R minimo

Marco Baroni

LUG BZ, 15 dicembre 2007

Scaletta

R

- `http://www.r-project.org/`
- Variante free, open source del linguaggio/ambiente statistico S di Venables e Ripley
- Gira sotto Linux, Mac e Windows
- Riga di comando e GUI (almeno per Mac e Windows)
  - Sotto Windows, la GUI `www.sciviews.org` è meglio di quella di default
- Utilizzo anche non-interattivio via scripting
- Grazie a modello open source, sono stati sviluppati moltissimi pacchetti che implementano una grande varietà di tecniche classiche e innovative per l'analisi dei dati
- Pacchetti disponibili attraverso il Comprehensive R Archive Network (CRAN):
  `http://cran.r-project.org/`

Libri su R

- Peter Dalgaard, *Introductory Statistics with R*, Springer, 2002
- Michael Crawley, *The R Book*, Wiley, 2007
- W.N. Venables & B.D. Ripley, *Modern Applied Statistics with S, 4th edition*, Springer, 2002
- ...
- Vari libri/tutorial disponibili sul sito; estesa documentazione distribuita con l'installazione

Scaletta

R come calcolatore

```
# per lanciare il programma
# digitare R (maiuscolo) sul terminale

> 1+1
[1] 2

> a <- 2

> a * 2
[1] 4

> log(a)        # log in base e
[1] 0.6931472

> log(a,2)      # log in base 2
[1] 1
```

Gestire la sessione

```
> setwd("path/to/data")

> ls()

> ls                                    # NB!!!

> quit()
> quit(save="yes")
> quit(save="no")

# NB: supporto, almeno in alcune versioni,
# per history recall, tab completion
```

Matematica vettoriale

```
> a <- c(1,2,3)  # c (per combine) crea vettori

> a * 2    # operatori applicati a ciascun elemento di vettore
[1] 2 4 6

> log(a)   # molte funzioni standard prendono vettori come input
[1] 0.0000000 0.6931472 1.0986123

> sum(a)
[1] 6

> length(a)
[1] 3

> sum(a)/length(a)
[1] 2
```

Inizializzare vettori

```
> a <- 1:100                    # sequenza di interi
> a

> a <- 10^(1:100)

> a <- rnorm(100)               # 100 numeri random

> a <- runif(100, 0, 5)         # come in Java etc.
```

Statistiche riassuntive

```
> length(a)

> summary(a)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.02717 0.51770 1.05200 1.74300 2.32600 9.11100

> mean(a)

> median(a)

> sd(a)

> quantile(a)
    0%    25%    50%    75%   100%
0.0272 0.5177 1.0518 2.3261 9.1107

> quantile(a,.75)
```

Scaletta

Grafici di base

```
> a<-2^(1:100)          # non dimenticare le parentesi!
> plot(a)

> x<-1:100              # y in funzione di x
> plot(x,a)

> plot(x,a,log="y")     # vari grafici su scala logaritmica
> plot(x,a,log="x")
> plot(x,a,log="xy")
> plot(log(x),log(a))
```

Grafici di base

```
> hist(rnorm(100))      # istrogramma e stima della densità
> hist(rnorm(1000))
> plot(density(rnorm(100000)))

> pdf("density.pdf")
> plot(density(rnorm(100000)))
> dev.off()
```

Grafici un po' meno di base

```
> a <- rbinom(10000,100,.5)
> hist(a)

> hist(a, probability=TRUE)
> lines(density(a))

> hist(a, probability=TRUE)
> lines(density(a), col="red", lwd=3)

> hist(a, probability=TRUE,
+ main="Some Distribution", xlab="value",
+ ylab="probability")
```

+ è prompt per comandi incompleti!

```
> lines(density(a), col="red", lwd=3)
```

Test statistici di base

```
> a <- rbinom(10000,100,.5)
> b <- rbinom(10000,100,.7)

> t.test(a,b)

> c <- a^2

> cor.test(a,c)
```

Scaletta

Input da file esterni

- Usare altri programmi per formattare dati in file di testo
- Vari formati accettabili, io di solito uso tab-delimited con header:

```
word frequency cat
dog 15 noun
bark 10 verb
```

- Importare in R come “dataframe”:

```
> brown <- read.table("brown.stats.txt",
+ header=TRUE)
```

- Come in vari linguaggi OO, si accede a colonne (vettori) di dataframe con $:

```
> summary(brown$frequency)
```

- In linea di massima, R ci indovina con il tipo dei dati, ma talvolta comandi della famiglia `as.X` (`as.character`, `as.factor`) si rendono necessari

R scripting

- Una lista di comandi R salvati in un file di testo sono già uno script di R (vari editor capiscono la sintassi R)
- Lanciare script in R:

```
> source("my_script.R")
```

- NB: A differenza che in sessioni interattive, occorre usare `print()` esplicitamente per visualizzare i contenuti di una variabile: `print(sd(a))` invece di `sd(a)`
- NB2: Anche se R permette loop da linguaggio di programmazione classico, algoritmi "vettorizzati" sono più efficienti (e.g., `sum(a)` invece di iterazione esplicita sui valori di `a`)

Help!

```
> help("hist")
> ?hist

> help.search("histogram")

> ?help.search

> help.start()    # documentazione in HTML

> demo("graphics")
```