

A short introduction to R

Kauê de Sousa

2021

Content

- R
- Data structures
- Creating objects and assigning values
- Indexing
- R packages
- Error and Warning messages
- Further reading and learning

Lecture

This lecture is available on [Youtube](#)

R

R

A free, open source programming language and software environment for statistical computing and graphics

Software for data science:

- experiment/survey design
- data retrieval
- data wrangling
- data analysis
- reporting

A programming language, so we can

- use existing functions to code up our data science tasks
- write new functions for customized/novel tasks

[1] [Click here to download R](#)

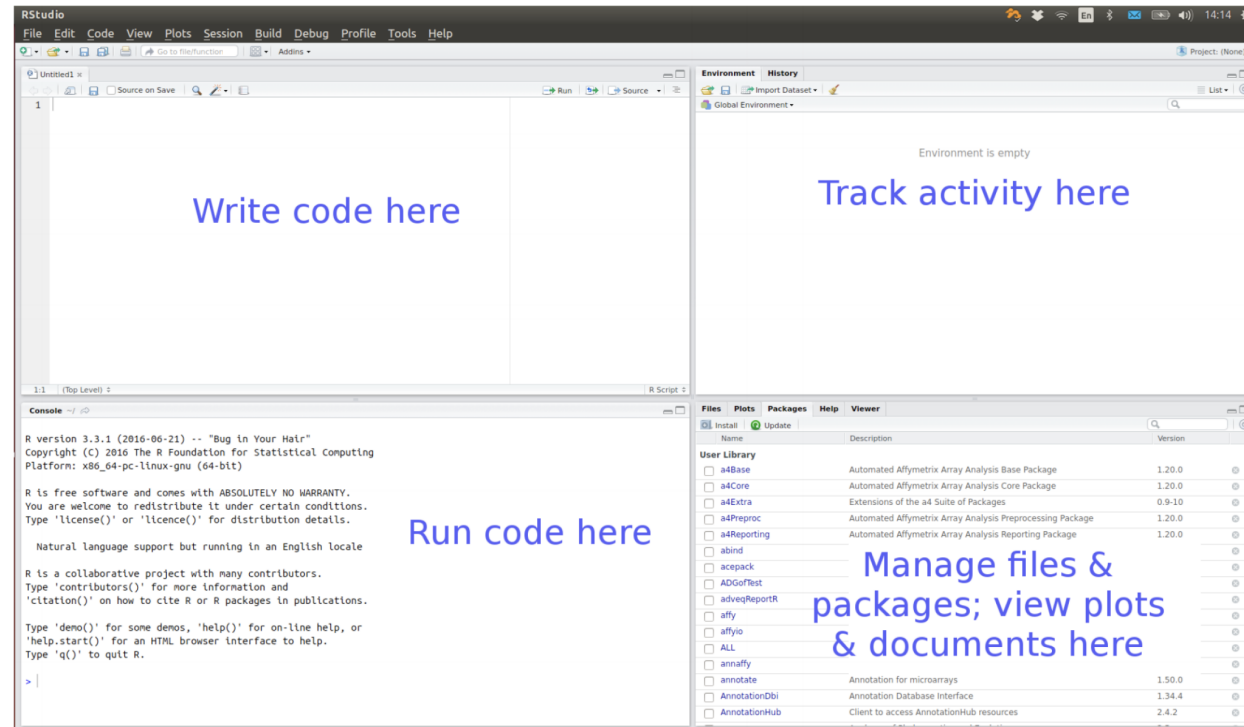
R code-along

We can type commands directly into the R console

```
3 + 4  
?"+" #look up help for "+"  
x <- 3 + 4  
y <- log(x)  
ls() # list of objects in the current workspace  
data() # find out what standard data sets there are  
plot(iris) # plot Fisher's iris data
```

RStudio

An integrated development environment for R



[1] [Click here to download RStudio](#)

Data structures

Data structures

R is a vector based language

Data structures are the building blocks of code. In R there are four main types of structure:

- vectors
- matrices and arrays
- lists
- data frames

Vectors

A single number is a special case of a numeric vector. Vectors of length greater than one can be created using the concatenate function `c()`.

