Applicazioni **PARTE II** Modelli Statistici II Corso di Laurea Magistrale in Biostatistica

Prof.ssa Fulvia Pennoni fulvia.pennoni@unimib.it

Novembre 2022

Contents

Distribuzione di Gauss Bivariata

Generazione di determinazioni dalla Normale Bivariata

La funzione `mvtnorm::rmvnorm` (dalla libreria `mvtnorm`) permette di generare delle realizzazioni da una distribuzione di densità bivariata (o multivariata) Gaussiana.

La funzione richiede come input il vettore delle medie e la matrice di varianza e covarianza Σ.

Considerando la seguente distribuzione congiunta per le due variabili causuali

$$(X_1, X_2) \sim N(\mu_1, \mu_2, \sigma_1^2, \sigma_2^2, \sigma_{12})$$

si fissano i seguenti valori dei parametri $N(0, 0, 3, 3, 2)$ e pertanto $\rho_{12} = 0.67$ (approsssimativamente) ovvero si tratta di variabili con una forte associazione lineare positiva.

La funzione `rmvnorm` richiede in input:

- la numerosità degli elementi da generare
- il vettore delle medie
- la matrice quadrata di varianze e covarianze **sigma1**

Nel seguente codice si noti che la matrice di varianza e covarianza viene specificata con come oggetto matrice con la funzoone `matrix`

```
require(mvtnorm)
sigma1 <- matrix(c(3,2,2,3), ncol=2); sigma1
#>      [,1] [,2]
#> [1,]    3    2
#> [2,]    2    3
```

Le realizzazioni dalla Normale doppia si ottengono nel modo seguente

```
set.seed(1234)
n <- 10
x <- rmvnorm(n,
             mean=c(0,0),
             sigma=sigma1)
x
```

```
#>              [,1]       [,2]
#>  [1,] -1.7816127 -0.2971177
#>  [2,]  0.3049418 -3.1251971
#>  [3,]  1.0070981  1.0840293
#>  [4,] -1.2677859 -1.2396778
#>  [5,] -1.4633761 -1.7889620
#>  [6,] -1.3891508 -1.9103445
#>  [7,] -1.2161674 -0.3754547
#>  [8,]  1.4843338  0.4145543
#>  [9,] -1.3899805 -1.7901664
#> [10,]  0.1384960  3.3915029
```

L'oggetto in **output** contiene le 10 realizzazioni da (X_1, X_2), dove la prima colonna è riferita a X_1 e la seconda a X_2.

Le matrici di *varianza e covarianza* e di *correlazione* empiriche per le realizzazioni ottenute sono le seguente

```
cov(x)
#>           [,1]      [,2]
#> [1,] 1.3874831 0.9333147
#> [2,] 0.9333147 3.4732530
cor(x)
#>           [,1]      [,2]
#> [1,] 1.0000000 0.4251538
#> [2,] 0.4251538 1.0000000
```

Si incrementa il numero di realizzazioni fino a 2000 fissando le **medie** delle due variabili casuali a zero, rispettivamente e lasciando invariata la matrice di varianza-covarianza.

```
set.seed(1234)
n <- 2000
x <- rmvnorm(n, mean=c(0,0), sigma=sigma1)
cov(x)
#>          [,1]     [,2]
#> [1,] 3.066812 2.043976
#> [2,] 2.043976 2.966497
cor(x)
#>           [,1]      [,2]
```

```
#> [1,] 1.0000000 0.6776575
#> [2,] 0.6776575 1.0000000
```

Si calcolano le statistiche descrittive dei valori empirici e la matrice di varianza-covarianza e quella di correlazione

```
skim_without_charts(x)
```

Table 1: Data summary

Name	x
Number of rows	2000
Number of columns	2
Column type frequency:	
numeric	2
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
V1	0	1	0.00	1.75	-6.55	-1.22	0.06	1.16	5.75
V2	0	1	0.01	1.72	-5.95	-1.15	0.01	1.19	5.32

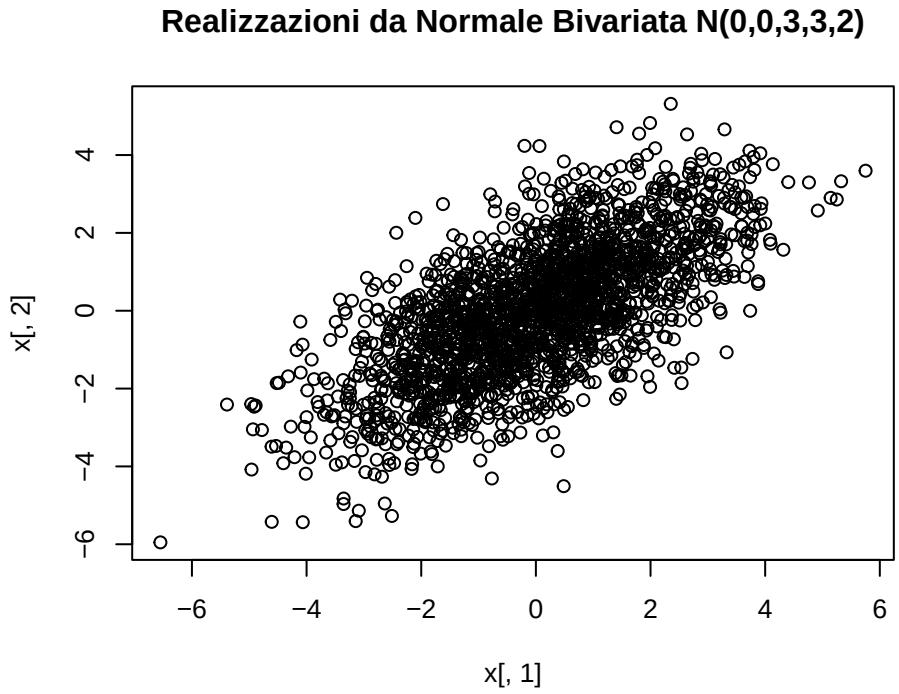
```
cov(x)
#>          [,1]     [,2]
#> [1,] 3.066812 2.043976
#> [2,] 2.043976 2.966497
cor(x)
#>           [,1]      [,2]
#> [1,] 1.0000000 0.6776575
#> [2,] 0.6776575 1.0000000
sd(x[,1])
#> [1] 1.751232
sd(x[,2])
#> [1] 1.722352
```

Si nota l'aderenza dei valori simulati a quelli della distribuzione teorica.

Diagramma a dispersione dei valori ottenuti

Le realizzazioni precedenti possono essere visualizzate anche attraverso il diagramma a dispersione bidimensionale

```
plot(x[,1],
     x[,2], main = "Realizzazioni da Normale Bivariata N(0,0,3,3,2)")
```



Si noti che i punti si dispongono a forma di **ellisse** con il fuoco centrato sulle medie.

Curve di livello

La **curve di livello** corrisponde al luogo geometrico dei punti del piano individuato dall'equazione $f(x_1, x_2) = k$ dove k è il livello. Il numero dei livelli è fissato di default dalla funzione `contour` in base alla densità.

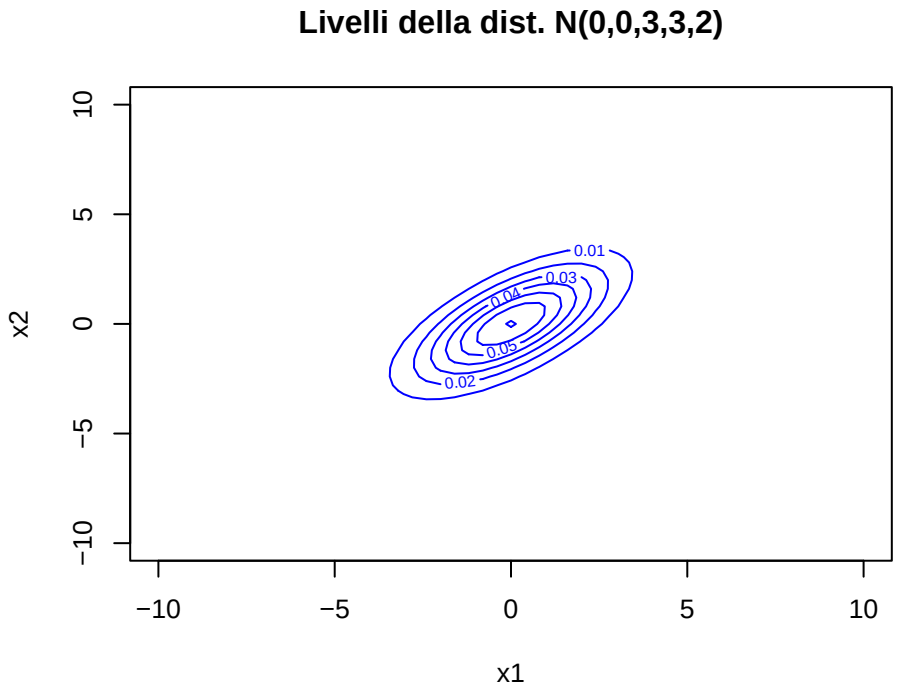
Si tratta delle *proiezioni ortogonali* delle curve ottenute intersecando il grafico precedente con un piano.

Al crescere di k ovvero della quota del piano le circonferenze si avvicinano al centro.

Si generano dei valori e si determia la densità

```r
x1 <- x2 <- seq(-10, 10, length = 51)

dens <- matrix(dmvnorm(expand.grid(x1, x2),
                sigma = sigma1),
                ncol = length(x1))
contour(x1,
        x2,
        dens,
        main = "Livelli della dist. N(0,0,3,3,2)",
        col="blue",
        xlab = "x1",
        ylab = "x2")
```



Si noti che geometricamente l'orientamento delle elissi permette di stabilire se l'associazione tra le due variabili casuali è positiva o negativa. E' di interesse anche il centro e la lunghezza dei semi assi.

Algoritmo Expectation-Maximization (EM)

Si consideri la seguente tabella di contingenza a doppia entrata

$X\|Z$	1	2	3	
1	y_{11}	y_{12}	y_{13}	$y_{1.}$
2	y_{21}	y_{22}	NA	$y_{2.}$
	$y_{.1}$	$y_{.2}$	$y_{.3}$	$y_{..}$

Nel seguente esempio si riprende il modello illustrato nelle dispense di teoria. I dati osservati sono $\mathbf{y} = \{y_{11}, y_{12}, y_{13}, y_{21}, y_{22}\}$ mentre il valore mancante è quelli della cella 23 ovvero $u = y_{23}$.

Il vettore dei dati **completi** è $\mathbf{c} = (\mathbf{y}, \mathbf{u})$.

In base ad un modello lineare

$$y_{ij} = \mu + \alpha_i + \beta_j + e_{ij}$$

dove μ è un intercetta, α è il parametro per la riga e β per la colonna, mentre l' ipotesi per il termine di errore è che questa componente segua una distribuzione Normale $e_{ij} \sim N(0, \sigma^2)$. Vincoli a somma zero sono posti sui parametri $\sum_i \alpha_i = 0$ e $\sum_j \beta_j = 0$.

