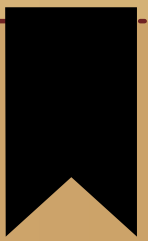


Laboratorio di Statistica



36 ore. 2 ore **effettive** a settimana

Orario Lezioni:

martedì ore 08:30 – 10:30, aula 309

mercoledì ore 08:30 – 10:30, aula 309

• Le lezioni del mercoledì sono la replica di quelle del martedì

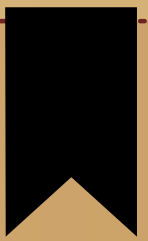
Sito del corso :

<http://frasca.di.unimi.it/stat15/stat.html>

Trovate:

- Slide e materiale didattico di ogni lezione
- Collegamenti a manuali R
- Eventuali avvisi

Laboratorio di Statistica



Argomenti del corso:

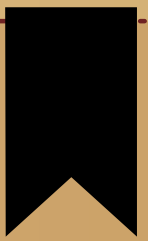
- Introduzione al linguaggio R

- Strutture dati R: vettori, matrici, liste, data frame
- Istruzioni di controllo del flusso
- Funzioni e script
- Operazioni di I/O
- Grafica

- Seconda parte: esercitazioni di statistica e calcolo delle probabilità

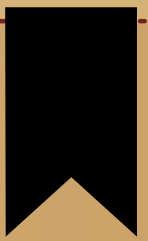


Ambiente R



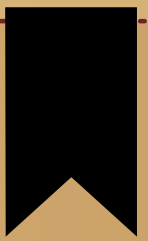
- Software gratuito disponibile al repository CRAN (Comprehensive R Archive Network)
 - Oltre all'ambiente R, fornisce documentazione e package R aggiornati
 - Disponibile per piattaforme UNIX, Windows and MacOS
- Linguaggio ad alto livello
- User-Friendly
- Disponibili numerosi package che implementano algoritmi in diversi ambiti: statistica, machine learning, bioinformatica, economia, medicina, etc.

Ambiente R



- Linguaggio interpretato
 - I comandi vengono eseguiti direttamente dall'interprete
 - Il codice non deve essere compilato
 - Carico aggiuntivo per la CPU (passaggio intermedio)
- Ottimo per l'elaborazione di dati statistici
 - Praticamente è tutto già implementato, nelle librerie base di R o in package installabili semplicemente con un comando (quasi sempre)
- Per avviare R:
 - Linux/Mac: digita 'R' dalla linea di comando
 - Windows: apri R dalla GUI (Graphical User Interface)

Ambiente R



- Uscire da R
 - `q()`
- Chiedere aiuto ad R
 - `help()` # chiede l'help generale
 - `help(nomecomando)` # chiede l'help del comando
 - `?nomecomando` # chiede l'help del comando
 - `help.start()` # lancia l'help in un browser

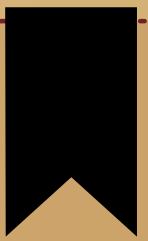


Ambiente R



- `ls()` # mostra il contenuto del workspace
- `rm(i)` # rimuove dal workspace la variabile *i*
- `rm(list=ls())` # rimuove tutto (dati, funzioni) dal workspace
- `save(a, b ,file="prova.rda")` # salva le variabili *a* e *b* nel file binario compresso 'prova.rda'
- `load("prova.rda")` # ricarica nel workspace i dati salvati in precedenza
- `setwd(nome_directory)` # sposta il controllo di R nella directory specificata
- `getwd()` # visualizza la directory corrente

Strutture dati fondamentali in R: Vettori



- Nei linguaggi di programmazione ad alto livello non si ha accesso diretto alla memoria fisica, ma ad una sua astrazione (*struttura dati*)
- Prescinde dai dati realmente utilizzati
- Specifica come sono organizzati i dati e come accedervi



Vettori

- I vettori rappresentano sequenze di elementi omogenei accessibili direttamente mediante un indice
- Un vettore v è rappresentabile tramite una struttura unidimensionale:

	1	2	3	4	5	6	7	8
v	4	8	2	1	3	1	5	0

- Gli indici partono da 1. Ad es. 6 è l'indice dell'elemento 1

Comando c

> c(1,4,5) # crea un vettore di interi
> c("A","B","C") # crea un vettore di caratteri
> c("gatto", "topo", "12") # crea un vettore di stringhe

La *funzione* c(arg1, arg2, arg3, ...) concatena i suoi argomenti

- Un vettore può essere *assegnato* ad una *variabile*

Variabili ed assegnamenti

```
> X <- c(1, 4, 5)           # il vettore <1 4 5> è assegnato alla variabile X
```

```
> X
```

```
[1] 1 4 5
```

La variabile X rappresenta ora il vettore <1 4 5>: si può pensare come un “contenitore” del vettore stesso

Un nuovo assegnamento cancella il contenuto precedente

```
> X <- c(4, 7)             # il vettore <4 7> è assegnato alla variabile X
```

```
> X
```

