01-stRutture di dati

Ottavia M. Epifania, Ph.D

Lezione di Dottorato @Università Cattolica del Sacro Cuore (MI)

8-9 Giugno 2023

Table of contents

Table of Contents

Vengono creati **c**oncatenando diverse variabili insieme

Si usa la funzione `c()`

Tutte le variaili all'interno della funzione `c()` vanno separate da una virgola

Diversi tipi di variabili $\rightarrow$ diversi tipi di vettori:

- `int`: vettori numerici (numeri interi)
- `num`: vettori numerici (numeri continui)
- `logi`: vettori logici
- `chr`: vettori character
- `factor`: vettori factor con diversi livelli

int & num

`int`: numeri interi: -3, -2, -1, 0, 1, 2, 3

```
mesi = c(5, 6, 8, 10, 12, 16)
```

```
[1]  5  6  8 10 12 16
```

`num`: tutti i valori numerici tra $-\infty$ e $+\infty$: 1.0840991, 0.8431089, 0.494389, -0.7730161, 2.9038161, 0.9088839

```
peso = seq(3, 11, by = 1.5)
```

```
[1]  3.0  4.5  6.0  7.5  9.0 10.5
```

logi

Valori logici possono essere veri `TRUE` (`T`) o falsi `FALSE` (`F`):

```
v_logi = c(TRUE, TRUE, FALSE, FALSE, TRUE)
```

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

Si usano per testare delle condizioni:

```
mesi > 12
```

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

chr & factor

chr: characters: a, b, c, D, E, F

```
v_chr = c(letters[1:3], LETTERS[4:6])
```

```
[1] "a" "b" "c" "D" "E" "F"
```

factor: Usa numeri o caratteri per identificare i livelli della variabile:

```
ses = factor(rep(c("low", "medium", "high"), each = 2))
```

```
[1] low    low    medium medium high   high
Levels: high low medium
```

Si può cambaire l'ordine dei livelli:

```
ses1 = factor(ses, levels = c("medium", "high", "low"))
```

```
[1] low    low    medium medium high   high
Levels: medium high low
```

Creare i vettori

Concatenare le variabili con `c()`: `vec = c(1, 2, 3, 4, 5)`

Utilizzando le sequenze:

```
-5:5 # vector of 11 numbers from -5 to 5
```

```
 [1] -5 -4 -3 -2 -1  0  1  2  3  4  5
```

```
seq(-2.5, 2.5, by = 0.5) # sequence in steps of 0.5
```

```
 [1] -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5
```

Ripetendo gli elementi:

```
rep(1:3, 4)
```

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

Creare i vettori II

```
rep(c("condA", "condB"), each = 3)
```

```
[1] "condA" "condA" "condA" "condB" "condB" "condB"
```

```
rep(c("on", "off"), c(3, 2))
```

```
[1] "on"  "on"  "on"  "off" "off"
```

```
paste0("item", 1:4)
```

```
[1] "item1" "item2" "item3" "item4"
```

Non mischiate i vettori! a meno che non lo vogliate davvero

`int` + `num` → `num`

`int/num` + `logi` → `int/num`

`int/num` + `factor` → `int/num`

`int/num` + `chr` → `chr`

`chr` + `logi` → `chr`

Vettori e operazioni

I vettori possono essere sommati/divisi/moltiplicati tra di loro o anche per un numero singolo

```
a = c(1:8) # vettore di lunghezza 8
a
```

```
[1] 1 2 3 4 5 6 7 8
```

```
b = c(4:1) # vettore di lunghezza 4
b
```

```
[1] 4 3 2 1
```

```
a - b # il vettore b è "riciclato" sul vettore a
```

```
[1] -3 -1  1  3  1  3  5  7
```

Se i vettori non hanno la stessa lunghezza (o uno non è un multiplo dell'altro) ottenete un warning

Vettori e operazioni II

Applicando una funzione a un vettore → viene applicata a **tutti** gli elementi del vettore

```
sqrt(a)
```

```
[1] 1.000000 1.414214 1.732051 2.000000 2.236068 2.449490 2.645751 2.82842
```

La stessa operazione si può applicare a ogni singolo elemento del vettore

```
(a - mean(a))^2 # squared deviation
```

```
[1] 12.25  6.25  2.25  0.25  0.25  2.25  6.25 12.25
```

Your turn!