The elements of the vector must be of the same type, common types are integer, numeric, character, factor and logical

```
c(1, 2, 3) # an integer
```

```
c("red", "yellow", "green") # a vector with characters (strings)
```

```
c(TRUE, FALSE, TRUE) # a logical
```

R is a case sensitive language, for example, TRUE is different than true or True

Vectors

Elements of a vector are separated to each other with a comma

```
c(1, 4, 8)
```

Numbers are native elements and are recognised by R without the need of quotation mark ". To assign strings (characters) to a vector you need to wrap each element with a quotation mark.

```
c("apple", "banana", "orange", 1, 2)
```

Missing values

Missing values (of any type) are represented by logical constant NA.

```
c("apple", "banana", "orange", NA)
```

The first vector below is different from the second vector because the first have a length 3 (three elements inside the vector), while the last has a length 4 (four elements), even though the last element is non-available. We can check the length of a vector with the function `length()`

```
length(c("apple", "banana", "orange"))
```

```
## [1] 3
```

```
length(c("apple", "banana", "orange", NA))
```

```
## [1] 4
```

Missing values

The function `is.na()` helps in finding the NAs in a vector. The function returns a logical vector where TRUE indicates the NAs and FALSE the available element.

```
is.na(c("apple", NA, "banana", "orange", NA))
```

```
## [1] FALSE TRUE FALSE FALSE TRUE
```

Lists

Lists are the R objects which contain elements of different types like numbers, characters, vectors or even another list inside it. List is created using `list()` function. And can contain elements of different lengths.

Each vector larger than 1 is concatenated with the function `c()`

Each element in the list is separated to each other with a comma ,

```
list(c(1, 2, 3, 4, 5),  
      c("A", "B", "C", NA, "S"),  
      TRUE)
```

```
## [[1]]  
## [1] 1 2 3 4 5  
##  
## [[2]]  
## [1] "A" "B" "C" NA  "S"  
##  
## [[3]]  
## [1] TRUE
```

Data frames

A data frame is the most common way of storing data in R and, generally, is the data structure most often used for data analyses. In other words, a data frame is a list of equal-length vectors and can be created with the function `data.frame()`

```
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species      : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1 1 1 1 ...
```

Here Species is a factor, a special data structure for categorical variables.

Click [here](#) to read more about factors.

Creating a data frame

```
data.frame(x = c(1:3),  
           y = c(3:5),  
           z = c("red", "blue", "black"),  
           w = c(TRUE, FALSE, TRUE))
```

```
##      x y      z      w  
## 1 1 3    red  TRUE  
## 2 2 4   blue FALSE  
## 3 3 5  black  TRUE
```

Creating objects and assigning values

Creating objects and assigning values

Objects are assigned using `<-`, an arrow to the left formed out of `<` and `-`. An equal sign `=` can also be used. Creating objects is helpful because R keeps them in the "Global Environment" for subsequent utilization.

```
mydf <- data.frame(x = c(1:3),  
                  y = c(3:5),  
                  z = c("red", "blue", "black"),  
                  w = c(TRUE, FALSE, TRUE))
```

```
mydf
```

```
##   x y   z   w  
## 1 1 3  red TRUE  
## 2 2 4  blue FALSE  
## 3 3 5 black  TRUE
```

Some tips in creating objects and assigning values

- Try to be objective and name your object clearly
- Don't use dots "." to separate name, as dots are used to parse **methods** (which is an advanced topic)

```
my_vector <- c("a", "y", "z") # good name
```

```
my.vector <- c("a", "y", "z") # not a good name
```

- Certain names should not be used to name objects (e.g. c, t, df, matrix, data), because these are names of functions in R. Type these on the Console and see what happens. Instead use, my_matrix, my_data, my_list or something similar.

