





1 Vettori

2 Fattori

3 Matrici

4 Dataframe







## Caratteristiche di un vettore

- **la lunghezza:** il numero di elementi da cui è formato il vettore
- **la tipologia:** la tipologia di dati da cui è formato il vettore. Un vettore infatti deve essere formato da **elementi tutti dello stesso tipo!**



















```
correct = c(1,2,3,NA, 4)
class(correct)
```

```
[1] "numeric"
```

correct

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



◀ ◻ ▶ ◀ ◼ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ 🔍 ↺











# 1 Vettori

- Indicizzazione

## 2 Fattori

## 3 Matrici

## 4 Dataframe



```
# Creo un vettore formato da 10 numeri casuali
my_vect = round(runif(n = 10,min = 1, max = 100))
my_vect
```

```
my_vect[1] # estraggo il primo elemento
```

◀ ◻ ▶ ◀ ▢ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ↺ 🔍 ↻



```
my_vect[1:5] # estraggo i primi 5 elementi
```

[1] 20 18 37 47 77

```
my_vect[c(1,4,2,9)] # estraggo elementi a scelta
```

[1] 20 47 18 85



◀ ◻ ▶ ◀ ◼ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ 🔍 ↺



[1] 3 4







1 Vettori

2 **Fattori**

3 Matrici

4 Dataframe



## Fattori

I fattori sono una tipologia di dato peculiare e per quanto simile a semplici **characters** in realtà sono un tipo di vettore **integer** con delle proprietà aggiuntive.











I fattori permettono di avere dei livelli `levels()` come metadati,

```
levels(my_fact)
```

```
[1] "hello" "ciao" "hola"
```







```
# come sarebbe my_fact2?
droplevels(my_fact2)
```

```
# modifico my_fact2 eliminando i livelli inutili
my_fact2 = droplevels(my_fact2)
my_fact2
```

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ 🔍 ↺







1 Vettori

2 Fattori

3 **Matrici**

4 Dataframe



















## Cosa fa R di default?

```
matrix(data = 1:10, ncol = 3, nrow = 3)
```

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

```
matrix(data = 1:2, ncol = 3, nrow = 3)
```

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

**warnings:** la funzione ci informa di qualcosa di potenzialmente problematico, ma (circa!!) tutto liscio



1 Vettori

2 Fattori

3 **Matrici**

- Matrici - Indicizzazione

4 Dataframe







E' possibile anche selezionare un'intera riga o colonna

```
my_mat[1,]
```

```
[1] 1 3 5 7 9
```

```
my_mat[,1]
```

[1] 1 2



















```
rowbind()
```

```
rbind(my_mat1, my_mat2)
```

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

Cosa notate di “strano”?



## Operazioni con le matrici

Come per i vettori, anche alle matrici si possono applicare operazioni matematiche:

```
my_mat = matrix(data = 1:4, nrow = 2, ncol = 2)
```

## # element-wise

```
my_mat*my_mat
```

```

      [,1] [,2]
[1,]    1    9
[2,]    4   16

```











- 1 Vettori
- 2 Fattori
- 3 Matrici
- 4 Dataframe**



## Dataframe

Il dataframe è la struttura più “complessa”, utile e potente di R.

- ogni elemento è un **vettore** con un **nome associato** (aka una colonna)
- ogni colonna deve avere lo stesso numero di elementi
- di conseguenza ogni riga ha lo stesso numero di elementi (**struttura rettangolare**)



1 Vettori

2 Fattori

3 Matrici

4 Dataframe

- Creazione
- Indicizzazione
- Combinare Dataframes
- Esportazione e importazione dati











1 Vettori

2 Fattori

3 Matrici

4 Dataframe

- Creazione
- Indicizzazione
- Combinare Dataframes
- Esportazione e importazione dati



























```
my_df
```

	numeri	lettere	normale
1	1	a	-0.0994276
2	2	b	-0.1670143
3	3	c	-1.0846619
4	4	d	1.0178860

```
my_df[my_df$numeri== 2, 2]
```



```
my_df
```

```
   numeri lettere  normale
1        1      a -0.0994276
2        2      b -0.1670143
3        3      c -1.0846619
4        4      d  1.0178860
```

```
my_df[my_df$numeri== 2, 2]
```

```
[1] "b"
```



```
my_df
```

	numeri	lettere	normale
1	1	a	-0.0994276
2	2	b	-0.1670143
3	3	c	-1.0846619
4	4	d	1.0178860

```
my_df[my_df$numeri == 2, "lettere"]
```



```
my_df
```

```
   numeri lettere  normale
1        1      a -0.0994276
2        2      b -0.1670143
3        3      c -1.0846619
4        4      d  1.0178860
```

```
my_df[my_df$numeri== 2, "lettere"]
```

```
[1] "b"
```



