Salvatore Ingrassia
Francesca Greselin
Isabella Morlini

Modelli Mistura e Algoritmo EM

Course on "Computational Statistics" - Nova Universitas
Macerata, 25-26 Settembre 2008

Indice

Capitolo 1

Introduzione ai Modelli Mistura

Ouverture

Mi capita spesso di domandare che nell'insegnamento si faccia un maggiore uso dei metodi che favoriscono l'intuizione e di sentirmi rispondere che innanzitutto bisogna sviluppare il senso del rigore. Questa risposta implica un malinteso che mi stupisce sempre; a mio avviso è un errore contrapporre intuizione e rigore. L'intuizione favorisce la scoperta ed aiuta a comprenderla; una volta che ci ha suggerito un enunciato, si tratta di dimostrarlo rigorosamente. L'intuizione è utile per arrivare ad un risultato come gli occhi lo sono al viandante che si serve dei suoi piedi per arrivare alla meta, ed è così assurdo il combatterla come lo sarebbe il raccomandare al viandante di chiudere gli occhi per arrivare più sicuramente alla meta. L'intuizione ed il rigore scientifico sono due qualità che si completano.

Paul Levy, *Théorie de l'addition des variables aléatoires*, Gauthier-Villars, Paris, 1954

In questa epoca gli uomini stanno provando l'esperienza emozionante che si ha quando si cerca di indovinare il modo in cui la natura si comporterà in una nuova situazione mai vista prima. Da esperimenti e informazioni in un certo campo si può indovinare quello che accadrà in una regione che nessuno ha ancora esplorato. E' un po' diverso dalla esplorazione normale per il fatto che sulla terra esplorata ci sono già abbastanza indizi per indovinare come sarà quella non ancora scoperta. Queste ipotesi, invece, sono spesso molto diverse da quello che si è già visto, e richiedono un grande sforzo di pensiero.

Cosa c'è nella natura che permette di indovinare che questo accada, rendendo possibile, conoscendo una parte, come si comporterà il resto? Questa è una domanda non scientifica, cui non so come rispondere, e perciò darò una risposta non scientifica: io credo che è perché la natura ha in sé una grande semplicità e perciò una grande bellezza.

Richard Feynman, *La Legge Fisica*, Boringhieri, 1971

1.1 Introduzione

I modelli mistura costituiscono uno degli argomenti più interessanti della modellistica statistica, come si può vedere da alcune belle ed interessanti monografie che negli ultimi 20 anni sono state dedicate all'argomento: Titterington *et al.* (1985), McLachlan and Basford (1988), McLachlan and Peel (2000), Frühwirth-Schnatter (2006). La storia comincia con un articolo di Pearson (1894) su una mistura di due densità normali univariate per modellare una distribuzione di misurazioni inerenti $N = 1000$ granchi raccolte nella spiaggia di Napoli da Weldon (1892, 1893)[1], vedi Figura 1.1 a); la stima venne ottenuta utilizzando il metodo dei momenti. Una pietra miliare nella costruzione di modelli mistura è stato l'articolo di Dempster *et al.* (1977) sull'algoritmo EM, che ha consentito la stima dei parametri del modello in accordo al metodo della massima verosimiglianza. L'algoritmo EM è uno strumento molto potente, oggetto di una monografia di McLachlan and Krishnan (2008).

Per introdurci all'argomento, consideriamo una popolazione Ω costituita da g sottogruppi, cioè $\Omega = \Omega_1 \cup \cdots \cup \Omega_g$, i cui elementi siano mescolati casualmente in proporzioni $\alpha_1, \ldots, \alpha_g$. Si assuma di essere interessati ad una certa quantità X che sia eterogenea fra i gruppi ed omogenea all'interno dei gruppi. Per tale motivo, la variabile aleatoria X avrà una distribuzione di probabilità differente in ogni gruppo, e possiamo assumere che le distribuzioni di probabilità all'interno di ciascun gruppo appartengano alla stessa famiglia parametrica $f(x;\boldsymbol{\theta})$, dove il parametro $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ varia da un gruppo all'altro. I gruppi possono essere indicizzati mediante una variabile discreta S a valori nell'insieme $\{1, \ldots, g\}$.

Quando si effettua un campionamento casuale da una popolazione di questo tipo, possiamo osservare non solo la quantità X, ma anche la variabile indicatrice S. La probabilità di scegliere il gruppo $S = j$ con $j \in \{1, \ldots, g\}$ viene indicata con $h(j)$, e risulta uguale a α_j ($j = 1, \ldots, g$), mentre condizionatamente a $S = j$, la quantità X è una variabile aleatoria avente distribuzione $f(x|\boldsymbol{\theta}_j)$, essendo $\boldsymbol{\theta}_j$ il parametro relativo al j-esimo gruppo. La densità congiunta $p(x, j)$ è data da:

$$p(x, j) = p(x|j)h(j) = f(x|\boldsymbol{\theta}_j)h(j)\ . \tag{1.1}$$

Un modello mistura finita di distribuzioni nasce nei casi in cui non è possibile osservare l'indicatore di gruppo S, e quindi si osserva unicamente la variabile aleatoria X. La densità marginale di X è allora data da:

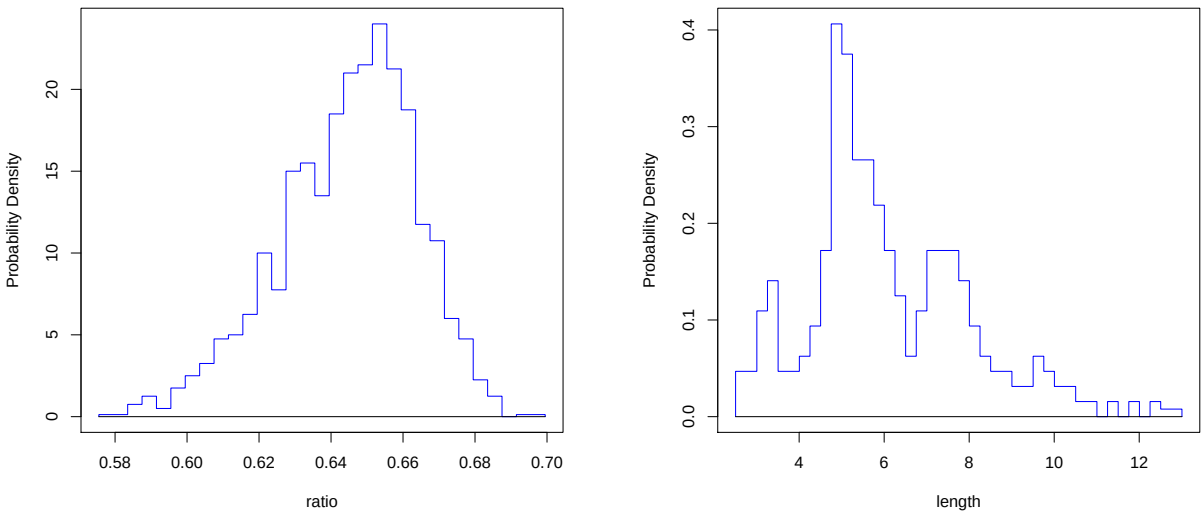
$$p(x;\boldsymbol{\psi}) = \sum_{j=1}^{g} p(x, j) = \alpha_1 f(x|\boldsymbol{\theta}_1) + \cdots + \alpha_g f(x|\boldsymbol{\theta}_g)\ .$$

dove con $\boldsymbol{\psi}$ abbiamo indicato il vettore di tutti i parametri del modello. Per comprendere meglio, consideriamo il file dati *cassie.csv* che contiene $N = 256$ osservazioni inerenti la lunghezza di pesci (in particolare, dentici), studiato in Cassie (1954), vedi Figura 1.1 b). L'istogramma mostra che la distribuzione è multimodale, ed una possibile spiegazione è che i pesci appartengano a diversi gruppi di età (e quindi abbiano lunghezza differente). Poichè non è possibile (o comunque è molto difficile) misurare l'età dei pesci, non possiamo osservare il gruppo di appartenenza e quindi l'eterogeneità non può essere osservata.

Nota Computazionale 1.1 Recentemente è stato rilasciato il pacchetto `mixdist` per R. Utilizzando la routine *plot.mix* possiamo disegnare l'istogramma di dati espressi come distribuzioni di frequenza:

```
library(mixdist)
data(pearson)
plot.mix(pearson)
```

[1] Dataset: *pearson.csv*.



a) *pearson* b) *cassie*

Figura 1.1: *Istogrammi delle distribuzioni dei dati* pearson.csv *e* cassie.csv.

ed analogamente

```
data(cassie)
plot.mix(cassie)
```

1.2 Definizioni e concetti preliminari

Definizione 1.2 Sia $X : \Omega \to \mathcal{X}$ una variabile aleatoria (v.a.) a valori in uno spazio campionario $\mathcal{X} \subseteq \mathbb{R}$. Diremo che X segue una *distribuzione mistura finita* se la sua legge può essere rappresentata mediante una funzione di densità di probabilità (fdp) del tipo

$$p(x) = \alpha_1 f_1(x) + \cdots + \alpha_g f_g(x) \qquad x \in \mathcal{X}\,.$$

dove

$$\alpha_j > 0 \quad (j = 1, \ldots, g), \qquad \alpha_1 + \cdots + \alpha_g = 1$$

e

$$f_j(\cdot) \geq 0\,, \qquad \int_{\mathcal{X}} f_j(x)dx = 1\,, \qquad j = 1, \ldots, g\,.$$

Più in generale possiamo considerare un vettore aleatorio $\mathbf{X}$ a valori in $\mathcal{X} \subseteq \mathbb{R}^q$ avente funzione di densità

$$p(\mathbf{x}) = \alpha_1 f_1(\mathbf{x}) + \cdots + \alpha_g f_g(\mathbf{x}) \qquad x \in \mathcal{X}\,. \tag{1.2}$$

con

$$f_j(\cdot) \geq 0\,, \qquad \int_{\mathcal{X}} f_j(\mathbf{x})d\mathbf{x} = 1\,, \qquad j = 1, \ldots, g\,.$$

La fdp $p(\cdot)$ viene chiamata *funzione di densità miscuglio (finito)*.

Nel seguito considereremo il caso generale q-dimensionale.

I parametri $\alpha_1, \ldots, \alpha_g$ sono chiamati *pesi della mistura* (mixing weights) e le $f_1(\cdot), \ldots, f_g(\cdot)$ sono le *densità componenti* della mistura. Si verifica immediatamente che la (1.2) definisce una funzione di densità di probabilità.

In molte situazioni, la forma parametrica delle $f_1(\cdot), \ldots, f_g(\cdot)$ risulta assegnata e pertanto possiamo specificare in maniera esplicita la (1.2) come segue:

$$p(\mathbf{x}; \boldsymbol{\psi}) = \alpha_1 f_1(\mathbf{x}; \boldsymbol{\theta}_1) + \cdots + \alpha_g f_g(\mathbf{x}; \boldsymbol{\theta}_g) \qquad \mathbf{x} \in \mathcal{X} \tag{1.3}$$

dove θ_j denota l'insieme dei parametri di $f_j(\cdot)$ e $\boldsymbol{\psi}$ denota l'insieme di tutti i parametri distinti del modello mistura.

In generale non si richiede che le densità componenti nella (1.3) debbano appartenere alla stessa famiglia parametrica, anche se ciò accade nella maggior parte delle applicazioni pratiche. In questo caso, la funzione di densità della mistura assume la forma seguente:

$$p(\mathbf{x}; \boldsymbol{\psi}) = \alpha_1 f(\mathbf{x}; \theta_1) + \cdots + \alpha_g f(\mathbf{x}; \theta_g) \qquad \mathbf{x} \in \mathcal{X} \tag{1.4}$$

dove $f(\cdot; \theta_j)$ ($j = 1, \ldots, g$) denota la generica componente della famiglia parametrica. Un'importante classe di misture parametriche è costituita dalle misture gaussiane, cioè da misture le cui componenti sono distribuzioni normali. Nel seguito la funzione di densità di una v.a. $X \sim N(\mu, \sigma^2)$ verrà denotata con $\phi(x; \mu, \sigma^2)$. L'argomento verrà approfondito nel paragrafo 1.6.

Dal punto di vista applicativo, possiamo considerare due diversi tipi di applicazione dei modelli mistura (anche se il confine non sempre risulta chiaramente delineato): come modello semiparametrico per la stima di densità non note (applicazioni indirette) oppure per la costruzione di modelli di *clustering* (approccio diretto).

1.2.1 Applicazioni dirette

Sono le applicazioni generate nel contesto in cui possiamo assumere che la popolazione Ω sia costituita da un certo numero g di *gruppi* o *categorie* o *sottopopolazioni*, cioè $\Omega = \Omega_1 \cup \Omega_2 \cup \cdots \cup \Omega_g$ e pertanto ciascuna unità statistica $\omega \in \Omega$ appartenga ad un solo gruppo Ω_j ($j = 1, \ldots, g$). In questo caso la densità $f_j(\cdot)$ descrive la distribuzione di X assumendo che l'unità ω provenga dalla categoria j e α_j denota la probabilità che ω provenga da questa categoria. Per comprendere meglio questo caso, consideriamo un insieme di dati[2] studiato in Campbell and Mahon (1974) concernente granchi della specie *Leptograpsus*. Ciascuna osservazione è basata su 5 variabili: ampiezza della bocca (*FL*), ampiezza dela parte posteriore (*RW*), lunghezza lungo la linea media (*CL*), massima ampiezza (*CW*) e profondità (*BD*) della carapace; i dati sono raggruppati per sesso, vedi figura Figure 1.2. Se rappresentiamo anche la funzione di densità, emergono i due gruppi, vedi Figura 1.3.

In questo contesto, la realizzazione $\mathbf{x}_1, \ldots, \mathbf{x}_N$ di un campione $\mathbf{X}_1, \ldots, \mathbf{X}_N$ di N v.a. iid aventi distribuzione miscuglio con g componenti con densità $p(\mathbf{x})$ data in (1.2) può essere ottenuto come segue. Sia Z_j una va categoriale a valori in $\{1, \ldots, g\}$ con probabilità rispettivamente uguali a $\alpha_1, \ldots, \alpha_g$, e supponiamo che la densità condizionata di X_n ($n = 1, \ldots, N$) dato $Z_n = j$ sia $f_j(x_n)$, ($j = 1, \ldots, g$). Allora la densità (non condizionata) di X_n (cioè la sua densità marginale) è data da $f_j(x_n)$. In questo contesto, la variabile Z_n può essere pensata come l'etichetta identificativa di X_n. In generale, è più conveniente lavorare con un vettore g-dimensionale $\mathbf{Z}_n = (Z_{nj}, j = 1, \ldots, g)$ al posto della singola variabile categoriale Z_n, dove $Z_{nj} = 1$ se $\mathbf{X}_n$ proviene dalla j-esima popolazione e $Z_{nj} = 0$ altri-

[2] Dataset *crabs.txt*. Nell'applicazione consideriamo solo le prime 100 osservazioni.

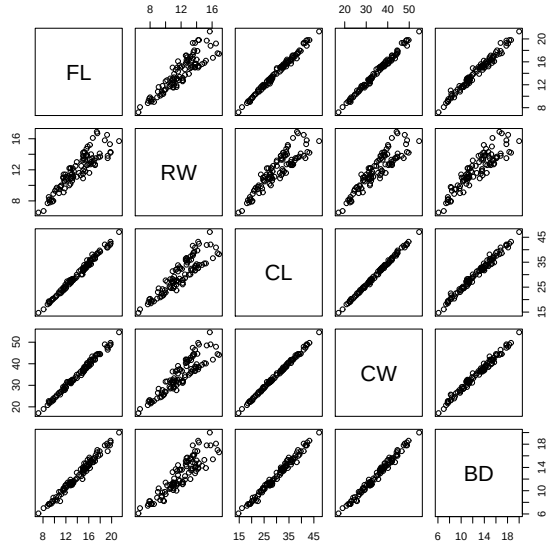


Figura 1.2: *Matrice di diagrammi a dispersione del crab dataset.*

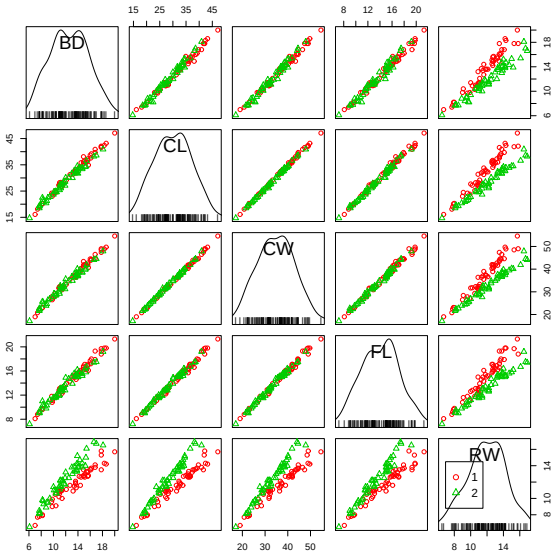


Figura 1.3: *Matrice di diagrammi a dispersione del crab dataset con grafico della funzione di densità per ciascuna variabile e classificazione dei dati.*

menti. Pertanto $\mathbf{Z}_n$ segue una distribuzione multinomiale[3] corrispondente all'estrazione di una delle categorie $1, \ldots, g$ con probabilità $\alpha_1, \ldots, \alpha_g$, cioè

$$P(\mathbf{Z}_n = \mathbf{z}_n) = \alpha_1^{z_{1n}} \alpha_2^{z_{2n}} \cdots \alpha_g^{z_{gn}}$$

e scriveremo

$$\mathbf{Z}_n \sim M_g(1, \boldsymbol{\alpha}) \qquad \text{con} \quad \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_g)' \, .$$

Per comprendere l'approccio sopra delineato, cosideriamo la generazione di un campione $\{\mathbf{X}_n\}_{n=1,\ldots,N}$ di v.a. iid dove $\mathbf{X}_n$ è estratto da una popolazione Ω costituita da g gruppi $\Omega_1, \ldots, \Omega_g$, – cioè $\Omega = \Omega_1 \cup \cdots \cup \Omega_g$ – in proporzioni rispettivamente $\alpha_1, \ldots, \alpha_g$. Se la densità di X_n in Ω_j è data da $f_j(\mathbf{x}_n)$, for $j = 1, \ldots, g$, allora la densità di $\mathbf{X}_n$ è una mistura di g componenti del tipo (1.2). In questo caso, i dati categorizzati possono essere rappresentati come segue:

$$\{\mathbf{y}_n = (\mathbf{x}_n, \mathbf{z}_n) \, : \, n = 1, \ldots, N\} \tag{1.5}$$

dove ciascun $\mathbf{z}_n = (z_{nj} \, : \, j = 1, \ldots, g)$ è il vettore indicatore g-dimensionale avente 1 nella posizione corrispondente alla categoria corretta e zero altrove. Questo argomento verrà approfondito in particolare nel Capitolo 5.