- Create un vettore di tipo `character`:
  - `condizioneA` ripetuto 3 volte
  - `condizioneB` ripetuto 2 volte
  - `condizioneC` ripetuto 5 volte
- Trasformate il vettore in `factor`
- Cambiate i livelli del vettore: `condizioneB`, `condizioneA`, `condizioneC`
- Create un vettore (`my_vector`) che vada da -3 a 3 a step di 0.2

Indicizzare i vettori

Come si va a "raggiungere" un particolare elemento all'interno del vettore?

```
nomi = c("Pasquale", "Egidio", "Debora", "Luca", "Andrea")
```

Pasquale	Egidio	Debora	Luca	Andrea
1	2	3	4	5

Indicizzare i vettori

Come si va a "raggiungere" un particolare elemento all'interno del vettore?

```
nomi = c("Pasquale", "Egidio", "Debora", "Luca", "Andrea")
```

Pasquale	Egidio	Debora	Luca	Andrea
1	2	3	4	5

`nome_vettore[indice]`

Indicizzare i vettori II

Pasquale	Egidio	Debora	Luca	Andrea
1	2	3	4	5

Indicizzare i vettori II

Pasquale	Egidio	Debora	Luca	Andrea
1	2	3	4	5

`nomi[1]` $\rightarrow$

Indicizzare i vettori II

Pasquale	Egidio	Debora	Luca	Andrea
1	2	3	4	5

`nomi[1]` $\rightarrow$ Pasquale

`nomi[3]` $\rightarrow$

Indicizzare i vettori II

Pasquale	Egidio	Debora	Luca	Andrea
1	2	3	4	5

`nomi[1]` $\rightarrow$ Pasquale

`nomi[3]` $\rightarrow$ Debora

`nomi[seq(2, 5, by = 2)]` $\rightarrow$

Indicizzare i vettori II

Pasquale	Egidio	Debora	Luca	Andrea
1	2	3	4	5

`nomi[1]` $\rightarrow$ Pasquale

`nomi[3]` $\rightarrow$ Debora

`nomi[seq(2, 5, by = 2)]` $\rightarrow$ Egidio, Luca

Indicizzare i vettori: Esempi

```
peso
```

```
[1]  3.0  4.5  6.0  7.5  9.0 10.5
```

```
peso[2]         # secondo elemento del vettore peso
```

```
[1] 4.5
```

```
(peso[6] = 15.2) # sostituisce il sesto elemento del v. peso
```

```
[1] 15.2
```

```
peso[seq(1, 6, by = 2)] # elementi 1, 3, 5
```

```
[1] 3 6 9
```

```
peso[2:6]       # dal 2 al 6 elemento di peso
```

```
[1]  4.5  6.0  7.5  9.0 15.2
```

```
peso[-2]       # vettore peso senza il secondo elemento
```

```
[1]  3.0  6.0  7.5  9.0 15.2
```

Indicizzare i vettori usando la logica

```
peso
```

```
[1]  3.0  4.5  6.0  7.5  9.0 15.2
```

Indicizzare i vettori usando la logica

```
peso
```

```
[1]  3.0  4.5  6.0  7.5  9.0 15.2
```

Quali sono i valori maggiori di 7?

```
peso > 7
```

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

Indicizzare i vettori usando la logica

```
peso
```

```
[1]  3.0  4.5  6.0  7.5  9.0 15.2
```

Quali sono i valori maggiori di 7?

```
peso > 7
```

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

Usiamo questa informazione per filtrare il nostro vettore:

```
peso[peso > 7] # valori in peso maggiori di 7
```

```
[1]  7.5  9.0 15.2
```

```
peso[peso >= 4.5 & peso < 8] # valori tra 4.5 e 8
```

```
[1] 4.5 6.0 7.5
```

Your turn!

- Prendete il vettore numerico che avete creato prima:
  - Estraete il terzo elemento
  - Estraete tutti gli elementi dispari del vettore e assegnateli a `my_vector1`
  - Estraete tutti gli elementi di `my_vector1` ≤ 0

Table of Contents

Un vettore che ci ha creduto abbastanza

Quel che basta per vincere una seconda dimensione

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

Crea una matrice 3×4 e la assegna all'oggetto `A`:

```
A = matrix(1:12, nrow=3, ncol = 4, byrow = FALSE)
A
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

WARNING: i dati all'interno della matrice devono essere tutti dello stesso tipo

Etichette

```
rownames(A) = c(paste("riga", 1:nrow(A), sep = "_"))

colnames(A) = c(paste("colonna", 1:ncol(A), sep = "_"))

A
```

```
       colonna_1 colonna_2 colonna_3 colonna_4
riga_1         1         4         7        10
riga_2         2         5         8        11
riga_3         3         6         9        12
```

Trasposta della matrice:

