Introduzione a “R”
Uno strumento per affrontare l’analisi dati

Andrea Messina
andrea.messina@roma1.infn.it

Perche' un programma di analisi dati ?

- E' indispensabile avere uno strumento software di riferimento per affrontare problemi legati alla manipolazione e all'analisi dati
  - Input/output dati e la loro manipolazione
  - Operazioni
  - Calcoli ed algoritmi complicati
  - Grafici, istogrammi
  - Tabelle
  - Simulazioni
  - Analisi statistica
- Queste esigenze vi accompagneranno durante tutto il corso di laurea ed oltre

Le caratteristiche fondamentali

- Facile da utilizzare indipendentemente dalla piattaforma software
- Gratuito
- Open source
- Deve essere immediato leggere e scrivere dati nella forma piu' adatta al problema in esame
- Deve avere un'interfaccia grafica efficace per produrre grafici e tabelle in vari formati (txt, pdf, png, jpg, …)
- Deve avere una base solida di librerie scientifiche (matematica, statistica, …)
- Deve essere semplice implementare funzioni specifiche definite dall'utente
- …….

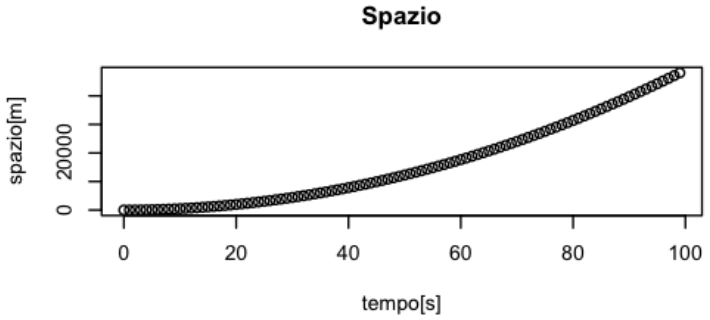
Obiettivi di questa presentazione

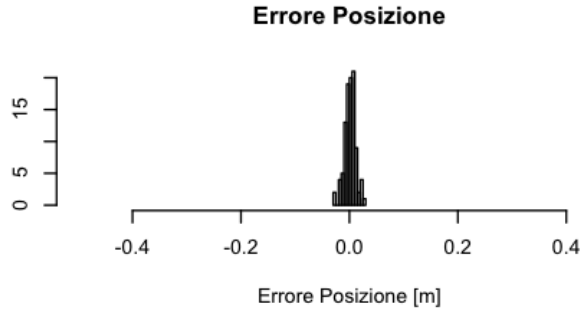
- Mostrarvi che uno strumento con le caratteristiche elencate prima vi aiutera' in tante situazioni pratiche
- Mostrarvi che "R" ha le caratteristiche per essere lo strumento di riferimento
- Darvi i rudimenti necessari per sfruttare questo strumento in modo profiquo
- ATTENZIONE: questa no e' una lezione sulle tecniche di programmazione o sul linguaggio "R"