1.2.2 Applicazioni indirette

In questo caso la distribuzione mistura viene considerata come semplice e flessibile strumento matematico per la stime di densità. In altre parole, i modelli mistura costituiscono un interessante approccio semiparametrico per la stima di forme distribuzionali non note come alternativa ai metodi *kernel*. Ad esempio, una mistura di due normali, in cui una componente presenta una varianza *inflated*, può essere utilizzata per descrivere densità con code pesanti. In generale è possibile rappresentare una grande varietà di funzioni di densità mediante combinazioni lineari convesse di funzioni di densità componenti. Nelle Figure 1.4,1.5 vengono riportati i grafici della mistura

$$p(x; \alpha, \Delta) = \alpha \, \phi(x; 0, 1) + (1 - \alpha) \, \phi(x; \Delta, 1) \tag{1.6}$$

per $\alpha = 0.5$ e $\alpha = 0.75$ e $\Delta = 1, 2, 3, 4$.

Nota Computazionale 1.3 La routine in R per generare i grafici delle misture (1.6) è data in Algoritmo 6.8. La procedura – che utilizza la funzione `mixt2.r`, vedi Algoritmo 6.1 – richiede in ingresso il peso α della prima componente $N(0, 1)$ e fornisce i diagrammi per la mistura avente come

[3] **Distribuzione multinomiale**. Sia $\mathbf{X}$ un vettore aleatorio categoriale, con g categorie codificate come $\{1, \ldots, g\}$. Diremo che $\mathbf{X} \sim \text{Mult}(n, p_1, \ldots, p_g)$ se ha la seguente funzione di densità:

$$p(x_1, \ldots, x_g) = \begin{cases} \dfrac{n!}{x_1! \cdots x_g!} p_1^{x_1} \cdots p_g^{x_g} & \text{se } x_1, \ldots, x_g \in \mathbb{N} \, : \, \sum_{j=1}^g x_j = n \\ 0 & \text{altrimenti} \end{cases}$$

Risulta:

$$\begin{aligned} \mathbb{E}(X_j) &= np_j \, , \\ \text{Var}(X_j) &= np_j(1 - p_j) \, , \\ \text{Cov}(X_i, X_j) &= E[(X_i - np_i)(X_j - np_j)] \, = \, -np_i p_j \, . \end{aligned}$$

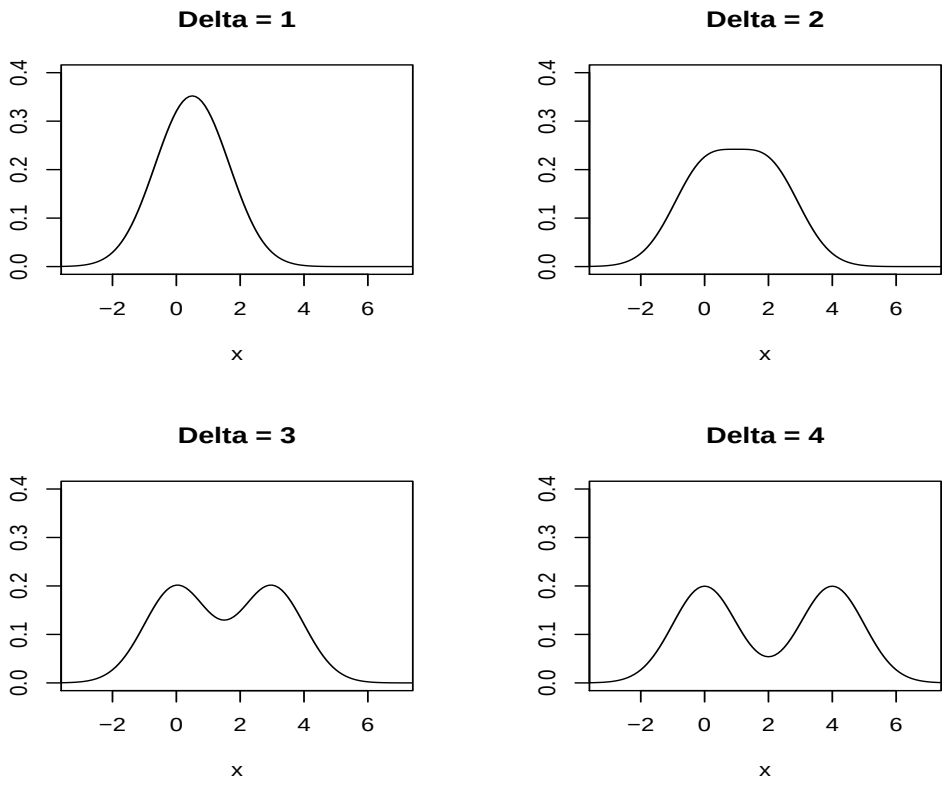


Figura 1.4: *Rappresentazioni grafiche della funzione di densità della mistura (1.6) per* $\alpha = 0.5$.

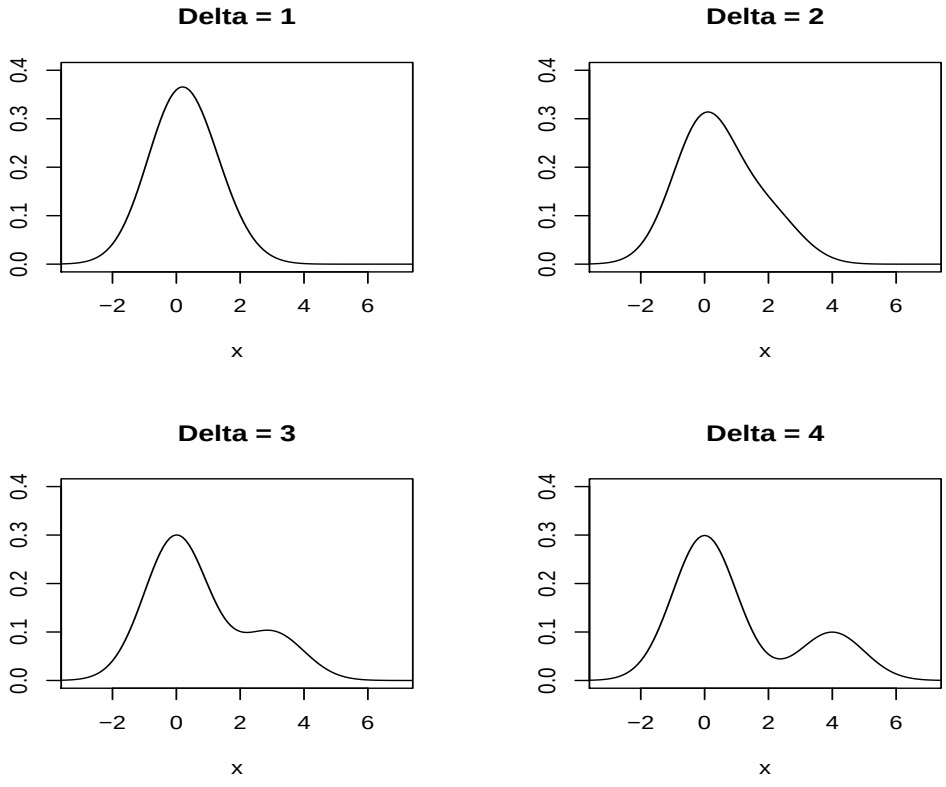


Figura 1.5: *Rappresentazioni grafiche della funzione di densità della mistura (1.6) per* $\alpha = 0.75$.

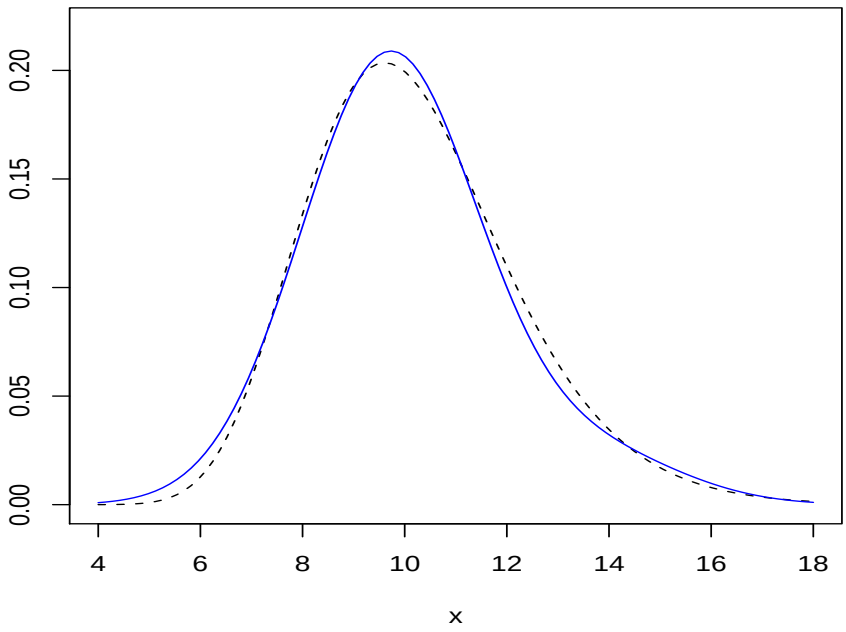


Figura 1.6: *Rappresentazioni grafiche delle fdp della distribuzione lognormale di parametri* $\mu = 10$ *e* $\sigma^2 = 0.04$ *(linea tratteggiata) e della mistura gaussiana a due componenti (1.8) (linea continua).*

seconda componente $N(\Delta, 1)$ per $\Delta = 1, 2, 3, 4$.

```
source("plotmixt.r")
source("mixt2.r")
plotmixt(0.5)
plotmixt(0.75)
```

Altri esempi sono dati in Figura 1.7 che riporta i grafici delle misture descritte in Tabella 1.1.

Un'importante applicazione indiretta concerne la modellizzazione di distribuzioni asimmetriche. Consideriamo la distribuzione lognormale avente densità

$$\frac{1}{\sqrt{2\pi}x\sigma} \exp\left(-\frac{1}{2}(\log x - \mu)^2/\sigma^2\right) . \tag{1.7}$$

In Figura 1.6 vengono confrontati i grafici delle funzioni di densità della distribuzione lognormale di parametri $\mu = \log(10)$ e $\sigma^2 = 0.04$ con quello di una mistura gaussiana a due componenti

$$p(x) = 0.9\,\phi(x; 9.7, 3) + 0.1\,\phi(x; 13.7, 3) . \tag{1.8}$$

La routine corrispondente è `plotlnorm.r`. Benchè le funzioni di densità siano molto prossime fra di loro, risulta chiaro il diverso significato dei due modelli (1.7) e (1.8): il primo assume che la popolazione sia omogenea, il secondo che sia eterogenea.

1.3 Generazione di campioni estratti con distribuzione mistura

Ci poniamo il problema di generare un campione casuale in base ad una distribuzione mistura. Lo schema in linea di principio è semplice, in accordo alla (1.1). Per ogni $n = 1, \ldots, N$, dapprima si

considera una realizzazione della v.a. categoriale S a valori in $\{1, \ldots, g\}$ con distribuzione di probabilità $(\alpha_1, \ldots, \alpha_g)$ e successivamente si considera una realizzazione della v.a. $\mathbf{X}_n$ con distribuzione $f(\cdot; \boldsymbol{\theta}_j)$.

Nota Computazionale 1.4 La funzione `genmixt.r` genera un campione di dimensione n da una mistura gaussiana a due componenti, vedi Algoritmo 6.9. Richiede in ingresso il peso α ed i parametri delle componenti (μ_1, σ_1), (μ_2, σ_2). Ad esempio per generare un campione di ampiezza $n = 50$ da una mistura di densità

$$p(x) = 0.3\,\phi(x; 1, 0.8^2) + 0.7\,\phi(x; 3, 2^2)$$

si esegue:

```
source("genmixt.r")
n<-50
alpha<-0.3
mu<-c(1,3)
sd<-c(0.8,2)
x<-genmixt(n,alpha,mu,sd)
x
```

E' anche interessante confrontare la distribuzione da cui sono stati generati i dati con la stima della funzione di densità. A tal scopo, la procedura precedente viene completata con alcune linee di programma. La nuova procedura è chiamata `genmixt2.r`:

```
source("genmixt2.r")
genmixt2(50,alpha,mu,sd)
genmixt2(100,alpha,mu,sd)
genmixt2(150,alpha,mu,sd)
genmixt2(200,alpha,mu,sd)
genmixt2(300,alpha,mu,sd)
```

♣

1.4 Identificabilità delle distribuzioni mistura

Un'importante proprietà di una famiglia parametrica di distribuzioni $\mathcal{F} = \{f(\cdot;\theta)\}_{\theta\in\Theta}$ indicizzata da un parametro $\theta \in \Theta$ $\mathcal{F} = \{f(\cdot;\theta)\}_{\theta\in\Theta}$ su un certo spazio campionario $\mathcal{X}$, è costituita dalla identificabilità. Diremo che la famiglia $\mathcal{F}$ è *identificabile* se e solo se, considerati due qualunque parametri $\theta_1, \theta_2 \in \Theta$, con $\theta_1 \neq \theta_2$, allora le due distribuzioni indicizzate da θ_1 e θ_2 sono diverse; in particolare risulta $f(\mathbf{x}; \theta_1) \neq f(\mathbf{x}; \theta_2)$ tranne al più per un insieme di punti $\mathbf{x} \in \mathcal{X}$ di misura nulla.

Ad esempio, siano X_1 e X_2 due v.a. indipendenti aventi distribuzione normale rispettivamente $N(\beta_1 + \nu, 1)$ e $N(\beta_2 + \nu, 1)$ con $\beta_1, \beta_2, \nu \in \mathbb{R}$. Posto $\theta = (\beta_1, \beta_2, \nu)$ come parametro della distribuzione di X_1 e X_2, allora il modello è identificabile se il sistema algebrico

$$\begin{cases} \beta_1 + \nu = a_1 \\ \beta_2 + \nu = a_2 \end{cases}$$

ammette un'unica soluzione per ogni $a_1, a_2 \in \mathbb{R}$. In realtà il sistema ammette infinite soluzioni, ad esempio se $\theta_1 = (0, 0, 2)$ e $\theta_2 = (1, 1, 1)$ allora le densità di X_1 e X_2 risultano uguali, pur essendo $\theta_1 \neq \theta_2$.

In generale, i modelli mistura non sono identificabili; in particolare qui consideriamo due diversi tipi di non identificabilità, rimandando a Frühwirth-Schnatter (2006) per ulteriori approfondimenti.

Non identificabilità dovuta ad invarianza per permutazione degli indici. Una prima causa di non identificabilità è dovuta all'invarianza della funzione di densità $p(\cdot;\boldsymbol{\psi})$ (1.3) per la permutazione di due o più valori dei parametri. Per comprendere ciò, supponiamo per semplicità che $p(\cdot;\boldsymbol{\psi})$ abbia due componenti, che indichiamo con $f(\cdot;\boldsymbol{\theta}_j)$ and $f(\cdot;\boldsymbol{\theta}_k)$ ($j,k=1,2$) che assumiamo appartenere alla stessa famiglia parametrica. La definizione di identificabilità implica che $p(\cdot;\boldsymbol{\psi}) = p(\cdot;\boldsymbol{\psi}^*)$ resta valida anche se le componenti con etichette j e k vengono scambiate. Pertanto, benchè la classe di misture può essere identificabile, il parametro $\boldsymbol{\psi}$ non lo è.

Consideriamo ad esempio una mistura di due distribuzioni normali

$$p(x;\boldsymbol{\psi}) = \alpha_1\phi(x;\mu_1,\sigma_1^2) + \alpha_2\phi(x;\mu_2,\sigma_2^2)$$

dove $\boldsymbol{\theta}_j = (\mu_j,\sigma_j^2) \in \boldsymbol{\Theta} = \mathbb{R}\times\mathbb{R}^+$ con $j=1,2$. Le due misture di parametri $(\alpha_1,\mu_1,\sigma_1^2) = (0.3,4,9),(\alpha_2,\mu_2,\sigma_2^2) = (0.7,2,5)$ e $(\alpha_1,\mu_1,\sigma_1^2) = (0.7,2,5),(\alpha_2,\mu_2,\sigma_2^2) = (0.3,4,9)$ – ottenute scambiando l'ordine delle componenti – definiscono ovviamente la stessa funzione di densità.

Ne segue che una mistura non è identificabile in senso stretto. Dalla (1.3) si nota che, se tutte le g densità componenti appartengono alla stessa famiglia, allora $f(\cdot;\boldsymbol{\psi})$ è invariante rispetto alle $g!$ permutazioni degli indici delle componenti di $\boldsymbol{\psi}$. In generale, si adotta la seguente definizione di identificabilità per distribuzioni mistura. Siano

$$f(\mathbf{x};\boldsymbol{\psi}) = \sum_{j=1}^{g}\alpha_j f_j(\mathbf{x};\boldsymbol{\theta}_j) \qquad \text{e} \qquad f(\mathbf{x};\boldsymbol{\psi}^*) = \sum_{j=1}^{g^*}\alpha_j f_j(\mathbf{x};\boldsymbol{\theta}_j^*)$$

due qualunque membri di una famiglia parametrica di densità mistura. Una mistura finita si dice *identificabile* rispetto a $\boldsymbol{\psi}\in\boldsymbol{\Psi}$ se risulta

$$f(\mathbf{x};\boldsymbol{\psi}) \equiv f(\mathbf{x};\boldsymbol{\psi}^*)$$

se e solo se $g=g^*$ e se possiamo permutare gli indici delle componenti in modo tale che si abbia

$$\alpha_j = \alpha_j^* \qquad \text{e} \qquad f_j(\mathbf{x};\boldsymbol{\theta}_j) = f_j(\mathbf{x};\boldsymbol{\theta}_j^*) \qquad j=1,\ldots,g.$$

In questo caso il simbolo $\equiv$ implica l'uguaglianza delle densità per quasi tutti i valori di $\mathbf{x}\in\mathbb{R}^q$ rispetto alla misura introdotta nello spazio $\mathbb{R}^q$ per $f(\mathbf{x};\boldsymbol{\psi})$. L'assenza di identificabilità di $\boldsymbol{\psi}$ dovuta a permutazioni degli indici delle componenti, da un punto di vista pratico viene usualmente affrontata imponendo appropriati vincoli su $\boldsymbol{\psi}$. Ad esempio, dopo aver stimato i parametri della mistura, si possono ordinare le componenti $\boldsymbol{\theta}_1\ldots,\boldsymbol{\theta}_g$ secondo l'ordine dei pesi

$$\alpha_1 \le \alpha_2 \le \cdots \le \alpha_g\,.$$

Non identificabilità dovuta a *overfitting*. Un'ulteriore causa di non identificabilità è dovuta ad un possibile sovradattamento (*overfitting*). Consideriamo una mistura finita di g distribuzioni, con $\boldsymbol{\psi} = (\alpha_1,\ldots,\alpha_g,\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_g)\in\boldsymbol{\Psi}_g$. Consideriamo poi una mistura di distribuzioni appartenenti alla stessa famiglia parametrica, ma contenente $g-1$ componenti anzichè g. Si può dimostrare che una qualunque mistura con $g-1$ componenti definisce un sottoinsieme non identificabile nello spazio $\boldsymbol{\Psi}_g$, che corrisponde a misture con g componenti dove o una componente è nulla o due componenti sono uguali.