```
A
```

```
       colonna_1 colonna_2 colonna_3 colonna_4
riga_1         1         4         7        10
riga_2         2         5         8        11
riga_3         3         6         9        12
```

```
t(A)
```

```
          riga_1 riga_2 riga_3
colonna_1      1      2      3
colonna_2      4      5      6
colonna_3      7      8      9
colonna_4     10     11     12
```

Creare le matrici (ancora)

Le matrici si possono anche creare concatenando vettori colonna:

```
cbind(a1 = 1:4, a2 = 5:8, a3 = 9:12)
```

```
     a1 a2 a3
[1,]  1  5  9
[2,]  2  6 10
[3,]  3  7 11
[4,]  4  8 12
```

o vettori riga:

```
rbind(a1 = 1:4, a2 = 5:8, a3 = 9:12)
```

```
   [,1] [,2] [,3] [,4]
a1    1    2    3    4
a2    5    6    7    8
a3    9   10   11   12
```

Indicizzare le matrici

Abbiamo due dimensioni:

	[,1]	[,2]	[,3]
[1,]	1, 1	1, 2	1, 3
[2,]	2, 1	2, 2	2, 3
[3,]	3, 1	3, 2	3, 3

```
my_matrix[righe, colonne]
```

Indicizzare le matrici: In pratica

```
A
```

```
        colonna_1 colonna_2 colonna_3 colonna_4
riga_1          1         4         7        10
riga_2          2         5         8        11
riga_3          3         6         9        12
```

`A[1, ]` →

`A[2, ]` →

`A[2, 3]` →

Indicizzare le matrici: In pratica

```
A
```

```
        colonna_1 colonna_2 colonna_3 colonna_4
riga_1          1         4         7        10
riga_2          2         5         8        11
riga_3          3         6         9        12
```

`A[1, ]` $\rightarrow$ 1, 4, 7, 10

`A[2, ]` $\rightarrow$

`A[2, 3]` $\rightarrow$

Indicizzare le matrici: In pratica

```
A
```

```
        colonna_1 colonna_2 colonna_3 colonna_4
riga_1          1         4         7        10
riga_2          2         5         8        11
riga_3          3         6         9        12
```

`A[1, ]` $\rightarrow$ 1, 4, 7, 10

`A[2, ]` $\rightarrow$ 2, 5, 8, 11

`A[2, 3]` $\rightarrow$

Indicizzare le matrici: In pratica

```
A
```

```
        colonna_1 colonna_2 colonna_3 colonna_4
riga_1          1         4         7        10
riga_2          2         5         8        11
riga_3          3         6         9        12
```

`A[1, ]` $\rightarrow$ 1, 4, 7, 10

`A[2, ]` $\rightarrow$ 2, 5, 8, 11

`A[2, 3]` $\rightarrow$ 8

Your turn!

- Create una matrice 3×3 con la tabellina del 3 (fino al 24, valori per riga)
- Assegnate i nomi alle colonne e alle righe
- Assegnate la trasposta della matrice all'oggetto `my_t`
- Estraete da `my_t`:
  - la prima riga
  - la seconda colonna
  - la terza cella della terza riga ([3, 3])

Table of Contents

Una matrice che ci ha creduto davvero

Davvero troppo

`array(data, c(nrow, ncol, ntab))`

Avendo 3 argomenti oltre i dati `nrow`, `ncol`, `ntab`, la loro indicizzazione prevede l'utilizzo di due virgole per accedere ai singoli argomenti: `nome_array[righe, colonne, tab]`

Un array

```
my_array = array(1:20, c(2, 5, 3)) # 2 x 5 x 3 array
my_array
```

```
, , 1

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10

, , 2

     [,1] [,2] [,3] [,4] [,5]
[1,]   11   13   15   17   19
[2,]   12   14   16   18   20

, , 3

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
```

Indicizzare l'array

```
my_array[1, , ]
```

```
my_array[, 2, ]
```

```
my_array[, , 3]
```

Indicizzare l'array

```
my_array[1, , ]
```

```
     [,1] [,2] [,3]
[1,]    1   11    1
[2,]    3   13    3
[3,]    5   15    5
[4,]    7   17    7
[5,]    9   19    9
```

```
my_array[, 2, ]
```

```
my_array[, , 3]
```

Indicizzare l'array

```
my_array[1, , ]
```

```
     [,1] [,2] [,3]
[1,]    1   11    1
[2,]    3   13    3
[3,]    5   15    5
[4,]    7   17    7
[5,]    9   19    9
```