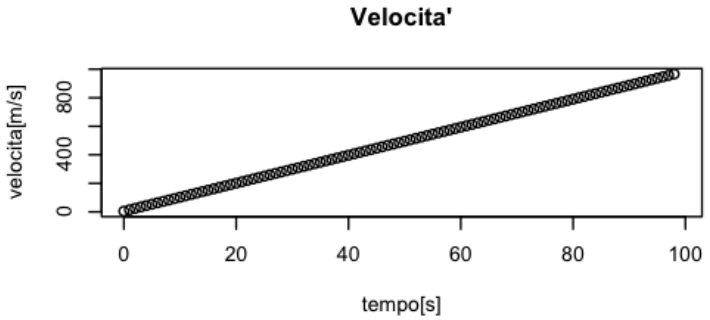
Come imparare ad usare questo strumento

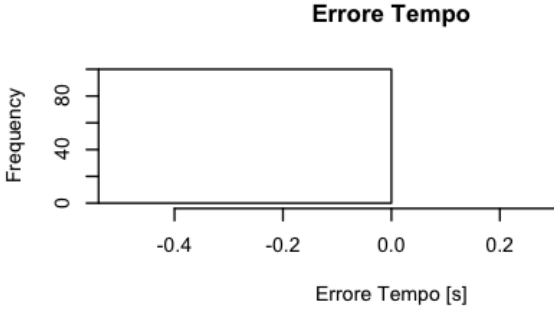
- Cercando di risolvere problemi pratici!
- “R” e’ uno strumento funzionale a raggiungere un obiettivo.
  - Non e’ utile studiare il suo funzionamento in tutti i dettagli, ma sapere come risolvere in modo adeguato il problema che si sta affrontanto
  - Se non si sapesse risolvere un problema, e’ fondamentale sapere dove reperire informazioni utili

Esempio pratico: simulazione di un punto materiale su un piano inclinato

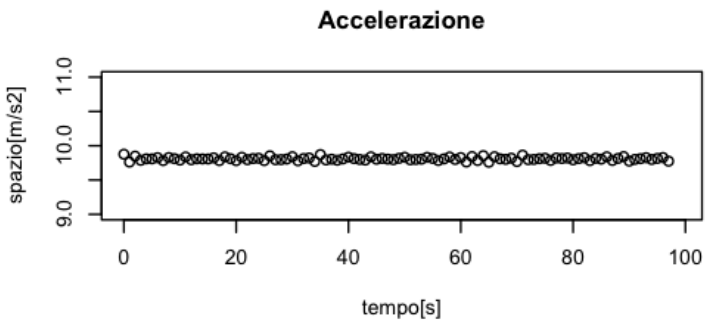


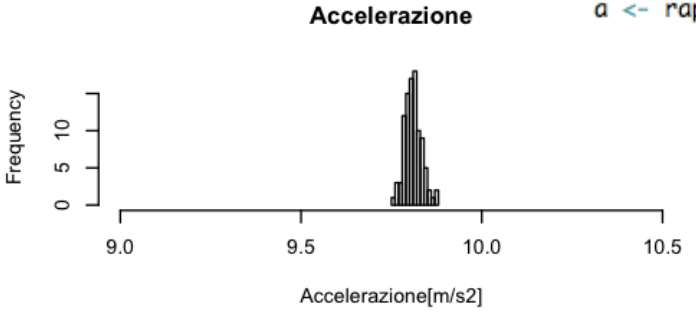







```
x0 <- 1.0; v0 <- 0.0
t <- seq(0, tmax-1, dt) # cercare nel help
x <- x0 + v0*t + 1/2 * 9.81 * t**2
x1 <- x; t1 <- t
ex <- rep(0,tmax); et <- rep(0,tmax);
if (withXerr == T) {
    ex <- rnorm(mean=0, n=tmax, sd=errx)
    x <- x+ex
}
if (withTerr == T) {
    et <- rnorm(mean=0, n=tmax, sd=errt)
    t <- t+et
}
v <- rapporto.incrementale(x,t, T)
a <- rapporto.incrementale(v,t, T)
```



