Calcoli ripetuti: ciclo di for

Possiamo ripetere dei segmenti di calcolo senza dover ripetere la scrittura dei comandi.

Per visualizzare i numeri da uno a 10 ed i loro logaritmi naturali possiamo scrivere

```
for (i in 1:10) {cat(i,log(i),"\n")}
```

```
for (indice del ciclo)
   {
    comandi
   }
```

```
> for (i in 1:10) {cat(i,log(i),"\n")}
1 0
2 0.6931472
3 1.098612
4 1.386294
5 1.609438
6 1.791759
7 1.94591
8 2.079442
9 2.197225
10 2.302585
```

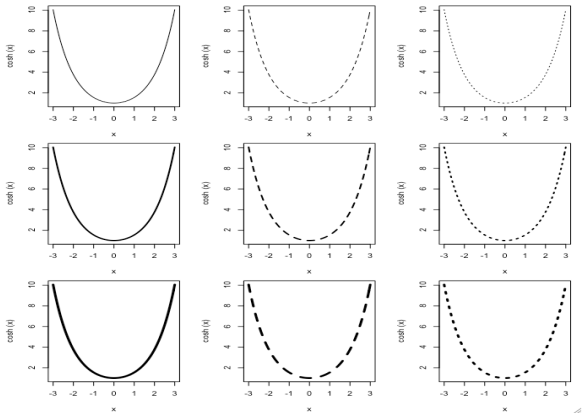
Ripeti i comandi tra la prima e l'ultima parentesi a graffe nel tanto che l'indice i scorre dal valore iniziale della sequenza al valore finale

Esempi

Il comando plot rappresenta le funzioni intrinseche, cioè in cui la x non è espressa nel comando.

Ecco alcuni esempi con diverse modalità per lo spessore della curva ed il tipo di linea

```
par(mfrow=c(3,3))
par(mar=c(4.2,4.8,0.5,1.0))
for(i in 1:3)
{for(j in 1:3)
  {plot
(cosh,-3,3,lwd=i,lty=j)
    }
  }
```

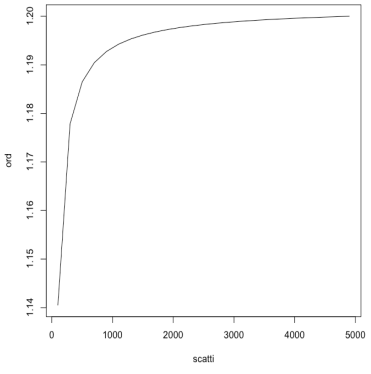


Ciclo di for e concetto di limite

$$\kappa = \sum_{k=2}^{\infty} \frac{1}{k^2} \sum_{j=1}^{k-1} \frac{1}{j} \quad (\approx 1.201)$$

```
scatti<-seq(100,5000,by=200)
conta<-0
m<-length(scatti)
ord<-matrix(0,m)
for (n in scatti)
{
kappa <- 0;conta<-conta+1
for (k in 2:n)
{ j <- 1:(k-1)
kappa <- kappa + sum(1/j)/k^2
}
ord[conta]<-kappa
cat(n,kappa,"\n")
}
plot(scatti,ord,type="l")
```

```
100 1.140516
300 1.177825
500 1.186488
700 1.190453
900 1.192751
1100 1.19426
1300 1.195331
1500 1.196132
1700 1.196755
1900 1.197255
2100 1.197664
2300 1.198006
2500 1.198297
2700 1.198547
2900 1.198764
3100 1.198955
3300 1.199124
3500 1.199275
3700 1.199410
3900 1.199533
4100 1.199644
4300 1.199745
4500 1.199837
4700 1.199923
4900 1.200001
```



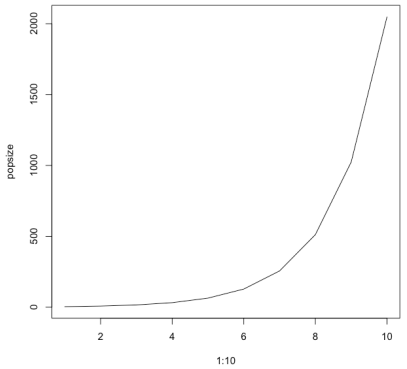
Esempio di ciclo di for

L'ipotesi è che la popolazione si raddoppi ogni anno

```
# Popolazione al livello iniziale
initsize=4;
# Creamo un vettore per memorizzare gli acrescimenti
popsize=rep(0,10); popsize[1]=initsize;
# Calcoliamo il livello della popolazione ai tempi da 2 e fino a 10
for (n in 2:10)
   {
    popsize[n]=2*popsize[n-1];
    x=log(popsize[n]);    cat(n,x,"\n");
    }
plot(1:10,popsize,type="l");
```

```
2 2.079442
3 2.772589
4 3.465736
5 4.158883
6 4.85203
7 5.545177
8 6.238325
9 6.931472
10 7.624619
```

N.B. le ordinate sono in scala logaritmica



Cicli annidati

I cicli possono essere annidati (uno interno all'altro, ma mai incrociati)

Un ciclo deve essere interamente interno ad un altro oppure avviarsi solo dopo che l'altro è finito

```
A=matrix(0,3,3);
for (row in 1:3)
   {
   for (col in 1:3)
      {
       A[row,col]=row*col
      }
   }
A;
```

Il risultato dovrebbe essere

```
[,1] [,2] [,3]
[1,] 1 2 3
[2,] 2 4 6
[3,] 3 6 9
```

Cicli annidati/2

```
p=rep(0,5)
for (init in c(1,5,9))
   {
   p[1]=init;
   for (n in 2:5)
      {
       p[n]=2*p[n-1]
       cat(init,n,p[n],"\n");
      }
   }
```

```
1 2 2
1 3 4
1 4 8
1 5 16
5 2 10
5 3 20
5 4 40
5 5 80
9 2 18
9 3 36
9 4 72
9 5 144
```

```
for (i in 1:10)
  {
   i1<-i+1
   for (j in i1:15)
      {
      cat(letters[i],LETTERS[j],"\n")
          }
}
```

```
> z<-matrix(nrow=2,ncol=4)
> for(i in 1:2){
+ for(j in 1:4) z[i,j]<-i+j}
> z
[,1] [,2] [,3] [,4]
[1,] 2  3  4  5
[2,] 3  4  5  6
```

Funzioni

Alcune elaborazioni debbono essere ripetute con dati diversi o con parametri diversi.

E' possibile scrivere più volte le stesse istruzioni, ma non è necessario dato che si possono compattare le istruzioni da reiterare in un blocco che si attiva su chiamata.

Il segmento di programma che contiene le istruzioni parametetrate è la function

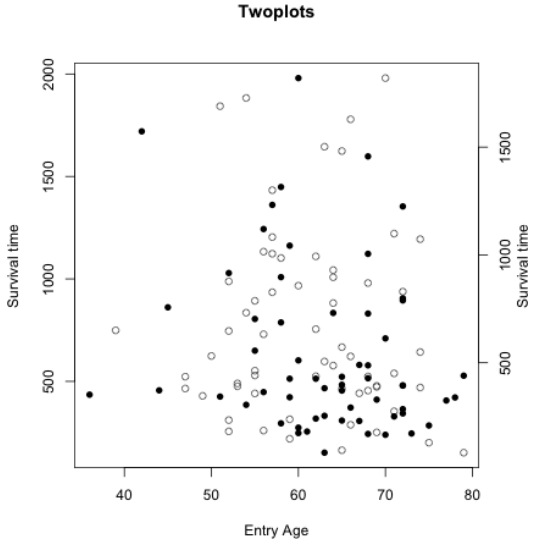
W<-function(x,y,z){}

- **W è il nome con cui il blocco verrà richiamato**
- **x, y e z sono gli arogmenti attraverso I quali saranno veicolati I valori su cui il blocco dovrà operre**
- **Tutte le istruzioni e gli oggetti comprese tra le due parentesi graffe fanno parte della funzione**

Esempio_1: funzioni

```
Twoplots<-
function(x1, y1, x2, y2, type = "l",xlim=NULL,ptitle="Twoplots",xlab
="x",ylab1="y1",ylab2 = "y2", ...)# Gli argomenti non debbono essere tutti
indicati. In mancanza, la routine usa i valori di default
{# Written by AE York Sept. 1996
# Plots y1 vs x1 and y2 vs x2 on the same plot
# Labels for y1 are on the left and labels for y2 on the right
        par(mar = rep(6, 4))    # margini del grafico
        if(missing(xlim))
                xlim <- range(pretty(range(c(x1, x2))))
        plot(x1, y1, lty = 1, pch = 1, type = type, xlim = xlim, xlab =
                xlab, ylab = ylab1, ...)
        par(new = T)
        plot(x2, y2, lty = 2, pch = 16, type = type, axes = F, xlim = xlim,
                xlab = "", ylab = "")
        axis(side = 4) # aggiunge l'asse a destra
        mtext(ylab2, side = 4, line = 2, outer = F)
        title(ptitle)}
Dat<-read.table("YJW.csv",sep=",",header=TRUE);attach(Dat);head(Dat)
J<-which(DummyAB==0) # Individuzione delle posizioni che hanno lo "0"
Asc1<-Entry.Age[J];Ord1<- Survtime[J]
Asc2<-Entry.Age[-J];Ord2<- Survtime[-J]
Twoplots(Asc1,Ord1,Asc2,Ord2,"p",xlim=c(min(min(Asc1,Asc2)),max(max
(Asc1,Asc2))),xlab="Entry Age",ylab1="Survival time",ylab2="Survival time")
```

Esempio_1: funzioni/2



Data coding

```
bmd <- c(-0.92,0.21,0.17,-3.21,-1.80,-2.60,
         -2.00,1.71,2.12,-2.11)
diagnosis <- bmd
diagnosis[bmd <= -2.5] <- 1
diagnosis[bmd > -2.5 & bmd <= 1.0] <- 2
diagnosis[bmd > -1.0] <- 3
data <- data.frame(bmd, diagnosis)
```

```
data
```

```
      bmd   diagnosis
1   -0.92         3
2    0.21         3
3    0.17         3
4   -3.21         1
5   -1.80         2
6   -2.60         1
7   -2.00         2
8    1.71         3
9    2.12         3
10 -2.11          2
```

```
diagnosis <- bmd
diagnosis <- replace(diagnosis, bmd <= -2.5, 1)
diagnosis <- replace(diagnosis, bmd > -2.5 & bmd <= 1.0,
2)
diagnosis <- replace(diagnosis, bmd > -1.0, 3)
```

Ricodifica

Alcune variabili sono state registrate con delle modalità nominali che potrebbero essere inadatte per le elaborazioni