Indexing

Indexing

Indexing is a way to access the element(s) in an R object using its position in the vector. Indices in R starts with 1 (the first element in the vector) and goes until the value that represents the length of the vector.

```
x <- c(3, 5, 6, 9, 10)
```

```
x[1] # access the element of position one
```

```
## [1] 3
```

```
x[4:5] # access the elements four and five of the vector
```

```
## [1] 9 10
```

```
x[c(2,4)] # access the elements two and four of the vector
```

```
## [1] 5 9
```

Indexing can also be done with `match()`, `which()` and `%in%`, but this is a bit more advanced. Click [here](#) to read more about it.

Indexing a data frame

Data frames are objects with two dimensions. The first represents the rows and the second represents the columns.

```
mydf[1] # take all the elements from the first column
```

```
##      x  
## 1 1  
## 2 2  
## 3 3
```

```
mydf[1,] # take all the elements from the first row
```

```
##      x y      z      w  
## 1 1 3 red TRUE
```

```
mydf[c(1,3), 3:4] # take the elements from rows one and three, and columns three and four
```

```
##          z      w  
## 1      red TRUE  
## 3     black TRUE
```

Indexing a data frame

An element of a data frame can be also indexed using its column name (the safer choice!)

```
# access column named x in mydf  
# this returns a data frame with one column  
mydf["x"]
```

```
##      x  
## 1  1  
## 2  2  
## 3  3
```

```
# dollar sign $ can be used to access elements inside a data frame or a list  
# this returns a vector  
mydf$x
```

```
## [1] 1 2 3
```

Indexing a data frame

An element of a data frame can be also indexed using its column name (the safer choice!)

```
# access lines one and two, and columns x and z in mydf  
mydf[1:2, c("x", "z")]
```

```
##      x      z  
## 1 1 red  
## 2 2 blue
```

R packages

R packages

A collection of R functions, compiled code and sample data. They are stored under a directory called "library" in the R environment

Most day-to-day work will require at least one contributed package.

The Comprehensive R Archive Network (CRAN) is where most of the packages are

To install a package from CRAN we use the function `install.packages()`

```
install.packages("ggplot2")
```

R packages, developer version

Develop a R package is often an ongoing project. Once a package's version is stable enough, the developer publishes it on CRAN.

Currently, most R packages under development are deposited on [GitHub](#). To install a developing version from GitHub we use the function `install_github()` from the R package "remotes"

```
install.packages("remotes")  
remotes::install_github("agrdatasci/gosset")
```

Library

You just need to install a R package once. Unless you change the R version or change the location of the library

To call a package we use the function `library()`. This should be done every time that you start a new R session. Packages names can be wrapped with quotation marks or not. The **best practice** is to use quotation marks.

```
library("ggplot2")  
library(gosset)
```

Error and Warning messages

Error messages

An error occurs when we try to run a function but provide (in some way) the wrong input(s). By consequence the object that we try to create is not generated and R returns an "error message".

```
cbind(iris, airquality)
```

```
> cbind(iris, airquality)
Error in data.frame(..., check.names = FALSE) :
  arguments imply differing number of rows: 150, 153
>
```

This error occurred because we tried to combine the data frames "iris" and "airquality" into a single object. But they have different dimensions (number of rows).

Warning messages

R returns a warning message to keep you aware of an unexpected outcome of the function that can affect the object that is created.

```
cbind(c(1:4), c(5:10))
```