Nel seguito si utilizza l'algortimo EM per la stima dei parametri del modello e si imputa il dato mancante attraverso i valori dei parametri ottenuti a convergenza dell'algoritmo.

Gli **step** della procedura di stima sono i seguenti:

- Si assegna un valore iniziale ai dati mancanti ovvero le frequenze non osservate $y_{23}^{(0)}$. Un modo generalmente utilizzato per inizializzare l'algoritmo è quello di utilizzare i valori delle statistiche dei dati osservati. Per esempio si assegna a $y_{23}^{(0)}$ il valore medio delle frequenze osservate.
- al passo **E** (Expectation): calcolano le stime dei parametri utilizzano le soluzioni note in forma chiusa con i valori iniziali che risultano le seguenti

$$\hat{\mu}^{(0)} = \bar{y}$$
$$\hat{\alpha}_i^{(0)} = \bar{y}_{i.} - \bar{y}$$
$$\hat{\beta}_j^{(0)} = \bar{y}_{.j} - \bar{y}.$$

Si imputa il valore mancante nel modo seguente $y_{23}^{(1)} = \hat{\mu}^{(0)} + \hat{\alpha}_2^{(0)} + \hat{\beta}_j^{(0)}$.

- al passo **M** (Maximization): si procede per via iterativa ricalcolando i valori dei parametri in base al valore imputato $y_{23}^{(1)}$ fino al raggiungimento del valore fissato per il criterio di convergenza.

Nell'esempio che segue si considerino le frequenze che riguardano due variabili: X assunzione di placebo o farmaco e Y stato di salute (pessimo, buono, ottimo). Si intende applicare il modello precedente e imputare il valore mancante relativo alla frequenza di coloro che hanno assunto il farmaco e hanno un ottimo stato di salute.

Si costruisce la tabella con la frequenze utilizzando la funzione `matrix` ed aggiungendo il valore mancante

```
y <- matrix(c(10,15, 17, 22, 23, NA),2,3,byrow=TRUE); y
#>      [,1] [,2] [,3]
#> [1,]   10   15   17
#> [2,]   22   23   NA
```

La seguente funzione denominata `em1` assegna un valore iniziale alla cella mancante della tabella e ricostruice il valore in base alle soluzioni in forma chiusa per i parametri del modello (queste sono riportate a pagina 69 delle dispense).

Si noti l'utilizzo della funzione `apply`

```
em1 <- function(y23, y){
  ystar <- y
  ystar[2,3] <- y23
  mu.hat <- mean(ystar)
  alpha.hat <- apply(ystar, MAR = 1,
                     mean) - mean(ystar)
  beta.hat  <- apply(ystar, MAR = 2,
                     mean) - mean(ystar)

  y23 <- mu.hat + alpha.hat[2] + beta.hat[3]

  return(c(mu = mu.hat,
           alpha = alpha.hat,
           beta = beta.hat,
           y23 = y23))
}
```

La funzione restituisce i valori dei parametri del modello ottenuti con le soluzioni in forma chiusa

```
em1(21,y)
#>     mu alpha1 alpha2  beta1  beta2  beta3    y23
#>     18     -4      4     -2      1      1     23
```

La seguente funzione `em.step` richiama la funzione precedente `em1` per ripetere la procedura di calcolo aggiornando le stime fino a convergenza.

Si ottengono in output le stime dei parametri di massima verosimiglianza ed il corrispondente valore del dato mancante.

La funzione richiede 2 input:

- i valori della tabella di contingenza
- il *livello di tolleranza* ϵ da scelglire per la convergenza dell'algoritmo EM. Tra i valori possibili si considera $\epsilon = 10^{-8}$ ovvero 0.00000001 indicato nel seguito con la notazione *1e-8*.

Nel nel corpo della funzione viene richiamata la funzione `dist` per calcola la distanza tra i valori contenuti nell'oggetto `trace` all'iterazione $h+1$ e all'itereazione h-esima (come illustrato nelle dispense di teoria).

Si noti l'ultilizzo del costrutto `while` per incrementare il contatore fino a convergenza

```
set.seed(1832)
em.step <- function(y, epsilon= 1e-8){
  trace <- NULL
  convergenza <- FALSE

  trace <- t(em1(y23 = mean(y, na.rm = TRUE), y = y))

  y23id <- grep("y23", colnames(trace))

  h <- 0
  while(!convergenza){
    h <- h + 1
    trace <- rbind(trace,
                   em1(y23 = trace[h, "y23"],
                       y = y))
    convergenza <- (dist(trace[h:(h+1), -y23id]) < epsilon)
  }

  return(trace)

}
```

Si applica la funzione alla matrice dei dati

```
em.step(y)
#>              mu    alpha1   alpha2     beta1        beta2      beta3      y23
#>  [1,] 17.40000 -3.400000 3.400000 -1.400000 1.600000e+00 -0.2000000 20.60000
#>  [2,] 17.93333 -3.933333 3.933333 -1.933333 1.066667e+00  0.8666667 22.73333
#>  [3,] 18.28889 -4.288889 4.288889 -2.288889 7.111111e-01  1.5777778 24.15556
#>  [4,] 18.52593 -4.525926 4.525926 -2.525926 4.740741e-01  2.0518519 25.10370
#>  [5,] 18.68395 -4.683951 4.683951 -2.683951 3.160494e-01  2.3679012 25.73580
#>  [6,] 18.78930 -4.789300 4.789300 -2.789300 2.106996e-01  2.5786008 26.15720
#>  [7,] 18.85953 -4.859534 4.859534 -2.859534 1.404664e-01  2.7190672 26.43813
#>  [8,] 18.90636 -4.906356 4.906356 -2.906356 9.364426e-02  2.8127115 26.62542
#>  [9,] 18.93757 -4.937570 4.937570 -2.937570 6.242951e-02  2.8751410 26.75028
#> [10,] 18.95838 -4.958380 4.958380 -2.958380 4.161967e-02  2.9167607 26.83352
#> [11,] 18.97225 -4.972254 4.972254 -2.972254 2.774645e-02  2.9445071 26.88901
#> [12,] 18.98150 -4.981502 4.981502 -2.981502 1.849763e-02  2.9630047 26.92601
#> [13,] 18.98767 -4.987668 4.987668 -2.987668 1.233175e-02  2.9753365 26.95067
#> [14,] 18.99178 -4.991779 4.991779 -2.991779 8.221170e-03  2.9835577 26.96712
#> [15,] 18.99452 -4.994519 4.994519 -2.994519 5.480780e-03  2.9890384 26.97808
#> [16,] 18.99635 -4.996346 4.996346 -2.996346 3.653853e-03  2.9926923 26.98538
#> [17,] 18.99756 -4.997564 4.997564 -2.997564 2.435902e-03  2.9951282 26.99026
#> [18,] 18.99838 -4.998376 4.998376 -2.998376 1.623935e-03  2.9967521 26.99350
#> [19,] 18.99892 -4.998917 4.998917 -2.998917 1.082623e-03  2.9978348 26.99567
#> [20,] 18.99928 -4.999278 4.999278 -2.999278 7.217488e-04  2.9985565 26.99711
#> [21,] 18.99952 -4.999519 4.999519 -2.999519 4.811659e-04  2.9990377 26.99808
#> [22,] 18.99968 -4.999679 4.999679 -2.999679 3.207772e-04  2.9993584 26.99872
#> [23,] 18.99979 -4.999786 4.999786 -2.999786 2.138515e-04  2.9995723 26.99914
#> [24,] 18.99986 -4.999857 4.999857 -2.999857 1.425677e-04  2.9997149 26.99943
#> [25,] 18.99990 -4.999905 4.999905 -2.999905 9.504511e-05  2.9998099 26.99962
#> [26,] 18.99994 -4.999937 4.999937 -2.999937 6.336340e-05  2.9998733 26.99975
#> [27,] 18.99996 -4.999958 4.999958 -2.999958 4.224227e-05  2.9999155 26.99983
#> [28,] 18.99997 -4.999972 4.999972 -2.999972 2.816151e-05  2.9999437 26.99989
#> [29,] 18.99998 -4.999981 4.999981 -2.999981 1.877434e-05  2.9999625 26.99992
#> [30,] 18.99999 -4.999987 4.999987 -2.999987 1.251623e-05  2.9999750 26.99995
#> [31,] 18.99999 -4.999992 4.999992 -2.999992 8.344152e-06  2.9999833 26.99997
#> [32,] 18.99999 -4.999994 4.999994 -2.999994 5.562768e-06  2.9999889 26.99998
#> [33,] 19.00000 -4.999996 4.999996 -2.999996 3.708512e-06  2.9999926 26.99999
#> [34,] 19.00000 -4.999998 4.999998 -2.999998 2.472341e-06  2.9999951 26.99999
#> [35,] 19.00000 -4.999998 4.999998 -2.999998 1.648228e-06  2.9999967 26.99999
#> [36,] 19.00000 -4.999999 4.999999 -2.999999 1.098818e-06  2.9999978 27.00000
#> [37,] 19.00000 -4.999999 4.999999 -2.999999 7.325456e-07  2.9999985 27.00000
#> [38,] 19.00000 -5.000000 5.000000 -3.000000 4.883637e-07  2.9999990 27.00000
#> [39,] 19.00000 -5.000000 5.000000 -3.000000 3.255758e-07  2.9999993 27.00000
#> [40,] 19.00000 -5.000000 5.000000 -3.000000 2.170505e-07  2.9999996 27.00000
#> [41,] 19.00000 -5.000000 5.000000 -3.000000 1.447004e-07  2.9999997 27.00000
#> [42,] 19.00000 -5.000000 5.000000 -3.000000 9.646691e-08  2.9999998 27.00000
```

```
#> [43,] 19.00000 -5.000000 5.000000 -3.000000 6.431127e-08  2.9999999 27.00000
#> [44,] 19.00000 -5.000000 5.000000 -3.000000 4.287418e-08  2.9999999 27.00000
#> [45,] 19.00000 -5.000000 5.000000 -3.000000 2.858279e-08  2.9999999 27.00000
#> [46,] 19.00000 -5.000000 5.000000 -3.000000 1.905519e-08  3.0000000 27.00000
#> [47,] 19.00000 -5.000000 5.000000 -3.000000 1.270346e-08  3.0000000 27.00000
#> [48,] 19.00000 -5.000000 5.000000 -3.000000 8.468973e-09  3.0000000 27.00000
#> [49,] 19.00000 -5.000000 5.000000 -3.000000 5.645983e-09  3.0000000 27.00000
```

Si ottiene l'oggetto che ha per riga il valore di ogni parametro ad ogni iterazione dell'algoritmo.

La convergenza rispetto al criterio stabilito $\epsilon < 1e^{-8}$ viene raggiunta dopo 49 iterazioni.