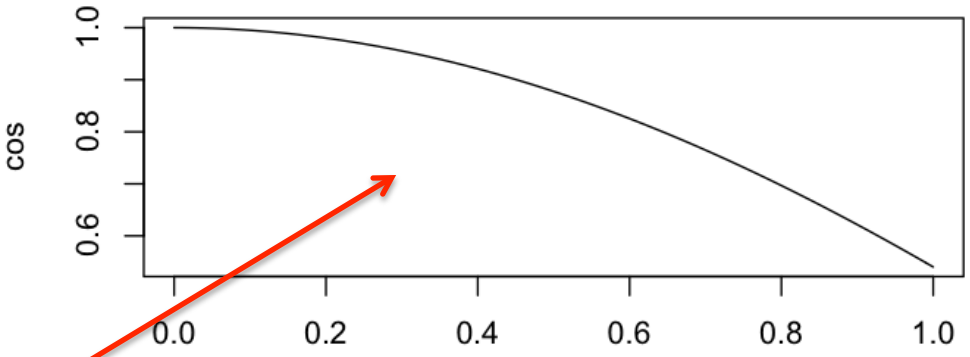


Istallare ed interagire con R

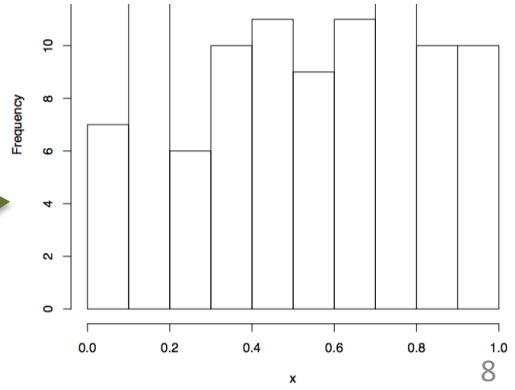
- http://www.r-project.org
- Esistono versioni precompilate per Windows, Linux, Mac OS X (si sceglie il mirror, si scarica, e si esegue!)
- http://cran.at.r-project.org (manuale)
- Caricare una libreria:
  - `install.packages("grid")` scarica il pacchetto
  - `library("grid")` carica la libreria in memoria
  - `require("grid")` controlla che la libreria ci sia altrimenti la carica

Interagire con R

```
iandrea2:R-statistics amessina$ R

R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin10.8.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> library("gridExtra")
Loading required package: grid
> log(10^23)
[1] 52.95946
>
>
> plot(cos)
> source("esercizio1.R")
```

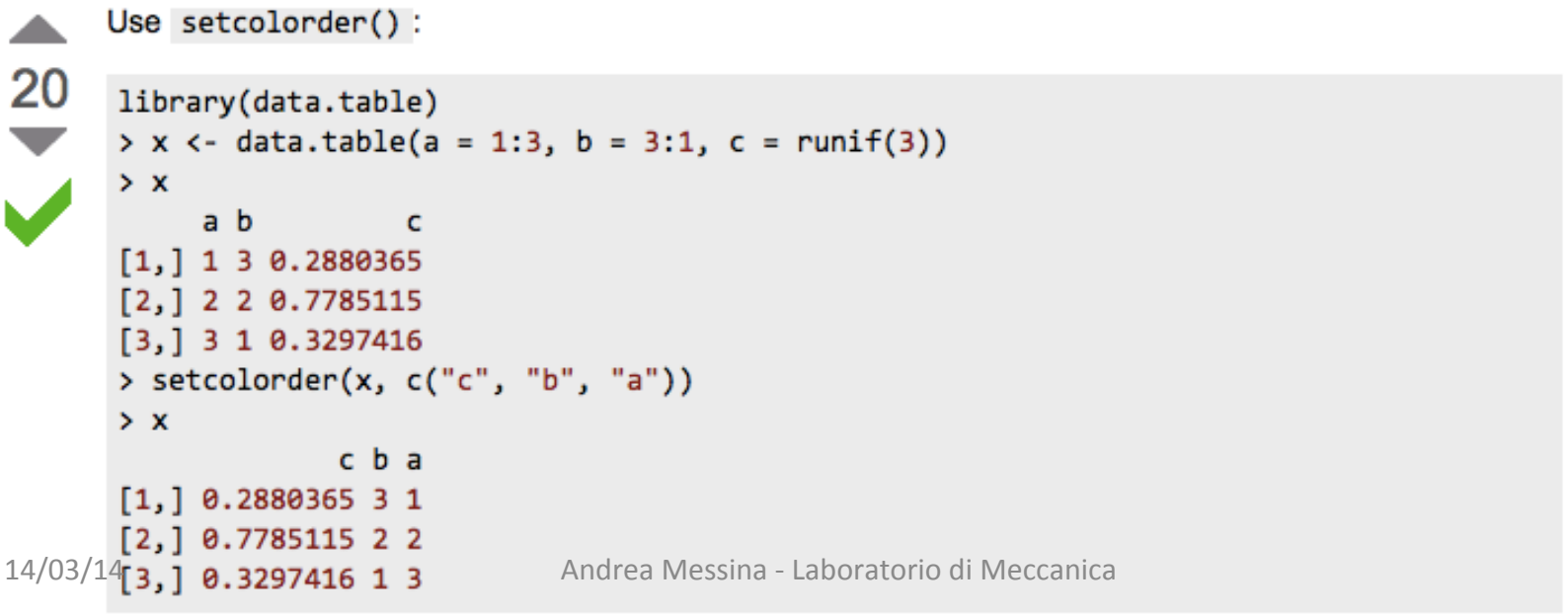



```
# 1. generare un capione di 100 valori distribuiti in modo uniforme tra 0 e 1

x <- runif(100, 0, 1)

# 2. farne un istogramma e salvarlo in pdf

pdf("istogramma.pdf")
hist(x)
dev.off()
```