Basso medio alto ----> 1 2 3

```
library(MASS)
data(Cars93)
A<-Cars93
print(A[,9])
A[,9]<-ifelse(A[,9]=="Driver & Passenger",3,ifelse(A[,9]==
"Driver only",2,1))
print(A[,9])
A[,11]<-ifelse(A[,11]=="rotary",6,A[,11])
A[,18]<-ifelse(A[,18]==2,3,A[,18]);A[,18]<-A[,18]-2;
A[,16]<-ifelse(A[,16]=="No",0,1)
A[,26]<-ifelse(A[,26]=="USA",1,0)
print(cbind(Cars93[,c(11,18,26)],A[,c(11,18,26)]))
```

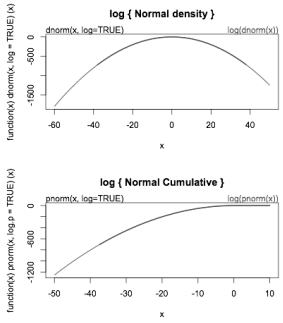
Modelli probabilistici univariati

Alcuni modelli per variabili continue o dense

Gaussiano con denominazione base "norm"

```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean = 0, sd = 1)
```

```
## Using "log = TRUE" for an extended range :
par(mfrow=c(2,1))
plot(function(x) dnorm(x, log=TRUE), -60, 50,
     main = "log { Normal density }")
curve(log(dnorm(x)), add=TRUE, col="red",lwd=2)
mtext("dnorm(x, log=TRUE)", adj=0)
mtext("log(dnorm(x))", col="red", adj=1)

plot(function(x) pnorm(x, log.p=TRUE), -50, 10,
     main = "log { Normal Cumulative }")
curve(log(pnorm(x)), add=TRUE, col="red",lwd=2)
mtext("pnorm(x, log=TRUE)", adj=0)
mtext("log(pnorm(x))", col="red", adj=1)
```

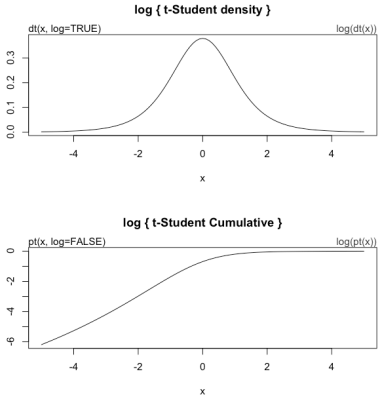


Modelli probabilistici univariati/2

t-Student con denominazione base “t”

```
dt(x, df, ncp, log = FALSE)
pt(q, df, ncp, lower.tail = TRUE, log.p = FALSE)
qt(p, df, ncp, lower.tail = TRUE, log.p = FALSE)
rt(n, df, ncp)
```

```
par(mfrow=c(2,1))
plot(function(x) dt(x, 5,0,log=FALSE),-5,5,
     main = "log { t-Student density }")
curve(log(dt(x)), add=TRUE, col="red",lwd=2)
mtext("dt(x, log=TRUE)", adj=0)
mtext("log(dt(x))", col="red", adj=1)

plot(function(x) pt(x, 5,0, log.p=TRUE), -5, 5
     main = "log { t-Student Cumulative }")
curve(log(pt(x)), add=TRUE, col="red",lwd=2)
mtext("pt(x, log=FALSE)", adj=0)
mtext("log(pt(x))", col="red", adj=1)
```

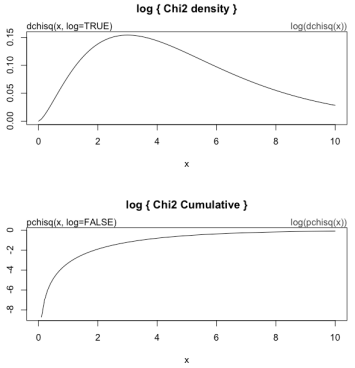


Modelli probabilistici univariati/3

chi2 con denominazione base “chisq”

```
dchisq(x, df, ncp=0, log = FALSE)
pchisq(q, df, ncp=0, lower.tail = TRUE, log.p = FALSE)
qchisq(p, df, ncp=0, lower.tail = TRUE, log.p = FALSE)
rchisq(n, df, ncp=0)
```

```
par(mfrow=c(2,1))
plot(function(x) dchisq(x, 5,0,log=FALSE),0,10,
     main = "log { Chi2 density }")
curve(log(dchisq(x)), add=TRUE, col="red",lwd=2)
mtext("dchisq(x, log=TRUE)", adj=0)
mtext("log(dchisq(x))", col="red", adj=1)

plot(function(x) pchisq(x, 5,0, log.p=TRUE), 0, 1
     main = "log { Chi2 Cumulative }")
curve(log(pchisq(x)), add=TRUE, col="red",lwd=2)
mtext("pchisq(x, log=FALSE)", adj=0)
mtext("log(pchisq(x))", col="red", adj=1)
```

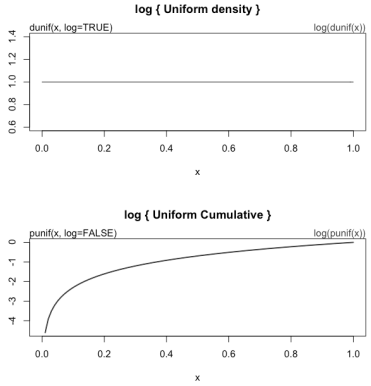


Modelli probabilistici univariati/4

Uniforme con denominazione base “unif”

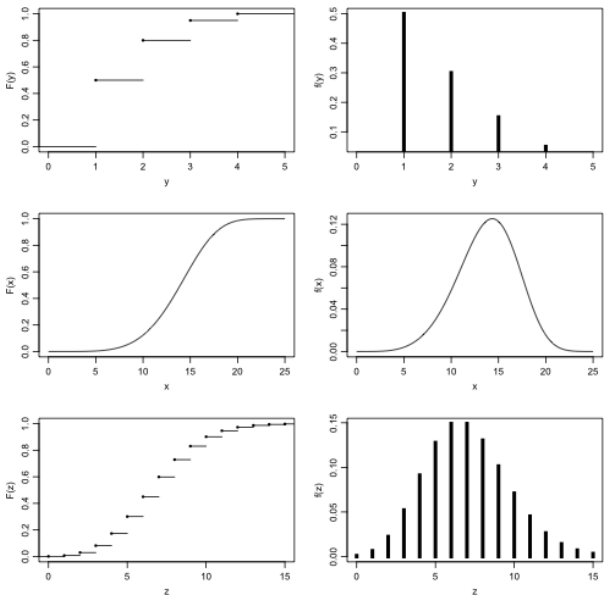
```
dunif(x, min=0, max=1, log = FALSE)
punif(q, min=0, max=1, lower.tail = TRUE, log.p = FALSE)
qunif(p, min=0, max=1, lower.tail = TRUE, log.p = FALSE)
runif(n, min=0, max=1)
```

```
par(mfrow=c(2,1))
plot(function(x) dunif(x, 0,1,log=FALSE),0,1,
     main = "log { Uniform density }")
curve(log(dunif(x)), add=TRUE, col="red",lwd=2)
mtext("dunif(x, log=TRUE)", adj=0)
mtext("log(dunif(x))", col="red", adj=1)

plot(function(x) punif(x, 0,1, log.p=TRUE), 0, 1,
     main = "log { Uniform Cumulative }")
curve(log(punif(x)), add=TRUE, col="red",lwd=2)
mtext("punif(x, log=FALSE)", adj=0)
mtext("log(punif(x))", col="red", adj=1)
```



Grafica per i modelli

```
 windows(width=6, height=7)
 par(mfcol=c(3,2),mai=c(0.6,0.5,0.1,0.1),mgp=c(2,0.7,0))
y<-c(1,2,3,4);Fy<-c(0,0.5,0.8,0.95,1.0)
plot(stepfun(y,Fy),xlab="y",ylab="F(y)",verticals=FALSE pch=16,main=NA)
# Diameter distribution, c.d.f.
 x<-seq(0,25,0.1);Fx<-pweibull(x,5,15)
plot(x, Fx, type="l", xlab="x", ylab="F(x)")
# Number of trees, c.d.f.
z<-seq(0,20,1);Fz<-ppois(z,7) # Distribuzione di Poisson
plot(stepfun(z,c(0,Fz)),xlab="z",ylab="F(z)",verticals=FALSE,pch=16,
main=NA,xlim=c(0,15))
# Tree species, density.
fy<-c(0.5,0.3,0.15,0.05)
plot(y, fy, type="n",xlim=c(0,5), xlab="y", ylab="f(y)")
sapply(1:4,function(i) lines(y[c(i,i)],c(0,fy[i]),lwd=4,lend="square"))
# Diameter distribution, density.
fx<-dweibull(x,5,15);plot(x,fx,type="l",xlab="x",ylab="f(x)")
 # Number of trees, density.
 fz<-dpois(z,7)
plot(z,fz,xlim=c(0,15),type="n",xlab="z",ylab="f(z)")
sapply(1:20,function(i) lines(z[c(i,i)],c(0,fz[i]),lwd=4,lend="square"))
```



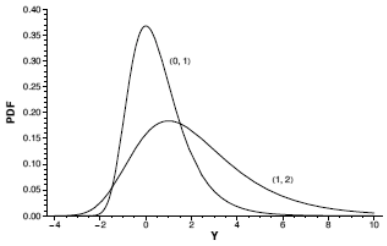
Esercizio

Effettuare lo stesso studio con I seguenti modelli

- **weibull**
- **Lognormale (lnorm)**
- **beta**
- **gamma**
- **Esponenziale (exp)**

Modelli non presenti in R

Gumbel(A,B) B > 0



Modella l'andamento del massimo di un campione estratto da una popolazione di tipo esponenziale

$$PDF = \frac{1}{B}\exp\left(\frac{A-y}{B}\right)\exp\left(-\exp\left(\frac{A-y}{B}\right)\right)$$

$$CDF = \exp\left(-\exp\left(\frac{A-x}{B}\right)\right)$$

Parameters -- A (ξ): Location, B (θ): Scale

Moments, etc.

$$Mean = A + \gamma B$$

$$Variance = \frac{1}{6}(\pi B)^2$$

$$Skewness = \frac{12\sqrt{6}\,\zeta(3)}{\pi^3} \approx 1.1395$$

$$Kurtosis = \frac{12}{5}$$

$$Mode = A$$


```
par(mfrow=c(2,1))
dgumbel <- function(x,a,b) 1/b*exp((a-x)/b)*
exp(-exp((a-x)/b))
pgumbel <- function(q,a,b) exp(-exp((a-q)/b))
qgumbel <- function(p,a,b) a-b*log(-log(p))
plot(function(x) dgumbel(x, 5,10),-10,30,
     main = "log { gumbelorm density }")
curve(log(dgumbel(x)), add=TRUE, col="red",lwd=2)
mtext("dgumbel(x, log=TRUE)", adj=0)
mtext("log(dgumbel(x))", col="red", adj=1)

plot(function(x) pgumbel(x, 5,10), -10, 30,
     main = "log { gumbelorm Cumulative }")
curve(log(pgumbel(x)), add=TRUE, col="red",lwd=2)
mtext("pgumbel(x, log=FALSE)", adj=0)
mtext("log(pgumbel(x))", col="red", adj=1)
```