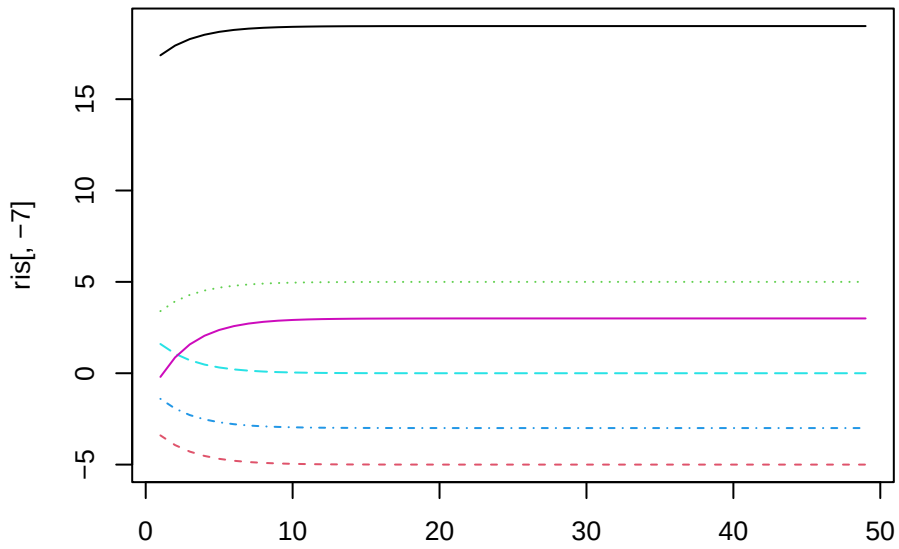
Le stime dei parametri del modello sono le seguenti:

$\hat{\mu} = 19, \quad \hat{\alpha}_1 = -5, \quad \hat{\alpha}_2 = 5, \quad \hat{\beta}_1 = -3 \quad \hat{\beta}_2 = 0, \quad \hat{\beta}_3 = 3; \quad \hat{y}_{23} = 27$

Trace plot

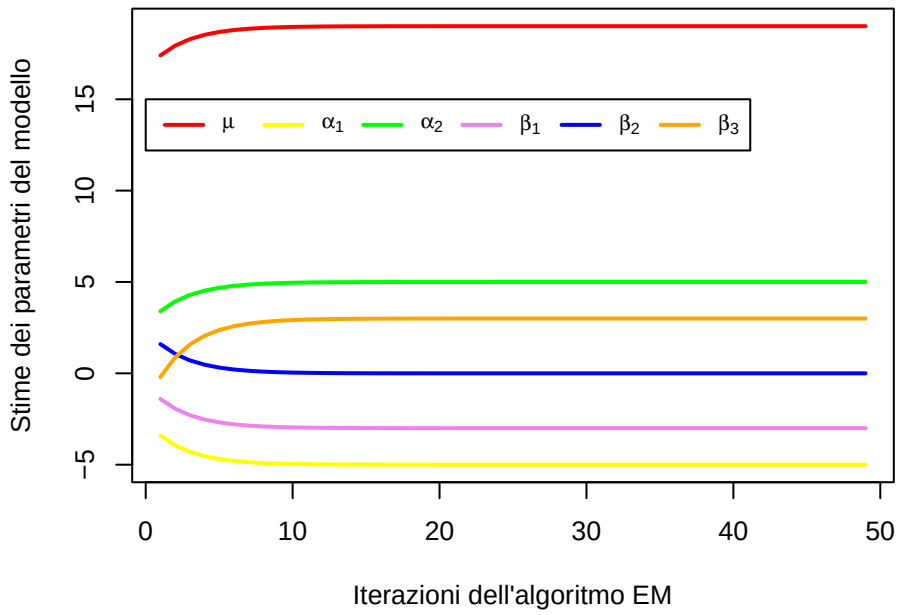
La rappresentazione grafica dei valori ottenuti ad ogni passo dell'algoritmo, chiamato anche trace plot, avviene utilizzando la funzione `matplot`

```
ris<- em.step(y)
matplot(ris[,-7], type = "l")
```



Con la seguente sintassi si definiscono le etichette ed i colori e si aggiungono al grafico insieme alla legenda.

```
names1 <- expression(mu,
                     alpha[1],
                     alpha[2],
                     beta[1],
                     beta[2],
                     beta[3])
pal1<- c("red", "yellow", "green", "violet", "blue", "orange")
matplot(ris[,-7],
        type = "l",
        col = pal1,
        lwd = 2,
        lty = 1,
        xlab = "Iterazioni dell'algoritmo EM",
        ylab = "Stime dei parametri del modello")
legend(x = 0,
       y = 15,
       legend  = names1,
       lwd = 2 ,
       col = pal1,
       lty = 1,
       horiz=TRUE,
       cex=0.8)
```



Il grafico permette di visualizzare tutti i valori di ogni parametro stimati ad ogni step.

I valori variano particolarmente nelle prime 10 iterazioni e nelle successive si ha un aggiustamento che riguarda le ultime cifre decimali.

Densità miscuglio di componenti Gaussiane

Si rappresenta graficamente la densità per alcuni modello miscuglio con due componenti Gaussiane per un valore specifico del peso di ogni componenete.

La funzione `funcmxn` calcola il valore della densità miscuglio considerando.

Il peso della prima componente $\pi \in (0, 1)$ è indicato con `p`,

I valori dei parametri μ_1, μ_2 e σ_1, σ_2 della prima e della seconda componente nei vettori `mu` e `sd`.

Si noti che la funzione resituisce il valore della densità miscuglio in un punto indicato con `f`.

```
funcmxn <- function(x, p, mu, sd){
  f1 <- dnorm(x, mu[1], sd[1])
  f2 <- dnorm(x, mu[2], sd[2])
  f <- p*f1 + (1-p)*f2
  f
}
```

Nel seguito si rappresenta la densità miscuglio per tre distinti scenari.

Scenario 1

Nel primo scenario si considera un miscuglio con due componenti X_1 e X_2 tali che $X_1 \sim N(1, 1)$ e $X_2 \sim N(4, 1)$ ovvero con stessa variabilità ma con con peso della prima componente pari a $\pi_1 = 0.4$ e della seconda $1 - \pi_1 = 0.6$.

Si determina il valore della densità nel punto $y = 0.5$

```
mu1 <- c(1,4)
sd1 <- c(1,1)

p1 <- 0.4

funcmxn(0.5,
        p1,
        mu1,
        sd1)
#> [1] 0.1413497
```

Si ottiene un valore della densità di 0.14.

Per una sequenza di valori $\mathbf{y}$ si ricava il corrispondente valore della densità miscuglio

```r
y1 <- seq(-5,10,0.01)

length(y1)
#> [1] 1501


pr1 <- funcmxn(x = y1,
               p = p1,
               mu = mu1,
               sd = sd1)

require(skimr)
skim_without_charts(pr1)
```

Table 3: Data summary

Name	pr1
Number of rows	1501
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

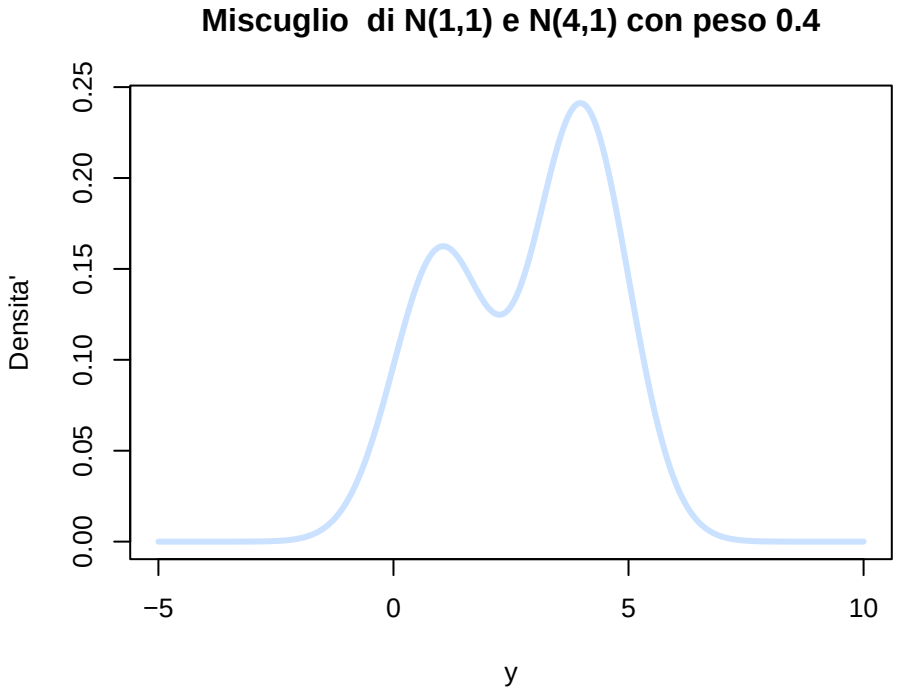
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	0.07	0.08	0	0	0.02	0.14	0.24

Si nota che la densità è concentrata tra i valori delimitata dalla *differenza interquartile*.

Si disegnano i valori ottenuti per visualizzare la forma della densità miscuglio

```r
plot(y1,
     pr1,
     xlab = "y",
     ylab="Densita'",
     lwd=3,
     col="lightsteelblue1",
     type = "l",
     main="Miscuglio  di N(1,1) e N(4,1) con peso 0.4")
```



Si nota la multimodalità.

Scenario 2

Nel secondo scenario si considerano le due seguenti componenti X_1 e X_2 tali che $X_1 \sim N(4,1)$ e $X_2 \sim N(4,64)$ che hanno stessa media ma la variabilità della seconda componente è molto più ampia di quella della prima, il peso della prima componente è $\pi_1 = 0.1$ mentre della seconda è $1 - \pi_1 = 0.9$

Per $y = 0.5$

```
mu2 <- c(4,4)
sd2 <- c(1,8)

p2 <-0.1

set.seed(1235)
funcmxn(0.5,
        p2,
        mu2,
        sd2)
#> [1] 0.04087215
```

il valore della densità è inferiore a quello calcolato nel precedente scenario.

