Generazione di numeri casuali: distribuzione uniforme

- **dunif**(x, min = *a*, max = b)
  - Densità della variabile, costante = 1/(b-*a*) per ogni *x*
- **punif**(q, min = *a*, max = b, lower.tail = TRUE)
  - Distribuzione della variabile, cioè P(X≤q) , lower.tail=FALSE restituisce P(X>q)
- **qunif**(p, min = *a*, max = b, lower.tail = TRUE)
  - p vettore di probabilità, restituisce i quantili corrispondenti
- **runif**(n, min = *a*, max = b)
  - Genera n numeri distribuiti uniformemente tra *a* e b.

Generazione di numeri casuali: distribuzione uniforme

```
> a <- 1; b <- 10
> dunif(2, min=a, max=b)
      [1] 0.1111111
> dunif(5, min=a, max=b)
    [1] 0.1111111
>  punif(1, min=a, max=b)
   [1] 0
> punif(1, min=a, max=b,
                    lower.tail=FALSE)
  [1] 1
> punif(5, min=a, max=b)
  [1] 0.4444444
```

```
> x <- runif(10, min=a, max=b)
> x
 [1]  5.128674 7.038350 1.766834 9.822729
      4.579058 4.326511 9.385128 8.203212
 [9]  7.819636 2.866502
>  p <- punif(x, min=a, max=b)
>  p
  [1]  0.45874153 0.67092774 0.08520376
  [4]  0.98030325 0.39767311 0.36961232
  [7]  0.93168087 0.80035687 0.75773729
  [10] 0.20738915
>qunif(p, min=a, max=b)
  [1] 7.363353 9.025403 2.468331 9.996508
  [5] 6.734821 6.423502 9.957992 9.641284
  [9] 9.471779 4.345912
```

Generazione di numeri casuali: distribuzione normale

- **dnorm**(x, mean = *a*, sd = b)
  - Densità della variabile normale a media *a* e deviazione standard *b*
- **pnorm**(q, mean = *a*, sd = b, lower.tail = TRUE)
  - Distribuzione della variabile, cioè P(X≤q) , lower.tail=FALSE restituisce P(X>q)
- **qnorm**(p, mean = *a*, ad = b, lower.tail = TRUE)
  - p vettore di probabilità, restituisce i quantili corrispondenti
- **rnorm**(n, mean = *a*, sd = b)
  - Genera n numeri distribuiti secondo la distribuzione normale di media *a* e deviazione satandard b.

Generazione di numeri casuali: distribuzione normale

```
> a <- 0; b <- 1 // normale standard
> dnorm(2, min=a, max=b)
    [1]  0.05399097

> dnorm(10, min=a, max=b)
    [1] 7.694599e-23
> dnorm(0, mean=a, sd=b)
   [1] 0.3989423 % 1/(b*sqrt(2*pi)) cap 5.5
> pnorm(-5, mean=a, sd=b)
   [1]  2.866516e-07
> pnorm(-5, mean=a, sd=b,
                    lower.tail=FALSE)
   [1] 0.9999997
> pnorm(5, mean=a, sd=b)
   [1]  [1] 0.9999997
```

```
> x <- rnorm(10, mean=a, sd=b)
> x
  [1]  1.4693717 -0.8008470  0.3110244
  [4] -0.6948257 -0.9855775  1.3804164
  [7] -1.3758086 -0.4970854  0.8554936
  [10] -0.8884784
> p<-pnorm(x, mean=a, sd=b)
> p
  [1] 0.92913400 0.21161011 0.62210895
  [4] 0.24358225 0.16217024 0.91627076
  [7] 0.08444044 0.30956442 0.80386102
  [10] 0.18714175
> qnorm(p, mean=a, sd=b)
  [1]  1.4693717 -0.8008470  0.3110244
  [4] -0.6948257 -0.9855775  1.3804164
  [7] -1.3758086 -0.4970854  0.8554936
  [10] -0.8884784
```

Generazione di numeri casuali: altre distribuzioni

- Si veda l'help in linea, R fornisce primitive per gestire altre distribuzioni
- Esponeziale (dexp, pexp, …)
- Poissoniana (dpois, ppois, …..)
- Binomiale (dbinom, pbinom...)