Consideriamo ad esempio una mistura di due distribuzioni normali con parametro $\boldsymbol{\psi} = (\alpha_1, \alpha_2, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2) \in \boldsymbol{\Psi}_2$. Una qualunque mistura di due normali può essere scritta come una mistura di tre distribuzioni normali aggiungendo una componente di peso $\alpha_3 = 0$:

$$\begin{aligned} p(x; \boldsymbol{\psi}) &= \alpha_1 \phi(x; \mu_1, \sigma_1^2) + \alpha_2 \phi(x; \mu_2, \sigma_2^2) \\ &= \alpha_1 \phi(x; \mu_1, \sigma_1^2) + \alpha_2 \phi(x; \mu_2, \sigma_2^2) + 0 \cdot \phi(x; \mu_3, \sigma_3^2) \, . \end{aligned}$$

Nello spazio dei parametri $\boldsymbol{\Psi}_3$, il parametro $\boldsymbol{\psi} = (\alpha_1, \alpha_2, 0, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2, \mu_3, \sigma_3^2) \in \boldsymbol{\Psi}_3$ corrisponde ad una mistura non identificabile in quanto la densità $p(x; \boldsymbol{\psi})$ è la stessa per qualunque valore di (μ_3, σ_3^2). Lo stesso caso di non identificabilità per misture con tre componenti si ottiene suddividendo una componente di una mistura di due distribuzioni normali

$$\begin{aligned} p(x; \boldsymbol{\psi}) &= \alpha_1 \phi(x; \mu_1, \sigma_1^2) + \alpha_2 \phi(x; \mu_2, \sigma_2^2) \\ &= \alpha_1 \phi(x; \mu_1, \sigma_1^2) + (\alpha_2 - \alpha_3) \phi(x; \mu_2, \sigma_2^2) + \alpha_3 \phi(x; \mu_2, \sigma_2^2) \, . \end{aligned}$$

In questo caso l'identificabilità può essere ottenuta, in modo formale, imponendo vincoli sullo spazio dei parametri in modo tale da assicurare l'identificabilità. In poche parole, l'identificabilità è ottenuta imponendo che la (1.3) sia una mistura di g componenti distinte, con pesi $\alpha_j > 0$ strettamente positivi.

1.5 Struttura delle misture con dati incompleti

Nel paragrafo 1.4 sono stati introdotti i vettori $\mathbf{Z}_n$ di variabili indicatrici al fine di definire, per ciascun $\mathbf{X}_n$, la componente nel modello mistura (1.3) da cui $\mathbf{X}_n$ si considera estratto. Per motivi che risulteranno via via più chiari quando affronteremo il problema della stima dei parametri di una mistura, da un punto di vista concettuale è opportuno associare un vettore di variabili indicatrici $\mathbf{Z}_n$ al vettore delle caratteristiche $\mathbf{X}_n$.

Sia $\mathbf{X}_1, \ldots, \mathbf{X}_n$ un campione di vettori aleatori i.i.d. con distribuzione (1.2), cioè

$$\mathbf{X}_1, \ldots, \mathbf{X}_n \overset{i.i.d}{\sim} p(\cdot; \boldsymbol{\psi}) \tag{1.9}$$

e sia $\mathbf{x}_1, \ldots, \mathbf{x}_N$ una realizzazione di $\mathbf{X}_1, \ldots, \mathbf{X}_N$. In questo contesto i dati possono essere considerati come incompleti poichè i corrispondenti vettori $\mathbf{z}_1, \ldots, \mathbf{z}_N$ di variabili indicatrici non sono disponibili. I dati completi sono pertanto

$$\mathbf{y}_n = (\mathbf{x}_n, \mathbf{z}_n) \qquad n = 1, \ldots, N \, . \tag{1.10}$$

I vettori $\mathbf{z}_1, \ldots, \mathbf{z}_N$ sono le realizzazioni dei vettori aleatori $\mathbf{Z}_1, \ldots, \mathbf{Z}_n$ che si assume siano distribuiti (non condizionatamente secondo la legge multinomiale) come

$$\mathbf{Z}_1, \ldots, \mathbf{Z}_n \overset{i.i.d}{\sim} M_g(1, \boldsymbol{\alpha}) \, . \tag{1.11}$$

La j-esima proporzione della mistura α_j può essere considerata come la probabilità a priori che il generico oggetto $\omega \in \Omega$ provenga dalla j-esima componente ($j = 1, \ldots, g$) mentre la probabilità a posteriori che lo stesso oggetto ω appartenga alla j-esima componente, dato il vettore di osservazioni $\mathbf{x}_n$, è espressa da:

$$\tau_j(\mathbf{x}_n) = P(\omega \in \Omega_j | \mathbf{x}_n) = P(Z_{nj} = 1 | \mathbf{x}_n) = \frac{\alpha_j f_j(\mathbf{x}_n)}{p(\mathbf{x}_n)} \qquad j = 1, \ldots, g; n = 1, \ldots, N \tag{1.12}$$

I modelli mistura possono essere inquadrati nel contesto di dati incompleti poichè i vettori di variabili indicatrici sono considerati come dati mancanti e dobbiamo stimare i parametri della distribuzione $p(\cdot; \boldsymbol{\psi})$ dai dati disponibili solo dalla distribuzione marginale di $\mathbf{X}_n$ piuttosto che dalla

distribuzione congiunta di $\mathbf{X}_n$ e $\mathbf{Z}_n$. Rileviamo che, se i dati fossero assegnati in forma completa, allora la stima dei parametri della distribuzione sarebbe abbastanza semplice in quanto i parametri $\boldsymbol{\theta}_j$ di ciascuna componente potrebbero essere stimati in base alle osservazioni nella classe Ω_j ($j = 1, \ldots, g$). Successivamente le proporzioni α_j del miscuglio potrebbero essere stimate in base ai dati correttamente classificati:

$$\hat{\alpha}_j = \frac{1}{N} \sum_{n=1}^{N} z_{jn} \qquad (j = 1, \ldots, g) \,. \tag{1.13}$$

1.6 Misture gaussiane

Le misture di distribuzioni normali costituiscono un esempio di grande rilevanza teorica ed applicativa. Nel caso di misture di distribuzioni normali multivariate si ha:

$$f(\mathbf{x}; \boldsymbol{\theta}_j) = \phi(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \qquad j = 1, \ldots, g$$

dove

$$\phi(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \frac{1}{|2\pi\boldsymbol{\Sigma}_j|^{1/2}} \exp\left\{(\mathbf{x} - \boldsymbol{\mu}_j)' \boldsymbol{\Sigma}_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j)\right\}$$

denota la densità della distribuzione normale multivariata con vettore delle medie $\boldsymbol{\mu}_j$ e matrice di covarianza $\boldsymbol{\Sigma}_j$ ($j = 1, \ldots, g$). In questo caso il vettore dei parametri $\boldsymbol{\psi}$ è:

$$\boldsymbol{\psi} = (\alpha_1, \ldots, \alpha_{g-1}, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_g) \,.$$

Nel caso di componenti normali omoscedastiche, le matrici di covarianza sono uguali

$$\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma} \qquad j = 1, \ldots, g \,.$$

Le misture gaussiane sono modelli molto flessibili. In Figura 1.7 vengono mostrati esempi di densità gaussiane univariate, ottenute da combinazioni delle componenti indicate in Tabella 1.1.

Nota Computazionale 1.5 I grafici delle misture gaussiane in Tabella 1.1 vengono generati utilizzando le routine degli Algoritmi 6.4-6.7.

```
source("plotmixt1.r")
source("plotmixt2.r")
source("plotmixt3.r")
source("plotmixt4.r")
source("mixt2.r")
source("mixt3.r")
plotmixt1()
plotmixt2()
plotmixt3()
plotmixt4()
```

1.7 Misture di distribuzioni t

In molte applicazioni si ha interesse a costruire modelli aventi code più pesanti di quelle della distribuzione normale. In questo contesto si possono considerare misture di distribuzioni appartenenti alla famiglia t-Student:

$$\mathbf{X} \sim \alpha_1 t(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \nu_1) + \cdots + \alpha_g t(\boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g, \nu_g) \tag{1.14}$$

Tabella 1.1: Funzioni di densità di quindici esempi di misture gaussiane.

	Densità	$f(x;\boldsymbol{\psi})$
1.	Gaussian	$N(0,1)$
2.	Skewed Unimodal	$\frac{1}{5}N(0,1)+\frac{1}{5}N(\frac{1}{2},(\frac{2}{3})^2)+\frac{3}{5}N(\frac{13}{15},(\frac{5}{9})^2)$
3.	Strongly Skewed	$\sum_{i=0}^{7} N(3\{(\frac{2}{3})^i-i\},(\frac{2}{3})^{2i})$
4.	Kurtotic Unimodal	$\frac{2}{3}N(0,1)+\frac{1}{3}N(0,(\frac{1}{10})^2)$
5.	Outlier	$\frac{1}{10}N(0,1)+\frac{9}{10}N(0,(\frac{1}{10})^2)$
6.	Bimodal	$\frac{1}{2}N(-1,(\frac{2}{3})^2)+\frac{1}{2}N(1,(\frac{2}{3})^2)$
7.	Separated Bimodal	$\frac{3}{4}N(-\frac{3}{2},(\frac{1}{2})^2)+\frac{1}{2}N(\frac{3}{2},(\frac{1}{2})^2)$
8.	Skewed Bimodal	$\frac{3}{4}N(0,1)+\frac{1}{4}N(\frac{3}{2},(\frac{1}{3})^2)$
9.	Trimodal	$\frac{9}{20}N(-\frac{6}{5},(\frac{3}{5})^2)+\frac{9}{20}N(\frac{6}{5},(\frac{3}{5})^2)+\frac{1}{10}N(0,(\frac{1}{4})^2)$
10.	Claw	$\frac{1}{2}N(0,1)+\sum_{i=0}^{4}\frac{1}{10}N(i/2-1,(\frac{1}{10})^2)$
11.	Double Claw	$\frac{49}{100}N(-1,(\frac{2}{3})^2)+\frac{49}{100}N(1,(\frac{2}{3})^2)+\sum_{i=0}^{6}\frac{1}{350}N((i-3)/2,(\frac{1}{100})^2)$
12.	Asymmetric Claw	$\frac{1}{2}N(0,1)+\sum_{i=-2}^{2}(2^{1-i}/31)N(i+\frac{1}{2},(2^{-i}/10)^2)$
13.	Asymmetric Double Claw	$\sum_{i=0}^{1}\frac{46}{100}N(2i-1,(\frac{2}{3})^2)+\sum_{i=1}^{3}\frac{1}{300}N(-i/2,(\frac{1}{100})^2)$ + $\sum_{i=0}^{1}\frac{7}{300}N(i/2,(\frac{7}{100})^2)$
14.	Smooth Comb	$\sum_{i=0}^{5}(2^{5-i}/63)N((65-96(\frac{1}{2})^i)/21,(\frac{32}{63})^2/2^{2i})$
15.	Discrete Comb	$\sum_{i=0}^{2}\frac{2}{7}N((12i-15)/7,(\frac{2}{7})^2)+\sum_{i=8}^{10}\frac{1}{21}N(2i/7,(\frac{1}{21})^2)$

dove $\mathbf{X}$ è un vettore aleatorio q-dimensionale e $t(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j,\nu_j)$ $(j=1,\ldots,g)$ è una distribuzione t multivariata avente ν_j gradi di libertà e di parametri $(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)$.

Le distribuzioni t multivariate stanno diventando sempre più popolari in statistica in quanto, per la caratteristica delle loro code, costituiscono un'alternativa più realistica ai modelli gaussiani. In particolare la distribuzione t costituisce un approccio più robusto rispetto all'adattamento mediante distribuzioni normali in presenza di valori anomali o di rumore.

Riassumiamo qui di seguito alcune note sulla distribuzione t multivariata.

1.7.1 Distribuzioni t multivariate

Un vettore aleatorio q-dimensionale $\mathbf{X}$ segue una distribuzione t-Student q-variata con ν gradi di libertà, parametro di locazione $\boldsymbol{\mu}$ e matrice prodotto interno definita positiva $\boldsymbol{\Sigma}$ se la sua funzione di densità congiunta è data da

$$t(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma},\nu)=\frac{\Gamma(\frac{\nu+q}{2})|\boldsymbol{\Sigma}|^{-1/2}}{(\pi\nu)^{p/2}\Gamma(\frac{\nu}{2})\{1+\delta(\mathbf{x},\boldsymbol{\mu};\boldsymbol{\Sigma})/\nu\}^{(\nu+q)/2}} \tag{1.15}$$

dove

$$\delta(\mathbf{x},\boldsymbol{\mu};\boldsymbol{\Sigma})=(\mathbf{x}-\boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) \tag{1.16}$$

denota la distanza di Mahalanobis fra $\mathbf{x}$ e $\boldsymbol{\mu}$ rispetto a $\boldsymbol{\Sigma}$. In questo caso scriveremo $\mathbf{X} \sim t_\nu(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

I gradi di libertà ν vengono anche chiamati *parametro di forma* in quanto la forma della (1.15) varia al variare di ν. In particolare se $(\nu + q)/2$ è un intero, allora la (1.15) è la distribuzione q-variata di Pearson di tipo VII. Per $\nu \to \infty$ la (1.15) tende alla distribuzione normale q-variata con vettore delle medie $\boldsymbol{\mu}$ e matrice di covarianza $\boldsymbol{\Sigma}$; Pertanto, la (1.15) può essere considerata come un'approssimazione della distribuzione normale multivariata per grandi valori di ν.

In Figura 1.8 viene rappresentata la densità della distribuzione t univariata per diversi gradi di libertà, confrontandola con quella della normale standard. Una situazione analoga si verifica nel caso multivariato. In Figura 1.9 vengono rappresentate le ellissi (o curve di livello) della funzione di densità 1.15, cioè il grafico della curva di equazione $t(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu) = c$, con $\boldsymbol{\mu} = \mathbf{0}$ e $\boldsymbol{\Sigma} = ((1, 0.8)', (0.8, 1)')$ per due differenti valori di c (rispettivamente $c = 0.01$ e $c = 0.1$) e per due differenti valori dei gradi di libertà: $\nu = 3$ and $\nu = 20$; nella stessa figura si confrontano queste curve con le corrispondenti curve della normale multivariata. Si noti che per $c = 0.01$ la curva della distribuzione normale è interna alla curva della distribuzione t; mentre la situazione opposta si osserva per $c = 0.1$.

Diamo alcune proprietà della distribuzione t multivariata. Sia $\mathbf{X}|U = u \sim N_q(\boldsymbol{\mu}, \boldsymbol{\Sigma}/u)$, con U v.a. $U \sim \chi^2_v/\nu$ dove $\nu \in \mathbb{R}^+$. Allora si ha:

1. $\mathbf{X} \sim t_q(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu)$.
2. $\mathbb{E}(\mathbf{X}) = \boldsymbol{\mu} \quad (\nu > 1)$
3. $\text{Cov}(\mathbf{X}) = \nu\boldsymbol{\Sigma}/(\nu - 2) \quad (\nu > 2)$.
4. $U|\mathbf{x} \sim \chi^2_{\nu+q}/(\nu + \delta(\mathbf{x}, \boldsymbol{\mu}; \boldsymbol{\Sigma}))$.

Per approfondimenti sulle proprietà matematiche e sui metodi statistici inerenti la distribuzione t-multivariata, si vedano Kotz & Nadarajah (2004), Nadarajah & Kotz (2005).

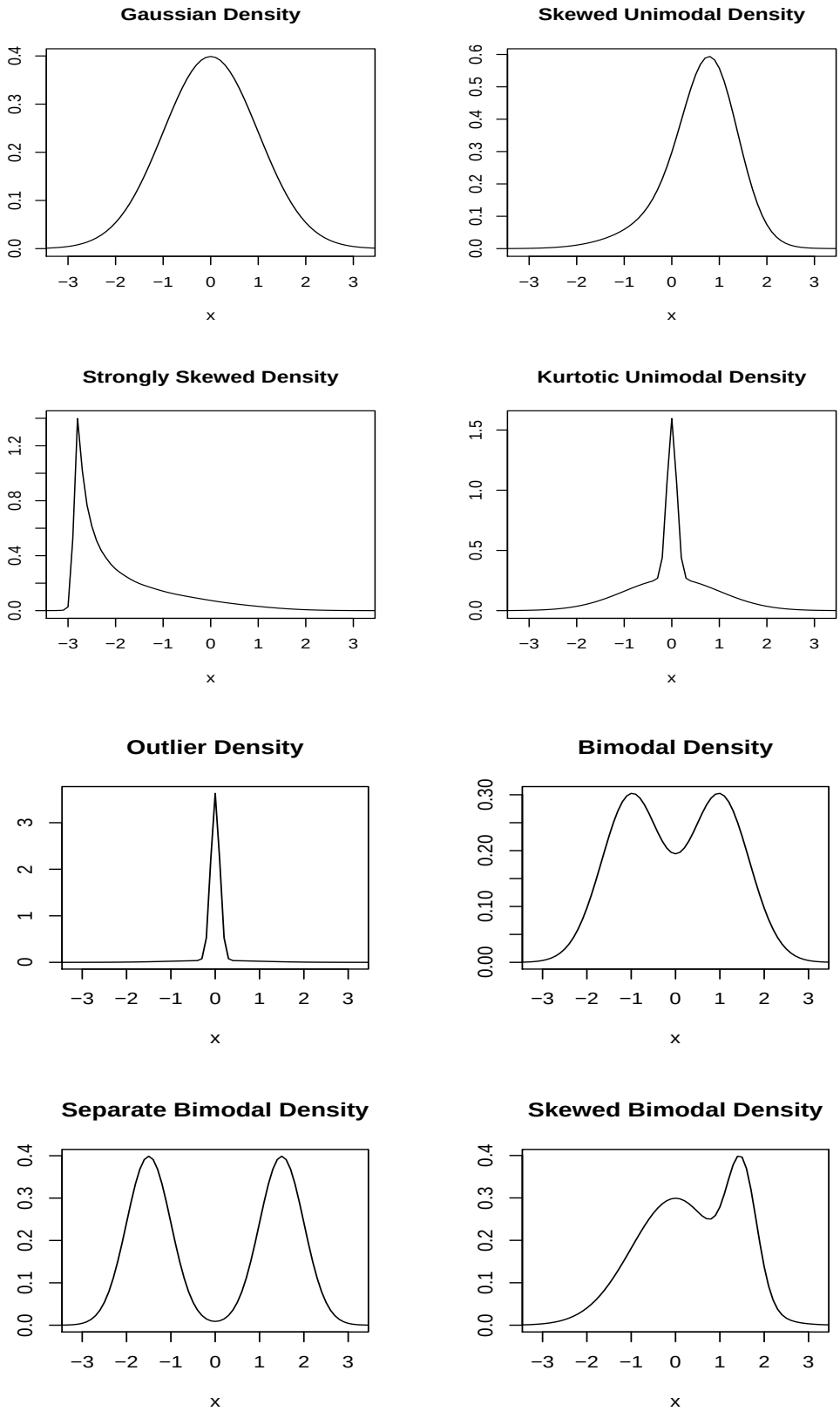


Figura 1.7: Rappresentazioni grafiche di misture gaussiane.