Si genera una sequenza di valori con più ampio campo di variazione

```
y2 <- seq(-30,40,0.01)
pr2 <- funcmxn(x = y2,
               p = p2,
               mu = mu2,
               sd = sd2)
skim_without_charts(pr2)
```

Table 5: Data summary

Name	pr2
Number of rows	7001
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

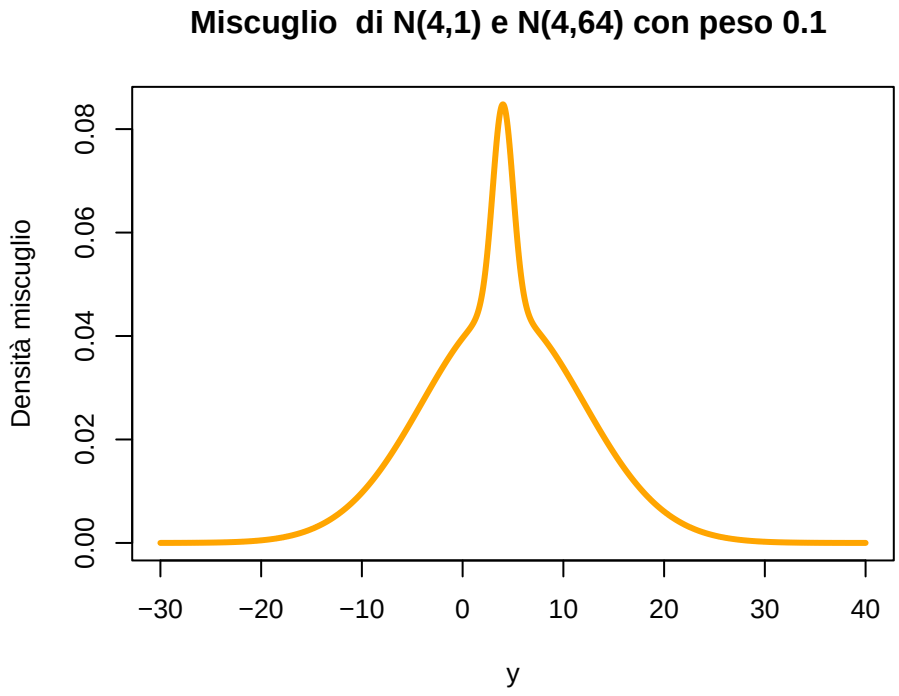
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	0.01	0.02	0	0	0	0.02	0.08

La densità è concentrata tra 0.0002 e 0.024.

Il grafico permette di rilevare la forma della densità miscuglio nel secondo scenario

```
plot(y2,
     pr2,
     xlab = "y",
     ylab="Densità miscuglio",
     lwd=3,
     col="orange",
     type = "l",
     main="Miscuglio  di N(4,1) e N(4,64) con peso 0.1 ")
```



In questo caso si osseva un unico massimo.

La densità è concentrata tra

Scenario 3

Nel terzo scenario si considera il valore della densità miscuglio di due componenti X_1 e X_2 tali che $X_1 \sim N(0,1)$ e $X_2 \sim N(0,9)$ che hanno stessa media ma la variabilità della seconda componente è più elevata, il peso della prima componente è uguale a quello della seconda $\pi_1 = 0.5$.

Per $y = 0.5$

```
mu3 <- c(0,0)
sd3 <- c(1,3)

p3 <-0.5

funcmxn(0.5,
        p3,
        mu3,
        sd3)
#> [1] 0.2416059
```

il valore è più elevato dei due ottenuti in esempio 1 ed esempio 2.

Per una sequenza di valori y (*si noti che la sequenza di valori deve essere generata in relazione al campo di variazione delle due variabili in modo poter avere il valore della densità miscuglio sul supporto considerato*) si calcola la corrispondente densità

```
y3 <- seq(-10,20,0.01)

pr3 <- funcmxn(x = y3,
               p = p3,
               mu = mu3,
               sd = sd3)

skim_without_charts(pr3)
```

Table 7: Data summary

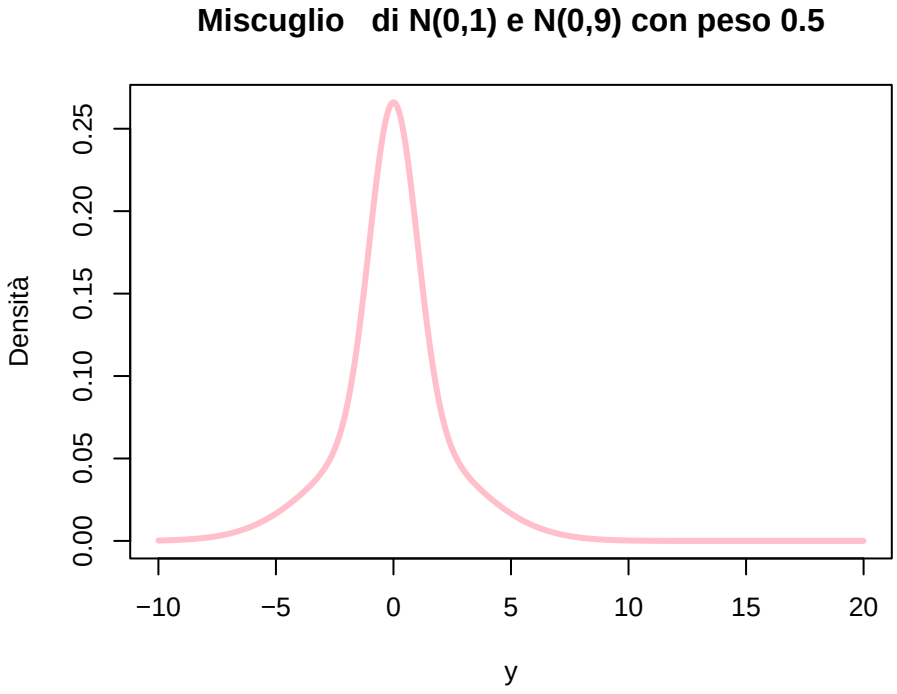
Name	pr3
Number of rows	3001
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	0.03	0.06	0	0	0	0.03	0.27

Il grafico permette di rilevare la forma della densità miscuglio

```
plot(y3,
     pr3,
     xlab = "y",
     ylab="Densità",
     lwd=3,
     col="Pink",
     type = "l",
     main="Miscuglio   di N(0,1) e N(0,9) con peso 0.5"
     )
```


Miscuglio di N(0,1) e N(0,9) con peso 0.5
Densità
0.00
0.05
0.10
0.15
0.20
0.25
−10
−5
0
5
10
15
20
y

Modello miscuglio univariato con due componenti Gaussiane

I dati riportati in `datacol.Rdata` riguardano alcuni individui selezionati per uno studio al fine di valutare il ruolo del colesterolo come fattore di rischio per le malattie cardiovascolari.

I valori del colesterolo (espressi in mg su dl di plasma) sono riferiti sia ai maschi (variabile binaria sex codificata con 1 per maschi) che alle femmine.

```
load('datacol.Rdata')
dim(datacol)
#> [1] 200   3
head(datacol)
```

ID	cholst	sex
1244	175	1
835	299	0
176	250	0
901	243	0
1972	150	1
1994	234	0

```
require(skimr)
skim_without_charts(datacol)
```

Table 10: Data summary

Name	datacol
Number of rows	200
Number of columns	3
Column type frequency:	
numeric	3
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
ID	0	1	1344.43	774.26	1	682.5	1367.5	2012.25	2616
cholst	0	1	220.49	43.92	133	191.0	214.0	249.00	357
sex	0	1	0.52	0.50	0	0.0	1.0	1.00	1

Il livello medio osservato nel campione per il colesterolo è 220.5 e presenta un'elevata varibilità media intorno alla media aritmetica è di 44.

Nel seguente chunck si riporta l'analisi descrittiva rispetto al genere

```
table(datacol$sex)
#>
#>   0   1
#>  97 103
require(dplyr)
datacol%>%
  dplyr::group_by(sex) %>%
  skim_without_charts()
```

Table 12: Data summary

Name	Piped data
Number of rows	200
Number of columns	3
_____	
Column type frequency:	
numeric	2
_____	
Group variables	sex

Variable type: numeric

skim_variable	sex	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
ID	0	0	1	1297.75	769.15	2	601	1206	1994	2616
ID	1	0	1	1388.39	780.22	1	831	1442	2069	2597
cholst	0	0	1	222.66	42.67	134	192	216	243	340
cholst	1	0	1	218.45	45.18	133	182	212	249	357

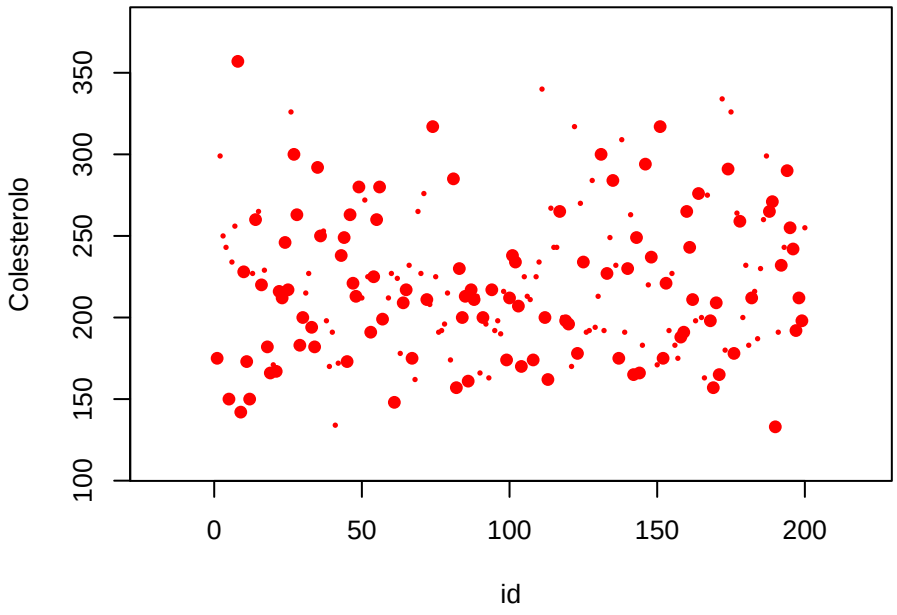
```
# oppure
# tapply(datacol$cholst, datacol$sex, summary)
# tapply(datacol$cholst, datacol$sex, sd)
```

Nel campione ci sono 97 donne e 103 uomini e si nota che il livello medio di colesterolo per le donne è 222.7 mentre per i maschi è inferiore pari a 218.4.

La variabilità media rispetto alla media aritmetica è pari a 42 per le donne e a 45 per i maschi.

La seguente rappresentazione mostra i valori osservati attraverso un diagramma a dispersione: i valori dei maschi sono rappresentati dal cerchio più grande

```
n <-dim(datacol)[1]
with(datacol,
     symbols(x=1:n,
             y=cholst,
             circles=sex,
             inches=1/30 ,
             xlab = "id",
             ylab = "Colesterolo",
             bg="red",
             fg=NULL))
```



Stima dei parametri del modello miscuglio

La libreria denominata `mclust` (Gaussian Mixture Modelling for Model-Based Clustering) permette di stimare i parametri del modello miscuglio con componenti Gaussiane.

```
require('mclust')
```

Funzione Mclust

La funzione principale della libreria `mclust::Mclust` richiede come argomenti:

- il dataframe dei dati,
- ovvero il numero di componenti del modello miscuglio (`G`),
- l'opzione `modelNames = "E"` per specificare se le componenti del modello sono assunte con stessa varianza, modello *omoschedastico* oppure `modelNames = "V"` se assunte con varianza specifica per ogni componente, modello *eteroschedastico*

```
mod1 <- Mclust(datacol$cholst,
       G = 2,
       modelNames = "E")
```

Si noti che la tolleranza per il criterio d'arresto dell'algoritmo può essere fissata con l'opzione `control = emControl(tol = 1.e-6)`

```
summary(mod1)
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust E (univariate, equal variance) model with 2 components:
#>
#>  log-likelihood   n df       BIC       ICL
#>       -1032.055 200  4 -2085.302 -2126.187
#>
#> Clustering table:
#>   1   2
#> 158  42
```

La funzione `summary` restituisce:

- il valore della funzione di **log-verosimiglianza** a convergenza dopo che i parametri sono stati stimati con l'algoritmo EM.
- **numero dei parametri** del modello liberi di variare (nel modello con due componenti sono dati dalla due medie, dalla varianza e dal peso).

