Introduzione a R

Giornata 1

Corsi ARCA - @DPSS

Filippo Gambarota

Primi passi con R

Installazione

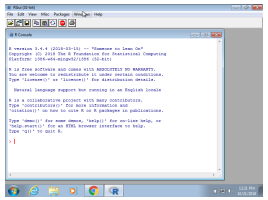
Per l'installazione trovate le indicazioni nella sezione Installare R e RStudio del libro. In generale i passaggi sono:

- scaricare R e installare **R** per il vostro sistema operativo
- scaricare e installare **RStudio**

Come si presenta R

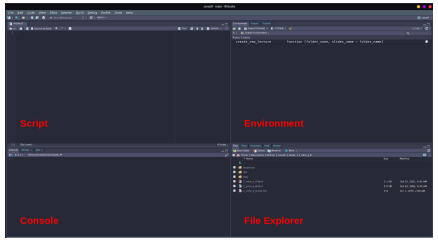
Console

```
put_image("R_console.png")
```



RStudio

```
knitr::include_graphics("img/rstudio.png")
```



I primi passi in R

R come calcolatrice

In R è possibile effettuare tutte le operazioni matematiche e algebriche dalle più semplici alle più avanzate

```
names_function <- c("Addizione", "Sottrazione", "Moltiplicazione", "Divisione",
                    "Resto della divisione", "Divisione intera", "Potenza",
                    "Valore assoluto","Segno di un'espressione", "Radice quadrata",
                    "Logaritmo naturale", "Esponenziale", "Funzioni trigonometriche",
                    "Fattoriale", "Coefficiente binomiale")

math_operators <- data.frame(
    formula = c(sprintf("x %s y", c("+", "-", "*", "/", "%%", "%/%", "^")),
                sprintf("%s(x)", c("abs", "sign", "sqrt", "log", "exp")),
                paste0(sprintf("%s(x)", c("sin", "cos", "tan", "asin", "acos", "atan")),
                       collapse = "<br />"),
                "factorial(x)", "choose(n, k)"),
    name = names_function,
    example = c(sprintf("> %s <br />[1] %s",
                        c("5 + 3", "7 - 2", "4 * 3", "8 / 3", "7 %% 5", "7 %/% 5",
                          "3 ^ 3","abs(3-5^2)", "sign(-8)", "sqrt(25)", "log(10)", "exp(1)"),
                        c("8", "5", "12", "2.666667", "2", "1", "27","22", "-1", "5",
                          "2.302585", "2.718282")),
                ">sin(pi/2) <br />[1]1 <br />>cos(pi/2) <br />[1]6.123234e-17",
                c(sprintf("> %s <br />[1] %s",
                          c("factorial(6)", "choose(5,3)"),
                          c("720", "10"))))
)
```

Operatori matematici

- Importante considerare l'**ordine delle operazioni** analogo alle regole della matematica: `2 x 3 + 1` prima `2 x 3` e poi `+ 1`. Analogalmente in R:

```
# Senza parentesi

2 * 3 + 1
```

```
## [1] 7
```

```
# Con le parentesi

(2 * 3) + 1
```

```
## [1] 7
```

```
# Con le parentesi forzando un ordine diverso

2 * (3 + 1)
```

```
## [1] 8
```

Operatori relazionali

Gli operatori relazionali sono molto utili dentro le **funzioni**, per **selezionare elementi dalle strutture dati** (vedremo più avanti) e in generale per **controllare** alcune sezioni del nostro codice:

```
3 > 4
```

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

```
3 >= 3
```

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

```
10 < 100
```

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

```
40 == 40
```

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

```
10 != 50
```

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

Operatori logici

Gli operatori logici permettono di **combinare espressioni relazionali** e ottenere sempre un valore `TRUE` o `FALSE`:

```
3 > 4 & 10 < 100
```

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

```
10 < 100 | 50 > 2
```

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

```
!5 > 4
```

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

R e gli oggetti

R e gli oggetti

> “Everything that exists in R is an object” - John Chambers

Il concetto di **oggetto** è fondamentale in R. Essenzialmente tutto quello che possiamo creare o utilizzare in R come un numero, un vettore, dei caratteri o delle funzioni sono creati come oggetti.

R e gli oggetti

- Come creare un oggetto?
- Oggetti e nomi
- Dove viene creato l'oggetto?

Come creare un oggetto?

La creazione di un oggetto avviene tramite il comando `<-` oppure `=` in questo modo: `nome <- oggetto`:

```
x
```

```
## Error in eval(expr, envir, enclos): object 'x' not found
```

```
10 # questo non è un oggetto, non è salvato
```

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

```
x <- 10 # ora il valore numerico 10 è associato al nome "x"

x
```

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

Convenzioni vs regole

Ci sono alcune cose da considerare quando si scrive codice ed in particolare si creano oggetti:

- **alcune modalità sono errate** --> R ci fornisce un messaggio di errore
- **alcune modalità sono sconsigliate** --> funziona tutto ma ci potrebbero essere problemi
- **alcune modalità sono stilisticamente errate** --> funziona tutto, nessun problema ma... anche l'occhio vuole la sua parte

Oggetti e nomi

Il nome di un oggetto è importante sia per l'utente che per il software stesso:

```
1 <- 10 # errore

_ciao <- 10 # errore

mean <- 10 # possibile ma pericoloso

`1` <- 10 # con i backticks si può usare qualsiasi nome ma poco pratico
```

```
## Error: <text>:4:1: unexpected input
## 3:
## 4: _
##    ^
```

```
my_obj <- 10

my.obj <- 10

My_obj <- 10 # attenzione a maiuscole e minuscole
```

Oggetti e nomi (proibiti)

In R ci sono anche dei nomi non solo sconsigliati ma proprio **proibiti** che nonostante siano sintatticamente corretti, non possono essere usati (per ovvie ragioni):

```
mean <- 10 # ok ma sconsigliato

function <- 10
```

```
## Error: <text>:4:10: unexpected assignment
## 3:
## 4: function <-
##              ^
```

```
TRUE <- 4
```

```
## Error in TRUE <- 4: invalid (do_set) left-hand side to assignment
```

```
T <- 2 # attenzione
```

Dove viene creato l'oggetto?