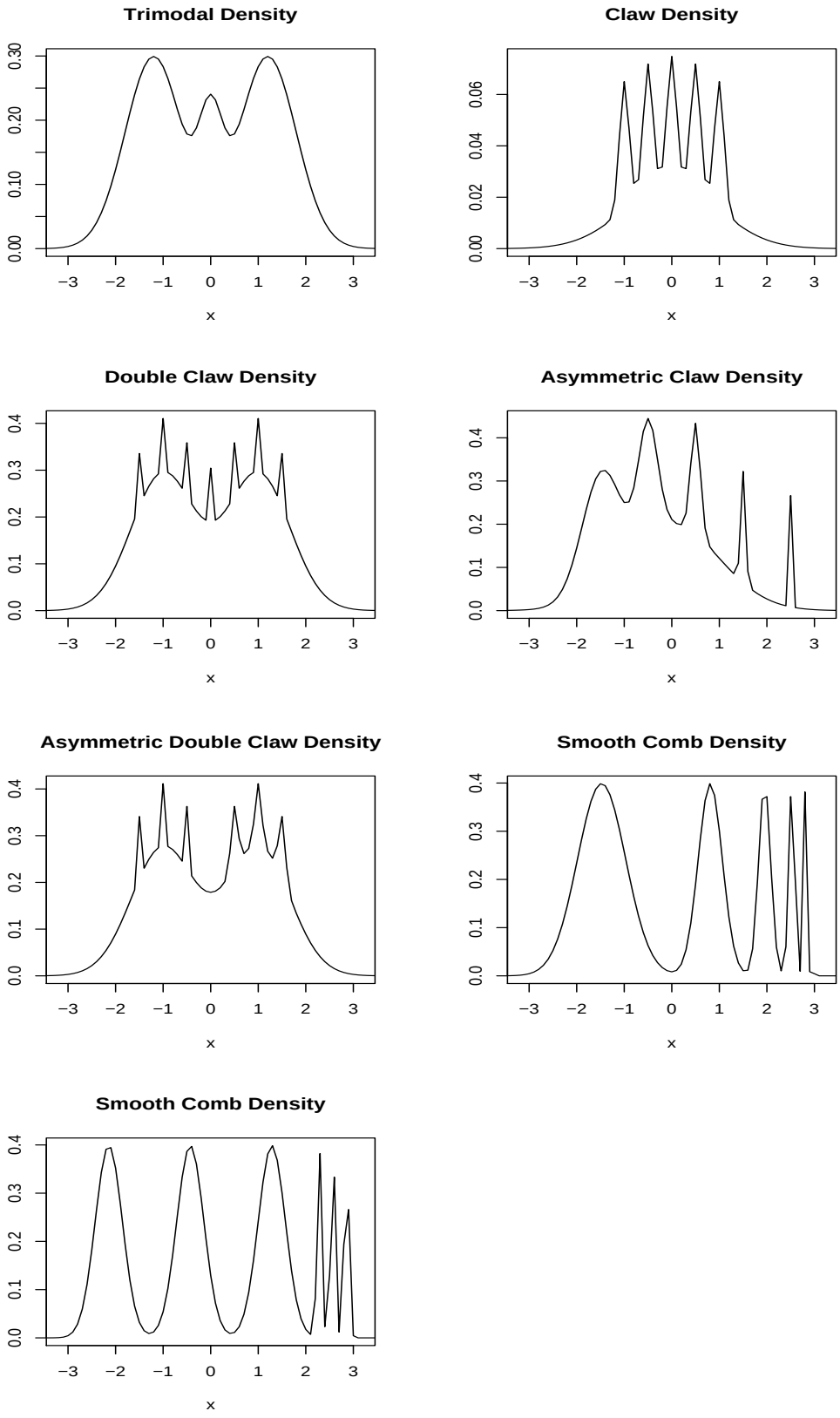


Figura 1.7: Rappresentazioni grafiche di misture gaussiane (segue).

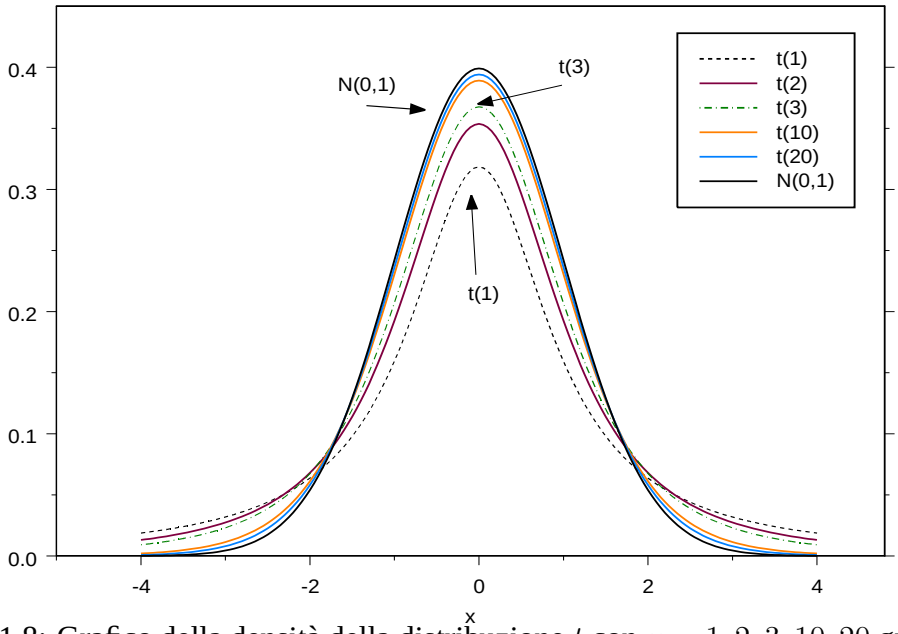


Figura 1.8: Grafico della densità della distribuzione t con $\nu = 1, 2, 3, 10, 20$ gradi di libertà.

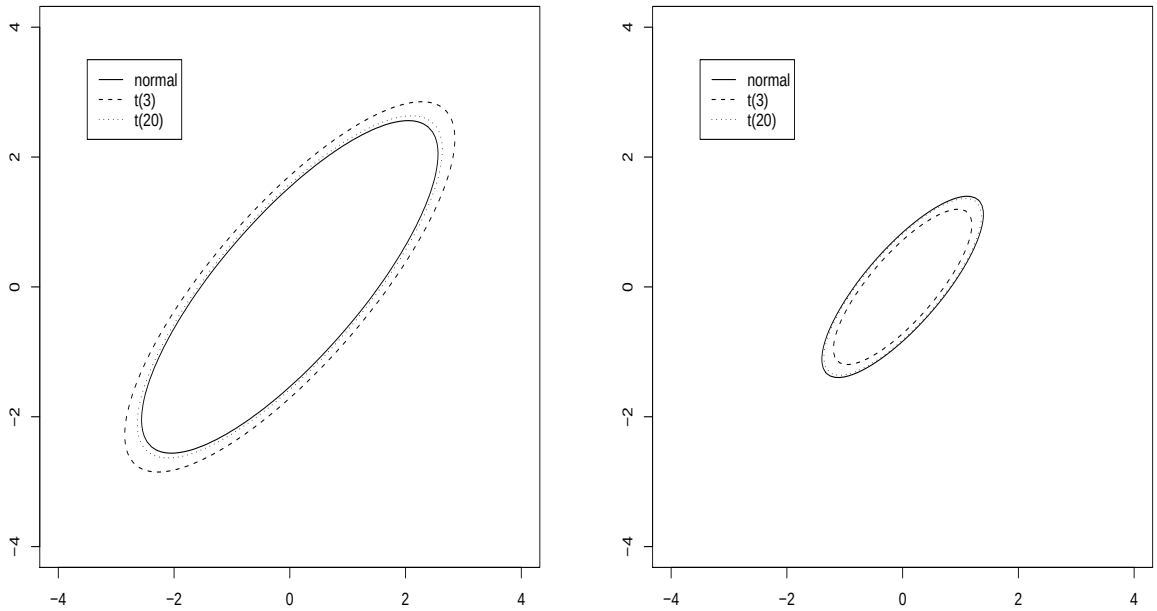


Figura 1.9: *Curve di livello della densità t bivariata con $\nu = 3, 20$ gradi di libertà e confronto con le coorspondenti curve della distribuzione normale bivariata (per gli stessi valori di $\boldsymbol{\mu}$ e $\boldsymbol{\Sigma}$). Grafico della curva $t(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu) = c$ per $c = 0.01$ (sinistra) e $c = 0.1$ (destra).*

Capitolo 2

La stima dei parametri

2.1 Stima di massima verosimiglianza

Definizione 2.1 Sia $\underset{\sim}{\mathbf{X}} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ la realizzazione di un campione $\mathbf{X}_1, \ldots, \mathbf{X}_N$ di vettori aleatori i.i.d aventi fdp $f(\mathbf{x}; \boldsymbol{\psi})$ dove $\boldsymbol{\psi} \in \boldsymbol{\Psi}$. La funzione

$$L(\boldsymbol{\psi}; \underset{\sim}{\mathbf{X}}) = f(\mathbf{x}_1; \boldsymbol{\psi}) \cdot f(\mathbf{x}_2; \boldsymbol{\psi}) \cdots f(\mathbf{x}_N; \boldsymbol{\psi}) = \prod_{n=1}^{N} f(\mathbf{x}_n; \boldsymbol{\psi}) \ , \tag{2.1}$$

considerata come funzione di $\boldsymbol{\psi}$, viene chiamata *funzione di verosimiglianza*.

Il metodo della massima verosimiglianza consiste nel trovare il valore $\hat{\boldsymbol{\psi}}(\underset{\sim}{\mathbf{X}})$ del parametro $\boldsymbol{\psi} \in \boldsymbol{\Psi}$ che più verosimilmente ha generato i dati.

Definizione 2.2 Il *principio di massima verosimiglianza* consiste nello scegliere quale stima di $\boldsymbol{\psi}$ un valore $\hat{\boldsymbol{\psi}}(\underset{\sim}{\mathbf{X}})$ che massimizza $L(\boldsymbol{\psi}; \underset{\sim}{\mathbf{X}})$, cioè trovare una funzione $\hat{\boldsymbol{\psi}} : \mathbb{R}^N \to \mathbb{R}^k$ (con $\boldsymbol{\psi} \in \boldsymbol{\Psi} \subseteq \mathbb{R}^k$ per un certo $k \in \mathbb{N}$) che soddisfa la condizione

$$L(\hat{\boldsymbol{\psi}}; \underset{\sim}{\mathbf{X}}) = \max_{\boldsymbol{\psi} \in \boldsymbol{\Psi}} L(\boldsymbol{\psi}; \underset{\sim}{\mathbf{X}}) \qquad \text{cioè} \qquad \hat{\boldsymbol{\psi}} = \arg_{\boldsymbol{\psi} \in \Psi} \max L(\boldsymbol{\psi}; \underset{\sim}{\mathbf{X}}) \ . \tag{2.2}$$

E' ben noto che, per motivi algebrici e numerici, è conveniente lavorare con il logaritmo della funzione di verosimiglianza, che viene chiamata *funzione di log-verosimiglianza*. Poichè il logaritmo è una funzione strettamente crescente (usualmente si considera il logaritmo naturale), allora $L(\boldsymbol{\psi})$ e $\log L(\boldsymbol{\psi})$ assumono il massimo (o i punti di massimo) in corrispondsenza degli stessi valori di $\boldsymbol{\psi}$. Per le proprietà del logaritmo, si ha:

$$\mathcal{L}(\boldsymbol{\psi}; \underset{\sim}{\mathbf{X}}) = \log L(\boldsymbol{\psi}; \underset{\sim}{\mathbf{X}}) = \log \left(\prod_{n=1}^{N} f(\mathbf{x}_n; \boldsymbol{\psi}) \right) = \sum_{n=1}^{N} \log f(\mathbf{x}_n; \boldsymbol{\psi}) \ . \tag{2.3}$$

Nel seguito, per semplicità, a volte in $L(\boldsymbol{\psi}; \underset{\sim}{\mathbf{X}})$ ometteremo il termine $\underset{\sim}{\mathbf{X}}$ e quindi scriveremo $L(\boldsymbol{\psi})$. Molte funzioni di verosimiglianza soddisfano opportune ipotesi di regolarità. In generale, sia $\boldsymbol{\Psi}$ un aperto di $\mathbb{R}^k$, e supponiamo che $f(\mathbf{x}; \boldsymbol{\psi})$ sia una funzione a valori positivi e differenziabile di $\boldsymbol{\psi} = (\psi_1, \psi_2, \ldots, \psi_k)$ (cioè, esistano le derivate parziali prime rispetto alle componenti di $\boldsymbol{\psi}$). Se esiste un punto di massimo $\widehat{\boldsymbol{\psi}}$, allora deve soddisfare le equazioni di verosimiglianza

$$\frac{\partial}{\partial \psi_i} \mathcal{L}(\boldsymbol{\psi}) \bigg|_{\boldsymbol{\psi} = \widehat{\boldsymbol{\psi}}} = 0 \qquad i = 1, 2, \ldots, k \tag{2.4}$$

cioè

$$\frac{\partial}{\partial\psi_i}\mathcal{L}(\boldsymbol{\psi}) = \frac{\partial}{\partial\psi_i}\log\left(\prod_{n=1}^{N} f(\mathbf{x}_n;\boldsymbol{\psi})\right) = \frac{\partial}{\partial\psi_i}\sum_{n=1}^{N}\log f(\mathbf{x}_n;\boldsymbol{\psi})$$
$$= \sum_{n=1}^{N}\frac{\partial}{\partial\psi_i}\log f(\mathbf{x}_n;\boldsymbol{\psi}) = 0\ .$$

Sinteticamente, il sistema di equazioni (2.4) può essere scritto

$$\frac{\partial\mathcal{L}(\boldsymbol{\psi};\underset{\sim}{\mathbf{X}})}{\partial\boldsymbol{\psi}} = \mathbf{0}\ . \tag{2.5}$$

2.2 Matrice di informazione

La matrice di informazione attesa di Fisher, per un certo valore $\boldsymbol{\psi} \in \boldsymbol{\Psi}$ del vettore dei parametri è definita come segue

$$\mathcal{I}(\boldsymbol{\psi}) = \mathbb{E}_{\boldsymbol{\psi}}\{\mathbf{S}(\underset{\sim}{\mathbf{X}};\boldsymbol{\psi})\mathbf{S}'(\underset{\sim}{\mathbf{X}};\boldsymbol{\psi})\} \tag{2.6}$$

dove

$$\mathbf{S}(\underset{\sim}{\mathbf{X}};\boldsymbol{\psi}) = \frac{\partial\mathcal{L}(\boldsymbol{\psi})}{\partial\boldsymbol{\psi}}$$

è il vettore gradiente della funzione di logverosimiglianza e $\underset{\sim}{\mathbf{X}} = (\mathbf{x}_1', \ldots, \mathbf{x}_N')'$ contiene i dati osservati. Sotto opportune condizioni, si può dimostrare che

$$\mathcal{I}(\boldsymbol{\psi}) = \mathbb{E}_{\boldsymbol{\psi}}\{\mathbf{I}(\boldsymbol{\psi};\underset{\sim}{\mathbf{X}})\} \tag{2.7}$$

dove

$$\mathbf{I}(\boldsymbol{\psi};\underset{\sim}{\mathbf{X}}) = -\frac{\partial^2\mathcal{L}(\boldsymbol{\psi})}{\partial\boldsymbol{\psi}\partial\boldsymbol{\psi}'} \tag{2.8}$$

è l'opposto della matrice Hessiana della funzione di verosimiglianza.