Le stime dei parametri si richiedono con

```
summary(mod1,parameters = TRUE)
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust E (univariate, equal variance) model with 2 components:
#>
#>  log-likelihood   n df       BIC       ICL
#>       -1032.055 200  4 -2085.302 -2126.187
#>
#> Clustering table:
#>   1   2
#> 158  42
#>
#> Mixing probabilities:
#>         1         2
#> 0.7777503 0.2222497
#>
#> Means:
#>        1        2
#> 203.7208 279.1730
#>
#> Variances:
#>        1        2
#> 935.6102 935.6102
```

Dove:

- **Mixing prob.** sono le stime dei pesi $\hat{\pi}$ di ogni componente: $\hat{\pi}_1 = 0.78$ della prima e $\hat{\pi}_2 = 0.22$ della seconda.
- **means** i valori delle medie delle due componenti. Dalle stime si nota che la prima componente è rappresentata da una variabile casuale Normale con valore medio $\hat{\mu}_1$=204, mentre la seconda $\hat{\mu}_2 = 279$.
- **Variances**: le stime delle varianze delle componenti. Dato che per ipotesi sono uguali si tratta della stima di un'unica varianza $\hat{\sigma}^2$ pari a 936. Questa indica che in media sia nella prima che nella seconda sottopopolzione la variabilità media rispetto al valore medio è di 30 mg su dl di plasma.

Dalla stima di questo modello miscuglio sappiamo che la popolazione è caratterizzata da un miscuglio di due sottopolazioni la prima descritta da $X_1 \sim N(204, 936)$ e la seconda da $X_2 \sim N(279, 936)$.

Classificazione delle unità

- **clustering table** restituisce il numero complessivo dei pazienti che sono stati classificati come appartenenti alla prima o alla seconda componente.

Nel campione osservato di 158 pazienti 42 pazienti vengono classificati nel primo gruppo ed i restanti nel secondo.

Si noti che questa assegnazione viene effettuata rispetto alle **probabilità a posteriori** che si trovano nell'oggetto z

```
head(mod1$z)
#>            [,1]         [,2]
#> [1,] 0.99863336 0.0013666414
#> [2,] 0.03184899 0.9681510115
#> [3,] 0.63194159 0.3680584100
#> [4,] 0.75129790 0.2487021035
#> [5,] 0.99981810 0.0001819027
#> [6,] 0.86199548 0.1380045218
```

In base a questo modello miscuglio con due componenti la prima unità ha una probabilità quasi certa sulla prima componente. La seconda unità invece ha una probabilità a posteriori maggiore di essere classificata nella seconda componente. Mentre la quarta unità ha una probabilità di 0.75 di essere classificata nella prima componente e di 0.25 di essere classificata nella seconda componente.

In base a questa incertezza della classificazione a posteriori viene calcolato il termine di penalizzazione presente nel criterio di informazione ICL, riportato nel summary del modello.

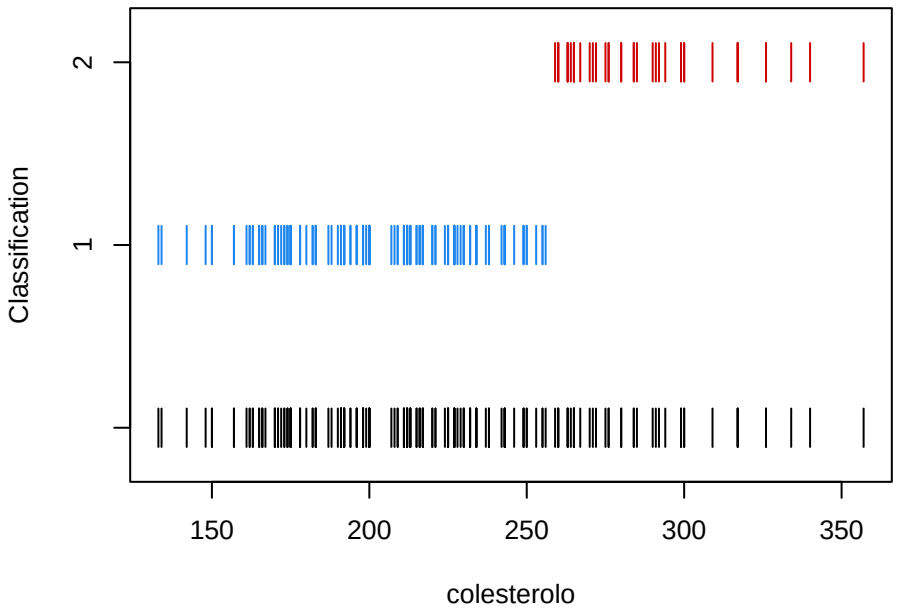
Si noti che è possibile adottare il criterio della massima probabilità a posteriori che verrà mostrato nel seguito anche applicando la funzione which.max

```
head(apply(mod1$z,1,which.max))
#> [1] 1 2 1 1 1 1
```

che restituisce il vettore di elementi 1 oppure 2 se l'id viene classificato nella prima o nella seconda componente rispettivamente.

Il grafico seguente permette invece di visualizzare tutti i valori osservati del colesterolo **nero**, e quali valori vengono classificati nel gruppo 2 **rosso** e nel gruppo 1 **blu**.

```
plot(mod1,
     what='classification', xlab = "colesterolo")
```



Dal grafico si nota il valore del colesterolo che discrimina le due sottopopolazioni è leggermente superiore a 250.

Possiamo vedere come sono state classificate le unità a seconda del genere maschio o femmina

```
class<-mod1$classification; head(class)
#> [1] 1 2 1 1 1 1
table(class,datacol$sex)
#>
#> class  0  1
#>     1 78 80
#>     2 19 23
```

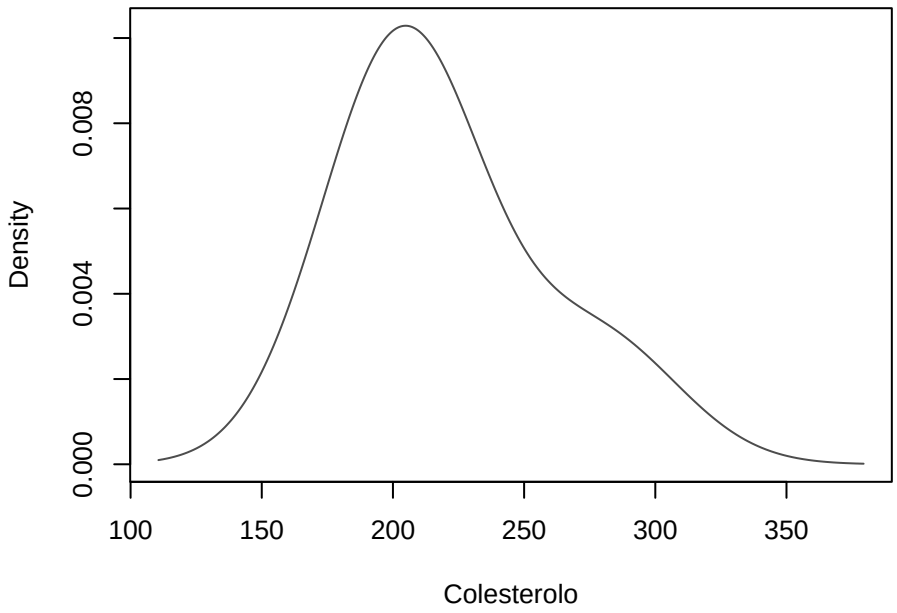
I due generi sono presenti in numerosità quasi uguale nei due gruppi.

Alla prima componente appartengono il 78% dei soggetti.

Rappresentazione della densità miscuglio

Il grafico della densità misuglio si ottiene nel modo seguente

```
plot(mod1,
     what='density', xlab = "Colesterolo")
```



Scelta del numero delle componenti

La funzione `mclus:mclustBIC` permette di visualizzare i valori del criterio di informazione Bayesiano (BIC) quando vengono stimati i modelli miscuglio con componenti di numerosità crescente.

Il BIC è calcolato rispetto a modelli miscuglio che differiscono per il numero delle componenti a seconda che queste abbiano stessa varianza (opzione E) o diversa variabilità (opzione V).

La funzione accetta come input il vettore dei dati

```
bayesinf <- mclustBIC(datacol$cholst)
bayesinf
#> Bayesian Information Criterion (BIC):
#>           E         V
#> 1 -2090.155 -2090.155
#> 2 -2085.302 -2089.269
#> 3 -2096.304 -2103.554
#> 4 -2104.550 -2118.857
#> 5 -2114.649 -2124.799
#> 6 -2123.391 -2149.002
#> 7 -2134.099 -2134.237
#> 8 -2144.769 -2142.100
#> 9 -2155.475 -2132.422
#>
#> Top 3 models based on the BIC criterion:
#>       E,2       V,2       E,1
#> -2085.302 -2089.269 -2090.155
```

Nella tabella vengono elencati i valori realizzati del BIC per ogni modello miscuglio con un numero di componenti che varia da 1 a 9 (righe). La prima riga è riferita al modello senza eterogenità ovvero senza componenti che assume una popolazione omogenea.

La prima colonna è riferita ai modelli in cui si ipotizza uguale variabilità per le componenti del miscuglio mentre la seconda ai modelli in cui si assume specifica variabilità per ogni componente.

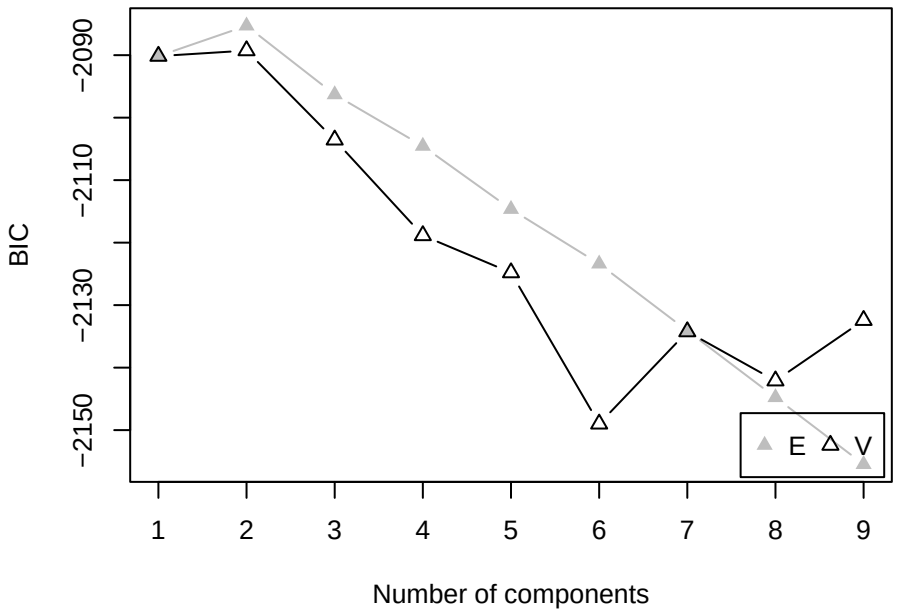
Il modello migliore è quello che presenta il valore minimo del BIC ovvero con maggiore log-verosimiglianza relativamente al numero dei parametri e al numero delle osservazioni.

