

Create a vector We present three different methods for creating a vector:

- by concatenation using function `c()`;
- as a sequence of consecutive integer numbers;
- or as a sequence in arithmetic progression for which we indicate the first number, the final one, and the length using function `seq()`.

```
> x = c(1, 3, 2, 15, 6, 21, 34, 54, 7)
> x = c(T, T, F, T, F)
> x
[1] TRUE TRUE FALSE TRUE FALSE
> x = -5:13
> x
[1] -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9
[16] 10 11 12 13
> x = seq(-3, 3, length=100)
```

You can access the elements of a vector like follows

```
> x = c(23, 21, 32, 25, 34, 19, 32, 45, 67)
> x[4] # the 4th element
[1] 25
x[2:6] # elements from 2 to 6, including 6
[1] 21 32 25 34 19
x[-3] # all elements except the 3rd
[1] 23 21 25 34 19 32 45 67
```

A vector can be edited using function `data.entry(vector)`.

1.3 Arithmetic operations and predefined functions

We can perform such operations from the prompt line using variables or constants

```
> sin(1)
[1] 0.841471
> log(2)
[1] 0.6931472
> x = 3
> x^2
[1] 9
> exp(x)
[1] 20.08554
```

Vector operations are done componentwise

```
> x = c(1, 3, 2, 15, 6, 21, 34, 54, 7)
> y = c(22, 11, 32, 25, 54, 13, 27, 36, 2)
> x^2
[1] 1 9 4 225 36 441 1156 2916 49
> x + y
[1] [1] 23 14 34 40 60 34 61 90 9
```

R has mathematical and statistical functions for manipulating vectors, matrices, or simple variables.

```
> x <- c(1, 3, 2, 15, 6, 21, 34, 54, 7)
> length(x)
> [1] 9
> sort(x)
> [1] 1 2 3 6 7 15 21 34 54
> sqrt(x)
[1] 1.000000 1.732051 1.414214 3.872983 2.449490
[6] 4.582576 5.830952 7.348469 2.645751
> exp(x)
[1] 2.718282e+00 2.008554e+01 7.389056e+00
[4] 3.269017e+06 4.034288e+02 1.318816e+09
[7] 5.834617e+14 2.830753e+23 1.096633e+03
```

Information about a certain function can be obtained using `help(function_name)` on the prompt line.

1.4 User defined functions

A function could be defined in the command line: suppose that we want to compute the variance of a distribution

```
> variance = function (x, p) {
+ expectation = sum(p*x);
+ variance = sum(p*(x - expectation)^2);
+ return (variance)
+ }
> y = c(23, 32, 31, 27, 27, 33, 25, 21)
> q = c(1/8, 1/16, 1/8, 1/16, 1/8, 1/16, 1/8, 5/16)
> variance(y, q)
```

It's more convenient to write such a function in an R script: **File** → **New File** → **R Script** , and than in the edit panel we write the code

```
variance = function(x, p) {
  expectation = sum(p*x);
  variance = sum(p*(x - expectation)^2);
  return (variance)
}
```

RStudio. After editing the script will be saved (**Ctrl+S**) with a name like "my_script.R" and can be loaded with **Code** → **Source File** (**Ctrl+Shift+O**¹ or **Ctrl+Shift+S**²) or from the command line with `source(script_file)`.

The same script can contain beside the definition of the function. a call to this function using different arguments; for example we can add to our script

```
> y = c(23, 32, 31, 27, 27, 33, 25, 21)
> q = c(1/8, 1/16, 1/8, 1/16, 1/8, 1/16, 1/8, 5/16)
> variance(y, q)
```

¹Ubuntu Linux.

²Windows.

RStudio. Once loaded the script, a function defined in it can be executed from the command line: `variance(y, q)` or from the edit panel like this: we select the required lines and we execute them with **Ctrl+Enter**; the entire script can be executed with **Ctrl+Alt+R**.

A function can be modified in the edit panel or from the command line using `fix(function_name)`

```
> fix(dispersie)
```

1.5 Manipulating data files

Suppose that a file "my_file" (which is in the current working directory, otherwise we must add the relative path to this file) contains an array of data (without any header); we can read the file and transform it in a vector.

```
> x = scan("my_file")
```

If the file contains a header (let us suppose that two columns have names "col1" and "col2"), then we execute

```
> y = read.table("my_file", header = T) # this object contains a header
> x1 = y[['col1']] # this vector contains only the data from coloumn "col1"
> x2 = y[['col2']] # this vector contains only the data from coloumn "col2"
```

We can read also csv (comma separated values) files:

```
> x = read.csv(file="date.csv", header = T)
```

1.6 Iterative and control structures

R has standard structures for iterations and control:

```
> if (condition){
+   statement
+ }else
+ {
+   alternative
+ }
```

```
> for (var in sequence){
+   statement
+ }
```

```
> while (condition){
+   statement
+ }
```

The following function uses such structures:

```
vector_sqrt = function(x) {
  for(i in 1:length(x)) {
    if(x[i] > 0)
      x[i] = sqrt(x[i])
    else)
      x[i] = sqrt(-x[i])
  }
}
```

2 Simulation of random variables (Illustrations of LLN and CLT)

2.1 Remarkable continuous distributions

Solved exercise. Graphically represent the density of the Exponential distribution, $Exp(\lambda)$ ($\lambda > 0$).