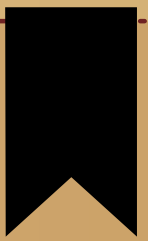
```
[1] 4 7
```

```
> Y <- 100                 # vettore formato da 1 solo elemento
```

```
> Y
```

```
[1] 100
```

Tipi elementari di vettori



- **Numerici**: numeri interi o in virgola mobile (floating point)
- **Caratteri**: singoli caratteri o stringhe (sequenze) di caratteri
- **Logici**: TRUE o FALSE



Operazioni con vettori aritmetici

- Le operazioni usuali dell'aritmetica vengono eseguite sui vettori **elemento per elemento**:

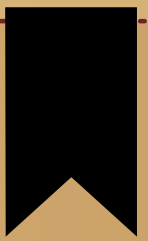
Addizione e sottrazione

```
> x <- c(1,2,3)
> y <- c(4,5,6)
> z <- x + y
> z
[1] 5 7 9
> d <- y - x
> d
[1] 3 3 3
```

Moltiplicazione e divisione

```
> x <- c(1,2,3)
> y <- c(4,5,6)
> p <- x * y
> p
[1] 4 10 18
> q <- y / x
> q
[1] 4.0 2.5 2.0
```

Esempi di funzioni applicate a vettori numerici



```
> x <- c(1, 3, 2, 5)
> max(x)
[1] 5
> min(x)
[1] 1
> range(x)
[1] 1 5
> sum(x)
[1] 11
> prod(x)
[1] 30
```

```
> sort(x) # ordinamento
[1] 1 2 4 5
> order(x) # indici corrispondenti agli
           elementi ordinati
[1] 1 3 2 4
```



Generazione di sequenze regolari

- R dispone di diversi comandi per generare automaticamente sequenze di numeri:

```
> c(1:10)
```

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

```
> c (5:1)
```

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

```
> seq (1,10)
```

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

```
> seq (from=1, to=4, by=0.5)
```

```
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0
```

- La funzione **seq()** può avere 5 argomenti (si veda l' help).

- Un' altra funzione per generare repliche di vettori è **rep()**:

```
> rep( c(1,2), times=4)
```

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

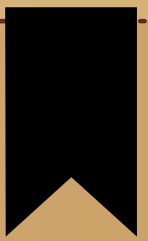
```
> rep( x, times) # ripete il vettore x  
times volte
```

Vettori ed operatori logici

- Sono vettori i cui elementi possono assumere valore **TRUE** o **FALSE**
- Si ottengono dall'applicazione di operatori logici
<, <=, >, >=, == (uguaglianza), != (disuguaglianza)
- Possibile trasformarli in valori numerici mediante il comando **as.numeric()**

```
> as.numeric(FALSE)
[1] 0
> as.numeric(TRUE)
[1] 1
```

Vettori ed operatori logici



```
> x<- 3:8
```

```
> x > 5
```

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

```
> x <= 5
```

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

```
> x == 5
```

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

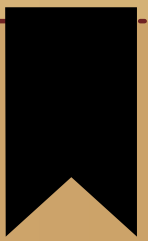
```
> x != 5
```

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

```
> x != c(5,6) # vale la “regola del riciclo”!
```

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


Dati mancanti



- In molte situazioni reali i componenti di un vettore possono essere “non noti” o comunque non disponibili.
- In questi casi R riserva il valore speciale **NA** (“Not Available”)
- In generale qualunque operazione che coinvolga valori NA ha come risultato NA.

```
> x <- c(1:4, NA)
```

```
> x + 2
```

```
[1] 3 4 5 6 NA
```



Dati mancanti

- Si noti che NA non è un valore ma un “marcatore” di una quantità non disponibile:

```
> x == NA  
[1] NA NA NA NA NA
```

- Per individuare quali elementi siano effettivamente NA in un vettore si deve usare la funzione `is.na()`:

```
> is.na(x)  
[1] FALSE FALSE FALSE FALSE TRUE
```

Vettori: selezione e accesso a sottoinsiemi di elementi



Esistono diverse modalità di accesso a singoli elementi o a sottoinsiemi di elementi di un vettore. In generale la selezione e l'accesso avviene tramite l' **operatore []** (parentesi quadre) :

- sottoinsiemi di elementi di un vettore sono selezionati collegando al nome del vettore un vettore di indici in parentesi quadre.