Il legame `nome-oggetto` viene creato nel **global environment** e rimane *salvato* per tutta a sessione di utilizzo. Potete visualizzare l'environment in RStudio oppure usando il comando `ls()`:

```
put_image("obj_environment.png")
```

Non solo numeri (anticipazione)

Non solo numeri (anticipazione)

In R possiamo usare oltre ai numeri (in senso matematico) anche le **stringhe** ovvero parole, lettere intepretate così come sono:

```
"ciao" # stringa formata da 5 caratteri
```

```
## [1] "ciao"
```

```
x <- "ciao" # associo la stringa ad un oggetto

x + 1 # operazioni matematiche con stringhe (ha senso?)
```

```
## Error in x + 1: non-numeric argument to binary operator
```

```
x == "ciao"
```

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

```
x > 10
```

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

Ambiente di lavoro

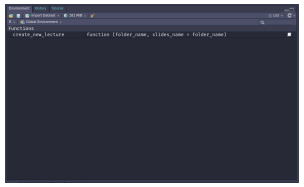
Ambiente di lavoro

- **Environment**
- **Working directory**
- **Packages**

Environment

Il **working environment** è la vostra *scrivania* quando lavorate in R. Contiene tutti gli oggetti (variabili) creati durante la sessione di lavoro.

```
knitr::include_graphics("img/environemnt.png")
```



Working Directory

La working directory è la posizione (cartella) sul vostro PC dove R sta lavorando e nella quale R si aspetta di trovare i vostri file, se non specificato altrimenti

```
knitr::include_graphics("img/working_directory.png")
```

Packages

In R è possibile installare e caricare pacchetti aggiuntivi che non fanno altro che rendere disponibili librerie di funzioni create da altri utenti. Per utilizzare un pacchetto:

- Installare il pacchetto con `install.packages("nomepacchetto")`
- Caricare il pacchetto con `library(nomepacchetto)`

Packages

```
knitr::include_graphics("img/packages.png")
```

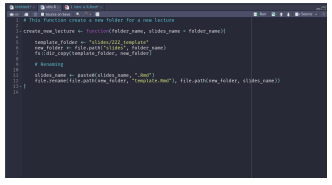
Come lavorare in R

Scrivere e organizzare script

- Lo script è un file di testo dove il codice viene salvato e puo essere lanciato in successione

- Nello script è possibile combinare **codice** e **commenti**

```
knitr::include_graphics("img/script.png")
```



R Projects

Gli `R projects` sono una feature implementata in R Studio per organizzare una cartella di lavoro

- permettono di impostare la **working directory** in automatico
- permettono di usare **relative path** invece che **absolute path**
- rendono più **riproducibile** e **trasportabile** il progetto
- permettono un **veloce accesso** ad un determinato progetto

Come risolvere i problemi ~~nella vita~~ in R

Come risolvere i problemi ~~nella vita~~ in R

In R gli errori sono:

- inevitabili
- parte del codice stesso
- educativi

Resta solo da capire come affrontarli

R ed errori

Ci sono diversi livelli di **allerta** quando scriviamo codice:

- **messaggi**: la funzione ci restituisce qualcosa che è utile sapere, ma tutto liscio
- **warnings**: la funzione ci informa di qualcosa di *potenzialmente* problematico, ma (circa) tutto liscio
- **error**: la funzione non solo ci informa di un **errore** ma le operazioni richieste non sono state eseguite

Come risolvere un errore?

- capire il messaggio
- leggere la documentazione della funzione
- cercare il messaggio su Google
- chiedere aiuto nei forum dedicati

Come risolvere un errore?

- Ogni funzione ha una pagina di documentazione accessibile con `?nomefunzione` o `??nomefunzione`
- Possiamo cercare anche la documentazione del pacchetto
- Possiamo cercare su Google il nome della funzione o l'eventuale messaggio che riceviamo

Come NON fare una domanda

```
put_image("bad_question.png")
```

Hi I'm trying to plot my data using ggplot2 but I'm getting this error I've never seen before... I was searching online and couldn't find anything useful.

This is my code:

```
##PACKAGES

library(ggplot2)
library(viridis)
library(hrbrthemes)

##IMPORT DATA
library(readxl)
TOC_LANG <- read_excel("PHD/LAB/LAB
RESULTS/Jul-2021- TOC/TOC- LAN and GROVE
AV/vario016_TOC_MOD.xlsx",
sheet = "R", range = "B3:E55")

#IMPORT DATA

library(readxl)
TOC_PORTGA <- read_excel("PHD/LAB/LAB
RESULTS/Jul-2021- TOC/TOC- LAN and GROVE
AV/vario016_TOC_MOD.xlsx",
sheet = "R", range = "A59:E105")

View(TOC_LANG)
```

```
put_image("bad_question_after.png")
```