Modelli probabilistici univariati/5

Alcuni modelli per variabili discrete

binom	Binomial (size, prob)
hyper	hypergeometric (m, n, k)
nbinom	negative binomial (size, prob)
pois	Poisson (lamda)
geom	geometric (prob)

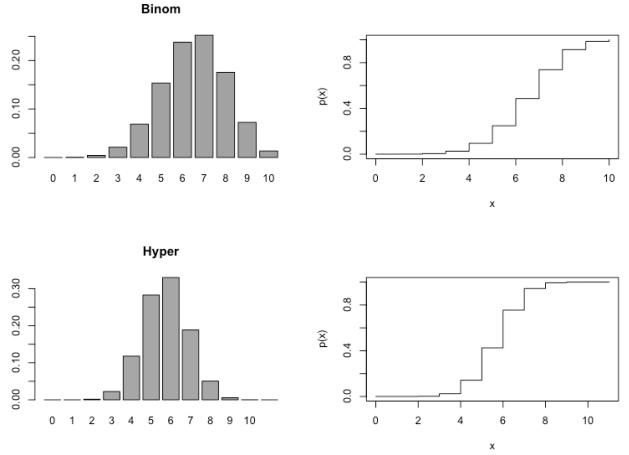
four all discrete distributions you can use the 4 functionspxxxx CDFdxxxx pmf (d for discrete)qxxxx quantilerxxxx random variates

Binomiale

```
plot(0:100,pbinom(0:100,size=100,prob=0.22))
#look at the figure and guess binomial distribution
dbinom(3,size=20,prob=0.4) #pmf(3) Binomial distribution, n=20 p=0.4
dbinom(3,20,0.4) #short version of the above command
dbinom(3.5,20,0.4)

pbinom(3,20,0.4) #CDF(x) Binomial distribution, n=20 p=0.4
sum(dbinom(1:3,20,0.4))
#these commands also work for vectors! why is it not the same as pbinom above??
sum(dbinom(0:3,20,0.4))# now it is the same

qbinom(0.93,20,0.4)#quantile (inverse CDF)
pbinom(11,20,0.4)
pbinom(10,20,0.4)
rbinom(100,20,0.4)#random vector of length 100 of binomial distribution

hist(rbinom(100,20,0.4))#show the histogram with frequencies
hist(rbinom(10000,20,0.4),freq=FALSE)#show the histogram with relative frequencies
plot(0:20,dbinom(0:20,20,0.6),type="s")#plot of the pmf; "s" stands dor stairs

# Calculate  Porb(X>=10) by simulation
 mean(runif(1000000)>0.8)
# calculates the empirical probability that uniform 0,1 variate larger than 0.8
```

Esempio

```
par(mfrow=c(2,2))
barplot(dbinom(0:10,10,0.65), col="coral", names.arg=0:10,main="Binom")
plot(0:10, pbinom(0:10, 10, 0.65), type="s", xlab="x", ylab="p(x)")

barplot(dhyper(0:11,12,9,10), col="salmon", names.arg=0:11,main="Hyper")
plot(0:11, phyper(0:11, 12, 9,10), type="s", xlab="x", ylab="p(x)")
```



Istogramma con curva

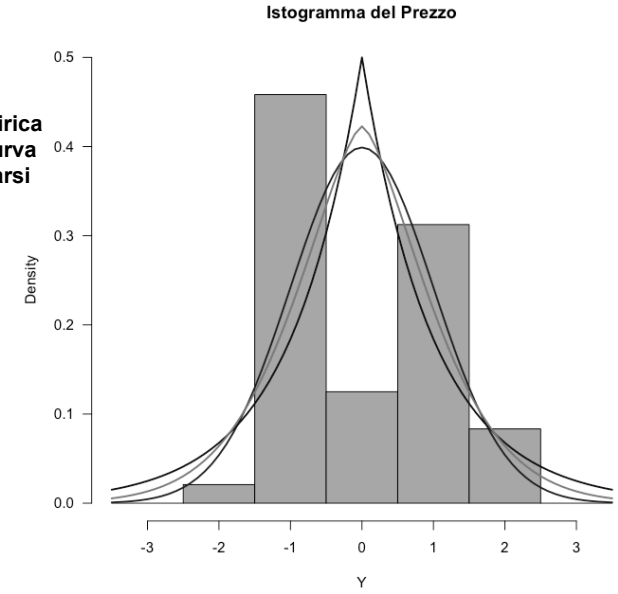
Prezzo medio per carato in dollari di un anello con diamanti sul mercato di Singapore.

```
library(normalp)
Y<-scan("PMKarat.txt")
Y<-scale(Y) # sottrazione della media e divisione per lo scarto
Y[Y < -3.5 | Y > 3.5] <- NA # Esclusione dei valori troppo grandi
x <- seq(-3.5, 3.5, .1)
dn <- dnorm(x)
par(mar=c(4.5, 4.1, 3.1, 0))
hist(Y, breaks=seq(-3.5, 3.5), ylim=c(0, 0.5), col="lightcoral", freq=FALSE,
main="Istogramma del Prezzo")
lines(x, dnorm(x), lwd=2,col="red4")
lines(x,dnormp(x,p=1),lwd=2,col="blue")
lines(x,dnormp(x,p=1.5),lwd=2,col="green4")
par(mar=c(5.1, 4.1, 4.1, 2.1))
```

Si tenta di modellare I dati con varie curve di tipo gaussiano esponenziale

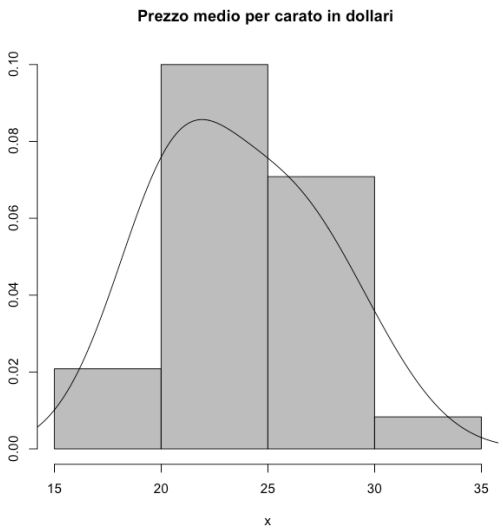
Esempio

Se la distributione empirica è bimodale, nessuna curva teorica potrà mai adattarsi ad essa in modo soddisfacente



Comando density

```
x<-scan("PMKarat.txt")
idq<-summary(x)[5]-
summary(x)[2]
truehist
(x,probability=TRUE,col
="plum",main=
"Prezzo medio per carato in
dollari ")
lines(density
(x,width=2*idq))
```



Esempio/1

```
#Istogramma delle frequenze e spline interpolante
par(mfrow=c(1,1),mar=c(4,4,2,2))
attach(attitude)
hist(rating,breaks="sturges",freq=FALSE,main="Istogramma
delle frequenze",xlab="Modalità osservate",ylab="Densità di
frequenza",col="lightblue",border="red")
Dens<-density(rating)
xval<-range(Dens$x);yval<-range(Dens$y)
# yval[2]<-yval[2]+0.0025
hist(rating,breaks="sturges",freq=FALSE,main="Istogramma
delle frequenze",xlab="Modalità osservate",ylab="Densità di
frequenza",col="lightblue",border="red",probability=TRUE,xlim
=xval,ylim=yval)
lines(Dens,col="blue",lwd=2,lty=2)
```

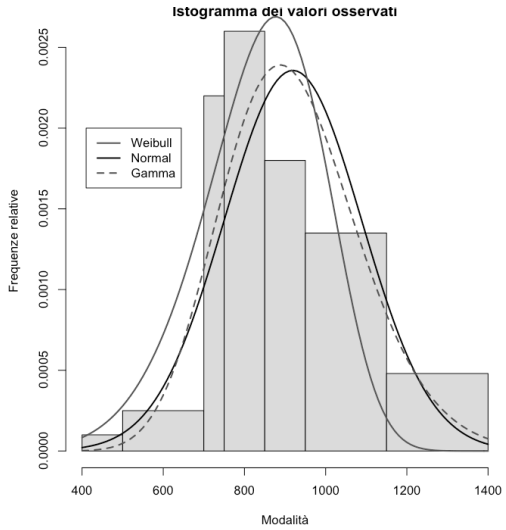
```
density(x, bw = "nrd0", adjust = 1,
        kernel = c("gaussian", "epanechnikov", "rectangular",
                   "triangular", "biweight",
                   "cosine", "optcosine"),
        weights = NULL, window = kernel, width,
        give.Rkern = FALSE,
        n = 512, from, to, cut = 3, na.rm = FALSE, ...)
```

Adattamento di una distribuzione teorica

```
hist(Nile,main="Istogramma dei valori
osservati",ylab="Frequenze relative",xlab="Modalità",breaks=c
(400,500,700,750,850,950,1150,1400 ),freq=FALSE,right=TRUE,col
="lightgreen",prob=TRUE)
sdn<-sd(Nile);xdn<-mean(Nile)
curve(dnorm(x,mean=xdn,sd=sdn),add=TRUE,lwd=2,col="navy")
cvn<-xdn/sdn^2;hdn<-xdn^2/sdn^2
curve(dweibull(x,shape=6.5,scale=900),from=400,
to=1400,add=TRUE,lwd=2,col="magenta")
curve(dgamma
(x,rate=cvn,shape=hdn,),add=TRUE,lwd=2,lty=2,col="brown")
legend(410,0.0020,legend=c("Weibull","Normal","Gamma"),col=c
("magenta","navy","brown"),lwd=c(2,2,2),lty=c(1,1,2))
```

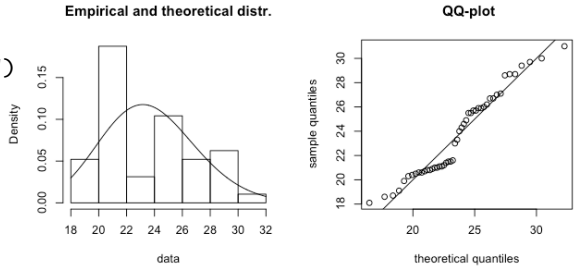
Esempio (continua)

Si può notare la unimodalità dell'istogramma che viene in effetti colta dai modelli. Nessuno di essi sembra però individuare la asimmetria verso i valori grandi

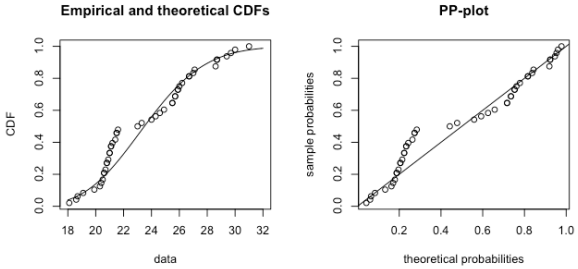


Adattamento di una distribuzione/2

```
library(fitdistrplus)
x<-scan("PMKarat.txt")
f1c <- fitdist(x,"gamma")
print(f1c)
plot(f1c)
```