Come cercare aiuto

- C'e' una letteratura sconfinata su web (in inglese!)
  - Google e' una risorsa eccezionale per trovare risposte a problemi specifici, provate a cercare "r table rearrange columns"



20

Use `setcolorder()`:

```
library(data.table)
> x <- data.table(a = 1:3, b = 3:1, c = runif(3))
> x
     a b         c
[1,] 1 3 0.2880365
[2,] 2 2 0.7785115
[3,] 3 1 0.3297416
> setcolorder(x, c("c", "b", "a"))
> x
             c b a
[1,] 0.2880365 3 1
[2,] 0.7785115 2 2
[3,] 0.3297416 1 3
```

Come cercare aiuto

- Help in linea:
  - `help()`
  - `help.start()`
  - `help("data.frame")`
- Pagina web di Giulio D'Agostini: http://www.roma1.infn.it/~dagos/R/index.html

Un po' di sintassi

- Case sensitive (`nome != Nome`)
- Caratteri alfanumerici piu' "`.`"
- Usuali operatori aritmetici: `+, -, *, /,`
  - E meno usuali `%%, %/%, ^, **, sqrt(x)`
- Operatori logici: `==, !=, >=, <=`
- `is.vector(v)` ?
- Spreadsheet: `> data.entry(x)` interattivo!!!
- '`;`' per separare istruzioni nella stessa riga, '`{}`' per raggrupparle
  - `If (x == 1) {….} else if (x <0) {…} else {}`

Un po' di sintassi

- `for (i in 1:10) {}`
- `while (condition == TRUE) {}`
- `switch()`
- Per assegnare un valore ad una variabile:
  - `x <- 2`
  - `X <- cos(pi)`
- Tutto cio' che segue il carattere '#' non viene eseguito
- `options()`
- `options()$digits`
- `ceiling(x), floor(x), trunc(x, ...), round(x, digits = 0) signif(x, digits = 6)`

Vettori (liste)

- `x <- c(0., 4.5, 1.3, 3.2)` c per 'combine'
  - I vettori cominciano da '1'
- `2*x` moltiplicazione per una costante
- `cos(x), x+y, x/y, x**2`
- `x[1:3]` le prime 3 componenti
- `x[-3]` la 3 componente e' rimossa
- `x[c(-1,-2)]` rimuove la prima e la seconda componente
- `rev(x), sort(x), sum(x), match(4.5, x), length(x)`

matrici

- `mdat1 < c(1, 2, 3, 4, 5, 6)`
  - `dim(mdat1)<-c(2,3)` definisce una matrice 2x3
  - `dim(mdat1)<-c(3,2)` definisce una matrice 3x2
- `mdat <- matrix(c(1,2,3, 11,12,13), nrow = 2, ncol = 3, byrow = TRUE, dimnames = list(c("row1", "row2"), c("C.1", "C.2", "C.3")))`

Funzioni

- ```
  My.square <- function(x)
  { return(x**2) }
  ```
- ```
  My.print2file <- function(x,
  filename="output.txt") {
     write(x, file, 10, FALSE, ", ")
  }
  ```

Grafici

- Ogni sistema operativo ha la sua funzione per gestire la finestra grafica:
  - `windows(), quartz(), X11()`
  - per salvare in un file: `pdf(), png(), jpeg()`
  - Finita l'operazione la funzione che gestisce la interfaccia grafica va chiusa: dev.off()

```
pdf("test.pdf", height=5, pointsize=10)
plot(0)
dev.off()
```

grafici

```
plot(x1)
```

scatter plot x1 on the horizontal axis and x2 on the vertical axis

```
plot(x1,x2)
```

the same but using a formula (x2 as a function of x1)

```
plot(x2 ~ x1)
```

shows all the available methods for plot (depending on the number of loaded packages).

```
methods(plot)
```

