

Modulo Didattico 1: Il concetto di Misura

Test per le organizzazioni

Ottavia M. Epifania
ottavia.epifania@unipd.it

Margherita Calderan
margherita.calderan@unipd.it

Università di Padova

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti
- 6 L'ambiente
- 7 Programmazione entry level

- 1 Il corso
 - Orari
 - Argomenti

2 Introduzione

3 RStudio

4 Operatori

5 Oggetti

6 L'ambiente

Lezioni: (METTIAMO I NOSTRI ORARI REALI)

- Martedì: 10:30-12:30 (orario reale: 10:30 - 12:00)
- Giovedì: 13:00-15:00 (orario reale: 13:30 - 15:00)
- Venerdì: 10:30-12:30 (orario reale: 10:30 - 12:00)

Ricevimento: su appuntamento, prevalentemente online

ottavia.epifania@unipd.it
margherit.calderan@unipd.it

- 1 Il corso
 - Orari
 - Argomenti

2 Introduzione

3 RStudio

4 Operatori

5 Oggetti

6 L'ambiente

Metterei una slide dove spieghiamo a grandi linee cosa vogliamo fare

Ti lascio le slide seguenti perché magari ti sono utili per la parte di R (o anche solo per vedere come suo io quarto per il pdf.)

- 1 Il corso
- 2 Introduzione**
- 3 RStudio
- 4 Operatori
- 5 Oggetti
- 6 L'ambiente
- 7 Programmazione entry level

1 Il corso

2 Introduzione

- Evoluzione e storia di R
- Caratteristiche di R

3 RStudio

4 Operatori

5 Oggetti

6 L'ambiente

7 Programmazione entry level

- Discende da S e S+
- Parte del progetto GNU
- Software libero sotto licenza GNU GPL
- Open Source
- <https://www.r-project.org/COPYING>

1 Il corso

2 Introduzione

- Evoluzione e storia di R
- Caratteristiche di R

3 RStudio

4 Operatori

5 Oggetti

6 L'ambiente

7 Programmazione entry level

Pros

Linguaggio object-oriented

Più dimestichezza nell'analisi dei dati, più conoscenza del dato, modelli più complessi

Permette di addentrarsi sempre di più nei linguaggi di programmazione

Cons

Difficile da imparare (all'inizio)

Non è intuitivo (all'inizio) → se non si ha già una vaga idea di dove partire non si riesce a fare nulla

- 1 Il corso
- 2 Introduzione
- 3 RStudio**
- 4 Operatori
- 5 Oggetti
- 6 L'ambiente
- 7 Programmazione entry level

Tol - master - RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

```
tolTempiModelli.R tolTempi.qmd dataPrep.R  
1 # DATA PREPARATION TOL  
2 # OTTAVIA, MAGGIO 2023  
3 #  
4 rm(list = ls())  
5 library(tubridate)  
6 library(dplyr)  
7  
8 setwd("H:/shortcut-targets-by-id/1740KGGGxve6z7MtNn0OcT3W23DXiim")  
9  
10 name.data.43 = paste0("AdapTo1_43/", # do not change  
11                       "to143_2023_05_08.csv") # change according to  
12 name.data.52 = paste0("AdapTo1_52/", # do not change  
13                       "to152_2023_05_08.csv") # change according to  
14 name.data.45 = paste0("AdapTo1_45/", # do not change  
15                       "to145_2023_05_08.csv") # change according to  
16 name.env = ls()  
17 name.env = name.env[grepl("name.data", name.env)]  
18  
19 for (i in 1:length(name.env)) {  
20   assign(gsub("name.", "", name.env[i]),  
21         read.csv(get(name.env[i]), header = T, sep = ","))  
22 }  
23  
1984 total_time
```

```
Console Terminal Background Jobs  
R 4.2.3 - H:/shortcut-targets-by-id/1740KGGGxve6z7MtNn0OcT3W23DXiim/V/PyrcAssist/  
> ggplot(all.data, aes(x = n_moves, y = as.numeric(preplanning))) + geom_point()  
Warning message:  
Removed 8 rows containing missing values (geom_point).  
> |
```