I tre modelli migliori rispetto a questo criterio vengono visualizzati in basso con la scritta **top 3 models**.

Il modello più parsimonioso che spiega la variabilità presente nei dati è quello con 2 componenti e stessa varianza, che è appunto il modello stimato in precedenza.

I valori della tabella possono anche essere rappresentati graficamente come segue.

```
plot(bayesinf)
```



Modello miscuglio multivariato con componenti Gaussiane

Si considerano i valori che rappresentano le rilevazioni rispetto a due esami clinici eseguiti su 200 pazienti (dati simulati). Nel file `data.Rdata` si supponga che Y_1 rappresenti il livello dei globuli bianchi e Y_2 il livello di emoglobina.

```
load("data.Rdata")
head(data)
```

	Y1.1	Y2.1
1	185.2594	222.6894
2	209.0801	248.5053
4	229.8731	205.3069
5	289.8899	218.1793
7	254.6754	222.6894
8	204.2028	188.8758

Per il primo paziente la rilevazione dei globuli bianchi è di 185 mentre quella di emoglobina di 223.

```
require(skimr)
skim_without_charts(data)
```

Table 15: Data summary

Name	data
Number of rows	200
Number of columns	2
_____	
Column type frequency:	
numeric	2
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
Y1.1	0	1	206.39	36.96	100.18	185.26	204.2	229.87	319.39
Y2.1	0	1	212.30	36.11	126.13	184.78	213.8	237.32	302.96

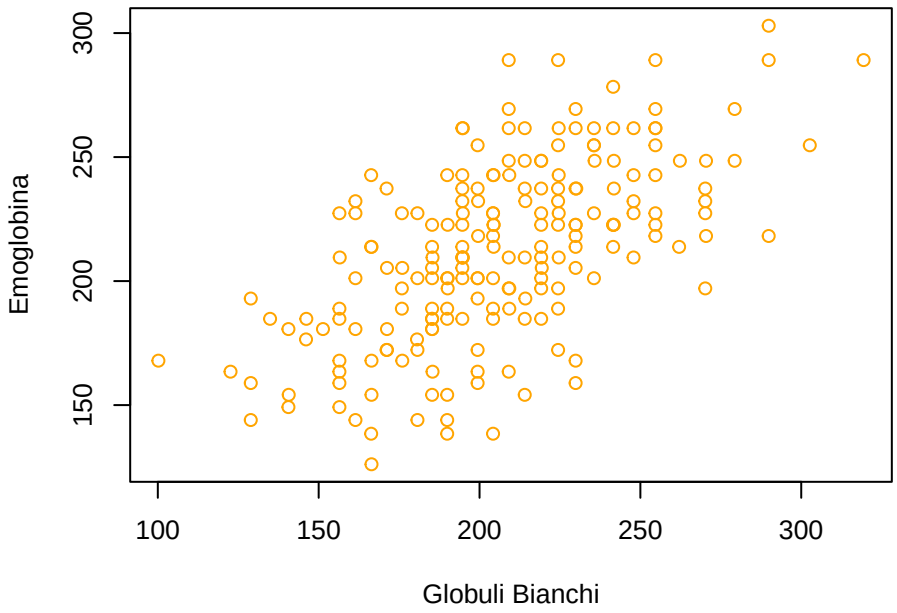
I pazienti hanno 206 come livello medio di globuli bianchi e 212 di emoglobina. La variabilità media intorno alla media è circa 37 per i globuli bianchi e 36 per l'emoglobina.

```
cov(data)
#>            Y1.1       Y2.1
#> Y1.1 1366.1894  812.4941
#> Y2.1  812.4941 1304.0416
cor(data)
#>           Y1.1      Y2.1
#> Y1.1 1.000000 0.608722
#> Y2.1 0.608722 1.000000
```

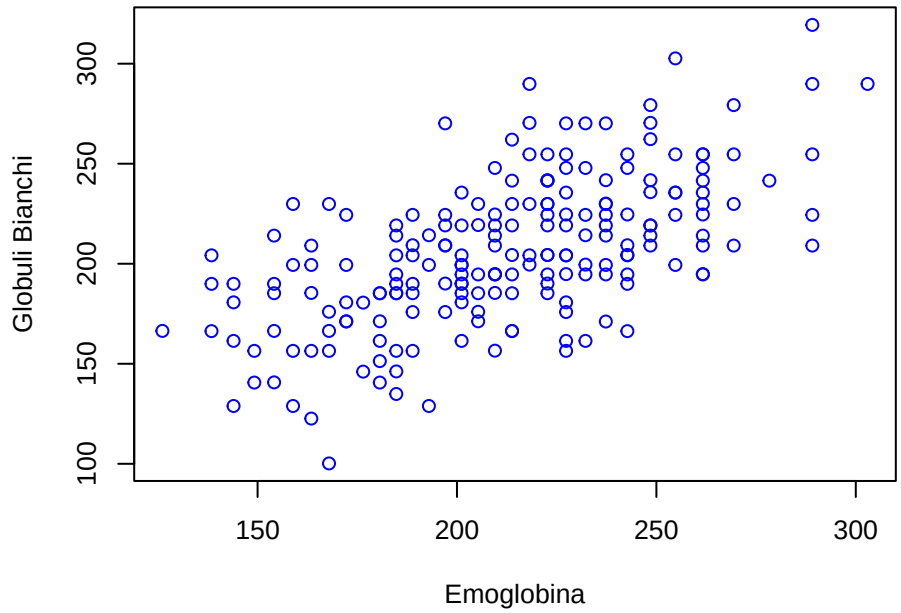
Il coefficiente di correlazione lineare campionario è 0.6 e suggerisce un'associazione positiva abbastanza elevata tra le due variabili.

Il due grafici evidenziano quest'associazione

```
plot(data$Y1.1, data$Y2.1, xlab = "Globuli Bianchi",
     ylab = "Emoglobina", col = "orange")
```




```
plot(data$Y2.1, data$Y1.1,
     xlab = "Emoglobina", ylab = "Globuli Bianchi", col = "blue")
```



Il modello miscuglio multivariato viene stimato con la libreria `mclust` utilizzando la **decomposizione spettrale** per la matrice di varianza-covarianza delle componenti. Inizialmente si considera il **modello sferico**.

Le specificazioni seguenti `EII` e `VII` definiscono il **modello sferico** ovvero il modello in cui le componenti sono indipendenti con uguale varianza `E` o diversa varianza `V`.

Selezione del numero delle componenti

Come nel caso univariato la selezione delle componenti avviene con il criterio d'informazione Bayesiano nel modo seguente

```
require(mclust)
mcc <-Mclust(data, modelNames = c("EII", "VII"))
mcc$BIC
#> Bayesian Information Criterion (BIC):
#>         EII       VII
#> 1 -4027.750 -4027.750
#> 2 -3972.182 -3977.513
#> 3 -3970.082 -3978.606
#> 4 -3974.086 -3988.241
#> 5 -3989.188 -4007.767
#> 6 -4002.328 -4025.495
#> 7 -4014.416 -4043.538
#> 8 -4030.288 -4058.401
#> 9 -4045.601 -4078.349
#>
#> Top 3 models based on the BIC criterion:
#>     EII,3     EII,2     EII,4
#> -3970.082 -3972.182 -3974.086
```

Valutando alcuni diversi modelli per un numero crescente di componenti con varianza uguale o diversa il criterio suggerisce 3 compomenti con uguale varianza uguale (E).

Stima dei parametri

La stima del modello selezionato come migliore avviene nel modo seguente

```
mc <-Mclust(data,
            G = 3,
            modelNames = c("EII"))
summary(mc,parameters = TRUE )
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust EII (spherical, equal volume) model with 3 components:
#>
```

```
#>  log-likelihood   n df       BIC       ICL
#>       -1961.199 200  9 -3970.082 -4053.728
#>
#> Clustering table:
#>   1   2   3
#> 105  55  40
#>
#> Mixing probabilities:
#>         1         2         3
#> 0.4897486 0.2813647 0.2288867
#>
#> Means:
#>          [,1]     [,2]     [,3]
#> Y1.1 202.5506 244.8475 167.3407
#> Y2.1 212.4403 246.9436 169.4106
#>
#> Variances:
#> [,,1]
#>          Y1.1     Y2.1
#> Y1.1 562.8875   0.0000
#> Y2.1   0.0000 562.8875
#> [,,2]
#>          Y1.1     Y2.1
#> Y1.1 562.8875   0.0000
#> Y2.1   0.0000 562.8875
#> [,,3]
#>          Y1.1     Y2.1
#> Y1.1 562.8875   0.0000
#> Y2.1   0.0000 562.8875
```

I parametri liberi di variare del modello *sono 9* perchè il modello è multivariato. I parametri sono i seguenti:

- I pesi di due componenti (mixing probabilities, il terzo si determina per differenza rispetto al vincolo a somma 1)
- I valori delle medie per le due variabili in ogni sottopolazione (6 medie)
- La varianza supposta uguale per tutte le componenti (1 varianza)

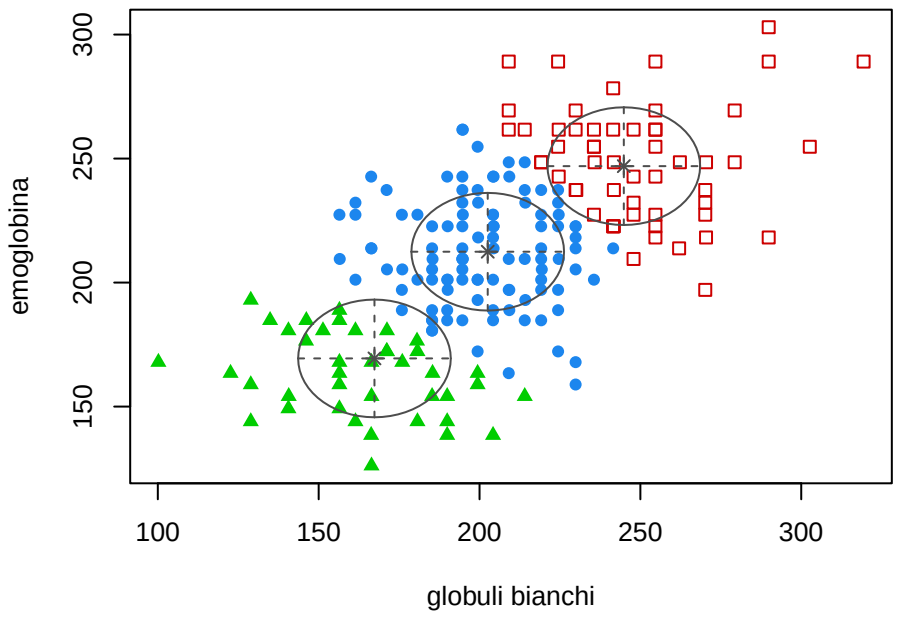
Dai valori stimati notiamo che:

- Il peso di ogni gruppo è il seguente 49% per il primo gruppo, 28% per il secondo, e 23% per il terzo.