## Indicizzazione subset()

```
my_df = data.frame(numeri = rep(1:3,each = 3),  
                   lettere = rep(letters[1:3],3),  
                   normale = rnorm(n = 9, mean = 0, sd = 1))
```

*#visualizzo le prime 5 righe attraverso il comando head*  
`head(my_df, n = 5)`

	numeri	lettere	normale
1	1	a	2.1359557
2	1	b	0.8155501
3	1	c	-0.9766140
4	2	a	2.2328673
5	2	b	0.1459335



## Indicizzazione subset()

```
subset(my_df, subset = lettere == "a" & numeri > 2)
```

	numeri	lettere	normale
7	3	a	0.07177012



Equivalente a:

```
my_df[my_df$lettere == "a" & my_df$numeri > 2,]
```

	numeri	lettere	normale
7	3	a	0.07177012



## subset(df, select = ...)

E' possibile anche selezionare colonne piuttosto che righe attraverso l'argomento *select*:

```
subset(my_df, select = c(lettere, numeri))
```

```
#visualizzo le prime tre righe attraverso il comando head  
head(subset(my_df, select = c(lettere, numeri)), n = 3)
```

	lettere	numeri
1	a	1
2	b	1
3	c	1



```
subset(df, subset = ..., select = ...)
```

Possiamo anche combinare le due cose:

```
subset(my_df, subset = lettere == "a" & numeri > 2,  
       select = c(lettere, numeri))
```

```
head(subset(my_df, subset = lettere == "a" & numeri > 2,  
           select = c(lettere, numeri)), n = 3)
```

```
lettere numeri  
7         a      3
```



La maggiorparte delle volte vi troverete ad accedere alle variabili tramite l'operatore `$`. Questo comando può essere utilizzato anche per creare una nuova variabile...

```
# creo una variabile che è la somma di numeri e normale  
my_df$somma = my_df$numeri + my_df$normale  
str(my_df)
```

```
'data.frame':  9 obs. of  4 variables:  
 $ numeri : int  1 1 1 2 2 2 3 3 3  
 $ lettere: chr  "a" "b" "c" "a" ...  
 $ normale: num  2.136 0.816 -0.977 2.233 0.146 ...  
 $ somma  : num  3.136 1.8156 0.0234 4.2329 2.1459 ...
```



Potete sia creare che modificare variabili accedendo attraverso \$

```
# Modifico la variabile num aggiungendo 1
my_df$numeri = my_df$numeri+1

# Creo una terza variabile composta dalla variabile num e let
my_df$both = paste(my_df$numeri,my_df$lettere, sep = "_") # ?paste

str(my_df)
```

```
'data.frame':  9 obs. of  5 variables:
 $ numeri : num  2 2 2 3 3 3 4 4 4
 $ lettere: chr  "a" "b" "c" "a" ...
 $ normale: num  2.136 0.816 -0.977 2.233 0.146 ...
 $ somma : num  3.136 1.8156 0.0234 4.2329 2.1459 ...
 $ both : chr  "2_a" "2_b" "2_c" "3_a" ...
```



1 Vettori

2 Fattori

3 Matrici

4 Dataframe

- Creazione
- Indicizzazione
- Combinare Dataframes
- Esportazione e importazione dati



## Combinare Dataframes

Essendo simili a delle matrici, i dataframe si possono combinare tra loro attraverso le funzioni `rbind()`:

```
my_df2 = data.frame(numero = 1:9, lettere = letters[1:9],  
                    normale = rnorm(9,mean = 0,sd = 1),  
                    somma = my_df$somma,  
                    both = paste(1:9,letters[1:9], sep = "_"))  
  
str(my_df2)
```

```
'data.frame':  9 obs. of  5 variables:  
 $ numero : int  1 2 3 4 5 6 7 8 9  
 $ lettere: chr  "a" "b" "c" "d" ...  
 $ normale: num  0.459 -0.43 -0.961 -1.358 0.245 ...  
 $ somma  : num  3.136 1.8156 0.0234 4.2329 2.1459 ...  
 $ both   : chr  "1_a" "2_b" "3_c" "4_d" ...
```



## Unisco i due dataframes

```
my_df3 = rbind(my_df,my_df2)
```

```
Error in `match.names()`:  
! names do not match previous names
```