Environment History Connections Tutorial

R - Global Environment

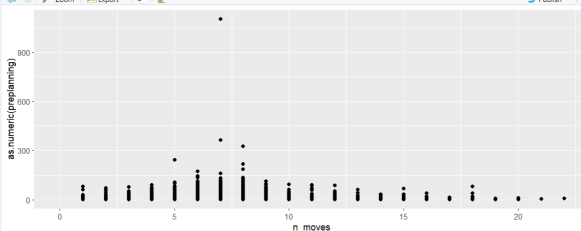
total_time43	378 obs. of 29 variables
total_time45	166 obs. of 44 variables
total_time52	412 obs. of 36 variables
wide.acc	412 obs. of 28 variables

Values

alleged.trials	num [1:3] 20 35 27
i	12L
name_desc	chr [1:3] "sbj_char43" "sbj_char45" "sbj_char52"
name.data.43	"AdapTo1_43/to143_2023_05_08.csv"
name.data.45	"AdapTo1_45/to145_2023_05_08.csv"
name.data.52	"AdapTo1_52/to152_2023_05_08.csv"
name.env	chr [1:3] "data.43" "data.45" "data.52"
names.print	chr [1:12] "accuracy43" "accuracy45" "accuracy52" "execution43" "execution45" ...

Files Plots Packages Help Git Viewer Presentation

Zoom Export



console vs. script

Console

I comandi nella console vengono eseguiti e non salvati

Per eseguire il comando → Invio

L'output è immediato ed appare nella console

console vs. script

Console

I comandi nella console vengono eseguiti e non salvati

Per eseguire il comando → Invio

L'output è immediato ed appare nella console

Script

è possibile salvare gli script con tutti i comandi salvati

Per eseguire il comando → Ctrl + Invio (cmd + Enter)

L'output è restituito nella console

Per passare alla console → ctrl + 2

Per passare allo script → ctr + 1

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori**
- 5 Oggetti
- 6 L'ambiente
- 7 Programmazione entry level

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori**
 - Matematici
 - Insiemistici
 - Tavole di verità
 - Funzioni matematiche di base
- 5 Oggetti

Simbolo	Significato
+	Addizione
-	Sottrazione
*	Moltiplicazione
/	Divisione
^	Potenza

Esempio:

$((4 * 2) - 7) + 2)^3$

Simbolo	Significato
+	Addizione
-	Sottrazione
*	Moltiplicazione
/	Divisione
^	Potenza

Esempio:

```
((4 * 2) - 7) + 2)^3
```

```
[1] 27
```

Simbolo	Significato
<	Minore
>	Maggiore
<=	Minore o uguale
>=	Maggiore o uguale
==	Uguale
!=	Diverso

Esempio:

```
(10 / 2) > 3
```

```
[1] TRUE
```

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori**
 - Matematici
 - Insiemistici
 - Tavole di verità
 - Funzioni matematiche di base
- 5 Oggetti

Simbolo	Significato
&	AND (congiunzione \wedge)
!	NOT (negazione \neg)
	OR (Disgiunzione \vee)

Esempio:

(2 > 3) & (5 < 10)

Simbolo	Significato
&	AND (congiunzione \wedge)
!	NOT (negazione \neg)
	OR (Disgiunzione \vee)

Esempio:

(2 > 3) & (5 < 10)

[1] FALSE

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori**
 - Matematici
 - Insiemistici
 - Tavole di verità
 - Funzioni matematiche di base
- 5 Oggetti

Table 4: Tavole di verità

(a) Congiunzione

a	b	$a \wedge b$
V	V	V
V	F	F
F	V	F
F	F	F

(b) Disgiunzione

a	b	$a \vee b$
V	V	V
V	F	V
F	V	V
F	F	F

(c) Negazione

a	$\neg a$
V	F
F	V

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori**
 - Matematici
 - Insiemistici
 - Tavole di verità
 - Funzioni matematiche di base
- 5 Oggetti