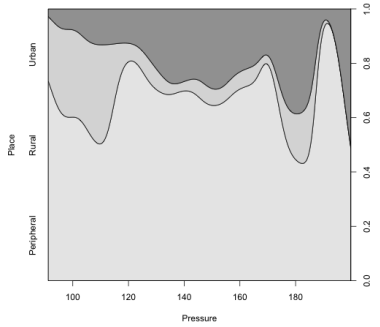


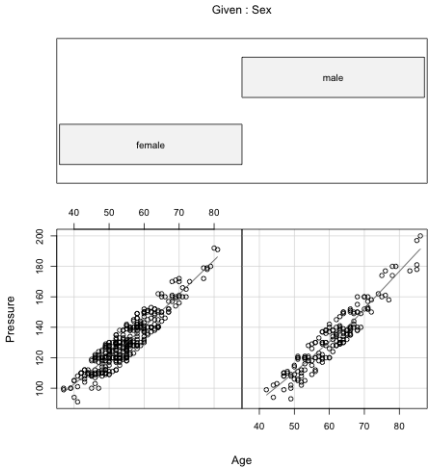
Se l'adattamento fosse buono, gli scarti nei grafici PP e QQ sarebbero piccoli



Densità condizionate

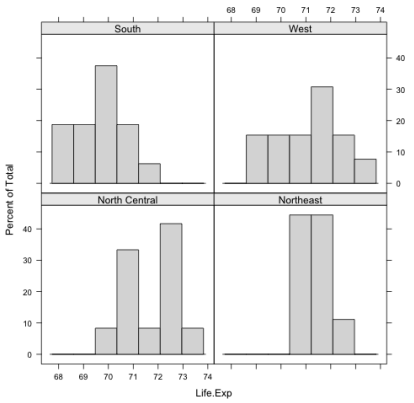
```
Older<-read.table("UrbGen.csv",header=T,sep=",")
attach(Older)
graphics.off()      # reset/close all graphical devices
cdplot(Place~Pressure,bw="nrd0",col=c("moccasin","gold","peru"))
coplot(Pressure ~ Age | Sex, panel=panel.smooth)
```

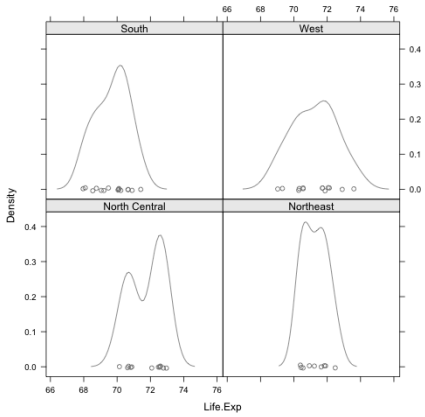




Confronto di ditribuzioni

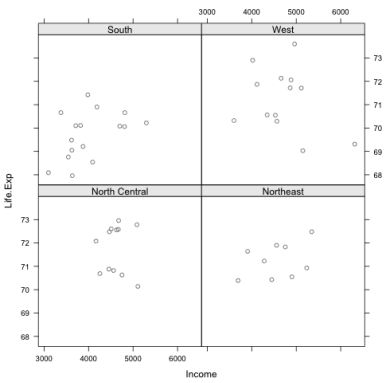
```
USstates <- read.table("USstates.txt",header=T,sep=",")
attach(USstates)
summary(USstates)  # summary stats
library(lattice)                # enhanced plotting environment
histogram(~ Life.Exp | Region)  # Life.exp split in 3 classes
densityplot(~ Life.Exp | Region)
```

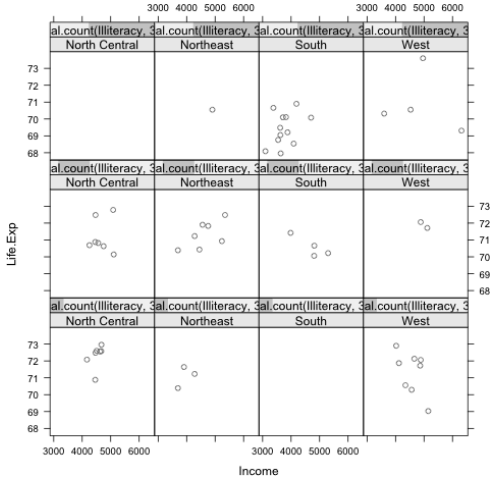




Confronto di relazioni

```
xyplot(Life.Exp ~ Income| Region)
xyplot(Life.Exp ~ Income| Region*equal.count(Illiteracy,3,0))
```





Integrazione numerica

Per calcolare il valore di un integrale definito di una certa funzione in un dato intrervallo si usa il comando **integrate**

```
integrate(f, lower, upper, ..., subdivisions=100,
rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol,
stop.on.error = TRUE, keep.xy = FALSE, aux = NULL)
```

$$\int_0^{\pi/2} 4\sin(2x)\exp(-x^2)dx$$

```
integrand<-function(x) 4*sin(2*x)*exp(-x^2)
print(Area<-integrate(integrand,0,pi/2))
```

2.190752 with absolute error < 2.4e-14

```
integrand<-function(x) 4*sin(2*x)*exp(-x^2)
print(Area<-integrate(integrand,0,Inf))
```

2.152318 with absolute error < 2.7e-05

Spiegate il risultato inferiore

Campionamento

```
Population<-c
(3,4,12,5,11,35,24,21,29,38,11,17,34,6,22,13,19,24,38,2,17,
+ 15,28,31,29,11,23,34,3,12,4,28,39,16,22,31,37,19)
N<-length(Population);n<-floor(0.20*N)
set.seed(262657)
Sam1<-sample(Population,n,replace=TRUE)
print(Sam1)
Sam2<-sample(Population,n,replace=FALSE)
print(Sam2)
Sam3<-sample(Population,n,replace=TRUE,prob=1/(Population^2))
print(Sam3)
Sam4<-sample(Population,n,replace=FALSE,prob=1/(Population^2))
print(Sam4)
Sam5<-sample(Population,n,replace=TRUE,prob=1/(Population^0.1))
print(Sam5)
Music<-c
("Puccini","Rossini","Verdi","Mascagni","Paisiello","Rendano","Donize
tti","Bellini","Leoncavallo","Monteverdi")
Best<-sample(Music,4,replace=FALSE)
print(Best)
```

Generazione di numeri pseudo casuali

La simulazione di numeri casuali si basa su sequenze di numeri che nascono da ben definite relazioni matematiche e per questo si parla di numeri pseudocasuali.

Pur conservando un precipuo carattere deterministico si comportano come una successione di variabili casuali.

L'evoluzione dei generatori di numeri ha una sua pietra miliare nei generatori congruenziali lineari introdotti da Lehmer (1949).

La loro formula è

$$X_n = (aX_{n-l} + c)\ Mod\ m; \quad n=1,2, ..., m$$

$$p \equiv q\ Mod\ m \text{ significa che } \quad p = q - \left\lfloor \frac{q}{m} \right\rfloor m$$

ovvero *p è il resto intero della divisione di q per m*

Generazione di numeri pseudo casuali/2

Le sequenze ottenute dalla sono deterministiche: dato un certo termine è possibile calcolarne qualsiasi altro che interverrà in successione,

La conseguenza è che ogni successione è riproducibile: per ripeterla tutta è sufficiente conservarne il valore iniziale.

Il comando in R è

RNGkind(kind = NULL, normal.kind = NULL)

Esistono diversi algoritmi per ottenere numeri pseudo casuali di qualità.
In particolare

kind="Wichmann-Hill" (da me sperimentato con successo in molte occasioni)

kind="Mersenne-Twister" (scelta di default) che ha un periodo molto elevato (2^19937)-1 e si mantiene efficace anche nel caso di generazioni multivariate.

Generazione di numeri pseudo casuali/3

Dal generatore si ottengono valori pseudo casuali nell'intervallo unitario

Per ottenere valori da altre distribuzioni ci si può basare sul seguente risultato teorico

Teorema
Se la variabile casuale X ha funzione di densità f(x) allora la variabile casuale

$$u = \int_{-\infty}^{x} f(t)dt$$

ha densità uniforme sull'intervallo [0,1].

Ad esempio, per i modelli che rientrano nella famiglia Burr

$$F(x) = \frac{1}{1+e^{\{-g(x)\}}} \qquad x = g^{-1}\left[Ln\left(u/(1-u) \right) \right]$$

Per ottenere una x con distribuzione F si usa una una uniforme u e la si trasforma

Esempio: distribuzione logistica

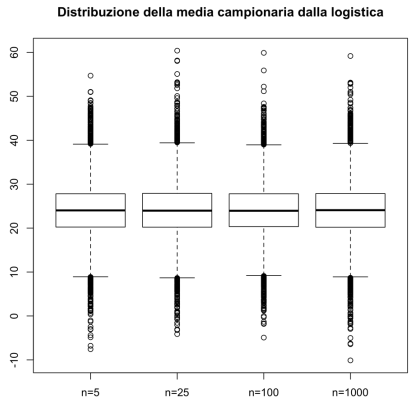
$$F(x;\mu,\sigma) = \frac{1}{1+\exp\left[-\left(\frac{x-\mu}{\sigma}\right)\right]}; \qquad \sigma > 0$$

$$x = \mu + \sigma \log\left(\frac{u}{1-u}\right) \qquad u \sim uniform\ [0,1]$$

```
Rlogis <- function(n,mu, sigma)
 {u <- rep(runif(1),n)
  return(mu + sigma* log(u/(1-u)))}
```

Simulazione della distribuzione di una media campionaria

```
Rlogis <- function(n,mu, sigma)
{u <- rep(runif(1),n)
 return(mu + sigma* log(u/(1-u)))}
set.seed(820731)
mu<-24;sigma<-3.5
Numsim<-10000
mean5<-rep(0,Numsim);mean25<-mean5
mean100<-mean5;mean1000<-mean5
for (i in 1:Numsim)
{mean5[i]<-mean(Rlogis(5,mu,sigma))
mean25[i]<-mean(Rlogis(25,mu,sigma)
mean100[i]<-mean(Rlogis(100,mu,sigma))
mean1000[i]<mean(Rlogis
(1000,mu,sigma))}
boxplot
(mean5,mean25,mean100,mean1000,nam
es=c("n=5","n=25","n=100","n=1000"))
title("Distribuzione della media campionaria
dalla logistica")
```