This distribution is null for negative values of the real axis, hence it will be enough to represent it on the positive semi-axis (we will use in fact an interval of type $[0, a]$).

```
density_exponential = function(lambda, n, a) {  
  x = seq(0, a, n);  
  y = dexp(x, lambda);  
  plot(x, y, type = 'l');  
}
```

Exercise to work.

1.1. Write three functions that graphically represent the densities of the following distributions:

- (a) $Gamma(\alpha, \lambda)$.
- (b) $Student(r)$.
- (c) $N(\mu, \sigma^2)$.

2.2 The Law of Large Numbers (LLN).

Let X_i , $1 \leq i \leq n$, be a sequence of independent and identically distributed random variables, their mean is

$$\bar{x}_n = \frac{X_1 + X_2 + \dots + X_n}{n},$$

Then, according to LLN, $\bar{x}_n \rightarrow \mu$, where $\mu = \mathbb{E}[X_i]$, $\forall 1 \leq i \leq n$.

Solved exercise. Verify the LLN by using the sequence of random variables $X_i : Poisson(\lambda)$, $1 \leq i \leq n$. (We know that $\mathbb{E}[X_i] = \lambda$.)

```
LLN_Poisson = function(lambda, n) {  
  sum = 0;  
  for(i in 1:n) {  
    u = rpois(1, lambda);  
    sum = sum + u;  
  }  
  return(sum/n);  
}
```

A more simple (and faster?) variant:

```
LLN_Poisson = function(lambda, n) {  
  return(mean(rpois(n, lambda)));  
}
```

Solved exercise. Check the LLN by using the sequence of random variables $X_i : Gamma(\alpha, \lambda)$, $1 \leq i \leq n$. (We know that $\mathbb{E}[X_i] = \alpha/\lambda$.)

We employ here just the faster variant:

```

LLN_Gamma = function(alfa, lambda, n) {
  return(mean(rgamma(n, alfa, lambda)));
}

```

Exercises to work.

2.1. Write a function that has to verify the LLN by using the sequence of random variables

(a) $X_i : Exponential(\lambda)$, $1 \leq i \leq n$. (We know that $\mathbb{E}[X_i] = 1/\lambda$.)

(b) $X_i : B(m, p)$, $1 \leq i \leq n$. (We know that $\mathbb{E}[X_i] = mp$.)

2.2 Solve the same exercise for the sequence of random variables $X_i : Student(r)$, $1 \leq i \leq n$. (We know that $\mathbb{E}[X_i] = 0$.) Compare the results with the exact values for the following parameters: $n \in \{1000, 10000, 100000, 1000000\}$ and $r \in \{2, 3, 4, 5\}$.

2.3 The Central Limit Theorem (CLT).

Let X_i , $1 \leq i \leq n$, be a sequence of independent and identically distributed random variables: $\mathbb{E}[X_i] = \mu$ and $Var[X] = \sigma^2$, $\forall 1 \leq I \leq n$. Let

$$\bar{x}_n = \frac{X_1 + X_2 + \dots + X_n}{n},$$

be their mean, then, according to CLT, \bar{x}_n (for large values of n) follows the distribution of $N(\mu, \sigma^2/n)$.

By using the standardization of the sample mean we get

$$\frac{\bar{x}_n - \mu}{\sigma/\sqrt{n}} : N(0, 1).$$

Solved exercise. Verify the CLT by using the sequence of random variables $X_i : Poisson(\lambda)$, $1 \leq I \leq n$. (We know that $\mathbb{E}[X_i] = \lambda$ and $Var[X_i] = \lambda$.)

The CLT says that

$$P\left(\frac{\bar{x}_n - \mu}{\sigma/\sqrt{n}} \leq z\right) \cong P(Z \leq z),$$

where $z \in \mathbb{R}$ and $Z : N(0, 1)$ - the standard normal law. The probability from the right is $pnorm(z)$, while the probability from the left can be approximated like this: take a large number,

N , of independent simple random samples of the same size n , $\left(X_i^k\right)_{i=1, n}^{k=1, N}$, and compute

$$P^N(z) = \frac{|\{k : \bar{x}_n^k \leq z\sigma/\sqrt{n} + \mu\}|}{N}.$$

(X_i^k are variates of the given distribution - say Poisson, for our exercise). Then, compare this probability with $pnorm(z)$.

```

CLT_Poisson = function(lambda, n, N, z) {
  expectation = lambda;
  st_dev = sqrt(lambda);
  upper_bound = z * st_dev/sqrt(n) + expectation;
  sum = 0;
  for(i in 1:N) {
    x_n = mean(rpois(n, lambda));
    if(x_n ≤ upper_bound) {
      sum = sum + 1;
    }
  }
  return(sum/N);
}

```

Remark: n must be at least 30; e. g., $n = 30$, $N = 10000$, $z \in \{0, 1, 1.5, 2\}$.

Exercises to work.

- 3.1. Write a function that has to verify the CLT by using the sequence of random variables $X_i : \text{Exponential}(\lambda)$, $1 \leq i \leq n$. (We know that $\mathbb{E}[X_i] = 1/\lambda$, $\text{Var}[X_i] = 1/\lambda^2$.)
- 3.2 Solve the same exercise for the sequence of random variables $X_i : \text{Gamma}(\alpha, \lambda)$, $1 \leq i \leq n$. (We know that $\mathbb{E}[X_i] = \alpha/\lambda$ and $\text{Var}[X_i] = \alpha/\lambda^2$.) Choose $n = 50$, $N \in \{5000, 10000, 20000\}$, and $z \in \{-1.5, 0, 1.5\}$.