E altre.....

qqnorm, qqline e qqplot

Funzioni per confrontare graficamente diverse distribuzioni:

- qqnorm(x):

produce un grafico quantile-quantile dei dati del vettore x rispetto ad una corrispondente distribuzione normale

- qqline(x):

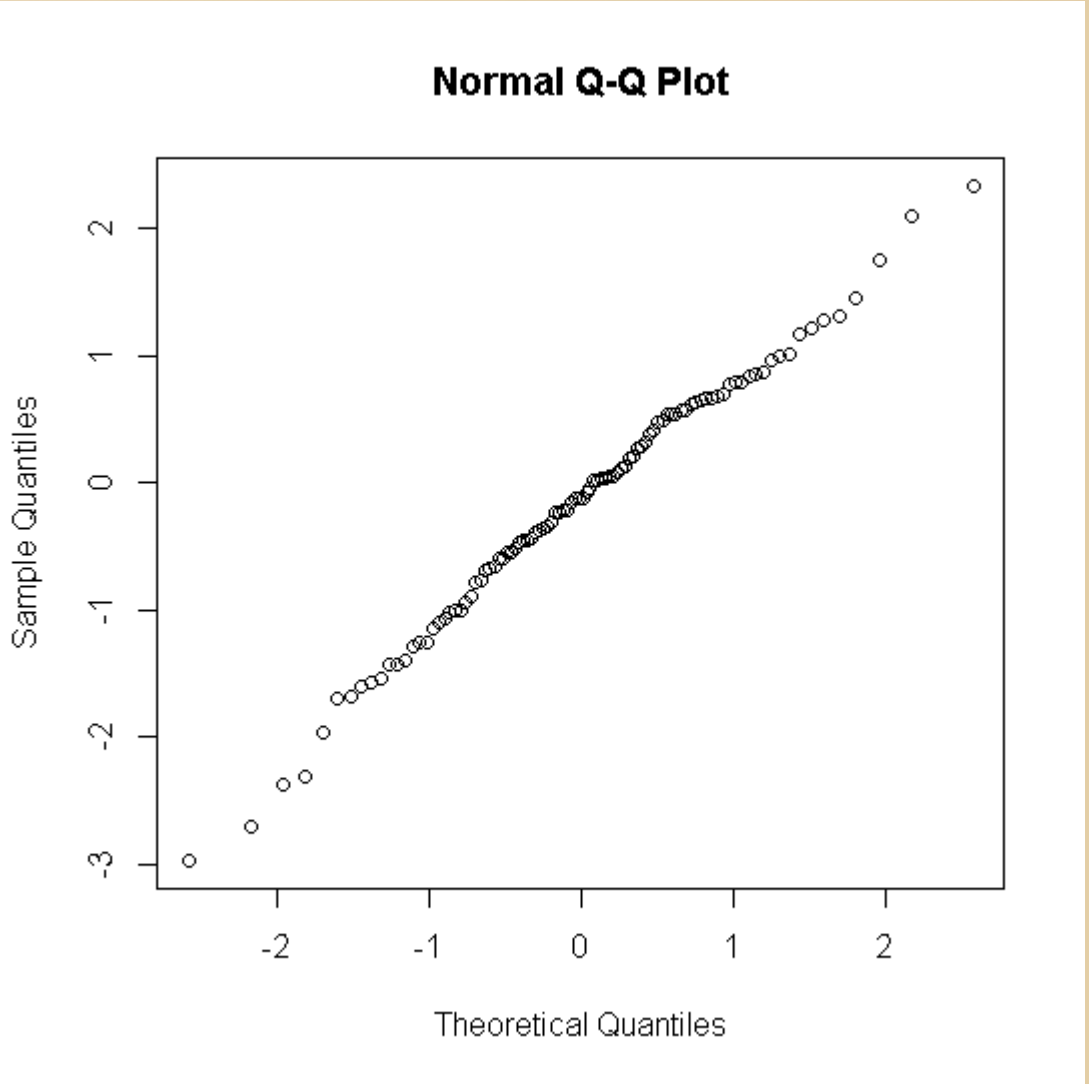
come qqnorm, ma viene aggiunta una linea che passa attraverso il primo e terzo quartile

- qqplot(x,y):

produce il grafico dei quantili dei dati del vettore x rispetto ai quantili dei dati del vettore y