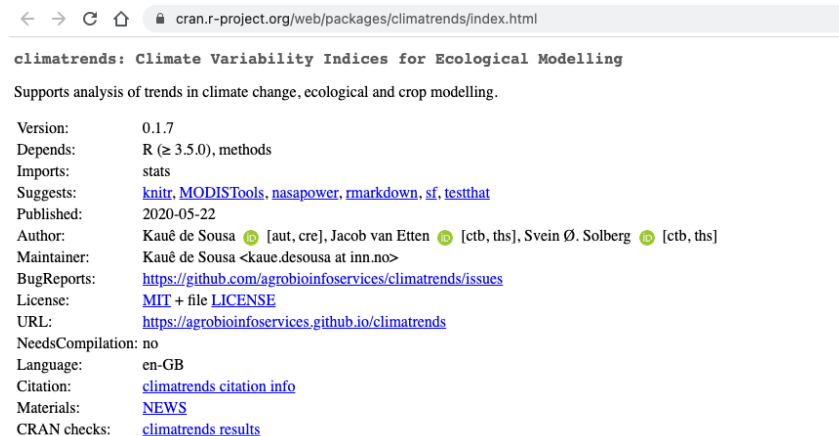
```
> cbind(c(1:4), c(5:10))
      [,1] [,2]
[1,]    1    5
[2,]    2    6
[3,]    3    7
[4,]    4    8
[5,]    1    9
[6,]    2   10
Warning message:
In cbind(c(1:4), c(5:10)) :
  number of rows of result is not a multiple of vector length (arg 1)
>
```

This message warns us that we are combining two vectors with different lengths. The first has a length of four and the second a length of six. R runs the command, repeats the elements of index 1 and 2 from the first vector, and returns the matrix with a warnings message.




Tips to deal with errors and warnings

Error and warning messages from native functions in R are often not so informative. But don't be afraid, most of them are, in some way, already reported on the web. Also the R community is very friendly and responsive. Here some tips:

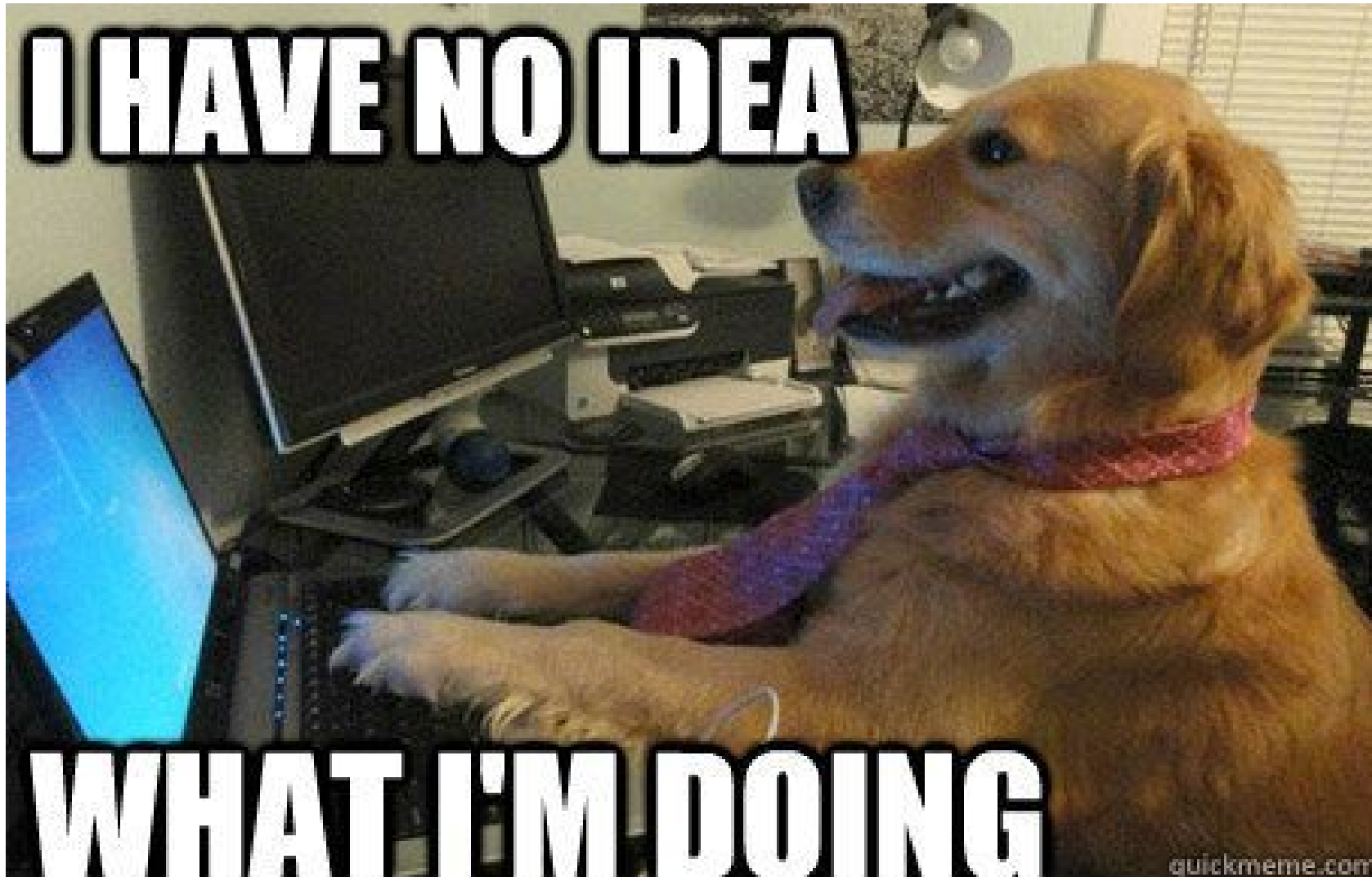
- Don't panic and read carefully the error/warning message
- Copy the message and search for it on Google. Is very likely that you will find someone that went through the same error.
- **Stack Overflow** is where most of the errors are solved.
- On Twitter, look for help using the hashtag **#Rstats**
- Lastly, try to contact the package's developer through the channel informed in the package's home page on CRAN (BugReports).