2.3 Stime di massima verosimiglianza per modelli mistura

Assegnato un campione $\mathbf{x}_1, \ldots, \mathbf{x}_N$ di N osservazioni indipendenti dal modello mistura (1.4), la corrispondente funzione di verosimiglianza si scrive

$$L(\boldsymbol{\psi};\underset{\sim}{\mathbf{X}}) = \prod_{i=1}^{n} p(\underset{\sim}{\mathbf{X}};\boldsymbol{\psi}) = \prod_{n=1}^{N}\left[\sum_{j=1}^{g}\alpha_j f_i(\mathbf{x}_n;\boldsymbol{\theta}_j)\right]\ . \tag{2.9}$$

Se consideriamo i dati categorizzati (1.5):

$$\underset{\sim}{\mathbf{Y}} = \{\mathbf{y}_n\ :\ n = 1, \ldots, N\} = \{(\mathbf{x}_n, \mathbf{z}_n)\ :\ n = 1, \ldots, N\}$$

allora, la verosimiglianza corrispondente a $\underset{\sim}{\mathbf{Y}} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ può essere scritta nella forma

$$L_c(\boldsymbol{\psi};\underset{\sim}{\mathbf{X}}) = \prod_{n=1}^{N}\prod_{j=1}^{g}\alpha_j^{z_{nj}} f_j(\mathbf{x}_n;\boldsymbol{\theta}_j)^{z_{nj}}\ , \tag{2.10}$$

dove $z_{nj} = 1$ se $\mathbf{x}_n$ proviene dalla j-esima popolazione e $z_{nj} = 0$ altrimenti. Consideriamo il logaritmo

$$\mathcal{L}_c(\boldsymbol{\psi}; \underset{\sim}{\mathbf{X}}) = \sum_{n=1}^{N} \mathbf{z}_n' \mathbf{V}(\boldsymbol{\alpha}) + \sum_{n=1}^{N} \mathbf{z}_n' \mathbf{U}_n(\boldsymbol{\theta}) \tag{2.11}$$

dove $\mathbf{V}(\boldsymbol{\alpha})$ è un vettore g-dimensionale (dipendente dal vettore dei pesi $\boldsymbol{\alpha}$), la cui j-esima componente è uguale a $\log \alpha_j$, e $\mathbf{U}_j(\boldsymbol{\theta})$ è un vettore g-dimensionale (dipendente dal vettore $\boldsymbol{\theta}$ dei parametri delle componenti) la cui j-esima componente è $\log f_j(\mathbf{x}_n; \boldsymbol{\theta}_j)$. La forma (2.9) della verosimiglianza mistura corrisponde alla densità marginale di $\mathbf{x}_1, \ldots, \mathbf{x}_N$ ottenuta sommando la (2.10) rispetto a $\mathbf{z}_1, \ldots, \mathbf{z}_N$. Ciò mette in risalto l'interpretazione della mistura come problema con dati incompleti in cui i vettori indicatori costituiscono i dati mancanti. In questa formulazione, la stima di massima verosimiglianza dei parametri della mistura può essere ottenuta mediante l'algoritmo di EM.

2.3.1 Singolarità della funzione di verosimiglianza di misture gaussiane

Si può dimostrare che la funzione di logverosimiglianza di $\boldsymbol{\psi}$, in corrispondenza di un assegnato campione $\mathbf{x} = (x_1, \ldots, x_N)$ di dimensione N estratto da una mistura di gaussiane, è non limitata. Consideriamo il caso più semplice costituito da una mistura di due distribuzioni normali univariate:

$$f(x; \mu, \sigma^2) = \frac{1}{2\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) + \frac{1}{2\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$$

dove sono assegnati solo i parametri (μ, σ^2) di una delle due densità, con $\alpha_1 = \alpha_2 = 1/2$.

Dimostriamo che assegnato un qualunque campione $x_1, \ldots, x_N$ e comunque scelto un valore $M \in \mathbb{R}$, esiste $\sigma = \sigma_0$ tale che per $\mu = x_1$ la logverosimiglianza risulta maggiore di M:

$$\begin{aligned}
\mathcal{L}(\mu = x_1, \sigma_0^2; \mathbf{x}) &= \sum_{n=1}^{N} \log p(x_n; \mu = x_1, \sigma^2) \\
&> \log\left(\frac{1}{2\sqrt{2\pi\sigma_0^2}}\right) + \sum_{n=2}^{N} \log\left(\frac{1}{2\sqrt{2\pi}}\right) \exp\left(-\frac{x_n^2}{2}\right) \\
&= -\log\sigma_0 - \sum_{n=2}^{N} \frac{x_i^2}{2} - N\log(2\sqrt{2\pi}) > M\,.
\end{aligned}$$

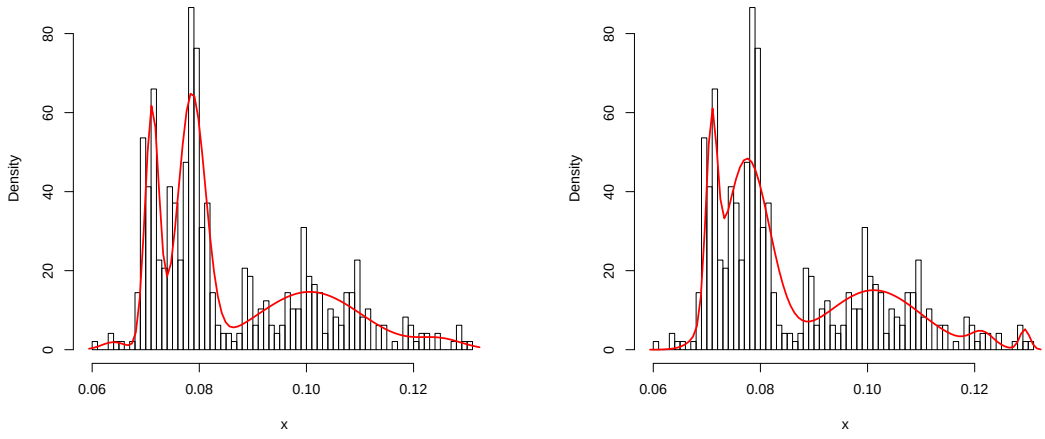
2.3.2 Punti di massimo locale per la funzione di verosimiglianza di misture gaussiane

Oltre a punti singolari, la funzione di verosimiglianza di misture gaussiane può presentare punti di massimo locale, vedi Figura 2.1. Redner & Walker (1984) dimostrano il seguente teorema (caso univariato).

Teorema 2.3 Si assuma che le derivate della funzione di densità $f(x; \boldsymbol{\psi})$ esistano almeno fino al terzo ordine e che soddisfino opportune condizioni di regolarità. Si assuma inoltre che la matrice di informazione di Fisher $\mathcal{I}(\boldsymbol{\psi}^0)$ valutata in corrispondenza del vero valore del parametro $\boldsymbol{\psi}^0 \in \boldsymbol{\Psi}$ sia ben definita e sia definita positiva. Allora esiste un'unica soluzione $\hat{\boldsymbol{\psi}}_N$ del sistema (2.5) in un certo intorno di $\boldsymbol{\psi}^0$ per una dimensione campionaria N sufficientemente grande. Tale stima $\hat{\boldsymbol{\psi}}_N$ costituisce un massimo locale della funzione di verosimiglianza e si ha:

$$\sqrt{N}(\widehat{\boldsymbol{\psi}}_N' - \boldsymbol{\psi}^0) \to \mathcal{N}(0, [\mathcal{I}(\boldsymbol{\psi}^0)]^{-1})$$

in distribuzione per $N \to \infty$.



$\mathcal{L}(\hat{\boldsymbol{\psi}}) = 1525$ $\mathcal{L}(\hat{\boldsymbol{\psi}}) = 1510$

$\mathcal{L}(\hat{\boldsymbol{\psi}}) = 1497.247$ $\mathcal{L}(\hat{\boldsymbol{\psi}}) = 1497.235$

Figura 2.1: Modelli mistura gaussiana ($g = 5$) per file dati *stamp.dat* ottenuti in corrispondenza di punti diversi di massimo locale della funzione di logverosimiglianza.

Teorema 2.4 Sotto condizioni leggermente più restrittive di quelle del Teorema precedente, $\widehat{\boldsymbol{\psi}}_N$ è l'unico stimatore di massima verosimiglianza fortemente consistente.

Pertanto, nel seguito, quale stimatore di massima verosimiglianza si considera un tale punto $\hat{\boldsymbol{\psi}}_N \in \boldsymbol{\Psi}$. Nel caso univariato

$$p(x;\boldsymbol{\psi}) = \alpha_1\phi(x;\mu,\sigma_1^2) + \cdots + \alpha_g\phi(x;\mu_g,\sigma_g^2) \qquad x \in \mathbb{R} \tag{2.12}$$

dove $\phi(x;\mu_j,\sigma_j^2)$ è la funzione di densità della distribuzione normale di parametri μ_j, σ_j^2 $(j = 1,\ldots,g)$, si può dimostrare che se il campione $\mathbf{x} = (x_1,\ldots,x_N)$ estratto con distribuzione (2.12) contiene almeno $k+1$ punti distinti, allora per $c \in (0,1]$, esiste un punto di massimo assoluto fortemente consistente in un sottoinsieme $\boldsymbol{\Psi}_c$ di $\boldsymbol{\Psi}$ che soddisfa il vincolo

$$\min_{i\neq j}\left(\frac{\sigma_j^2}{\sigma_i^2}\right) \geq c \;>\; 0\,.$$

Nel caso multivariato, considerata la funzione di densità di una mistura di g normali multivariate

$$p(\mathbf{x};\boldsymbol{\psi}) \;\;=\;\; \alpha_1\phi(\mathbf{x};\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1) + \cdots + \alpha_g\phi(\mathbf{x};\boldsymbol{\mu}_g,\boldsymbol{\Sigma}_g) \qquad \mathbf{x} \in \mathbb{R}^q \tag{2.13}$$

dove $\phi(\mathbf{x};\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)$ è la funzione di densità di una normale multivariata di parametri $\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j^2$ $(j = 1,\ldots,g)$, si può dimostrare che, per opportuni valori positivi $a, b > 0$, se gli autovalori delle matrici di covarianza $\boldsymbol{\Sigma}_j$ soddisfano il vincolo:

$$a \leq \lambda_i(\boldsymbol{\Sigma}_j) \leq b \qquad i = 1,\ldots,p\,, j = 1,\ldots,g\,. \tag{2.14}$$

allora la funzione di logverosimiglianza, massimizzata nel sottospazio $\boldsymbol{\Psi}_{a,b}$ of $\boldsymbol{\Psi}$:

$$\begin{aligned}\boldsymbol{\Psi}_{a,b} = \{&(\alpha_1,\ldots,\alpha_g,\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_g,\boldsymbol{\Sigma}_1,\ldots,\boldsymbol{\Sigma}_g) \in \mathbb{R}^{g[1+p+(p^2+p)/2]} \;: \\ &\alpha_1+\cdots+\alpha_g = 1,\; \alpha_j \geq 0,\; a \leq \lambda_i(\boldsymbol{\Sigma}_j) \leq b, \quad \text{for } j = 1,\ldots,g\}\,,\end{aligned}$$

conduce ad un problema di ottimizzazione ben posto avente un'unica soluzione globale vincolata ed un numero inferiore di massimi locali spuri (cioè massimi locali della funzione di verosimiglianza diversi dalla stima di massima verosimiglianza, vedi Ingrassia (2004)).

2.4 Classificazione mediante modelli mistura

Il problema della classificazione è uno dei problemi più studiati in statistica. In termini generali si tratta di determinare – sulla base di alcune caratteristiche quantitative e/o qualitative osservato su una popolazione statistica $\Omega = \{\omega_1,\omega_2,\ldots,\omega_N\}$ – un certo numero di gruppi da cui le osservazioni provengono, cioè $\Omega = \Omega_1 \cup \cdots \cup \Omega_g$, aventi la proprietà di essere il più possibile omogenei al loro interno ed eterogenei fra di loro. Esempi di applicazione di questo tipo di problematica sono: diagnosi medica automatica, riconoscimento di reperti archeologici, riconoscimento di caratteri, classificazione di forme, riconoscimento di immagini digitali, etc.

Aspetti generali e regole di classificazione. Sia Ω una popolazione ed assumiamo che essa sia costituita da g sottopopolazioni o gruppi $\Omega = \Omega_1,\ldots,\Omega_g$, $g \geq 2$. Il problema che ci poniamo è quello di costruire un criterio per assegnare un oggetto $\omega \in \Omega$ ad uno dei g gruppi Ω_i sulla base delle informazioni contenute in un dato vettore di caratteristiche $\mathbf{x} = \mathbf{x}(\omega)$ osservate relativamente ad ω. Per esempio, le sottopopolazioni potrebbero essere costituite da gruppi di malati affetti da differenti

patologie e $\mathbf{x}$ in questo caso si riferisce ai valori ottenuti in un certo insieme di analisi cliniche. In questo caso il problema è quello di diagnosticare la malattia del paziente – fra le g malattie possibili – solo in base ai risultati delle analisi cliniche effettuate.

Naturalmente è desiderabile fare il minor numero di errori possibile, nel senso che verrà precisato più avanti. Considerata una partizione dello spazio $\mathcal{X}$, cioè un insieme di g regioni mutuamente esclusive $\mathcal{X}_1, \ldots, \mathcal{X}_g$, con $\cup_{j=1}^{g} \mathcal{X}_j = \mathcal{X}$, una *regola di decisione* $d(\mathbf{x})$ è una funzione $d : \mathbb{R}^p \to \{1, 2, \ldots, g\}$.

Siano assegnate le *probabilità a priori* $\alpha_1, \ldots, \alpha_g$ delle g classi Ω_j:

$$\alpha_j = P(\omega \in \Omega_j) \qquad j = 1, \ldots, g$$

con $\alpha_j > 0$ e $\sum_{j=1}^{g} \alpha_j = 1$. Per semplicità nel seguito scriveremo anche $P(\Omega_j)$ anzichè $P(\omega \in \Omega_j)$. Tali probabilità possono ad esempio essere valutate in base al rapporto fra il numero n_j di elementi campionati dalla classe Ω_j rispetto al numero totale di elementi $N = n_1 + \cdots + n_g$.

In mancanza di ulteriori informazioni, la classificazione viene effettuata in base a tali quantità e la migliore regola di classificazione sarà quella di assegnare l'oggetto ω alla classe Ω_j per cui risulta massima la probabilità a priori α_k, cioè:

$$\text{si assegna } \omega \text{ in } \Omega_k \iff \alpha_k \geq \alpha_j\,, \qquad j = 1, \ldots, g,\ j \neq k\,. \tag{2.15}$$

Nel caso in cui $P(\Omega_j) = P(\Omega_k)$ per qualche j, allora l'oggetto ω può essere assegnato indifferentemente alla classe Ω_j o Ω_k. La regola (2.15) minimizza la probabilità di errata classificazione.

Usualmente abbiamo però l'informazione che proviene dal modello $\mathbf{x}$. In base a quanto detto prima, sono noti i valori di $\mathbf{x} \in \mathcal{X}_j$ associati agli oggetti $\omega \in \Omega_j$, anche se uno stesso modello $\mathbf{x}$ può essere associato, in generale, ad oggetti provenienti da classi diverse. Per tale ragione, dobbiamo considerare la distribuzione condizionale $P(\Omega_j|\mathbf{x})$. Supponiamo che il vettore $\mathbf{x}$ abbia una funzione di densità (o di probabilità) $f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^q$. Siano inoltre note le funzioni di densità (o di probabilità) $f_1(\mathbf{x}), \ldots, f_g(\mathbf{x})$ del vettore di variabili aleatorie associate ad $\Omega_1, \ldots, \Omega_g$, cioè $f_i(\mathbf{x})$ è la funzione di densità di $\mathbf{x}$ quando $\omega \in \Omega_j$, per $j = 1, \ldots, g$. Allora per il teorema di Bayes si ha:

$$P(\Omega_j \,|\, \mathbf{x}) = \frac{\alpha_j\, f_j(\mathbf{x})}{p(\mathbf{x})} \tag{2.16}$$

dove $f_j(\mathbf{x})$ è la verosimiglianza e $p(\mathbf{x})$ è funzione di densità di $\mathbf{x}$ che è data da:

$$p(\mathbf{x}) = \sum_{j=1}^{g} f_j(\mathbf{x})\, \alpha_j\,. \tag{2.17}$$

In questo caso si considera la regola di classificazione nota come *regola bayesiana di minimo errore* in base alla quale si assegna l'oggetto ω alla classe Ω_k per cui risulta:

$$\alpha_k\, f_k(\mathbf{x}) \geq \alpha_j\, f_j(\mathbf{x}) \quad j = 1, \ldots, g; j \neq k\,. \tag{2.18}$$

Nel caso in cui la distribuzione a priori $\alpha_1, \ldots, \alpha_g$ è uniforme, cioè $\alpha_j = 1/g$ per ogni $j = 1, \ldots, g$, allora la (2.18) viene chiamata *regola discriminante di massima verosimiglianza*.

L'importanza del teorema di Bayes risiede nel fatto che esso consente di esprimere le probabilità a posteriori in termini di quantità che sono spesso più semplici da calcolare. Si può dimostrare che la regola bayesiana di minimo errore rende minima la probabilità di errata classificazione, vedi McLachlan (1992).

La regola (2.18) introduce un nuovo concetto. Consideriamo un punto $\mathbf{x} \in \mathbb{R}^q$ tale che $\alpha_k\, f_k(\mathbf{x}) = \alpha_j\, f_j(\mathbf{x})$ per $j \neq k$. L'oggetto ω, in base alla sua caratteristica $\mathbf{x}$, potrà essere assegnato indifferentemente alle due classi Ω_j o Ω_k. Ne segue che l'insieme dei punti $\{\mathbf{x} \in \mathbb{R}^q \;:\; \alpha_k\, f_k(\mathbf{x}) = \alpha_j\, f_j(\mathbf{x})\}$ costituisce la *superficie o frontiera di decisione* fra Ω_j e Ω_k. In tal modo, lo spazio delle caratteristiche viene suddiviso in g regioni $\mathcal{R}_1, \ldots, \mathcal{R}_g$ tale che se $\mathbf{x}$ ricade in $\mathcal{R}_j$, allora il corrispondente oggetto ω viene assegnato alla classe Ω_j.

Si noti che tali regioni non corrispondono con gli insiemi $\mathcal{X}_1, \ldots, \mathcal{X}_g$ introdotti in precedenza: $\mathcal{X}_j$ è costituito dall'insieme dei punti $\mathbf{x} \in \mathbb{R}^q$ provenienti da oggetti ω appartenenti alla classe Ω_j; l'insieme $\mathcal{R}_j$ è l'insieme dei punti di $\mathbb{R}^q$ tali che se $\mathbf{x} \in \mathcal{R}_j$ allora ω si assegna a Ω_j.

Classificazione mediante modelli mistura Nel caso di modelli mistura (1.3), la probabilità a posteriori $\tau_j(\mathbf{x}; \boldsymbol{\psi})$ che l'oggetto ω appartenga alla classe Ω_j è data da

$$\tau_j(\mathbf{x}; \widehat{\boldsymbol{\psi}}) = \frac{\alpha_j f_j(\mathbf{x}; \widehat{\boldsymbol{\theta}}_j)}{p(\mathbf{x}; \widehat{\boldsymbol{\psi}})} \quad j = 1, \ldots, g\ . \tag{2.19}$$

in corrispondenza di una stima $\widehat{\boldsymbol{\psi}}$ del parametro $\boldsymbol{\psi}$. In base alla regola di decisione (2.18), l'oggetto ω viene assegnato alla classe per cui è massima la probabilità a posteriori (2.19).

2.5 L'algoritmo EM

L'algoritmo EM è una procedura iterativa di ampio utilizzo per il calcolo della stima di massima verosimiglianza in condizioni di informazione parziale (dati incompleti).