```
my_array[, 2, ]
```

```
     [,1] [,2] [,3]
[1,]    3   13    3
[2,]    4   14    4
```

```
my_array[, , 3]
```

Indicizzare l'array

```
my_array[1, , ]
```

```
     [,1] [,2] [,3]
[1,]    1   11    1
[2,]    3   13    3
[3,]    5   15    5
[4,]    7   17    7
[5,]    9   19    9
```

```
my_array[, 2, ]
```

```
     [,1] [,2] [,3]
[1,]    3   13    3
[2,]    4   14    4
```

```
my_array[, , 3]
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
```

Table of Contents

Un array con più senso

Sono dei contenitori per diversi tipi di oggetti (e.g., vettori, data frames, altre liste, matrici, array ecc.)

Ai loro elementi possono essere assegnati dei nomi:

```
my_list = list(w = peso, m = mesi, s = ses1, a = A)
names(my_list)
```

```
[1] "w" "m" "s" "a"
```

```
str(my_list)
```

```
List of 4
 $ w: num [1:6] 3 4.5 6 7.5 9 15.2
 $ m: num [1:6] 5 6 8 10 12 16
 $ s: Factor w/ 3 levels "medium","high",..: 3 3 1 1 2 2
 $ a: int [1:3, 1:4] 1 2 3 4 5 6 7 8 9 10 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "riga_1" "riga_2" "riga_3"
  .. ..$ : chr [1:4] "colonna_1" "colonna_2" "colonna_3" "colonna_4
```

Indicizzare le liste

Gli elementi della lista possono essere indicizzati con $ (se la lista ha dei nomi):

```
my_list$m # vettore dei mesi
```

```
[1]  5  6  8 10 12 16
```

oppure con [[]]:

Nome dell'elemento

```
my_list[["m"]]
```

```
[1]  5  6  8 10 12 16
```

Posizione dell'elemento:

```
my_list[[2]]
```

```
[1]  5  6  8 10 12 16
```

Your turn!

- Create una lista che contenga:
  - La matrice originale con la tabellina del 3
  - La trasposta della matrice
  - Tutti gli elementi ≥ 0 di `my_vector1`
- Date un nome ad ogni elemento all'interno della lista

Table of Contents

Una lista più ordinata

I data frames sono delle liste di vettori di uguale lunghezza

I diversi vettori possono contenere informazioni di diverse natura

I data frame più comuni sono i data frame in versione wide (i.e., *soggetti* $\times$ *variabili*) $\rightarrow$ `nrow(data)` $=$ numero di soggetti:

```
id = paste0("sbj", 1:6)
babies = data.frame(id, mesi, peso)
```

```
babies
```

```
    id mesi peso
1 sbj1    5  3.0
2 sbj2    6  4.5
3 sbj3    8  6.0
4 sbj4   10  7.5
5 sbj5   12  9.0
6 sbj6   16 15.2
```

Indicizzare i data frame

Vale tutto quello visto per le matrici:

Prima riga del data frame `babies`

```
babies[1, ]
```

Prima colonna del data frame

```
babies
babies[, 1]
```

In più:

```
babies$mesi # colonna mesi di babies

babies$mesi[2] # secondo elemento del vettore colonna

babies[, "id"] # column id

babies[2, ] # second row of babies (obs on baby 2)
```

Logic applies:

```
babies[babies$peso > 7, ] # filtra per tutte le righe con
```

```
    id mesi peso
4 sbj4   10  7.5
5 sbj5   12  9.0
6 sbj6   16 15.2
```

```
    #peso > 7
babies[babies$id %in% c("sbj1", "sbj6"), ] # restituisce le o
```

```
    id mesi peso
1 sbj1    5  3.0
6 sbj6   16 15.2
```

```
    # di questi due soggetti
```

Working with data frames II

```
dim(babies) # data frame con 6 righe e 3 colonne
```

```
[1] 6 3
```

```
names(babies) # = colnames(babies)
```

```
[1] "id"   "mesi" "peso"
```

```
head(babies) # fa vedere le prime sei righe del data frame
```

```
    id mesi peso
1 sbj1    5  3.0
2 sbj2    6  4.5
3 sbj3    8  6.0
4 sbj4   10  7.5
5 sbj5   12  9.0
6 sbj6   16 15.2
```

```
View(babies) # open data viewer
```

Your turn!

- Create un data frame con 10 osservazioni e le seguenti colonne:
  - `id`: `character`, id dei soggetti
  - `ses`: `factor`, livello socio economico dei soggetti con 3 livelli, `low`, `medium`, `high` (3 `low`, 5 `medium`, 2 `high`)
  - `income`: `numeric`
- Filtrate il data set:
  - Soggetti con `high ses`
  - Soggetti con `income` $>$ 2000