The screenshot shows the CRAN page for the 'climatrends' package. The browser address bar displays 'cran.r-project.org/web/packages/climatrends/index.html'. The page title is 'climatrends: Climate Variability Indices for Ecological Modelling'. Below the title, it states 'Supports analysis of trends in climate change, ecological and crop modelling.' The package details are listed in a table-like format with labels on the left and values on the right. The values include version numbers, dependencies, import packages, suggested packages with links, publication dates, author and maintainer names with ORCID icons, bug reports link, license, URL, and CRAN check results.

Version:	0.1.7
Depends:	R (≥ 3.5.0), methods
Imports:	stats
Suggests:	knitr , MODISTools , nasapower , rmarkdown , sf , testthat
Published:	2020-05-22
Author:	Kauê de Sousa  [aut, cre], Jacob van Etten  [ctb, ths], Svein Ø. Solberg  [ctb, ths]
Maintainer:	Kauê de Sousa < kaue.desousa@inn.no >
BugReports:	https://github.com/agrobioinfoservices/climatrends/issues
License:	MIT + file LICENSE
URL:	https://agrobioinfoservices.github.io/climatrends
NeedsCompilation:	no
Language:	en-GB
Citation:	climatrends citation info
Materials:	NEWS
CRAN checks:	climatrends results

R is a language with a steep learning curve. If you have no idea what you are doing, don't worry I also felt that way at the beginning. The tip is practice and look for help in the web forums.



Further reading and learning

R Basics

Professor Norman Matloff from UC Davis prepared "fasteR" a R tutorial for people with no background in coding. Read it here <https://github.com/matloff/fasteR>

Advanced learning

The 2nd Edition of "Advanced R" by Hadley Wickham provides great insights and exercises to improve your skills in R. Read it here <https://adv-r.hadley.nz/index.html>

The e-book "Introduction to Data Science" by Rafael A. Irizarry also provides great insights. Read it here <https://rafalab.github.io/dsbook/>

RStudio have a series of recorded webinars that can help you in learning more about R <https://rstudio.com/resources/webinars/>

Thank you!



Alliance



@desousakaue



k.desousa@cgiar.org

[Back to the main page](#)