E' stato proposto da Dempster, Laird and Rubin in un famoso articolo del 1977. Una trattazione aggiornata ed esauriente dell'argomento è costituita dalla monografia di McLachlan & Krishnan (2008). In questo contesto, il vettore dei dati osservati $\mathbf{x}$ è visto come incompleto ed è considerato come una funzione osservabile rispetto ai cosidetti *dati completi*. La nozione di *dati incompleti* include il ben noto caso dei dati mancanti, ma si applica anche a situazioni in cui i dati completi rappresentano ciò che si vorrebbe disponibile in base a qualche ipotetico esperimento. In quest'ultimo caso, i dati completi possono contenere alcune variabili che non sono mai osservabili nella realtà. Indichiamo pertanto con $\mathbf{y}$ il vettore contenente i dati completi e sia $\mathbf{z}$ il vettore contenente i dati addizionali, che si riferisce ai dati non osservabili o mancanti. Evidenziamo qui che il calcolo della stima di massima verosimiglianza è spesso notevolmente facilitato riformulando come problemi con dati incompleti anche problemi che, a prima vista, non appaiono come tali.

L'idea sottostante l'algoritmo EM è abbastanza intuitiva; inoltre algoritmi simili all'EM sono stati proposti ed applicati in vari lavori prima dell'articolo di Dempster *et al.* (1977), ma è solo in questo articolo che è stata proposta una formulazione generale dell'algoritmo, investigandone le proprietà fondamentali, vedi Meng & van Dyk (1997).

Formulazione dell'algoritmo. Sia $\mathbf{X}$ un vettore aleatorio avente funzione di densità $p(\mathbf{x}; \boldsymbol{\psi})$, dove $\boldsymbol{\psi} \in \boldsymbol{\Psi}$ è un vettore di parametri; sia $\mathbf{x}$ una realizzazione di $\mathbf{X}$ (i dati osservati).

Denotiamo con $f_c(\mathbf{y}; \boldsymbol{\psi})$ la funzione di densità di un vettore aleatorio $\mathbf{Y}$ corrispondente al vettore con dati completi di $\mathbf{X}$, ad esempio $\mathbf{Y} = (\mathbf{X}, \mathbf{Z})$. Allora la corrispondente funzione di logverosimiglianza può essere espressa come

$$\log L_c(\boldsymbol{\psi}) = \log f_c(\mathbf{x}; \boldsymbol{\psi})\ .$$

Da un punto di vista formale, abbiamo due spazi campionari $\mathcal{Y}$ e $\mathcal{X}$ e più funzioni suriettive da $\mathcal{Y}$ a $\mathcal{X}$. Invece di osservare il vettore dei dati completi $\mathbf{y}$ in $\mathcal{Y}$, si osserva il vettore dei dati incompleti $\mathbf{x} = \mathbf{x}(\mathbf{y})$ in $\mathcal{X}$. Ne segue che

$$f(\mathbf{x}; \boldsymbol{\psi}) = \int_{\mathcal{Y}(\mathbf{x})} f_c(\mathbf{y}; \boldsymbol{\psi})\, d\mathbf{y}\,,$$

dove $\mathcal{Y}(\mathbf{x})$ è il sottoinsieme di $\mathcal{Y}$ determinato dall'equazione $\mathbf{x} = \mathbf{x}(\mathbf{y})$.

L'algoritmo EM consente di risolvere, in maniera indiretta, il sistema di equazioni di verosimiglianza con dati incompleti (2.4) mediante una procedura iterativa che opera sulla funzione di logverosimiglianza con dati completi $\log L_c(\boldsymbol{\psi})$. Poichè questa è non osservabile, essa viene sostituita dalla sua aspettativa condizionata dato $\mathbf{x}$, utilizzando la stima $\boldsymbol{\psi}^{(k)}$ di $\boldsymbol{\psi}$ ottenuta al passo k.

Più esattamente, sia $\boldsymbol{\psi}^{(0)}$ una stima iniziale di $\boldsymbol{\psi}$. Allora, nella prima iterazione, nel passo E si calcola la quantità

$$Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(0)}) = \mathbb{E}_{\boldsymbol{\psi}^{(0)}}\{\log L_c(\boldsymbol{\psi}|\underset{\sim}{\mathbf{X}})\}\,,$$

dove $\mathbb{E}_{\boldsymbol{\psi}^{(0)}}\{\cdot\}$ indica la speranza matematica rispetto al parametro $\boldsymbol{\psi}^{(0)}$.

Nel passo M, si effettua la massimizzazione di $Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(0)})$ rispetto a $\boldsymbol{\psi}$ nello spazio dei parametri $\boldsymbol{\Psi}$. Cioè, si sceglie la stima $\boldsymbol{\psi}^{(1)}$ tale che

$$Q(\boldsymbol{\psi}^{(1)}; \boldsymbol{\psi}^{(0)}) \geq Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(0)})$$

per ogni $\boldsymbol{\psi} \in \boldsymbol{\Psi}$.

Si eseguono nuovamente i passi E ed M, utilizzando la stima $\boldsymbol{\psi}^{(1)}$ al posto di $\boldsymbol{\psi}^{(0)}$. In generale, nell'iterazione $(k+1)$, i passi E ed M vengono eseguiti come segue:

E-Step. Si calcola $Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(k)})$, dove

$$Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(k)}) = \mathbb{E}_{\boldsymbol{\psi}^{(k)}}\{\log L_c(\boldsymbol{\psi}|\underset{\sim}{\mathbf{X}})\}\,. \tag{2.20}$$

M-Step. Si sceglie il valore $\boldsymbol{\psi}^{(k+1)} \in \boldsymbol{\Psi}$ che massimizza $Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(k)})$; cioè,

$$Q(\boldsymbol{\psi}^{(k+1)}; \boldsymbol{\psi}^{(k)}) \geq Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(k)}) \tag{2.21}$$

per ogni $\boldsymbol{\psi} \in \boldsymbol{\Psi}$. In maniera equivalente, la (2.21) può essere espressa come segue:

$$\boldsymbol{\psi}^{(k+1)} = \arg\max_{\boldsymbol{\psi} \in \boldsymbol{\Psi}} Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(k)})\,. \tag{2.22}$$

I passi E ed M vengono eseguiti alternativamente finchè la differenza

$$L(\boldsymbol{\psi}^{(k+1)}) - L(\boldsymbol{\psi}^{(k)})$$

risulta inferiore ad un certo valore di tolleranza fissato a priori.

Nel paragrafo seguente dimostriamo la monotonia dell'algoritmo EM, cioè la proprietà che la successione di stime $\{\boldsymbol{\psi}^{(k)}\}_k$ genera una successione non decrescente dei valori di logverosimiglianza $\{\mathcal{L}(\boldsymbol{\psi}^{(k)})\}_k$:

$$\mathcal{L}(\boldsymbol{\psi}^{(k+1)}) \geq L(\boldsymbol{\psi}^{(k)}) \qquad k = 0, 1, 2, \ldots, \tag{2.23}$$

Si noti che non è necessario specificare la funzione esatta da $\mathcal{Y}$ in $\mathcal{X}$ né la corrispondente rappresentazione densità dei dati incompleti f in termini della densità dei dati completi p_c. Quello che si richiede è la specificazione del vettore dei dati completi $\mathbf{y}$ e la densità condizionata di $\mathbf{Y}$ dato il vettore dei dati osservati $\underset{\sim}{\mathbf{X}}$. La specificazione della densità condizionata è necessaria per eseguire il passo E.

2.6 Monotonia dell'algoritmo EM

Dimostriamo la monotonia della successione dei valori della funzione di logverosimiglianza $\{\mathcal{L}(\boldsymbol{\psi}^{(k)})\}_k$, cioè la disuguaglianza (2.23). Sia

$$k(\mathbf{y}|\mathbf{x};\boldsymbol{\psi}) = \frac{p_c(\mathbf{y};\boldsymbol{\psi})}{p(\mathbf{x};\boldsymbol{\psi})}$$

la densità condizionata di $\mathbf{Y}$ dato $\mathbf{X} = \mathbf{x}$. Allora la logverosimiglianza è data da

$$\begin{aligned}\log L(\boldsymbol{\psi}) &= \log p(\mathbf{x};\boldsymbol{\psi}) = \log p_c(\mathbf{y};\boldsymbol{\psi}) - \log k(\mathbf{y}|\mathbf{x};\boldsymbol{\psi}) \\ &= \log L_c(\boldsymbol{\psi}) - \log k(\mathbf{y}|\mathbf{x};\boldsymbol{\psi})\,. \end{aligned} \tag{2.24}$$

Calcolando i valori attesi di entrambi i termini della (2.24) rispetto alla distribuzione condizionata di $\mathbf{Y}$ dato $\mathbf{X} = \mathbf{x}$, utilizzando la stima corrente $\boldsymbol{\psi}^{(k)}$ di $\boldsymbol{\psi}$, si ha

$$\begin{aligned}\log L(\boldsymbol{\psi}) &= \mathbb{E}_{\psi^{(k)}}\{\log L_c(\boldsymbol{\psi})|\mathbf{x}\} - \mathbb{E}_{\psi^{(k)}}\{\log k(\mathbf{Y}|\mathbf{x};\boldsymbol{\psi})|\mathbf{x}\} \\ &= Q(\boldsymbol{\psi};\boldsymbol{\psi}^{(k)}) - H(\boldsymbol{\psi};\boldsymbol{\psi}^{(k)}) \end{aligned} \tag{2.25}$$

dove

$$\begin{aligned}Q(\boldsymbol{\psi};\boldsymbol{\psi}^{(k)}) &= \mathbb{E}_{\psi^{(k)}}\{\log L_c(\boldsymbol{\psi})|\mathbf{x}\} \\ H(\boldsymbol{\psi};\boldsymbol{\psi}^{(k)}) &= \mathbb{E}_{\psi^{(k)}}\{\log k(\mathbf{Y}|\mathbf{x};\boldsymbol{\psi})|\mathbf{x}\}\,. \end{aligned}$$

Dalla (2.25), segue

$$\begin{aligned}&\log L(\boldsymbol{\psi}^{(k+1)}) - \log L(\boldsymbol{\psi}^{(k)}) = \\ &\qquad \{Q(\boldsymbol{\psi}^{(k+1)};\boldsymbol{\psi}^{(k)}) - Q(\boldsymbol{\psi}^{(k)};\boldsymbol{\psi}^{(k)})\} - \{H(\boldsymbol{\psi}^{(k+1)};\boldsymbol{\psi}^{(k)}) - H(\boldsymbol{\psi}^{(k)};\boldsymbol{\psi}^{(k)})\}\,. \end{aligned} \tag{2.26}$$

La prima differenza nel termine a destra della (2.26) è non negativa poichè $\boldsymbol{\psi}^{(k+1)}$ è scelto in maniera tale che valga la (2.21)

$$Q(\boldsymbol{\psi}^{(k+1)};\boldsymbol{\psi}^{(k)}) \geq Q(\boldsymbol{\psi};\boldsymbol{\psi}^{(k)}) \tag{2.27}$$

per ogni $\boldsymbol{\psi} \in \boldsymbol{\psi}$. Pertanto la (2.23) è verificata se la differenza nel secondo termine a destra della (2.26) risulta non positiva; cioè se

$$H(\boldsymbol{\psi}^{(k+1)};\boldsymbol{\psi}^{(k)}) - H(\boldsymbol{\psi}^{(k)};\boldsymbol{\psi}^{(k)}) \leq 0\,. \tag{2.28}$$

Osserviamo che, per ogni $\boldsymbol{\psi} \in \boldsymbol{\Psi}$, si ha

$$\begin{aligned}H(\boldsymbol{\psi};\boldsymbol{\psi}^{(k)}) - H(\boldsymbol{\psi}^{(k)};\boldsymbol{\psi}^{(k)}) &= \mathbb{E}_{\psi^{(k)}}\left[\log \frac{k(\mathbf{Y}|\mathbf{x};\boldsymbol{\psi})}{k(\mathbf{Y}|\mathbf{x};\boldsymbol{\psi}^{(k)})}\middle|\mathbf{x}\right] \\ &\leq \log \mathbb{E}_{\psi^{(k)}}\left[\frac{k(\mathbf{Y}|\mathbf{x};\boldsymbol{\psi})}{k(\mathbf{Y}|\mathbf{x};\boldsymbol{\psi}^{(k)})}\middle|\mathbf{x}\right] \\ &= \log \int_{\mathcal{Y}(\mathbf{x})} k(\mathbf{y}|\mathbf{x};\boldsymbol{\psi})\,d\mathbf{y} \\ &= 0\,, \end{aligned} \tag{2.29}$$

dove la disuguaglianza nella (2.29) segue da un'applicazione della disuguaglianza di Jensen[1] e dalla concavità della funzione logaritmo.

Ciò dimostra la (2.29), e quindi la disuguaglianza (2.23), mostrando che la verosimiglianza $L(\boldsymbol{\psi})$ non decresce dopo una qualunque iterazione dell'algoritmo di EM. La verosimiglianza cresce se la diseguaglianza (2.27) è verificata in senso stretto. Pertanto per una successione limitata di valori di verosimiglianza $\{L(\boldsymbol{\psi}^{(k)})\}_k$, $L(\boldsymbol{\psi}^{(k)})$ converge monotonicamente verso un certo valore L^*, e ciò conduce alla stima $\hat{\boldsymbol{\psi}}$ di massima verosimiglianza: $L^* = L(\hat{\boldsymbol{\psi}})$. □

2.7 Alcune proprietà dell'algoritmo EM

L'algoritmo di EM presenta alcune interessanti proprietà che lo rendono preferibile rispetto ad altri algoritmi iterativi di ottimizzazione quali, ad esempio, quello di Newton-Raphson. Alcuni dei vantaggi dell'algoritmo di EM, rispetto ad altri, sono riassunti qui di seguito:

1. L'algoritmo di EM è numericamente stabile rispetto a ciascuna iterazione in cui la verosibiglianza cresce (tranne in prossimità di punti di singolarità).
2. Sotto condizioni abbastanza generali, l'algoritmo di EM mostra buone capacità di convergere al massimo globale della funzione di verosimiglianza. Cioè, assegnata una stima iniziale $\boldsymbol{\psi}^{(0)}$ nello spazio dei parametri, si ottiene quasi sempre una convergenza verso un massimo locale, eccetto alcuni casi dovuti a scelte critiche della stima iniziale di $\boldsymbol{\psi}^{(0)}$ oppure a particolari caratteristiche (patologiche) della funzione di logverosimiglianza.
3. L'implementazione dell'algoritmo di EM è in genere molto semplice, essendo basata sull'utilizzo di dati completi: ad ogni iterazione, *i*) il passo E implica unicamente il calcolo di valori attesi rispetto alle distribuzioni condizionali basate sui dati completi e *ii*) il passo M richiede solamente il calcolo della stima di massima verosimiglianza, che spesso può essere ricavata in forma chiusa.
4. La scrittura di programmi inerenti l'algoritmo di EM è molto semplice, in quanto non si richiede alcun calcolo della funzione di verosimiglianza né delle sue derivate.
5. L'algoritmo di EM richiede l'utilizzo di spazi limitati di memoria. Ad esempio, nella generica iterazione non si richiede il calcolo della matrice di informazione né della sua inversa.
6. Il costo computazionale per iterazione è abbastanza limitato, e ciò può compensare il maggior numero di iterazioni che l'algoritmo di EM richiede per la convergenza rispetto ad altri algoritmi di ottimizzazione.
7. Controllando la monotonia degli incrementi della funzione di verosimiglianza da ciascuna iterazione alla successiva, è semplice monitorare la convergenza e verificare la presenza di (macroscopici) errori di programmazione.
8. L'algoritmo di EM può essere utilizzato anche per ottenere stime di dati mancanti.

Alcuni elementi critici dell'algoritmo di EM sono:

[1] **Funzione convessa**. Una funzione continua $g : \mathbb{R} \to \mathbb{R}$ si dice *convessa* se risulta

$$g\left(\alpha x + (1-\alpha)y\right) \leq \alpha g(x) + (1-\alpha)g(y)$$

per ogni $x, y \in \mathbb{R}$ con $0 < \alpha < 1$. Se la disuguaglianza precedente è verificata strettamente, diremo che g è *strettamente convessa*.

Disuguaglianza di Jensen. Sia X una v.a. avente speranza matematica $\mathbb{E}(X)$ finita e sia $g(\cdot)$ una funzione convessa. Allora si ha $\mathbb{E}[g(X)] \geq g[\mathbb{E}(X)]$.

1. La convergenza dell'algoritmo di EM può risultare lenta anche nel caso di problemi apparentemente semplici ed in casi in cui c'è molta informazione incompleta.
2. Non viene garantita la convergenza verso il massimo assoluto della funzione di verosimiglianza nel caso in cui siano presenti più punti distinti di massimo locale, essendo l'algoritmo di EM una procedura di tipo deterministico (come per tutti gli algoritmi di ottimizzazione, quale ad esempio il metodo di Newton); la stima finale dipende pertanto dalla scelta della stima iniziale.

2.8 Estensioni dell'algoritmo EM

Due importanti estesioni dell'algoritmo EM sono gli algoritmi ECM e ECME. L'algoritmo ECM, proposto da Meng and Rubin (1993), costituisce un'estensione dell'algoritmo EM per i casi in cui il processo di massimizzazione al passo M diviene relativamente più semplice se si condiziona rispetto ad una qualche funzione dei parametri da stimare. Nell'algoritmo ECM, pertanto, il passo M dell'algoritmo EM viene sostituito da un certo numero di passi di massimizzazione condizionata (CM) che risultano computazionalmente più semplici. Come conseguenza, usualmente richiede un maggior numero di iterazioni rispetto all'algoritmo EM ma, nel complesso, raggiunge la convergenza in tempi inferiori.

Un'altra estensione dell'algoritmo EM è il cosidetto algoritmo ECME (*Expectation-Conditional Maximization Either*), vedi Liu and Rubin (1994). Qui il termine *either* si riferisce al fatto che alcuni dei (o tutti i) passi CM nell'algoritmo ECM vengono sostituiti da passi in cui si massimizza o la speranza condizionata della verosimiglianza basata su dati completi oppure la funzione di logverosimiglianza (con dati incompleti). Liu and Rubin (1994) hanno rilevato che l'algoritmo ECME risulta quasi sempre più veloce sia dell'EM che dell'ECM in termini di numero di iterazioni e di tempo di calcolo.

Per maggiori dettagli, oltre agli articoli sopra citati, si può vedere il capitolo 5 di McLachlan & Krishnan (2008).