- La terza sottopopolazione (gruppo 3) è formata da coloro che hanno i valori medi più bassi per entrambe le variabili.
- La prima sottopopolazione rappresenta coloro che hanno valori medi dei globuli bianchi ed emoglobina 202 e 212 rispettivamente.
- La seconda sottopopolazione è caratterizzata da coloro che hanno i valori medi più alti e pari a 245 e 247.
- La variabilià media intorno alla media è uguale per tutte le componenti ed è 23.

Il grafico di classificazione seguente mostra come vengono allocate le unità nei tre gruppi.

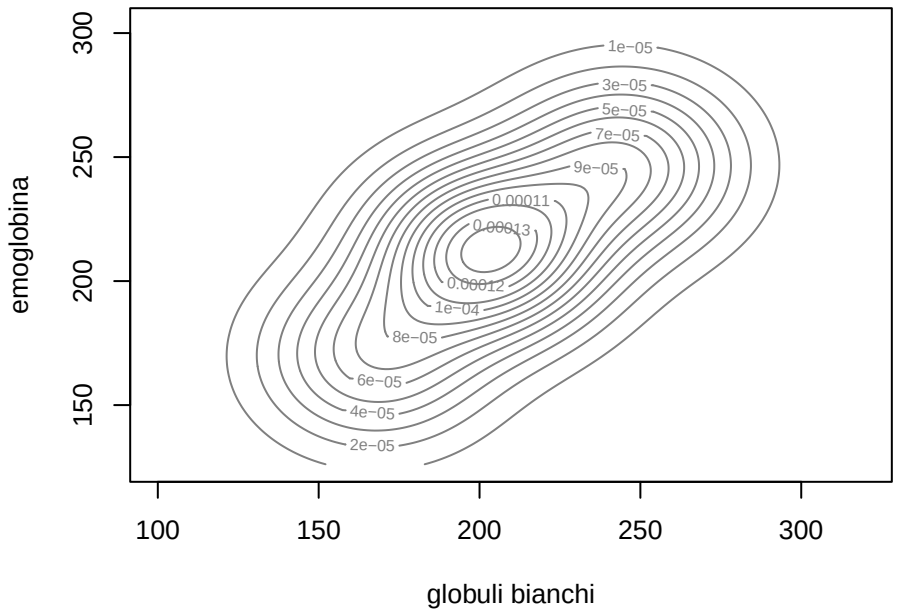
```
plot(mc,"classification", xlab = "globuli bianchi",
     ylab = "emoglobina")
```



Le ellissi rappresentano la densità univarita di ogni componente: hanno stessa forma con fuoco corrispondente alle medie stimate di gruppo e non hanno orientamento.

Le curve di livello della densità miscuglio si ottengono nel modo seguente.

```
plot(mc,"density", xlab = "globuli bianchi", ylab = "emoglobina")
```



La densità maggiore è concentrata nel punto di intersezione delle medie campionarie. Le ellissi sono orientate a destra dato che esiste correlazione positiva tra le due variabili.

Modello miscuglio con componenti sferiche e varianze specifiche per ogni componente

Il modello con distribuzione non omogenea tra i gruppi postula varianze diverse per ogni componente e si specifica con l'opzione `VII`

```
mc1 <-Mclust(data, G = 3, modelNames = c( "VII"))
summary(mc1, parameters = TRUE)
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust VII (spherical, varying volume) model with 3 components:
#>
#>  log-likelihood   n df       BIC       ICL
#>       -1960.162 200 11 -3978.606 -4064.025
#>
#> Clustering table:
#>  1  2  3
#> 88 66 46
#>
#> Mixing probabilities:
```

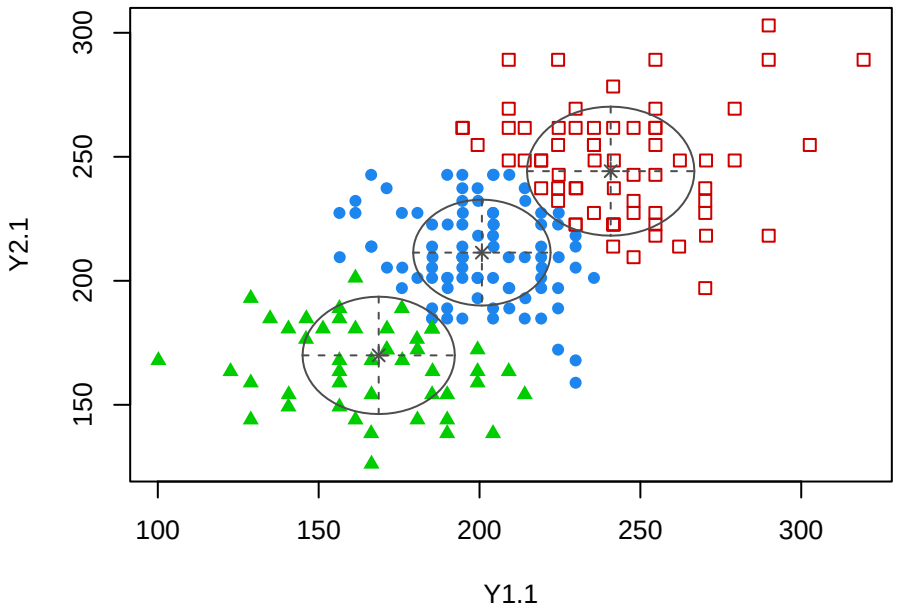
```
#>         1         2         3
#> 0.4174926 0.3375221 0.2449853
#>
#> Means:
#>          [,1]     [,2]     [,3]
#> Y1.1 200.7391 240.7749 168.6567
#> Y2.1 211.3657 244.2084 169.9287
#>
#> Variances:
#> [,,1]
#>          Y1.1     Y2.1
#> Y1.1 455.8909   0.0000
#> Y2.1   0.0000 455.8909
#> [,,2]
#>          Y1.1     Y2.1
#> Y1.1 676.6679   0.0000
#> Y2.1   0.0000 676.6679
#> [,,3]
#>          Y1.1     Y2.1
#> Y1.1 560.0236   0.0000
#> Y2.1   0.0000 560.0236
```

Dai risultati si nota che:

- I parametri liberi di variare *sono 11* ovvero 3 in più rispetto al modello stimato in precedenza.
- Il valore del BIC è superiore a quello del modello precedente
- La percentuale di pazienti appartenenti alla prima componente è inferiore a quello stimato nel modello precedente
- I pesi delle componenti non sono molto diversi da quelli del modello precedente
- I valori delle medie di ogni componente sono simili a quelli stimati in precedenza
- La seconda componente rappresenta la sottopolazione di pazienti con valori più altri dei globuli biachi e emoglobina ma anche con maggiore variabilità rispetto alle altre due componenti. La deviazione standard stimata è 26.
- L'ampiezza dell'ellissi del gruppo 3 (rosso) è maggiore rispetto a quella del gruppo 2 (blu), a causa della diversa variabilità e per ipotesi di modello sferico le ellissi non hanno orientamento.

Si noti che il seguente grafico mostra ellissi di ampiezza diversa che rappresentano le tre componenti aventi diversa variabilità.

```
plot(mc1,"classification")
```



Modello miscuglio non sferico

Nel caso in cui si supponga anche una covarianza tra le variabili risposta diversa da zero specifica per ogni componente si stima il modello miscuglio con l'opzione `VEE` che assume la stessa forma per le densità e stesso orientamento

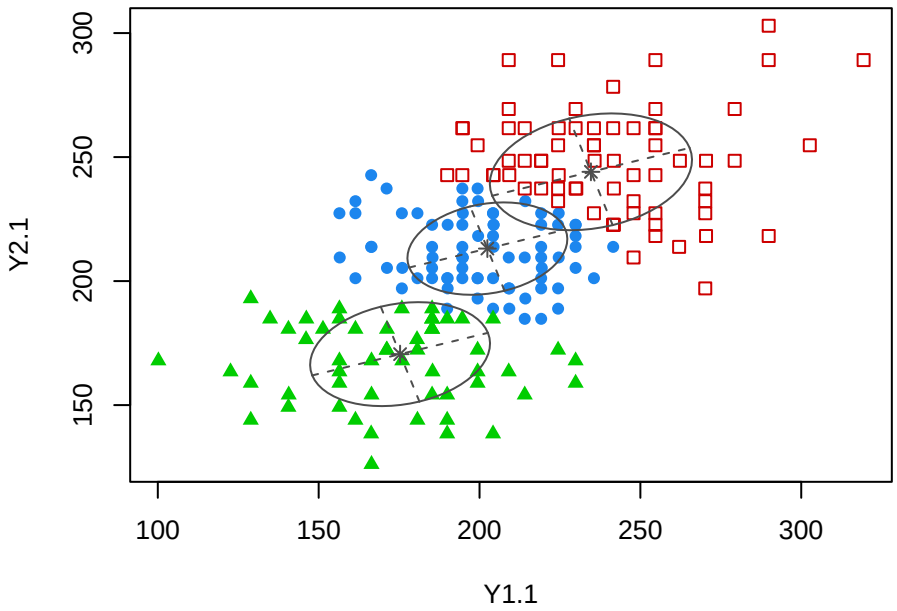
```
mc2 <-Mclust(data, G = 3, modelNames = c( "VEE"))
summary(mc2, parameters = TRUE)
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust VEE (ellipsoidal, equal shape and orientation) model with 3 components:
#>
#>  log-likelihood   n df       BIC       ICL
#>       -1956.648 200 13 -3982.174 -4089.575
#>
#> Clustering table:
```

```
#>  1  2  3
#> 77 69 54
#>
#> Mixing probabilities:
#>         1         2         3
#> 0.3615714 0.3586530 0.2797756
#>
#> Means:
#>           [,1]     [,2]     [,3]
#> Y1.1 202.4390 234.6283 175.3050
#> Y2.1 213.1036 244.0587 170.5467
#>
#> Variances:
#> [,,1]
#>            Y1.1      Y2.1
#> Y1.1 617.83969  94.39148
#> Y2.1  94.39148 347.15298
#> [,,2]
#>           Y1.1     Y2.1
#> Y1.1 987.4120 150.8535
#> Y2.1 150.8535 554.8090
#> [,,3]
#>           Y1.1     Y2.1
#> Y1.1 782.0067 119.4724
#> Y2.1 119.4724 439.3955
```

Dai risultati si nota che

- i parametri liberi di variare *sono 13* ovvero 2 in più rispetto al modello precedemte (5 in più rispetto al modello iniziale), la covarianza tra le variabili risposta è maggiore nella seconda componente (quella in cui le medie delle due variabili sono più grandi)

```
plot(mc2,"classification")
```



Bootstrap per errori standard e intervalli di confidenza

Gli intervalli di confidenza per i parametri del modello si ottengono con il bootstrap applicando il metodo del percentile.