Esempio: medie progressive

```
#esponenziale: F(x)=1-e(-lambda x)
lambda<-1;numsim<-200
mean5<-rep(0,numsim);mean25<-mean100<-mean5;mean200<-mean5
for (i in 1:numsim){
        mean5[i]<-mean(rexp(5,lambda))
        mean25[i]<-mean(rexp(25,lambda))
        mean100[i]<-mean(rexp(100,lambda))
        mean200[i]<-mean(rexp(200,lambda))
        }
boxplot(mean5,mean25,mean100,mean200,names=c
("n=5","n=25","n=100","n=200"))
title=("distribuzione della media per campioni dalla esponenziale")
######
Medprog<-function(x){
        n<-length(x)
        ret<-rep(0,n)
        for (k in 1:n){ret[k]<-mean(x[1:k])}
        return(ret)}
x<-rnorm(100, mean=0, sd=1)
y<-rcauchy(100, location = 0, scale = 1)
par(mfrow=c(1,2));plot(Medprog(x),type="l")
title("Medie progressive dalla normale standardizzata")
plot(Medprog(y),type="l")
title("Medie progressive dalla Cauchy")
```

Integrazione Monte Carlo

Vogliamo calcolare un integrale che ha una funzione integranda complicata

$$I = \int_a^b g(x)dx$$

Generiamo n valori indipendenti da una uniforme in [a,b] e calcoliamo

$$\hat{I} = (b-a)\frac{1}{n}\sum_{i=1}^{n} g(\mathrm{X}_i)$$

Per la legge dei grandi numeri si ha

$$\hat{I}_n \to (b-a)\mathbf{E}\big[g(\mathbf{X})\big]$$

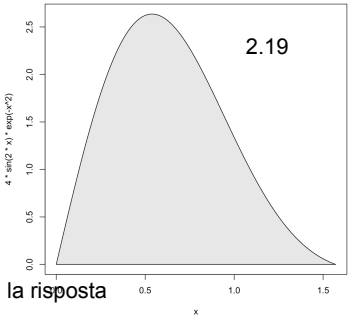
Il valore atteso di g(x) in [a,b] è dato da

$$\mathrm{E}\big[g(\mathrm{X})\big] = \int_a^b g(x)\frac{1}{b-a}dx = \left(\frac{1}{b-a}\right)I$$

Quindi I_n è una approssimazione di I che migliora all'aumentare di n

Esempio

$$\int_0^{\pi/2} 4\sin(2x)\exp(-x^2)dx$$




```
curve(4*sin(2*x)*exp(-x^2),0,pi/2,201)
x1<-seq(0,pi/2,length=201)
y1<-4*sin(2*x1)*exp(-x1^2)
x2<-seq(0,pi/2,length=201)
y2<-4*sin(2*x2)*exp(-x2^2)
polygon(c(x1,x2),c(y1,y2),col="lavender")
u <- runif(1000000, min=0, max=pi/2)
pi/2*mean(4*sin(2*u)*exp(-u^2))
# a=0, b=pi/2, so b–a=pi/2#
Se non fissa il seed del generatore di numeri casuali la risposta
# sarà diversa per la stessa chiamata
u <- runif(1000000, min=0, max=pi/2)
pi/2*mean(4*sin(2*u)*exp(-u^2)) # a=0, b=pi/2, so b–a=pi/2
# Fissato il seed il calcolo sarà lo stesso ad ogni call. Ad esempio
set.seed(820731)
u <- runif(1000000, min=0, max=pi/2)
pi/2*mean(4*sin(2*u)*exp(-u^2)) # a=0, b=pi/2, so b–a=pi/2
set.seed(820731)
u <- runif(1000000, min=0, max=pi/2)
pi/2*mean(4*sin(2*u)*exp(-u^2)) # a=0, b=pi/2, so b–a=pi/2
```

Verifica della normalità

Una parte considerevole delle studio inferenziale si regge sull'ipotesi di normalità delle osservazioni.

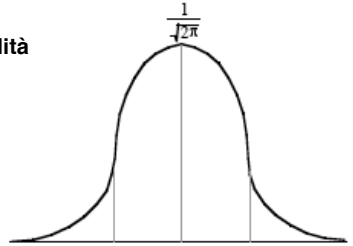
Se I valori non sono normali molte procedure conservano ancora efficienza purché

Gli errori si mostrino simmetrici

Le code si mostrino leggere

Grazie anche a queste condizioni la normalità diventa un'ipotesi plausibile per campioni moderatamente grandi.

E' quindi opportuno asseverare se i valori osservati siano riconducibili alla ipotesi di normalità



QQ_plot

Il grafico è ottenuto riportando in un sistema di assi cartesiano I valor (standardizzati ed ordinati) ed i quantili della Normale corrispondenti (da cui: grafico quantile-quantile)

In dettaglio

$$\left[Z(p_i),\ \frac{\hat{e}_{(i)}}{\hat{\sigma}_i}\right]; \quad i = 1,2,\ \ldots,n$$

Il pedice tra parentesi indica valori ordinati in senso ascendente.

$Z(p_i)$ sono i quantili della normale standardizzata corrispondenti alla frequenza cumulata p_i associata con $e_{(i)}$ osservato.

Se il grafico somiglia molto ad una retta che passa per l'origine e con inclinazione unitaria (la bisettrice del quadrante per intenderci), l'approssimazione normale è considerata buona.

Posizioni grafiche

La scelta dei quantili cui far corrispondere $e_{(i)}$ stimato è controversa.

Scartata la soluzione $p_i=i/n$ che è inadatta per un supporto infinito (raggiunge subito l'unità per i=n), ci si è orientati sulla seguente formula generale

$$p_i(\alpha,\beta) = \frac{i-\alpha}{n+1-(\alpha+\beta)}$$

Dove α e β sono tali da assicurare valori crescenti compresi tra zero ed uno.

I valori di questi parametri cambiano da distribuzione a distribuzione.

Spesso si sceglie $\alpha=\beta$

Scelta delle posizioni grafiche

Name	α	β	$p_i(\alpha,\beta)$
0) *Hazen*	0.5	0.5	$\left(\frac{i-0.5}{n}\right)$
1) *Weibull*	0	0	$\left(\frac{i}{n+1}\right)$
2) *Blom*	0.375	0.375	$\left(\frac{i-0.375}{n+0.25}\right)$
3) *Landwehr*	0.35	0.65	$\left(\frac{i-0.35}{n}\right)$
4) *Tukey*	0.3333	0.3333	$\left(\frac{i-0.3333}{n+0.3333}\right)$
5) *Cunnane*	0.4	0.4	$\left(\frac{i-0.4}{n+0.2}\right)$
6) *Benard*	0.3	0.3	$\left(\frac{i-0.3}{n+0.4}\right)$
7) *Filliben*	0.3175	0.3175	$\left(\frac{i-0.3175}{n+0.65}\right)$
8) *Gringorten*	0.44	0.44	$\left(\frac{i-0.44}{n+0.12}\right)$
9) *Larsen*	0.567	0.567	$\left(\frac{i-0.3175}{n-0.134}\right)$

La formula di Weibull gode di una certa popolarità.

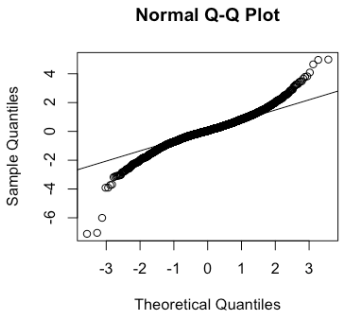
Quella che si consiglia è la Landwehr

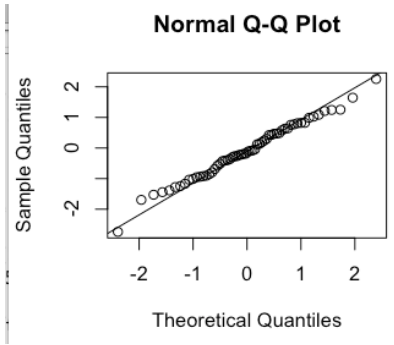
$p_i(\alpha,\beta)$ è una approssimazione dei quantili della distribuzione uniforme nell'intervallo unitario.

Grazie al teorema dell'inversione possiamo ottenere i quantili di moltissime distributioni a partire proprio da $p_i(\alpha,\beta)$

Esempio

```
x<-rnorm(60)
qqnorm(x);qqline(x)
###
library(MASS)
SP500
qqnorm(SP500)
qqline(SP500)
```



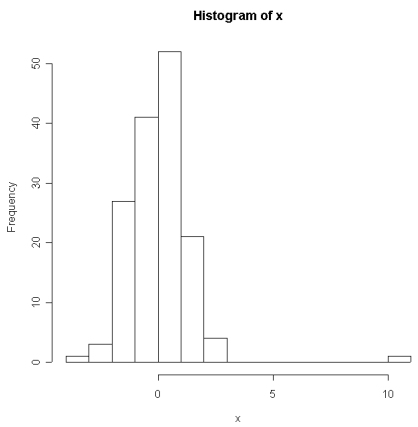


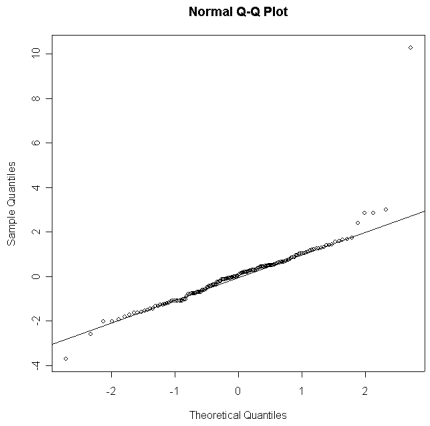
R utilizza le posizioni di Weibull

Il grafico Q-Q è un semplice ed utile ausilio per accertare l'aderenza alla Normale ovvero a qualsiasi distribuzione di cui siano noti i quantili.