2.9 Stima dei parametri di una mistura di densità normali

Ricordando che in questo caso, $\mathbf{z}_1, \ldots, \mathbf{z}_n$ sono le quantità mancanti, dalla (2.11) il passo E, espresso dalla (2.20), può essere descritto come segue:

$$Q(\boldsymbol{\psi}; \boldsymbol{\psi}^{(k)}) = \mathbb{E}_{\psi^{(k)}}\{\mathcal{L}_c(\boldsymbol{\psi}|\mathbf{x})\} = \sum_{n=1}^{N} \boldsymbol{\tau}_n^{(k)\prime} \mathbf{V}(\boldsymbol{\alpha}) + \sum_{n=1}^{N} \boldsymbol{\tau}_n^{(k)\prime} \mathbf{U}_n(\boldsymbol{\theta})$$

avendo suddiviso il parametro $\boldsymbol{\psi}$ nel vettore dei pesi $\boldsymbol{\alpha}$ e nel vettore dei parametri delle componenti $\boldsymbol{\theta}$, dove

$$\boldsymbol{\tau}_n^{(k)} = \boldsymbol{\tau}_n(\boldsymbol{\psi}^{(k)}) = \mathbb{E}_{\psi^{(k)}}(\mathbf{Z}_n|\mathbf{x}_n; \boldsymbol{\psi}^{(k)}) \,,$$

cioè

$$\tau_{nj}^{(k)} = [\boldsymbol{\tau}_n^{(k)}]_j = \frac{\alpha_j^{(k)} f_j(\mathbf{x}_n|\boldsymbol{\theta}_j^{(k)})}{p(\mathbf{x}_n|\boldsymbol{\psi}^{(k)})} \quad n = 1, \ldots, N, \; j = 1, \ldots, g.$$

I pesi $\tau_{nj}^{(k)}$ sono le probabilità a posteriori della categoria di appartenenza della n-esima osservazione, dato $\mathbf{x}_n$ e data la stima dei parametri $\boldsymbol{\psi}^{(k)}$ all'iterazione k.

Se le z_{nj} fossero osservabili, allora la stima di massima verosimiglianza di α_j sarebbe data semplicemente da

$$\widehat{\alpha}_j = \frac{1}{N}\sum_{n=1}^{N} z_{nj} \qquad j = 1, \ldots, g\,.$$

Il passo E, nella $(k+1)$-esima iterazione richiede semplicemente la sostituzione di z_{nj}, nella precedente relazione, con il valore $\tau_{nj}(\boldsymbol{\psi}^{(k)})$. Nel passo M si procede all'aggiornamento della stima dei parametri. La stima di α_j è ottenuta sostituendo nella precedente relazione z_{nj} con $\tau_{nj}(\boldsymbol{\psi}^{(k)})$:

$$\alpha_j^{(k+1)} = \frac{1}{N}\sum_{n=1}^{N} \tau_{nj}^{(k)} \qquad j = 1, \ldots, g\,. \tag{2.30}$$

Le stime delle medie $\mu_1, \ldots, \mu_g$ e delle varianze $\sigma_1^2, \ldots, \sigma_g^2$ alla $k+1$-esima iterazione vengono ottenute tenendo conto che $\boldsymbol{\theta}^{(k+1)}$ è una soluzione dell'equazione

$$\sum_{j=1}^{g}\sum_{n=1}^{N} \tau_{nj}^{(k)} \frac{\partial \log f(\mathbf{x}_n; \boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}} = \mathbf{0}\,. \tag{2.31}$$

Un interessante aspetto dell'algoritmo EM è che la soluzione della (2.31) esiste in forma chiusa, ottenendo le stime di μ_j e σ_j al passo $k+1$. Tenendo infatti conto che

$$\frac{\sum_{n=1}^{N} z_{nj} x_n}{\sum_{n=1}^{N} z_{ij}} \qquad \text{e} \qquad \frac{\sum_{n=1}^{N} z_{nj}(x_n - \mu_j)^2}{\sum_{n=1}^{N} z_{nj}}$$

sono le stime di massima verosimiglianza rispettivamente di μ_j e di σ_j^2, se le z_{nj} fossero note, allora, poichè $\log L_c(\boldsymbol{\psi})$ è lineare rispetto a z_{nj}, ne segue che queste vengono sostituite dalle loro stime $\tau_{nj}^{(k)}$ e ciò conduce a

$$\begin{aligned}
\mu_j^{(k+1)} &= \frac{\sum_{n=1}^{N} \tau_{nj}^{(k)} x_n}{\sum_{n=1}^{N} \tau_{nj}^{(k)}} \quad j = 1, \ldots, g \\
\sigma_j^{2(k+1)} &= \frac{\sum_{n=1}^{N} \tau_{nj}^{(k)} (x_n - \mu_j^{(k+1)})^2}{\sum_{n=1}^{N} \tau_{nj}^{(k)}} \quad j = 1, \ldots, g\,.
\end{aligned}$$

In maniera analoga, nel caso multivariato, si può dimostrare che le stime dei vettori delle medie $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_g$ e delle matrici di covarianza $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_g$ sono date da

$$\begin{aligned}
\boldsymbol{\mu}_j^{(k+1)} &= \frac{\sum_{n=1}^{N} \tau_{nj}^{(k)} \mathbf{x}_n}{\sum_{n=1}^{N} \tau_{nj}^{(k)}} \quad j = 1, \ldots, g \\
\boldsymbol{\Sigma}_j^{(k+1)} &= \frac{\sum_{n=1}^{N} \tau_{nj}^{(k)} (\mathbf{x}_n - \boldsymbol{\mu}_j^{(k+1)})(\mathbf{x}_n - \boldsymbol{\mu}_j^{(k+1)})'}{\sum_{n=1}^{N} \tau_{nj}^{(k)}} \quad j = 1, \ldots, g\,.
\end{aligned}$$

2.10 Stima dei parametri di una mistura di densità t-Student

Consideriamo ora misture di g componenti di distribuzioni t. Sia assegnato un campione $\underset{\sim}{\mathbf{X}}$ estratto con distribuzione

$$p(\mathbf{x}; \boldsymbol{\psi}) = \sum_{j=1}^{k} \alpha_j f(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, \nu_j) \tag{2.32}$$

dove $f(\mathbf{x};\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j,\nu_j)$ è la funzione di densità della distribuzione t multivariata (1.15) con parametro di locazione $\boldsymbol{\mu}_j$, matrice prodotto interno definita positiva $\boldsymbol{\Sigma}_j$, con ν_j gradi di libertà. In tale contesto si ha

$$\boldsymbol{\psi} = (\alpha_1,\ldots,\alpha_g,\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_g,\boldsymbol{\Sigma}_1,\ldots,\boldsymbol{\Sigma}_g,\nu_1,\ldots,\nu_g)$$

e lo spazio parametrico è

$$\boldsymbol{\Psi} = \{\boldsymbol{\psi}\in\mathbb{R}^{g[2+q+(q^2+q)/2]} : \alpha_1+\ldots+\alpha_g = 1, \alpha_j\geq 0, |\boldsymbol{\Sigma}_j|>0 \qquad j=1,\ldots,g\}.$$

Prima di descrivere l'algoritmo di EM, è necessario scrivere la funzione di verosimiglianza in maniera opportuna. Il vettore di dati completi è dato da

$$\mathbf{x}_c = (\mathbf{x}',\mathbf{z}_1',\ldots,\mathbf{z}_N',u_1,\ldots,u_N)'$$

dove $\mathbf{x} = (\mathbf{x}_1',\ldots,\mathbf{x}_N')$ indica i dati osservati; $\mathbf{z}_1,\ldots,\mathbf{z}_N$ sono i vettori indicatori che indicano la componente di $\mathbf{x}_1,\ldots,\mathbf{x}_N$, dove $z_{nj} = (\mathbf{z}_n)_j$ è uguale a uno o zero a seconda che $\mathbf{x}_n$ appartenga o meno alla j-esima sottopopolazione. E' conveniente considerare i dati osservati, incrementati mediante il vettore $\mathbf{z}_n$, come ancora incompleti ed introdurre ulteriori dati mancanti $u_1,\ldots,u_N$ definiti da:

$$\mathbf{X}_n \,|\, u_n, z_{nj}=1 \overset{ind}{\sim} N(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j/u_n)\ n=1,\ldots,N;\ j=1,\ldots,g$$

e

$$U_n \,|\, z_{nj}=1 \overset{iid}{\sim} \text{gamma}(\nu_j/2,\nu_j/2)\ n=1,\ldots,N; j=1,\ldots,g. \tag{2.33}$$

Assegnati $\mathbf{z}_1,\ldots,\mathbf{z}_N$, le v.a. $U_1,\ldots,U_N$ sono i.i.d. con distribuzione (2.33).

La funzione di verosimiglianza $L_c(\boldsymbol{\psi})$, basata sui dati completi, può essere fattorizzata nel prodotto delle densità marginali delle $\mathbf{Z}_n$, le densità condizionate delle U_n date le $\mathbf{z}_n$, e le densità condizionate delle $\mathbf{X}_n$ date le u_n e le $\mathbf{z}_n$. In base a ciò, la logverosimiglianza per dati completi può essere scritta:

$$\log L_c(\boldsymbol{\psi}) = \log L_{1c}(\boldsymbol{\alpha}) + \log L_{2c}(\boldsymbol{\nu}) + \log L_{3c}(\boldsymbol{\theta}) \tag{2.34}$$

dove $\boldsymbol{\alpha} = (\alpha_1,\ldots,\alpha_g)'$ e $\boldsymbol{\theta}$ contiene le $\boldsymbol{\mu}_j$ e $\boldsymbol{\Sigma}_j$ ($j=1,\ldots,g$) e

$$\log L_{1c}(\boldsymbol{\alpha}) = \sum_{n=1}^{N}\sum_{j=1}^{g} z_{nj}\log\alpha_j \tag{2.35}$$

$$\log L_{2c}(\boldsymbol{\nu}) = \sum_{n=1}^{N}\sum_{j=1}^{g} z_{nj}\left\{-\log\Gamma\left(\frac{\nu_j}{2}\right)+\frac{\nu_j}{2}\log\left(\frac{\nu_j}{2}\right)+\frac{\nu_j}{2}(\log u_n - u_n)-\log u_n\right\} \tag{2.36}$$

$$\log L_{3c}(\boldsymbol{\theta}) = \sum_{n=1}^{N}\sum_{j=1}^{g} z_{nj}\left\{-\frac{q}{2}\log(2\pi)-\frac{1}{2}\log|\boldsymbol{\Sigma}_j|-\frac{1}{2}u_n(\mathbf{x}_n-\boldsymbol{\mu}_j)'\boldsymbol{\Sigma}_j^{-1}(\mathbf{x}_n-\boldsymbol{\mu}_j)\right\}. \tag{2.37}$$

Il passo E. Il passo E, nella $(k+1)$-esima iterazione dell'algoritmo EM, richiede il calcolo della speranza condizionata della logverosimiglianza basata su dati completi $\log\mathcal{L}_c(\boldsymbol{\psi})$, indicata con $Q(\boldsymbol{\psi};\boldsymbol{\psi}^{(k)})$, calcolata in base alla stima $\boldsymbol{\psi}^{(k)}$ di $\boldsymbol{\psi}$. A tal scopo, è necessario calcolare le quantità $\mathbb{E}_{\boldsymbol{\psi}^{(k)}}(\mathbf{Z}_{nj}|\mathbf{x}_n)$, $\mathbb{E}_{\boldsymbol{\psi}^{(k)}}(U_n|\mathbf{x}_n,z_n)$ e $\mathbb{E}_{\boldsymbol{\psi}^{(k)}}(\log U_n|\mathbf{x}_n,z_n)$, al fine di ottenere nuove valutazioni per le probabilità a posteriori τ_{nj} e per le u_{nj}. Si dimostra che (la speranza condizionata può quindi essere espressa come (vedi ad esempio McLachlan & Peel (2000), Shoham (2002) per dettagli)

$$Q(\boldsymbol{\psi};\boldsymbol{\psi}^{(k)}) = Q_1(\boldsymbol{\alpha};\boldsymbol{\psi}^{(k)}) + Q_2(\boldsymbol{\nu};\boldsymbol{\psi}^{(k)}) + Q_3(\boldsymbol{\theta};\boldsymbol{\psi}^{(k)}) \tag{2.38}$$

dove

$$Q_1(\boldsymbol{\alpha};\boldsymbol{\psi}^{(k)}) = \sum_{j,n} \hat{\tau}_{nj}^{(k)} \ln \alpha_j \tag{2.39}$$

$$Q_2(\boldsymbol{\nu};\boldsymbol{\psi}^{(k)}) = \sum_{j,n} \hat{\tau}_{nj}^{(k)} Q_{2n}(\nu_j;\boldsymbol{\psi}^{(k)}) \tag{2.40}$$

$$Q_3(\boldsymbol{\theta};\boldsymbol{\psi}^{(k)}) = \sum_{j,n} \hat{\tau}_{nj}^{(k)} Q_{3n}(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j;\boldsymbol{\psi}^{(k)}), \tag{2.41}$$

dove:

$$\begin{aligned} &Q_{2n}(\nu_j;\boldsymbol{\psi}^{(k)}) \\ &\quad = -\ln\Gamma(\nu_j/2) + (\nu_j/2)\ln(\nu_j/2) \\ &\quad + \frac{\nu_j}{2}\left\{\sum_{n=1}^{N}[\ln u_{nj}^{(k)} - u_{nj}^{(k)}] + \{\psi[(\nu_j^{(k)}+q)/2] - \ln[(\nu_j^{(k)}+q)/2]\}\right\} \end{aligned}$$

e:

$$\begin{aligned} &Q_{3n}(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j;\boldsymbol{\psi}^{(k)}) \\ &\quad = \frac{1}{2}\left\{-q\ln(2\pi) - \ln|\boldsymbol{\Sigma}_j| + q\ln u_{nj}{}^{(k)} - u_{nj}(\mathbf{x}_n-\boldsymbol{\mu}_j)'\boldsymbol{\Sigma}_j^{-1}(\mathbf{x}_n-\boldsymbol{\mu}_j)\right\}. \end{aligned}$$

Il passo M. Il passo M, nella $(k+1)$-esima iterazione, effettua la massimizzazione della speranza condizionata della verosimiglianza con dati completi. Dalla (2.38) segue che le stime $\boldsymbol{\alpha}^{(k+1)}$, $\boldsymbol{\theta}^{(k+1)}$ e $\boldsymbol{\nu}^{(k+1)}$ possono essere ottenute indipendentemente l'una dall'altra, operando massimizzazioni separate sulle (2.39), (2.40) e (2.41). Le soluzioni per $\alpha_j^{(k+1)}$, $\boldsymbol{\mu}_j^{(k+1)}$ e $\boldsymbol{\Sigma}_j^{(k+1)}$ vengono ricavate in forma chiusa:

$$\begin{aligned} \alpha_j^{(k+1)} &= \frac{1}{N}\sum_{n=1}^{N}\tau_{nj}^{(k)} \\ \boldsymbol{\mu}_j^{(k+1)} &= \frac{\sum_{n=1}^{N}\tau_{nj}^{(k)}u_{nj}^{(k)}\mathbf{x}_n}{\sum_{n=1}^{N}\tau_{nj}^{(k)}u_{nj}^{(k)}} \\ \boldsymbol{\Sigma}_j^{(k+1)} &= \frac{\sum_{n=1}^{N}\tau_{nj}^{(k)}u_{nj}^{(k)}(\mathbf{x}_n-\boldsymbol{\mu}_j^{(k+1)})(\mathbf{x}_n-\boldsymbol{\mu}_j^{(k+1)})'}{\sum_{n=1}^{N}\tau_{nj}^{(k)}} \end{aligned}$$

per $j = 1,\ldots,g$. La stima del numero di gradi di libertà $\nu_j^{(k+1)}$ si ottiene risolvendo l'equazione

$$\left\{-\phi\left(\frac{\nu_j}{2}\right) + \log\left(\frac{\nu_j}{2}\right) + 1 + \frac{1}{n_j^{(k)}}\sum_{n=1}^{N}\tau_{nj}^{(k)}(\log u_{nj}^{(k)} - u_{nj}^{(k)}) + \phi\left(\frac{\nu_j^{(k)}+q}{2}\right) - \log\left(\frac{\nu_j^{(k)}+q}{2}\right)\right\} = 0$$

dove

$$\phi(s) = \frac{1}{\Gamma(s)}\left\{\frac{\partial\Gamma(s)}{\partial s}\right\}$$

è la funzione digamma, $n_j^{(k)} = \sum_{n=1}^{N}\tau_{nj}^{(k)}$ e

$$u_{nj}^{(k)} = \frac{\nu_j^{(k)}+q}{\nu_j^{(k)} + \delta(\mathbf{x}_n,\boldsymbol{\mu}_j^{(k)};\boldsymbol{\Sigma}_j^{(k)})}$$

con $\delta(\mathbf{x},\boldsymbol{\mu};\boldsymbol{\Sigma}) = (\mathbf{x}-\boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})$, introdotto in (1.16).

Capitolo 3

Stima del numero di componenti

3.1 Il problema

La stima del numero di componenti g in una mistura costituisce un importante aspetto della costruzione del modello, che però ancora non ha trovato una soluzione definitiva. Abbiamo visto nel Capitolo 1 che i modelli mistura sono utilizzati o come modello semiparametrico per la stima di densità non note (applicazioni indirette) oppure per la costruzione di modelli di *clustering* (approccio diretto).

In quest'ultimo caso, si utilizzano criteri di selezione del modello quali ad esempio l'AIC (*Akaike Information Criterion* – Akaike, 1974) oppure il BIC/SBC (Bayesian Information Criterion – Schwarz, 1978) che sono basati sulla funzione di logverosimiglianza, ma che tengono conto anche della complessità del modello, vedi ad esempio Kadane & Lazar (2004) per una rassegna su questi indici e delle loro basi teoriche sia in contesto bayesiano che frequentista.

Va osservato che, nel contesto della costruzione di modelli di *cluster*, la stima del numero g di componenti della mistura assume una particolare importanza in quanto coincide con la stima del numero di gruppi nella popolazione in esame. Un primo criterio, in assenza di informazioni a priori, è quello di considerare il numero di mode della distribuzione a partire da un'analisi esplorativa con metodi grafici. In ogni caso, una difficoltà di carattere pratico nella stima del numero di componenti nasce dal fatto che qualunque distribuzione di interesse può essere approssimata da numerose altre che risultanto indistinguibili dal punto di vista empirico (tenendo conto di una data numerosità campionaria), vedi Donoho (1988). Nel caso delle misture una mistura con g componenti può essere indistinguibile empiricamente da una con un numero di componenti inferiore o superiore. Pertanto, da un punto di vista pratico, il problema è quello di stimare il numero di componenti di un modello mistura in termini del modello più semplice (cioè con il minor numero g di componenti diverse fra loro e con proporzioni $\alpha_j > 0$, con $j = 1, \ldots, g$) che offre un buon adattamento ai dati osservati.