Esistono 4 diverse *modalità di selezione/accesso*:

1. Vettori di **indici interi positivi**
2. Vettore di **indici interi negativi**
3. Vettore di **indici logici**
4. Vettori di **indici a caratteri**

Selezione ed accesso tramite vettori di indici interi positivi

- Gli elementi di un vettore x sono selezionati tramite un vettore y di indici positivi racchiuso fra parentesi quadre: $x[y]$
- I corrispondenti elementi sono selezionati e concatenati.

```
> x <- 5:10  
> x[1] # selezione di un singolo  
elemento  
[1] 5  
> x[5]  
[1] 9
```

```
> length(x) # numero elementi nel vettore  
[1] 6  
> x[7] # accesso “fuori range”  
[1] NA  
> x[2:4]  
[1] 6 7 8  
> x[c(1,3,5)]  
[1] 5 7 9
```

Selezione ed accesso tramite vettori di indici interi negativi

- Sono selezionati gli elementi di un vettore x che devono essere esclusi tramite un vettore y di indici negativi racchiuso fra parentesi quadre : $x[y]$

```
> y <- rep(c("G","A","T","T"), times=3)
> y
[1] "G" "A" "T" "T" "G" "A" "T" "T" "G" "A" "T" "T"
> z <- y[-(1:5)]    # selezionati tutti gli elementi di y eccetto primi 5
> z
[1] "A" "T" "T" "G" "A" "T" "T"
> z <- z[-length(z)] # cancellazione dell' ultimo elemento di z
> z
[1] "A" "T" "T" "G" "A" "T"
```

Selezione ed accesso tramite vettori indice logici



- Il vettore degli indici deve essere della stessa lunghezza del vettore i cui elementi devono essere selezionati. Sono selezionati gli elementi corrispondenti a TRUE ed omessi gli altri

```
> x <- c(1:5,NA,NA)
> x
[1] 1 2 3 4 5 NA NA
> i <- c(rep(TRUE,times=3),rep(FALSE,times=4))
> i
# i è il vettore indice logico
[1] TRUE TRUE TRUE FALSE FALSE FALSE
FALSE
> x[i] # selez. elementi tramite vett. indice logico
[1] 1 2 3
```

```
> x[!is.na(x)]
# selezione elementi
che non sono NA
[1] 1 2 3 4 5
> x[!is.na(x) & x > 2]
[1] 3 4 5
```

& AND logico
| OR logico

Selezione ed accesso tramite vettori di indici a caratteri

- E' applicabile quando è stato impostato l'attributo *names* del vettore
- Associa una stringa ad ogni elemento
- Un sottovettore di *names* seleziona le componenti corrispondenti

```
> campione <- c(45, 180, 70, 2007, 3)
> names(campione) <- c("Eta", "Altezza", "Peso", "AnnoIns", "Ricoveri")
> campione
```

Eta	Altezza	Peso	AnnoIns	Ricoveri
45	180	70	2007	3

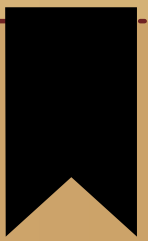
```
> Eta.Anno <- campione[c("Eta", "AnnoIns")]
> Eta.Anno
```

Eta	AnnoIns
45	2007

Comandi utili

- **sample**(*x*, *n*, *replace* = *FALSE*) # estrae *n* elementi dal vettore *x* in maniera casuale e senza reinserimento ---> *n* ? *length(x)*.
 - Se *replace* = *TRUE* , allora può accadere *n* > *length(x)*
 - > sample(1:6, 3)
[1] 4 1 3
 - > sample(1:6, 7) # errore!
 - > sample(1:6, 7, replace=TRUE)
[1] 6 3 6 1 1 5 3
- **floor**(*x*), **ceiling**(*x*), **trunc**(*x*), **round**(*x*, digits=3)
- **paste**("stringa1", "stringa2", sep="-") # concatenazione di stringhe
[1] "stringa1-stringa2"
- **setdiff**(*vettore1*, *vettore2*) # differenza insiemistica, elementi in *vettore1* non in *vettore2*

Esercizi



1) Inserire i seguenti valori numerici in tre distinti vettori:

- 2.5, 2, 5, 1, 11, -0.1, 10, 9
- 10, 9.5, 9, 8.5, 8, 7.5, 7, 6.5
- 1, 2, 3, 4, 5, 6, 7, 8

Quindi

- a) Calcolare minimo e massimo di ogni vettore
- b) Calcolare la somma degli elementi di ciascun vettore
- c) Calcolare la somma dei primi due vettori
- d) Calcolare la differenza dei tre vettori
- e) Calcolare il prodotto degli elementi del secondo vettore
- f) Concatenare i tre vettori ed estrarre un campione casuale pari al 30% di tutti gli elementi



Esercizi

2) I seguenti dati rappresentano il numero di giorni in cui ciascuno di 20 lavoratori si è assentato per malattia nelle ultime sei settimane:

2, 2, 0, 0, 5, 8, 3, 4, 1, 0, 0, 7, 1, 7, 1, 5, 4, 0, 4, 0

➤ I codici dei lavoratori sono “Lav0”, “Lav1”, ..., “Lav19”

Quindi

- Creare un vettore contenente i giorni di malattia e nel campo *names* i corrispondenti codici
- Selezionare il numero di giorni di malattia del lavoratore “Lav6”
- Determinare quanti lavoratori hanno fatto 0 giorni di malattia e stamparne i codici
- Estrarre il sottovettore contenente i giorni di malattia di tutti i lavoratori eccetto “Lav2” e “Lav11”