Casi di disnormalità: valori remoti

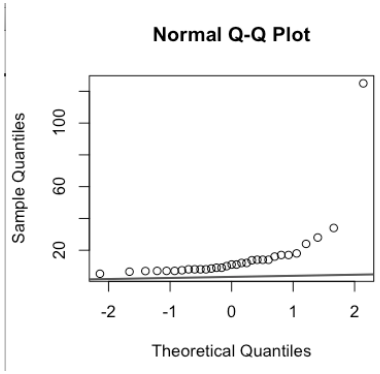
I casi di osservazioni estreme (alla luce del campione osservato) sono subito evidenti nel grafico QQ

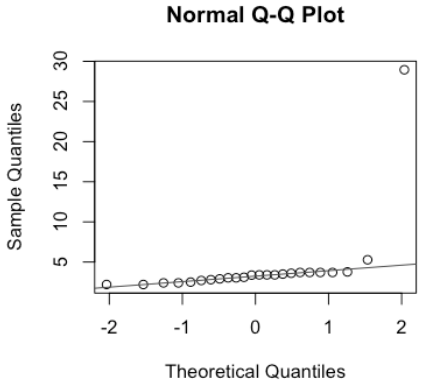




Esempio

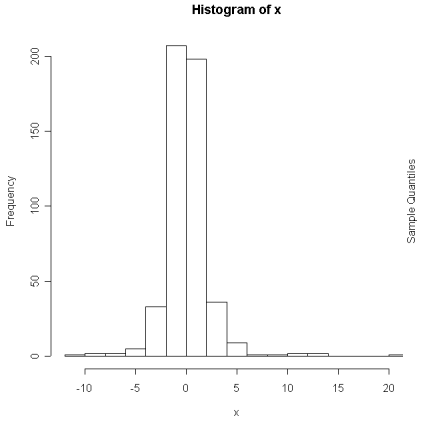
```
library(MASS)
qqnorm(chem)
qqline(chem)
###
qqnorm(abbey)
qqline(chem,col = 2,lwd=2)
```

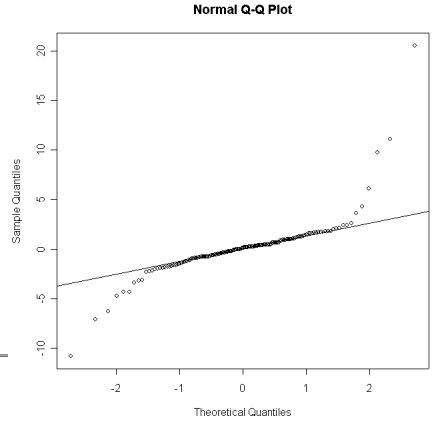




Casi di disnormalità: code pesanti

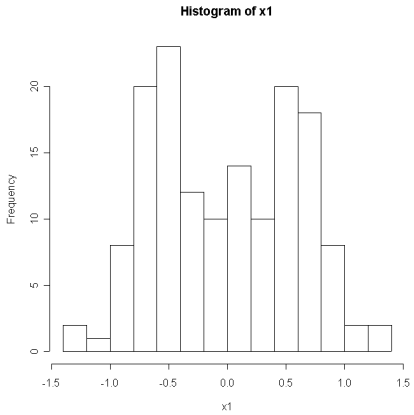
Le code pesanti si mostrano con la presenza di varie osservazioni per valori troppo bassi e/o troppo alti rispetto a ciò che ci si aspetta in un fenomeno normale

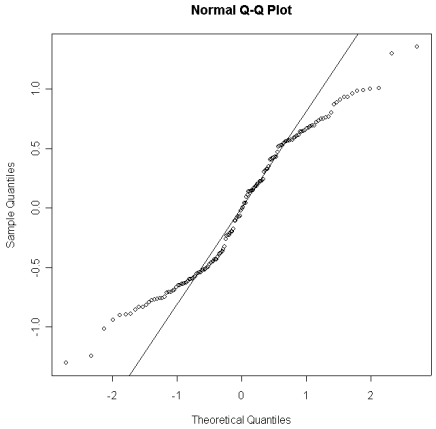




Casi di disnormalità: code smorzate

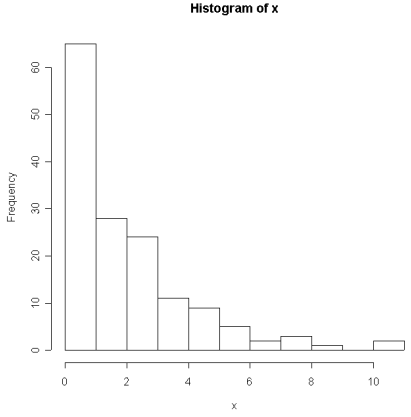
Le code smorzate si mostrano con l'assenza di osservazioni per valori bassi e/o alti laddove un fenomeno normale produrrebbe diversi valori sia pure con frequenza decrescente

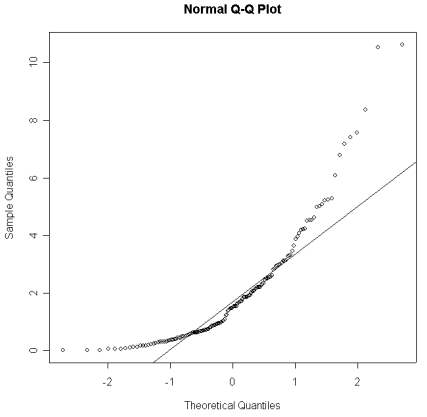




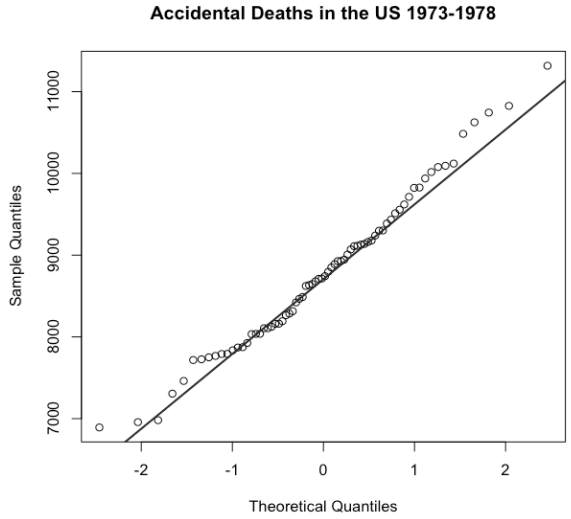
Casi di disnormalità: asimmetria

L'asimmetria della distribuzione si riscontra nella presenza di una forte curvatura del diagramma QQ.





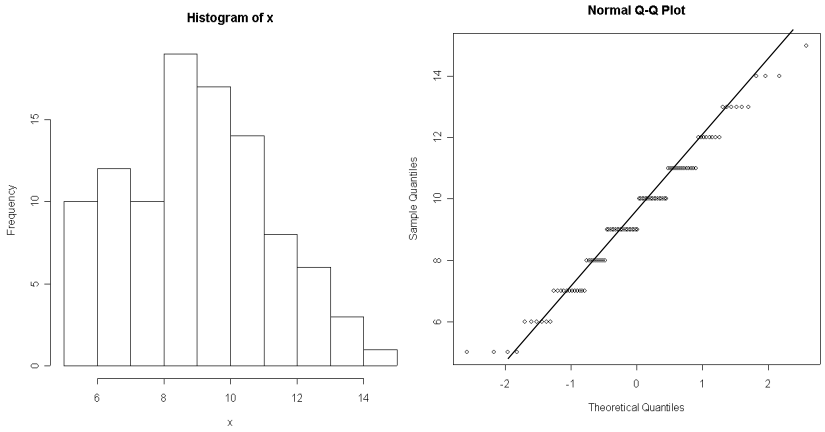
```
library(MASS)
qqnorm(accdeaths,main="Accidental Deaths in the US 1973-1978")
qqline(accdeaths,col = 4,lwd=2)
```



Casi di disnormalità: livellamenti

Il verificarsi di valori ripetuti (a causa di unità di misura troppo grezze) si traduce in una deviazione dalla normalità.

Anche la presenza di gap (vuoti) tra i valori è segno di disnormalità



Algebra delle matrici

E' la tabella che si ottiene ricopiando la matrice con le righe al posto delle colonne.

L'operatore è t()

A cosa è uguale
C<-t(t(A)) ?

```
> A<-matrix(1:12,3,4,byrow=TRUE)
> A
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
> B<-t(A)
> B
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

Trasposta di una matrice

E' la tabella che si ottiene ricopiando la matrice con le righe al posto delle colonne.

L'operatore è t()

A cosa è uguale
C<-t(t(A)) ?

```
> A<-matrix(1:12,3,4,byrow=TRUE)
> A
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
> B<-t(A)
> B
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

Uso dell'operatore diag()

La diagonale delle matrici quadrate ha sempre un ruolo speciale nelle applicazioni pratiche dell'algebra lineare

Matrice identità

$$I = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

```
> A<-matrix(0,5,5);diag(A)<-1
> A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    1    0    0    0
[3,]    0    0    1    0    0
[4,]    0    0    0    1    0
[5,]    0    0    0    0    1
```

A<-diag(5)

Recupero della diagonale

```
> A<-matrix(c(1,-17,4,pi,0,8,2,-3/4,-exp(1)),3,3,byrow=TRUE)
> A
         [,1]   [,2]      [,3]
[1,] 1.000000 -17.00  4.000000
[2,] 3.141593   0.00  8.000000
[3,] 2.000000  -0.75 -2.718282
> D<-diag(A)
> D
[1]  1.000000  0.000000 -2.718282
```

$$A = \begin{pmatrix} 1 & -17 & 4 \\ \pi & 0 & 8 \\ 2 & -3/4 & -e \end{pmatrix}$$

Traccia e determinante di una matrice

La traccia è data dalla somma sugli elementi della diagonale

$$Tr(A) = \sum_{i=1}^{n} a_{i,i}$$

Il determinante di una matrice è una somma di prodotti degli elementi della matrice combinati secondo un procedimento ben preciso

A partire dalle definizioni

$$X = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & -2 \\ -1 & 0 & 7 \end{bmatrix} \quad Y = \begin{bmatrix} 6 & 0 & 0 \\ -3 & 4 & 0 \\ 0 & -5 & 2 \end{bmatrix}$$

Calcoliamo la traccia ed il determinante delle due matrici

- TrX<-sum(diag(X)); TrY<-sum(diag(Y));
- DtX<-det(X);DtY<-det(Y)

Matrici triangolari

Le matrici triangolari sono strutture che si incontrano in varie situazioni, ad esempio nella soluzione dei sistemi di equazioni

$$T_L = \begin{bmatrix} a & e & h & j \\ 0 & b & f & i \\ 0 & 0 & c & g \\ 0 & 0 & 0 & d \end{bmatrix}; \quad T_{L\not d} = \begin{bmatrix} 0 & a & e & g \\ 0 & 0 & b & f \\ 0 & 0 & 0 & c \\ 0 & 0 & 0 & 0 \end{bmatrix}; \quad T_U = \begin{bmatrix} a & 0 & 0 & 0 \\ e & b & 0 & 0 \\ h & f & c & 0 \\ j & i & g & d \end{bmatrix}; \quad T_{U\not d} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ e & b & 0 & 0 \\ g & f & c & 0 \end{bmatrix};$$

```
H<-matrix(c(1:5,2:6,3:7,4:8,5:9),5,5);H
#
HL<-H;HL[upper.tri(H,diag=F)]<-0HL
HLd<H;HLd[upper.tri(H,diag=T)]<-0;HLd
HU<H;HU[lower.tri(H,diag=F)]<-0;HU
HUd<-H;HUd[lower.tri(H,diag=T)]<-0;HUd
```

```
> # Hankel matrix
> H<-matrix(c(1:5,2:6,3:7,4:8,5:9),5,5);H
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]    2    3    4    5    6
[3,]    3    4    5    6    7
[4,]    4    5    6    7    8
[5,]    5    6    7    8    9
```

Prodotto (interno) di Matrici

Si realizza grazie all'operatore

% * %

Attenzione! Il solo asterisco definisce il prodotto di HADAMARD tra matrici.