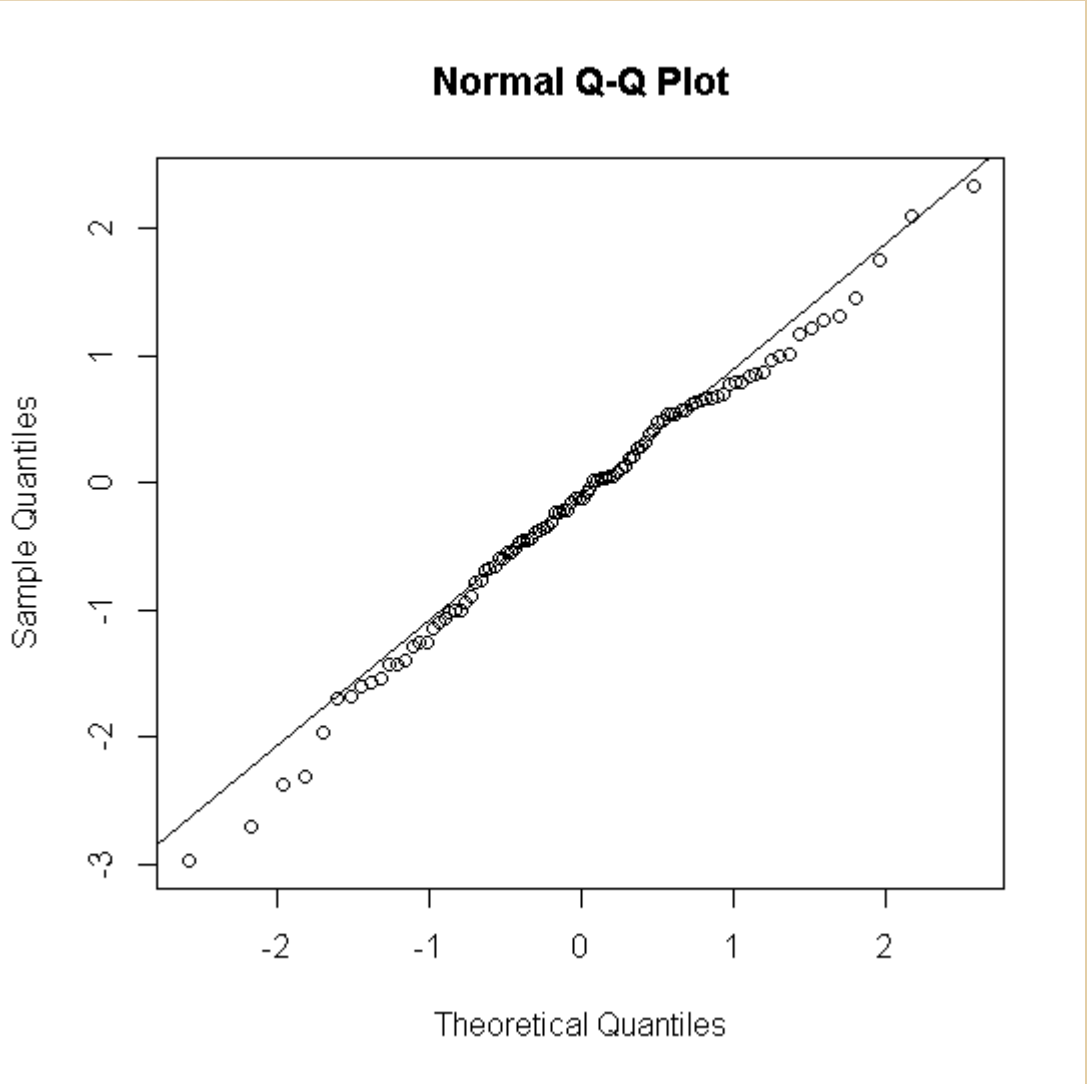
qqnorm, qqline e qqplot: esempio

```
> z<-rnorm(100);
>  qqnorm(z)
```



qqnorm, qqline e qqplot: esempio

```
> z<-rnorm(100);
>  qqline(z)
```



qqnorm, qqline e qqplot: esempio

```
> z<-rnorm(100);
> x<- runif(100)
> qqplot(x, z)
```