## Combinare Dataframes

```
str(my_df)
```

```
'data.frame':  9 obs. of  5 variables:
 $ numeri : num  2 2 2 3 3 3 4 4 4
 $ lettere: chr  "a" "b" "c" "a" ...
 $ normale: num  2.136 0.816 -0.977 2.233 0.146 ...
 $ somma : num  3.136 1.8156 0.0234 4.2329 2.1459 ...
 $ both : chr  "2_a" "2_b" "2_c" "3_a" ...
```

```
str(my_df2)
```

```
'data.frame':  9 obs. of  5 variables:
 $ numero : int  1 2 3 4 5 6 7 8 9
 $ lettere: chr  "a" "b" "c" "d" ...
 $ normale: num  0.459 -0.43 -0.961 -1.358 0.245 ...
 $ somma : num  3.136 1.8156 0.0234 4.2329 2.1459 ...
 $ both : chr  "1_a" "2_b" "3_c" "4_d" ...
```



## Unisco i due dataframes

- I dataframes devono avere lo stesso numero di colonne
- I nomi delle colonne devono essere identici



## Sistemo i nomi

```
names(my_df2)
```

```
[1] "numero" "lettere" "normale" "somma"   "both"
```

```
names(my_df)
```

```
[1] "numeri" "lettere" "normale" "somma"   "both"
```

```
# voglio che i names di my_df2 corrispondano ai names di my_df  
names(my_df2) = names(my_df)
```



## Unisco i dataframe

```
my_df3 = rbind(my_df, my_df2)
str(my_df3)
```

```
'data.frame':  18 obs. of  5 variables:
 $ numeri : num  2 2 2 3 3 3 4 4 4 1 ...
 $ lettere: chr  "a" "b" "c" "a" ...
 $ normale: num  2.136 0.816 -0.977 2.233 0.146 ...
 $ somma : num  3.136 1.8156 0.0234 4.2329 2.1459 ...
 $ both : chr  "2_a" "2_b" "2_c" "3_a" ...
```



Potrebbe anche capitarvi di dover raccogliere differenti tipi di dato dallo stesso partecipante, e successivamente combinare le informazioni raccolte...



## Combinare Dataframes

```
df_rt = data.frame(subj = factor(rep(c("caio", "tizio"), each = 400)),
                    cond = factor(rep(c("easy", "hard"),
                                      each = 200, times = 2)),
                    rt = c(rlnorm(n = 400, meanlog = -1, sdlog = .25),
                          rlnorm(n = 400, meanlog = -.7, sdlog = .3)))

str(df_rt)
```

```
'data.frame': 800 obs. of 3 variables:
 $ subj: Factor w/ 2 levels "caio","tizio": 1 1 1 1 1 1 1 1 1 1 1 ...
 $ cond: Factor w/ 2 levels "easy","hard": 1 1 1 1 1 1 1 1 1 1 1 ...
 $ rt : num 0.361 0.33 0.533 0.31 0.333 ...
```



Dataframe contente l'età:

```
df_age = data.frame(subj = factor(c("caio","tizio")), age = c(20,3))  
str(df_age)
```

```
'data.frame':  2 obs. of  2 variables:  
 $ subj: Factor w/ 2 levels "caio","tizio": 1 2  
 $ age : num  20 3
```



In questo caso, è possibile utilizzare la funzione `merge()`:

```
df_all_1 = merge(x = df_rt, y = df_age, by="subj")  
str(df_all_1)
```

```
'data.frame':  800 obs. of  4 variables:  
 $ subj: Factor w/ 2 levels "caio","tizio": 1 1 1 1 1 1 1 1 1 1 ...  
 $ cond: Factor w/ 2 levels "easy","hard": 1 1 1 1 1 1 1 1 1 1 ...  
 $ rt  : num  0.361 0.33 0.533 0.31 0.333 ...  
 $ age : num  20 20 20 20 20 20 20 20 20 20 ...
```



oppure la funzione `_join()`:

```
library(dplyr) # carico il pacchetto dplyr
```

```
df_all_2 = left_join(x = df_rt, y = df_age, by = c("subj"))  
str(df_all_2)
```

```
'data.frame':  800 obs. of  4 variables:  
 $ subj: Factor w/ 2 levels "caio","tizio": 1 1 1 1 1 1 1 1 1 1 ...  
 $ cond: Factor w/ 2 levels "easy","hard": 1 1 1 1 1 1 1 1 1 1 ...  
 $ rt  : num  0.361 0.33 0.533 0.31 0.333 ...  
 $ age : num  20 20 20 20 20 20 20 20 20 20 ...
```



1 Vettori

2 Fattori

3 Matrici

4 Dataframe

- Creazione
- Indicizzazione
- Combinare Dataframes
- Esportazione e importazione dati



















## Ora facciamo un po' di pratica!

Aprirete e tenete aperto questo link:

<https://etherpad.wikimedia.org/p/arca-corsoR>