E' diverso dal prodotto interno perché moltiplica le matrici elemento per elemento corrispondente

```
> x
  [,1] [,2] [,3] [,4]
A    1    2    3    4
B    5    6    7    8
C    9   10   11   12
> x * x
  [,1] [,2] [,3] [,4]
A    1    4    9   16
B   25   36   49   64
C   81  100  121  144
> x %*% t(x)
    A   B   C
A  30  70 110
B  70 174 278
C 110 278 446
```

Illustrazione

I costi unitari di trasferimento ed il numero di spedizioni da quattro magazzini per tre punti vendita sono i seguenti

Costi unitari

	M_1	M_2	M_3	M_4
V_1	10	15	9	7
V_2	14	8	12	8
V_3	6	14	22	17

Spedizioni

	M_1	M_2	M_3	M_4
V_1	4	8	7	2
V_2	1	0	2	9
V_3	0	5	3	6

Costi totali

	M_1	M_2	M_3	M_4
V_1	40	120	63	14
V_2	14	0	24	72
V_3	0	70	66	102

La matrice dei costi totali è ottenuta come prodotto di Hadamard tra la matrice dei costi unitari per la matrice del numero di spedizioni

```
CU<-matrix(c(10,15,9,
7,14,8,12,8,6,14,22,17),nrow=3,ncol=4,byrow=TRUE,dimnames=list(c("V.
1","V.2","V.3"),c("M1","M2","M3","M4")))
```

1) Costruire la Matrice
2) Realizzare il prodotto di Hadamard
3) Realizzare il prodotto t(CU) per SPE e interpretare il risultato

1) Mostrare che le due matrici

$$A = \begin{bmatrix} 1 & 5 & -5 \\ 3 & 2 & -5 \\ 6 & -2 & -5 \end{bmatrix}; \quad B = \begin{bmatrix} -3 & 2 & -6 \\ -3 & 5 & -7 \\ -2 & 3 & -4 \end{bmatrix}$$

Hanno lo stesso determinante. (-5)

2) E' plausibile che due matrici diverse abbiano lo stesso determinante?

3) Calcolare il determinante delle due trasposte e del prodotto delle due matrici

1.4 Norms

A *norm* on a vector space V over $\mathbb{C}$ is a real valued function $\|\cdot\|$ on V satisfying three axioms:

1. $\|v\| \geq 0$ for all $v \in V$ and $\|v\| = 0$ if and only if $v = 0$.
2. $\|cv\| = |c|\,\|v\|$ for all $c \in \mathbb{C}$ and $v \in V$.
3. $\|u + v\| \leq \|u\| + \|v\|$ for all $u, v \in V$ (*triangle inequality*).

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|$$

$$\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2}$$

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|$$

1) Manhattan: > N1<-sum(abs(A))
2) Euclidea
3) Minkowski
4) Thebychev

Frobenius: tr(A'A)

$$\|A\|_{\mathrm{F}} = \left(\sum_i \sum_j |a_{ij}|^2 \right)^{1/2}$$

INVERSA

```
> X
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    2    0
[3,]    0    0    3
```

La funzione `solve()` serve per risolvere sistemi di equazioni lineari, ma può essere utilizzata per il calcolo della matrice inversa

```
> solve(X) #l'inversa
     [,1] [,2]      [,3]
[1,]    1  0.0 0.0000000
[2,]    0  0.5 0.0000000
[3,]    0  0.0 0.3333333
> X%*%solve(X) #....verifica
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

essendo `%*%` l'operatore 'prodotto-matriciale' (riga × colonna)

In una matrice è possibile sostituire completamente una linea (riga o colonna), ammesso che le dimensioni corrispondano. Ad esempio, per la seconda riga:

```
> x[2,]<-rep(2,5)
> x
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    2    2    2    2
```

A differenza dei vettori, le matrici sono caratterizzate da una coppia di numeri, e la funzione `dim()` è utilizzata per restituire il numero di righe e colonne

library(MASS)
ginv(A)

Soluzione di un sistema lineare

$$\begin{cases} 5x_1 + 3x_2 + 2x_3 = 4 \\ 3x_1 + 9x_2 + 8x_3 = 6 \\ 4x_1 + 2x_2 + 2x_3 = 3 \end{cases}$$

```
A<-matrix(c(5,3,2,3,9,8,4,2,2),nrow=3, byrow=T)
b<-c(4,6,3)
print(A);print(b)
Sol<-solve(A,b)
print(round(Sol,2))
library(MASS)
A1<-ginv(A)
Sol1<-A1 %*% b
print(round(Sol1,2))
```

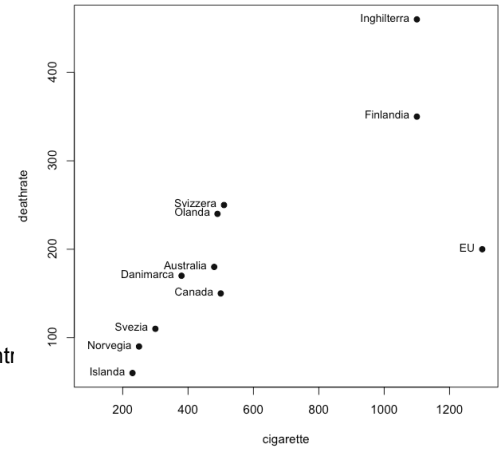
```
> A<-matrix(c(5,3,2,3,9,8,4,2,2),nrow=3, byrow=T)
> b<-c(4,6,3)
> print(A);print(b)
     [,1] [,2] [,3]
[1,]    5    3    2
[2,]    3    9    8
[3,]    4    2    2
[1] 4 6 3
> Sol<-solve(A,b)
> print(round(Sol,2))
[1] 0.5 0.5 0.0
```

```
> library(MASS)
> A1<-ginv(A)
> Sol1<-A1 %*% b
> print(round(Sol1,2))
     [,1]
[1,]  0.5
[2,]  0.5
[3,]  0.0
```

Scatterplot

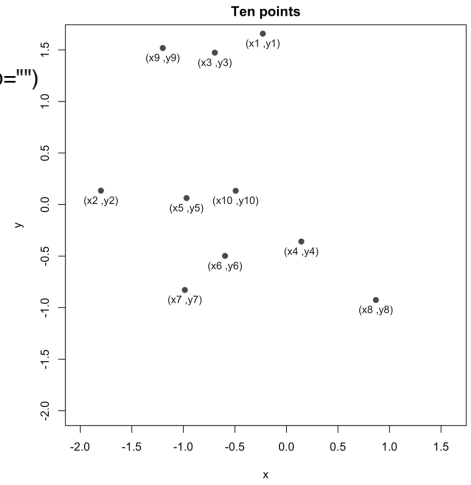
Consumo di sigarette pro-capite nel 1930 e morti di tumore al polmone nel 1950

```
country<-c("Australia" , "Canada",
"Danimarca" ,"Finlandia",
"Inghilterra" , "Islanda" , "Olanda" ,
"Norvegia", "Svezia", "Svizzera",
"EU" )
cigarette<-c(480,500,380,1100,
1100,230,490,250,300,510,1300)
deathrate<-c(180,150,170,350,
460,60,240,90,110,250,200)
plot(cigarette,deathrate,pch=19,
col="blue",xlim=c(100,1300))
text(cigarette,deathrate,label=countr
cex=0.9,pos=2)
```



Esempio

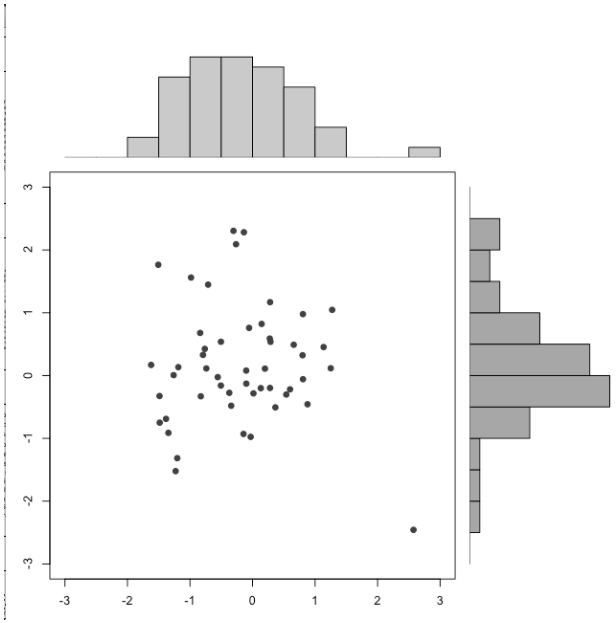
```
par(mfrow=c(1,1),mar=c(4,4,2,2))
set.seed(820731)
x <- rnorm(10)
y <- rnorm(10)
 l <- paste("(x",1:10," ,y",1:10,")", sep="")
plot(x,y,xlim=c(-2,1.6),ylim=c(-2,1.6),
pch=19,col="brown")
 text(x,y,l,pos=1,col="blue",cex=0.9)
title("Ten points")
```



Scatterplot con istogrammi a lato

```
set.seed(271828)
x <- pmin(3, pmax(-3, stats::rnorm(50)))
y <- pmin(3, pmax(-3, stats::rnorm(50)))
xhist <- hist(x, breaks=seq(-3,3,0.5), plot=FALSE)
yhist <- hist(y, breaks=seq(-3,3,0.5), plot=FALSE)
top <- max(c(xhist$counts, yhist$counts))
xrange <- c(-3,3);yrange <- c(-3,3)
nf <- layout(matrix(c(2,0,1,3),2,2,byrow=TRUE), c(3,1), c
(1,3), TRUE)
layout.show(nf)par(mar=c(3,3,1,1))
plot(x, y, xlim=xrange, ylim=yrange, xlab="",
ylab="",pch=19,col="red")
par(mar=c(0,3,1,1))barplot(xhist$counts, axes=FALSE, ylim=c
(0, top), space=0,col="skyblue")
par(mar=c(3,0,1,1))barplot(yhist$counts, axes=FALSE, xlim=c
(0, top), space=0, horiz=TRUE,col="sienna1")
par(def.par)#- reset to default
```

Notare l'impatto del valore anomalo



Esempio sulla regressione

Input the data from the cigarette example