Funzioni matematiche di base

Funzione	Significato
<code>log()</code>	Logaritmo naturale
<code>exp()</code>	Esponenziale
<code>abs()</code>	Valore assoluto
<code>sqrt()</code>	Radice quadrata
<code>round()</code>	Arrotondamento
<code>mean()</code>	Media
<code>min(), max()</code>	Minimo, Massimo
<code>sd(), var()</code>	Deviazione st., Varianza
<code>sum()</code>	Somma

Esempio:

```
abs(round(-1 * (10 / 3), 2))
```

```
[1] 3.33
```

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti**
- 6 L'ambiente
- 7 Programmazione entry level

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti**
 - Variabili, costanti, assegnazione in R
 - Matrici
 - Indicizzare e selezionare
 - Dataframe
 - Liste

Assegnazione e concatenazione

- Assegnazione: `<-` oppure `=`
- Concatenazione: `c()` serve per creare vettori di valori

```
c(1, 2, 25, 10)
```

```
[1] 1 2 25 10
```

Assegnazione e concatenazione

- Assegnazione: `<-` oppure `=`
- Concatenazione: `c()` serve per creare vettori di valori

```
c(1, 2, 25, 10)
```

```
[1] 1 2 25 10
```

- Esempi di variabili: `A`, `a`, `Giorgio`, `X3`
- Costanti:
 - `pi` (3.14...)
 - `TRUE`, `FALSE`

Variabili, costanti, assegnazione in R

```
A <- 12  
a = 6  
(A / a > 0) != TRUE  
  
[1] FALSE
```


Variabili, costanti, assegnazione in R

```
A <- 12
```

```
a = 6
```

```
(A / a > 0) != TRUE
```

```
[1] FALSE
```

```
A <- 12.8 # variabile numerica
```

```
nome <- "giorgio" # variabile character
```

```
b1 <- c(12, 0.3, 5, 778.3) # vettore numerico
```

```
Nomi3 <- c("giorgio", "ugo", "anna") # vettore character
```

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti**
 - Variabili, costanti, assegnazione in R
 - Matrici
 - Indicizzare e selezionare
 - Dataframe
 - Liste

Sono dei vettori concatenati di dimensione r (righe) $\times c$ (colonne)

```
matrix(data, nrow, ncol, byrow = FALSE)
```

Sono dei vettori concatenati di dimensione r (righe) $\times c$ (colonne)

```
matrix(data, nrow, ncol, byrow = FALSE)
```

Creazione matrice:

```
A <- c(1, 8, 5, 0)
M <- matrix(A, 2, 2)
M
```

	[,1]	[,2]
[1,]	1	5
[2,]	8	0

Sono dei vettori concatenati di dimensione r (righe) $\times c$ (colonne)

```
matrix(data, nrow, ncol, byrow = FALSE)
```

Creazione matrice:

```
A <- c(1, 8, 5, 0)
M <- matrix(A, 2, 2)
M
```

```
      [,1] [,2]
[1,]     1     5
[2,]     8     0
```

```
?matrix # Accede alla documentazione della funzione
```

Esercizio

- Scaricare il file `LaboratorioRmod1_Esercizi.pdf` dalla piattaforma Moodle del corso
- Il file contiene la traccia degli esercizi

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti**
 - Variabili, costanti, assegnazione in R
 - Matrici
 - Indicizzare e selezionare
 - Dataframe
 - Liste

Indicizzare e selezionare

vettore[i], matrice[r, c]

[i]: l'indice (la posizione) dell'oggetto che si vuole selezionare

[r, c]: riga, colonna (restituisce la cella specifica della matrice)

```
peso = c(3.3, 4, 5, 6.4, 3.5, 4.2)  matrice[1,]
peso[3]                               [1] 3.3 5.0 3.5
[1] 5                                matrice[, 2]
matrice = matrix(peso, nrow = 2)      [1] 5.0 6.4
matrice                                matrice[2,2]
                                     [1] 6.4

      [,1] [,2] [,3]
[1,] 3.3  5.0 3.5
[2,] 4.0  6.4 4.2
```


- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti**
 - Variabili, costanti, assegnazione in R
 - Matrici
 - Indicizzare e selezionare
 - Dataframe
 - Liste

```
data.frame(. . .)
```

Una matrice con r righe e c colonne

Può contenere variabili di tipo diverso

```
sexo = c(rep(c("M", "F"), length=length(peso)))  
data = data.frame(peso, sexo)  
data
```

	peso	sexo
1	3.3	M
2	4.0	F
3	5.0	M
4	6.4	F
5	3.5	M
6	4.2	F

```
data[r, c], data$nome_variabile
```

```
data[r, c], data$nome_variabile
```

```
data
```

	peso	sezzo
1	3.3	M
2	4.0	F
3	5.0	M
4	6.4	F
5	3.5	M
6	4.2	F

```
data[2, 1]
```

```
[1] 4
```

```
data$peso
```

```
[1] 3.3 4.0 5.0 6.4 3.5 4.2
```

```
data$peso[2]
```

```
[1] 4
```

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti**
 - Variabili, costanti, assegnazione in R
 - Matrici
 - Indicizzare e selezionare
 - Dataframe
 - Liste

```
list(name1 = obj1, name2 = obj2, name3 = obj3, . . .)
```

Permette di creare liste di oggetti diversi (matrici, vettori, variabili) di diverso tipo (numeric, character, integer)

```
list(name1 = obj1, name2 = obj2, name3 = obj3, . . .)
```

Permette di creare liste di oggetti diversi (matrici, vettori, variabili) di diverso tipo (numeric, character, integer)

```
A <- c(1,8,5,0)
M <- matrix(A, 2, 2)
Nomi3 <- c("giorgio", "ugo", "anna")
LV <- c(TRUE, FALSE, TRUE, TRUE)
OgLista <- list(a = A, Mat = M, nomi = Nomi3, X = LV)
str(OgLista)
```

List of 4

```
$ a      : num [1:4] 1 8 5 0
$ Mat    : num [1:2, 1:2] 1 8 5 0
$ nomi   : chr [1:3] "giorgio" "ugo" "anna"
$ X      : logi [1:4] TRUE FALSE TRUE TRUE
```

```
lista[[i]], lista$nome_oggettpo
```

[[i]]: indice dell'oggetto all'interno della lista a cui si vuole accedere

```
OgLista[[2]]
```

	[,1]	[,2]
[1,]	1	5
[2,]	8	0

```
$nome_oggetto
```

```
OgLista$Mat
```

	[,1]	[,2]
[1,]	1	5
[2,]	8	0

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti
- 6 L'ambiente**
- 7 Programmazione entry level

Ogni variabile che viene creata viene salvata nell'ambiente di R, visibile in ogni momento:

```
ls() # lista tutti gli oggetti nell'ambiente
```

```
[1] "a"           "A"           "b1"          "congiunzione"  
[6] "disgiunzione" "LV"          "M"           "matrice"  
[11] "nome"        "Nomi3"       "OgLista"     "peso"
```

```
rm(OgLista) # rimuove OgLista dall'ambiente
```

```
rm(list=ls()) # rimuove tutto dall'ambiente
```

- 1 Il corso
- 2 Introduzione
- 3 RStudio
- 4 Operatori
- 5 Oggetti
- 6 L'ambiente
- 7 Programmazione entry level**

function()

```
funzione = function(arg1, arg2, arg3,. . .) {  
  operazione1 = codice che fa cose  
  operazione2 = altro codice che fa cose  
  output = operazione1 e operazione2 fanno cose  
  return(output)  
}
```

Esempio

```
area = function(lato1, lato2) {  
  if (lato1 == lato2) {  
    warning("è un quadrato")  
  }  
  area = lato1*lato2  
  return(area)  
}
```

Esempio

```
area = function(lato1, lato2) {  
  if (lato1 == lato2) {  
    warning("è un quadrato")  
  }  
  area = lato1*lato2  
  return(area)  
}
```

```
area(lato1 = 5, lato2 = 2)
```

```
[1] 10
```