Nel seguito si illusta l'applicazione del bootstrap per il primo modello che assume struttura sferica e assenza di correlazione (`EII`).

La funzione `mclust::bootClust` permette di ottenere i risultati del campionamento boostrap utilizzando di default $B = 999$ campioni bootstrap

```
bootClust <- MclustBootstrap(mc)
```

La funzione summary restituisce i risultati riportando gli errori standard stimati per ogni parametro

```
summary(bootClust, what = "se")
#> ---------------------------------------------------------
#> Resampling standard errors
#> ---------------------------------------------------------
#> Model                       = EII
```

```
#> Num. of mixture components = 3
#> Replications               = 999
#> Type                       = nonparametric bootstrap
#>
#> Mixing probabilities:
#>         1         2         3
#> 0.06830526 0.09941863 0.07588911
#>
#> Means:
#>             1        2        3
#> Y1.1 7.563568 16.60598 7.475757
#> Y2.1 8.214820 11.32911 5.311737
#>
#> Variances:
#> [,,1]
#>         Y1.1    Y2.1
#> Y1.1 48.2646  0.0000
#> Y2.1  0.0000 48.2646
#> [,,2]
#>         Y1.1    Y2.1
#> Y1.1 48.2646  0.0000
#> Y2.1  0.0000 48.2646
#> [,,3]
#>         Y1.1    Y2.1
#> Y1.1 48.2646  0.0000
#> Y2.1  0.0000 48.2646
```

e gli l'intervallo di confidenza calcolati al livello di confidenza del 95%.

```
summary(bootClust, what = "ci")
#> ------------------------------------------------------------
#> Resampling confidence intervals
#> ------------------------------------------------------------
#> Model                      = EII
#> Num. of mixture components = 3
#> Replications               = 999
#> Type                       = nonparametric bootstrap
#> Confidence level           = 0.95
#>
```

```
#> Mixing probabilities:
#>                1          2         3
#> 2.5%  0.3655952 0.03388683 0.1262837
#> 97.5% 0.6446172 0.39446999 0.4370823
#>
#> Means:
#> [,,1]
#>           Y1.1     Y2.1
#> 2.5%  195.5367 201.2128
#> 97.5% 224.9258 233.4173
#> [,,2]
#>           Y1.1     Y2.1
#> 2.5%  234.1689 239.6560
#> 97.5% 297.3753 285.3773
#> [,,3]
#>           Y1.1     Y2.1
#> 2.5%  152.7594 162.7716
#> 97.5% 181.2393 184.0687
#>
#> Variances:
#> [,,1]
#>           Y1.1     Y2.1
#> 2.5%  453.0529 453.0529
#> 97.5% 639.8634 639.8634
#> [,,2]
#>           Y1.1     Y2.1
#> 2.5%  453.0529 453.0529
#> 97.5% 639.8634 639.8634
#> [,,3]
#>           Y1.1     Y2.1
#> 2.5%  453.0529 453.0529
#> 97.5% 639.8634 639.8634
```

Modello a classi latenti

Si considerino i dati seguenti che riguardano le risposte a due test somministrati ai pazienti di un reparto di oncologia per valutare la *depressione* e l'*ansia*. (I dati sono tratti da Zigmond, A. and Snaith, R. (1983), The hospital anxiety and depression scale, *Acta Psychiatrika Scandinavica*, 67, 361-370.)

```
load("psico.Rdata")
dim(Y)
#> [1] 201   2
head(Y)
#>      [,1] [,2]
#> [1,]    1    1
#> [2,]    0    0
#> [3,]    0    0
#> [4,]    1    1
#> [5,]    0    0
#> [6,]    0    0
tail(Y)
#>        [,1] [,2]
#> [196,]    1    0
#> [197,]    2    1
#> [198,]    1    1
#> [199,]    0    0
#> [200,]    3    3
#> [201,]    0    0
```

Le osservazioni nella prima colonna sono riferite al primo item del questionario che misura la *depressione* e quelle della seconda colonna sono riferite al secondo item che misura *l'ansia*.

Gli item sono misurati su una scala ordinale con quattro livelli codificati da 0 a 3: 0 basso, 1 medio-basso, 2 medio, 3 alto.

201 pazienti hanno risposto al questionario e la frequenza relativa di ogni categoria può essere visualizzata con la funzione apply nel modo seguente

```
n<-dim(Y)[1]
apply(Y,2,table)/n
#>          [,1]       [,2]
#> 0 0.35323383 0.39800995
#> 1 0.52736318 0.46268657
```

```
#> 2 0.07960199 0.09950249
#> 3 0.03980100 0.03980100
```

Dalle frequenze relative per ogni categoria di risposta si nota che il 52% dei pazienti ha una depressione medio-bassa ed il 46% un livello di ansia medio-basso. Solo il 4% presenta alti livelli di depressione e di ansia.

Il modello a classi latenti con distribuzione non parametrica per la variabile latente con supporto discreto e supposta con un numero di punti di supporto finito viene stimato con la libreria `MultiLCIRT` (Multidimensional Latent Class Item Response Theory) e la funzione `est_multi_poly`.

La funzione richiede che si determini il pattern delle configurazioni di risposta utilizzando la funzione `aggr_data` nel modo seguente

```
require(MultiLCIRT)
Yout <- aggr_data(Y)
S <- Yout$data_dis;S
#>       [,1] [,2]
#>  [1,]    1    1
#>  [2,]    0    0
#>  [3,]    1    0
#>  [4,]    2    2
#>  [5,]    2    1
#>  [6,]    0    1
#>  [7,]    1    2
#>  [8,]    0    2
#>  [9,]    2    0
#> [10,]    3    1
#> [11,]    3    3
#> [12,]    3    2
#> [13,]    2    3
#> [14,]    1    3
yv <- Yout$freq; yv
#>  [1] 77 59 20  6  6  7  8  5  1  3  4  1  3  1
cbind(S,yv)
#>           yv
#>  [1,] 1 1 77
#>  [2,] 0 0 59
#>  [3,] 1 0 20
#>  [4,] 2 2  6
#>  [5,] 2 1  6
#>  [6,] 0 1  7
```

```
#>  [7,] 1 2  8
#>  [8,] 0 2  5
#>  [9,] 2 0  1
#> [10,] 3 1  3
#> [11,] 3 3  4
#> [12,] 3 2  1
#> [13,] 2 3  3
#> [14,] 1 3  1
```

L'oggetto `S` restituito dalla funzione contiene le configurazione congiunte dei livelli delle due variabili risposta, mentre il vettore `yv` associa le rispettive frequenze campionarie. Pertanto vi sono 79 pazienti che hanno un livello di depressione e di ansia medio-basso. Vi sono 4 pazienti che presentano un livelli di depressione e di ansia alto.

La funzione per stimare il modello richiede `S` e `yv` e richiede di specificare il numero di classi latenti k. Supponendo per il momento l'esistenza di due classi latenti la funzione restituisce il seguente output

```
mod2 <- est_multi_poly(S,yv,k=2)
#> *-------------------------------------------------------------------------*
#> Link of type =                  0
#> Discrimination index =          0
#> Constraints on the difficulty = 0
#> Type of initialization =        0
#> *-------------------------------------------------------------------------*

summary(mod2)
#>
#> Call:
#> est_multi_poly(S = S, yv = yv, k = 2)
#>
#> Log-likelihood:
#> [1] -374.62
#>
#> AIC:
#> [1] 775.23
#>
#> BIC:
#> [1] 818.18
#>
#> Class weights:
#> [1] 0.4782 0.5218
```

```
#>
#> Conditional response probabilities:
#> , , class = 1
#>
#>          item
#> category      1      2
#>        0 0.7387 0.8324
#>        1 0.2474 0.1024
#>        2 0.0138 0.0652
#>        3 0.0000 0.0000
#>
#> , , class = 2
#>
#>          item
#> category      1      2
#>        0 0.0000 0.0000
#>        1 0.7839 0.7928
#>        2 0.1399 0.1309
#>        3 0.0763 0.0763
mod2$np
#> [1] 13
```

Il valore della log-verosimiglianza a convergenza dell'algoritmo EM è -374.62 ed il modello con due classi latenti ha 13 parametri liberi di variare, ed il valore del criterio d'informazione Bayesiano è 818.18.

Dalle stime dei parametri notiamo che il peso della prima componente è 0.48 mentre quello della seconda componente è 0.52.

Dalle stime delle probabilità condizionate possiamo caratterizzare la *prima classe latente* come quella riferita alla sottopopolazione dei pazienti a basso rischio di depressione ed ansia in quanto la probabilità di riportare un livello alto di depressione o di ansia è nulla ed è elevata la probabilità di risposta nella prima categoria di entrambe le variabili.

La *seconda classe latente* rappresenta più della metà della popolazione dei pazienti ed è caratterizzata da coloro che sono maggiormente inclini alla depressione e all'ansia in quanto la probabilità della prima categoria è nulla mentre è più alta la probabilità per le restanti categorie.

La classificazione dei pazienti nei due gruppi avviene con le probabilità a posteriori. Questa si richiede specificando `output = TRUE` nella funzione di stima e poi le probabilità sono contenute nell'oggetto `Pp` del modello.

```
mod2 <- est_multi_poly(S,yv,k=2,
                       output = TRUE)
#> *-------------------------------------------------------------------------*
#> Link of type =                   0
#> Discrimination index =           0
#> Constraints on the difficulty = 0
#> Type of initialization =         0
#> *-------------------------------------------------------------------------*
round(mod2$Pp,2)
#>       [,1] [,2]
#>  [1,] 0.04 0.96
#>  [2,] 1.00 0.00
#>  [3,] 1.00 0.00
#>  [4,] 0.04 0.96
#>  [5,] 0.01 0.99
#>  [6,] 1.00 0.00
#>  [7,] 0.13 0.87
#>  [8,] 1.00 0.00
#>  [9,] 1.00 0.00
#> [10,] 0.00 1.00
#> [11,] 0.00 1.00
#> [12,] 0.00 1.00
#> [13,] 0.00 1.00
#> [14,] 0.00 1.00
```

Si nota che per coloro che hanno un pattern di risposta 1,1 (prima riga della matrice S) le probabilità stimate a posteriori sono 0.036 e 0.964 di appartenere alla prima e alla seconda classe latente rispettivamente. Con la regola della massima probabilità a posteriori questi pazienti vengono allocati nella seconda classe latente.

Considerando coloro che hanno pattern di risposta 1, 2 (riga 7 di S) si nota che alla riga 7 di `Pp` si ha 0.126 e 0.874. Per cui coloro che hanno un livello di depressione medio-basso ed un livello di ansia medio vengono allocati con probabilità 0.9 nella seconda classe latente.