```
cig<-c(480,500,380,1100,1100,230,490,250,300,510,1300)
 death<-c(180,150,170,350,460,60,240,90,110,250,200)
cor(cig,death)
```

will produce the correlation 0.737345

To estimate the parameters use the lm command ('linear model')
lm(death~cig)

The output is

Coefficients:
(Intercept) cig
   67.5609   0.2284

summary(lm(death~cig)) gives a more detailed output

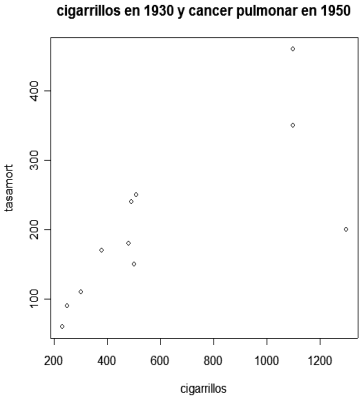
Aggiunta di informazioni

Adding titles. After obtaining the plot with

plot(cigarette,deathrate)
we write

title(main="cigarettes in 1930 and lung cancer in 1950")

b. In some plots, the title is included in the command itself.
hist(cigarette, main="percapita cigarette consumption in 1930")



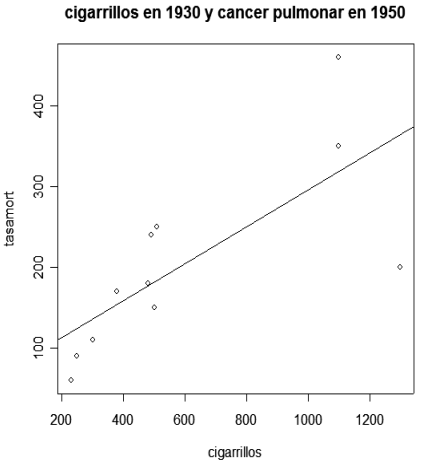
Aggiunta di informazioni/2

After we have obtained the scatter plot we can use the command **abline** to add a line; we need to indicate the intercept and the slope of the line.

Example:
First to get the plot

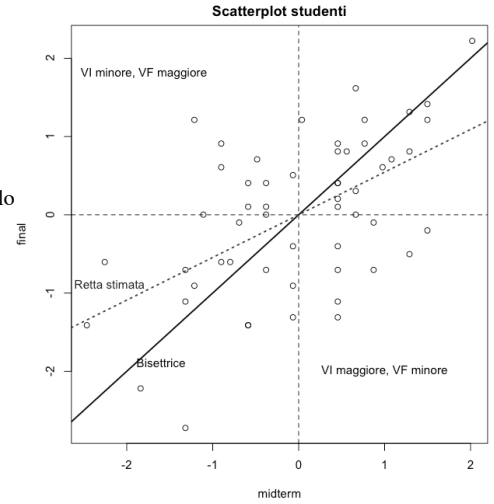
plot(cigarette,deathrate)
and then we write

abline(67.56 , 0.22844)



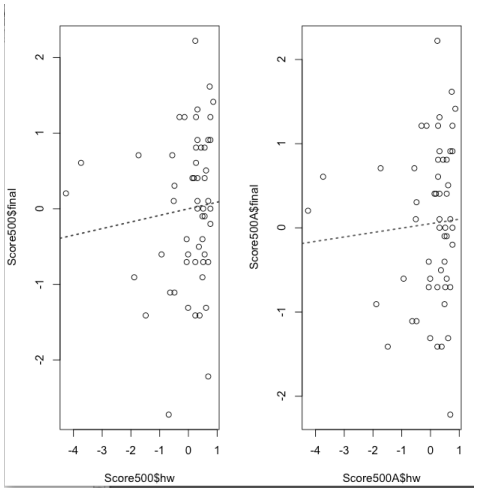
Regressione OLS

```
par(mfrow=c(1,1),mar=c(4,4,2,2))
library(faraway)
data(stat500)
Score500<-scale(stat500)
Score500<-data.frame(Score500)
names(Score500)
plot
(final~midterm,Score500,main="Scatterplo
t studenti")
abline(h=0,lty=2);
abline(v=0,lty=2)
text(-1.8,1.8,"VI minore, VF maggiore")
text(1,-2,"VI maggiore, VF minore")
abline(0,1,col="blue",lwd=2)
text(-1.6,-1.9,"Bisettrice",col="navy")
Stim<-lm(final~midterm,Score500)
abline(Stim$coef,lty=3,lwd=2,col="red")
text(-2.2,-0.9,"Retta stimata",col="red4")
```



Regressione OLS/2

```
par(mfrow=c(1,2),mar=c(4,4,2,2))
cor(Score500)
plot(midterm,total)
#########################
par(mfrow=c(1,2),mar=c(4,4,2,2))
plot(Score500$hw,Score500$final)
Ols<-lm(final~hw,data=Score500)s
ummary(Ols)
abline(Ols$coef,lty=3,lwd=2,col="red")
Score500
A<-data.frame(Score500[-47,])
plot(Score500A$hw,Score500A$final)
OlsA<-lm(final~hw,data=Score500A)
summary(OlsA)
abline(OlsA
$coef,lty=3,lwd=2,col="darkgreen")
```

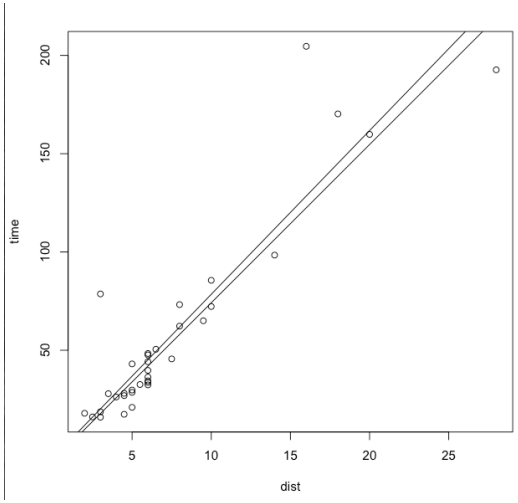



```
### ******** Lecture Code ********** ####

## Pickup data visualization
pickup <- read.csv("pickup.csv");names(pickup)

## R's summary function is pretty clever
summary(pickup)
# Histograms
par(mfrow=c(1,3)) # break the plot into 1x3 matrix (mfrow="multiframe row-wise")
hist(pickup$year, col=grey(.4), border=grey(.9))
hist(pickup$miles, col=grey(.4), border=grey(.9))
hist(pickup$price, col=grey(.4), border=grey(.9))
# Scatterplots
par(mfrow=c(1,2))
plot(price~year, col=make, data=pickup, pch=20)
plot(price~miles, col=make, data=pickup, pch=20)
legend("topright", fill=1:3, legend=levels(pickup$make), cex=.7)
# Boxplot
par(mfrow=c(1,1))
plot(log(price)~make, data=pickup, col=8)
#### Housing data: just price (in $100,000) vs size (in 1000 sq.ft.) ######
size <- c(.8,.9,1,1.1,1.4,1.4,1.5,1.6,1.8,2,2.4,2.5,2.7,3.2,3.5)
price <- c(70,83,74,93,89,58,85,114,95,100,138,111,124,161,172)
plot(size, price, pch=20)
print( n <- length(size) )

## Simple regression
reg <- lm(price ~ size)
b1 <- cor(price,size)*sd(price)/sd(size)
b0 <- mean(price) - mean(size)*b1
cbind(b0,b1)
```

```
# Labstat1. Esempi anche con regressione descrittiva
#### ******** Example 1: Rent Data ********* #####
rent <- read.csv("rent.csv")
par(mfrow=c(1,3))
boxplot(Rent ~ Bathrooms, data=rent)
boxplot(Rent ~ AC, data=rent)
boxplot(Rent ~ Parking, data=rent)
## Bathrooms looks like the most influential factor
plot(Rent ~ Rooms, data=rent, pch=20, col=as.factor(Bathrooms))
plot(Rent ~ YearBuilt, data=rent, pch=20, col=as.factor(Bathrooms))
plot(Rent ~ SqFt, data=rent, pch=20, col=as.factor(Bathrooms))
legend("topright", fill=1:3, legend=levels(as.factor(rent$Bathrooms)), title="Baths")
#
## Sq Ft looks the most influential, and it also correlates well with the # of bathrooms
par(mfrow=c(1,3))
boxplot(Rent, col=7, xlab="marginal", data=rent)
boxplot(Rent ~ Bathrooms, xlab="Bathrooms",main = "Rent Distribution", col=7, data=rent)
boxplot(Rent ~ Rooms, xlab="Rooms", col=7, data=rent)
# Looks like rent increases with the # of (bath)rooms
## There are some very high SqFt places which look like outliers
## removing these in R is very easy, but you could also just do it in excel
rent <- rent[rent$SqFt < 25]
## run the regression with 'lm'
reg <- lm(Rent ~ SqFt, data=rent)
summary(reg)
## Forecasting
yhat = reg$coef[1] + 14.8*reg$coef[2]
## clunky, but useful later with high-dimensional data
predict(reg, newdata=list(SqFt=14.8), se.fit=TRUE, interval="prediction")
```

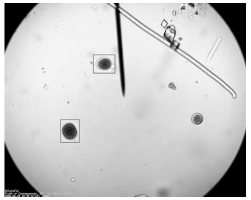
Regressione OLS e regressione robusta

```
library(MASS)
data(hills)
attach(hills)
plot(dist,time)
hillslm1<- lm(time~dist)
summary(hillslm1)
lines(abline(hillslm1))
hillslm2<- rlm(time~dist)
summary(hillslm2)
lines(abline(hillslm2,col="blue"))
```



Esempio_2: regressione lineare semplice

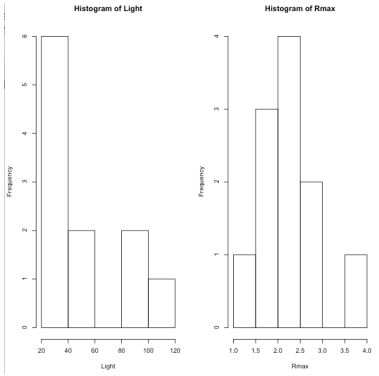
Dati sul saggio massimo di accrescimento (Rmax) di una alga *Chlorella vulgaris* in relazione alla intensità della luce (Light) misurata in µE per m^2 al secondo



```
Light<-c(20,20,20,20,21,24,44,60,90,94,101)
Rmax<-c
(1.73,1.65,2.02,1.89,2.61,1.36,2.37,2.08,2.69,2.32,3.67)
```

Primo passo di ogni indagine è l'analisi grafica. In Rmax c'è un interessante moda

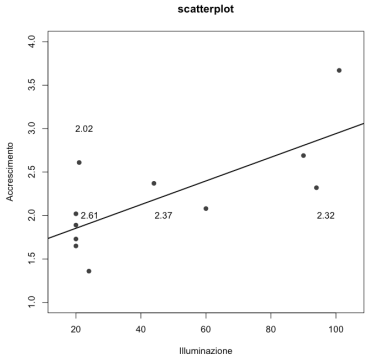
```
par(mfrow=c(1,2),cex=0.7)

# cex è un fattore di espansione/
riduzione rispetto allo standard
(minore di 1, rimpicciolisce)

hist(Light);hist(Rmax)
```



Esempio_2: continua

```
fit<-lm(Rmax~Light);summary(fit)
summary(fit)
```

lm è l'istruzione che esegue la regressione (linear model). A sinistra la dipendente, una tilde di separazione, e poi il o i regressori. Infine, c'è il sommario. L'esito del calcolo è tutto nell'oggetto fit




```
Call:
lm(formula = Rmax ~ Light)

Residuals:
    Min      1Q  Median      3Q     Max
-0.5478 -0.2607 -0.1166  0.1783  0.7431

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.580952   0.244519   6.466 0.000116 ***
Light       0.013618   0.004317   3.154 0.011654 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4583 on 9 degrees of freedom
Multiple R-squared: 0.5251, Adjusted R-squared: 0.4723
F-statistic: 9.951 on 1 and 9 DF,  p-value: 0.01165
```