Tabelle

- E’ facile produrre tabelle come colenne di vettori affiancate:

```
>a= 1:10; b=1:5; d=1:7
>length(b) = length(d) = length(a)
>d <-data.frame(a,b,d)  # no error now
>d
```

Si possono fare cose molto piu’ sofisticate… vedi esempio tabella.R

Numeri casuali e funzioni di probabilità

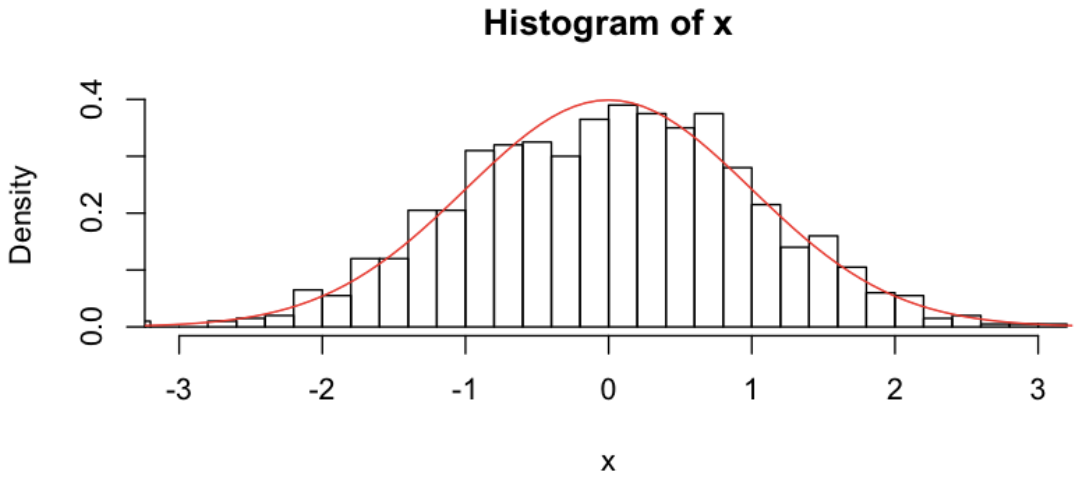
- Un calcolatore puo' generare numeri pseudocasuali, equiprobabili in un certo intervallo.
  - Con certi metodi che verranno illustrati piu' avanti in questo corso e' possibile trasformare un generatore di numeri equiprobabili in un generatore che segue una distribuzione diversa
  - p(x)=cost=G(f(x)) dove G e' la funzione che trasforma le distribuzione a piacere f in p
- Pseudocasuali vuol dire che una sequenza di numeri casuali e' univocamente identificata da un indice/ numero, il seme usato nel generatore. Per semi diversi sequenze diverse

Numeri casuali e funzioni di probabilità

- R conosce le principali pdf:
- dnome(x) =>pdf f(x)
- pnome(x) => cumulativa F(x)
- qnome(p) => quantili: x tale che p=F(x)
- rnome(n) => n numeri pseudo-casuali
- nome: binom, pois, norm, unif, geom, chisq

Istogrammi

- `X <-dnorm(1000)`
- `hist(x)`
- `hist(x, nc=30, prob=TRUE, xlim=c(-3, 3))`
- `plot(dnorm, -4, 4, ty='l', add=TRUE, col='red')`



Esempi

- tabella.R
- plot.R
- funzione.R
- write_file.R
- read_file.R
- covarianza.R

Esercizo

1. generare un capione di 100 valori distribuiti in modo uniforme tra 0 e 1
2. 2. farne un istogramma e salvarlo in pdf
3. varianti con titolo, media e deviazione standard del campione
4. generare un secondo campione di valori a piacere
5. costruire una tabella con i 2 campioni di valori
6. salvare la tabella in un file ed uscire dal programma
7. creare un terzo capmione di dati che sia funzione dei primi 2 ed
8. aggiungerlo alla tabella
7. fare un grafico del primo verso il terzo campione
8. costruire una funzione per calclolare la matrice di covarianza