3.2 Un esempio introduttivo

Consideriamo il file *stamp.txt* che concerne misurazioni dello spessore di $N = 485$ francobolli (in mm). Izenman & Sommer (1988) hanno studiato il problema di modellare tali osservazioni mediante una mistura gaussiana, vedi Figura 3.1. Consideriamo la costruzione di modelli mistura gaussiani con un numero g di componenti ($g = 1, \ldots, n$) sia nel caso di eteroschedasticità che in quello di omoschedasticità utilizzando la routine `EMuniv.r`. Per la rappresentazione grafica dell'istogramma consideriamo una suddivisione in 60 sottointervalli. Per il caso eteroschedastico (`vareq = F`) si ha:

```
y<-read.table('dati/stamp.txt', header=T)
```

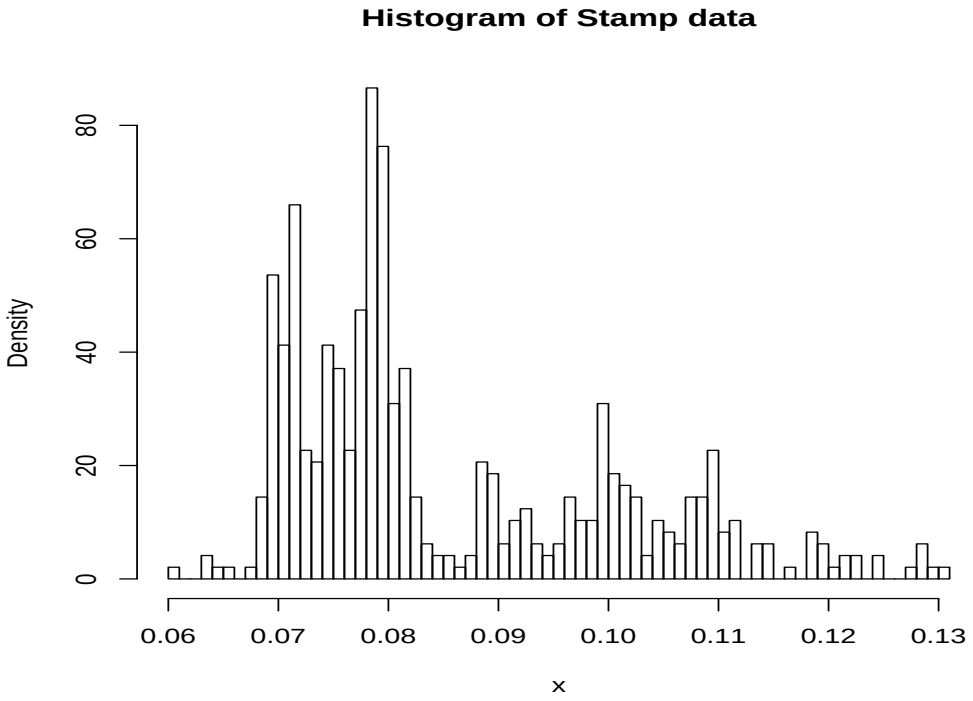


Figura 3.1: Istogramma della distribuzione dei dati del file *stamp.txt*.

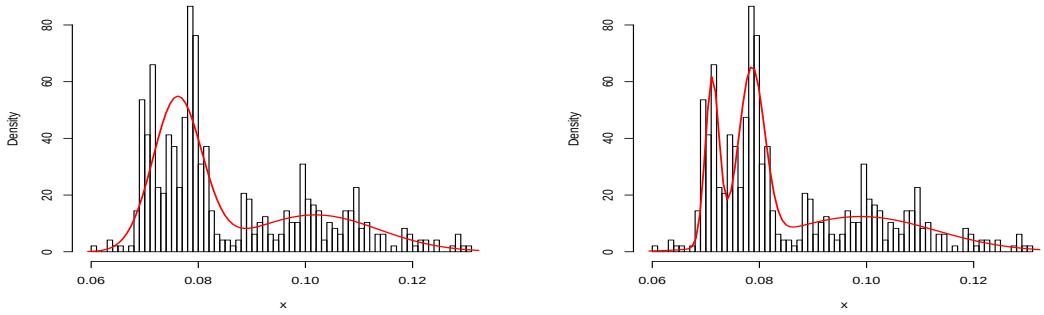
```
x<-y[,1]
EMuniv(x,1,60,F)
EMuniv(x,2,60,F)
...
EMuniv(x,8,60,F)
EMuniv(x,9,60,F)
```

e per il caso omoschedastico (`vareq = T`):

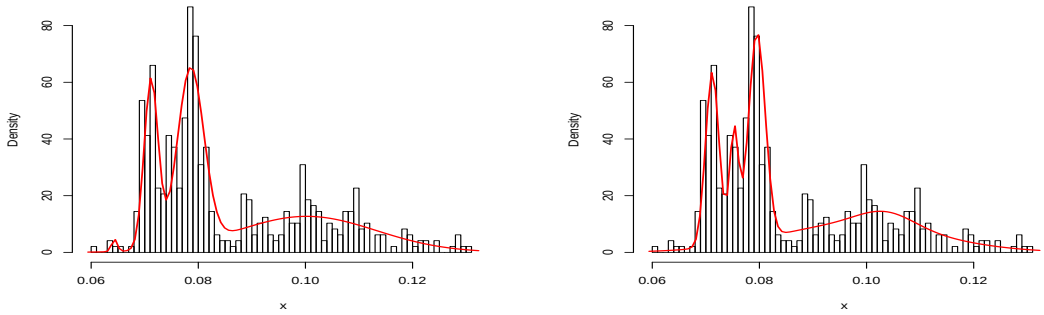
```
EMuniv(x,1,60,T)
EMuniv(x,2,60,T)
...
EMuniv(x,8,60,T)
EMuniv(x,9,60,T)
```

Possiamo vedere come varia la funzione di logverosimiglianza e la stima della funzione di densità in dipendenza del numero di componenti g, per $g = 1, \ldots, 9$, vedi Tabella 3.1 e Figura 3.2. Risultati delle simulazioni mostrano innanzitutto che, in corrispondenza di valori della funzione di logverosimiglianza uguali, o molto prossimi, si possono ottenere modelli molto diversi fra loro, vedi ad esempio i due modelli in Figura 2.1. Per visualizzare, ad esempio si può considerare:

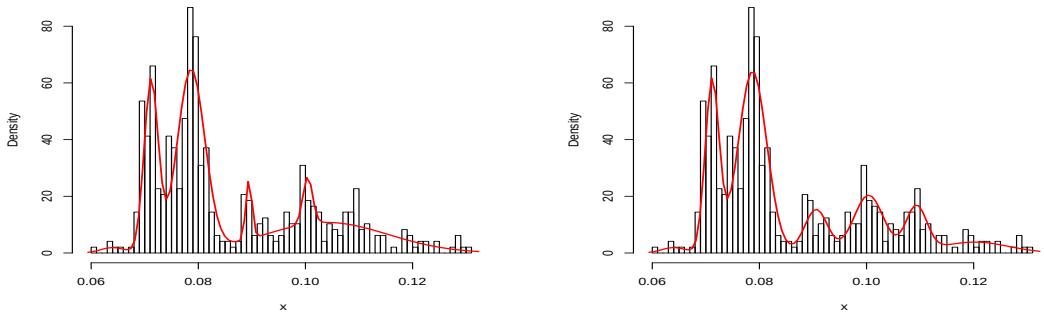
```
par(mfrow=c(2,2))
EMuniv(x,6,60,F)
EMuniv(x,6,60,F)
EMuniv(x,6,60,F)
EMuniv(x,6,60,F)
```



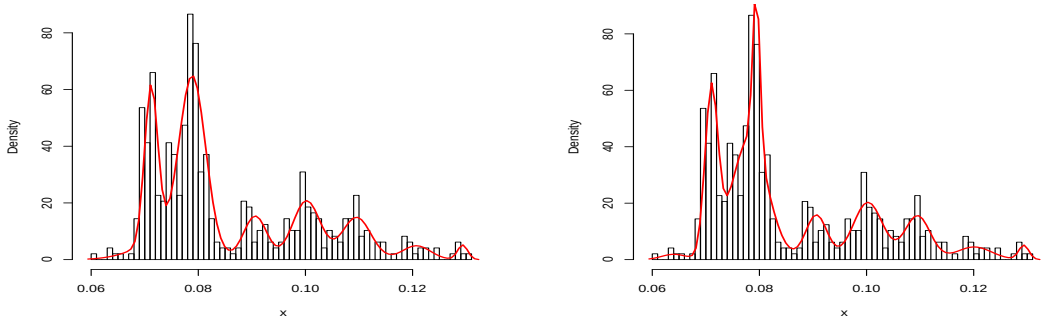
$g = 2,\ \mathcal{L}(\hat{\boldsymbol{\psi}}) = 1484.7$ $g = 3,\ \mathcal{L}(\hat{\boldsymbol{\psi}}) = 1518.8$



$g = 4,\ \mathcal{L}(\hat{\boldsymbol{\psi}}) = 1522.3$ $g = 5,\ \mathcal{L}(\hat{\boldsymbol{\psi}}) = 1531.6$



$g = 6,\ \mathcal{L}(\hat{\boldsymbol{\psi}}) = 1535.3$ $g = 7,\ \mathcal{L}(\hat{\boldsymbol{\psi}}) = 1539.8$



$g = 8,\ \mathcal{L}(\hat{\boldsymbol{\psi}}) = 1543.0$ $g = 9,\ \mathcal{L}(\hat{\boldsymbol{\psi}}) = 1550.7$

Figura 3.2: Modello mistura per dati *stamp.txt*: grafico della funzione di densità stimata per $g = 2, \ldots, 9$ (modelli eteroschedastici).

Numero g componenti	Varianze diverse (eteroschedasticità)	Varianze uguali (omoschedasticità)
1	1350.3	1350.3
2	1484.7	1439.0
3	1518.8	1474.5
4	1522.3	1487.0
5	1531.6	1487.3
6	1535.3	1505.8
7	1539.8	1525.3
8	1543.0	1526.0
9	1550.7	1535.2

Tabella 3.1: Modello mistura per dati *stamp.txt*: valori della logverosimiglianza per $g = 1, \ldots, 9$

Bisogna inoltre valutare se il modello ottenuto è compatibile con i dati in esame. Ad esempio, se un modello mistura viene utilizzato per descrivere una densità univariata multimodale, potrebbe risultare inaccettabile che una componente abbia una varianza molto maggiore di quella delle altre componenti, vedi ad esempio Figura 3.2, con $g = 3, 4, 5$ e quindi valori che si trovano agli estremi della distribuzione vengano assegnati alla stessa classe, come si può verificare utilizzando la routine `EMunivc.r`.

3.3 Metodi per la stima del numero di componenti

Nella maggior parte degli studi sulle misture, il problema della stima del numero di componenti g è separato da quello della stima dei parametri: dopo aver stimato il numero di componenti si passa alla stima dei parametri per g fissato. Approcci più complessi riguardano la stima congiunta di numero di componenti e parametri del modello, vedi McLachlan and Peel (2000) per ulteriori dettagli.

Possiamo distinguere vari approcci: nel seguito ci soffermeremo brevemente su quello basato sul test del rapporto di verosimiglianza mentre nel paragrafo successivo ci soffermeremo su criteri di selezione del modello basati su forme penalizzate di verosimiglianza). E' possibile poi considerare approcci di tipo non parametrico, ad esempio basati su test sul numero di mode o sul metodo dei momenti.

Test del rapporto di verosimiglianza. Un importante metodo per la stima del numero di componenti è basato su una verifica di ipotesi, dove assegnato un campione casuale $X_1, \ldots, X_N$ estratto con legge (1.3), si considera l'ipotesi nulla che la mistura abbia g_0 componenti rispetto all'alternativa che la mistura sia costituita da g_1 componenti:

$$H_0 \,:\, g = g_0 \qquad \text{vs} \qquad H_1 \,:\, g = g_1$$

con $g_1 > g_0$. In tale contesto si potrebbe utilizzare il test del rapporto di verosimiglianza dove la distribuzione asintotica di riferimento per l'ipotesi nulla è la distribuzione chi-quadrato.

Purtroppo, le necessarie ipotesi di regolarità che tale approccio richiede non sono verificate nei problemi con modelli mistura, poichè – sotto l'ipotesi nulla – le proporzioni della mistura in generale

giacciono sulla frontiera dello spazio dei parametri e pertanto i parametri stessi non sono identificabili sotto l'ipotesi nulla, vedi paragrafo 1.4. Vari approcci sono stati considerati al fine di affrontare il problema, compresi metodi bootstrap, vedi McLachlan and Peel (2000). Fra i lavori più recenti segnaliamo l'articolo di Lo *et al.* (2001) in cui – con riferimento a distribuzioni gaussiane omoschedastiche univariate – si considera un approccio basato sull'informazione di Kullback-Leibler dove, sotto opportune ipotesi di regolarità, la statistica test del rapporto di verosimiglianza segue asintoticamente la distribuzione di una somma pesata di variabili chi-quadrato con un grado di libertà.

3.4 Criteri di selezione del modello

Sia K il numero di parametri del modello (gradi di libertà). In generale questi criteri di selezione del modello hanno una forma del tipo

$$-2\mathcal{L}(\widehat{\boldsymbol{\psi}}) + \mathcal{C}_K \tag{3.1}$$

dove $\mathcal{L}(\widehat{\boldsymbol{\psi}})$ denota la funzione di logverosimiglianza calcolata in corrispondenza della stima finale $\widehat{\boldsymbol{\psi}}$ e $\mathcal{C}_K$ indica un termine di complessità che rappresenta una penalizzazione che cresce all'aumentare del numero di parametri del modello: se il modello in esame è troppo semplice, per offrire un buon adattamento ai dati, allora la (3.1) assumerà un grande valore in quanto si avrà un bassa verosimiglianza; al contrario, se il modello è troppo complesso, allora la (3.1) assumerà un grande valore a causa del termine $\mathcal{C}_K$.

3.4.1 L'indice di Akaike (AIC)

Un primo criterio che rientra nella famiglia (3.1) è quello di Akaike, l'*Akaike Information Criterion*, vedi Akaike (1973, 1974). Premettiamo alcuni concetti inerenti l'informazione di Kullback-Leibler, che sono alla base dell'AIC, vedi Burnham & Anderson (2004).

Sia ϕ la funzione di densità soggiacente il fenomeno in esame (ovviamente ϕ non ha parametri, possiamo chiamarla la "realtà") ed indichiamo con p una funzione di densità (modello) che approssima ϕ. L'*informazione di Kullback-Leibler* $\mathcal{I}(\phi, p)$ è l'informazione che viene persa quando il modello p viene utilizzato per approssimare la distribuzione ϕ:

$$\mathcal{I}(\phi, p) = \int \phi(\mathbf{x}) \log \left(\frac{\phi(\mathbf{x})}{p(\mathbf{x}; \boldsymbol{\psi})} \right) d\mathbf{x} \, . \tag{3.2}$$

Chiaramente il modello migliore (relativamente ad altri in esame) è quello per cui risulta minima la (3.2). Si noti che l'informazione di Kullback-Leibler può essere concettualizzata come distanza fra ϕ e $p(\cdot)$. La realtà ϕ è ovviamente considerata come fissa, mentre il modello ϕ varia all'interno di una classe $\mathcal{P}$ indicizzata con $\boldsymbol{\psi} \in \boldsymbol{\Psi}$. Il criterio (3.2) non può essere utilizzato direttamente nella selezione del modello in quanto richiede la conoscenza di ϕ e della famiglia $\mathcal{P}$. Si noti che la (3.2) può essere espressa anche in una delle due forme:

$$\mathcal{I}(\phi, p) = \int \phi(\mathbf{x}) \log \phi(\mathbf{x}) d\mathbf{x} - \int \phi(\mathbf{x}) \log[p(\mathbf{x}; \boldsymbol{\psi})] d\mathbf{x} \, . \tag{3.3}$$

$$= \mathbb{E}_\phi[\log \phi(\mathbf{X})] - \mathbb{E}_\phi[\log(p(\mathbf{X}; \boldsymbol{\psi})] \tag{3.4}$$

dove la speranza matematica $\mathbb{E}_\phi(\cdot)$ è calcolata rispetto alla densità ϕ e $\mathbf{X}$ è il vettore aleatorio associato a $\mathbf{x}$. Nella (3.4) la prima quantità non dipende dal modello p, pertanto, in corrispondenza di un dato $\boldsymbol{\psi} \in \boldsymbol{\Psi}$, il problema è quello di minimizzare

$$\mathbb{E}_\phi[\log(p(\mathbf{X}; \boldsymbol{\psi})] = \int \phi(\mathbf{x}) \log[p(\mathbf{x}; \boldsymbol{\psi})] d\mathbf{x} \, . \tag{3.5}$$

Sia assegnata la realizzazione $\mathbf{x}_1, \ldots, \mathbf{x}_N$ di un campione $\mathbf{X}_1, \ldots, \mathbf{X}_N$ e sia $\widehat{\boldsymbol{\psi}}$ la stima di massima verosimiglianza di $\boldsymbol{\psi}$ ottenuta in corrispondenza di $\mathbf{x}_1, \ldots, \mathbf{x}_N$. La (3.5) si scrive pertanto

$$\eta = \mathbb{E}_\phi[\log(p(\mathbf{X}; \widehat{\boldsymbol{\psi}})] = \int \phi(\mathbf{x}) \log[p(\mathbf{x}; \widehat{\boldsymbol{\psi}})] d\mathbf{x} \tag{3.6}$$

In Akaike (1973, 1974) si dimostra che lo stimatore $\widehat{\eta}$ di (3.5) ottenuto in corrispondenza di un campione $\mathbf{X}_1, \ldots, \mathbf{X}_N$

$$\widehat{\eta} = \frac{1}{N} \sum_{n=1}^{N} \log p(\mathbf{X}_n; \widehat{\boldsymbol{\psi}}) = \frac{1}{N} \log L(\widehat{\boldsymbol{\psi}}) \tag{3.7}$$

è distorto e fornisce una sovrastima della (3.6); in particolare la distorsione è asintoticamente uguale al numero K di parametri indipendenti del modello p. Pertanto una stima asintoticamente corretta della (3.5) è data da:

$$\frac{1}{N} \log L(\widehat{\boldsymbol{\psi}}) - K \ . \tag{3.8}$$

In letteratura, usualmente i criteri di selezione del modello sono espressi in termini del doppio dell' opposto della quantità precedente, e quindi si ottiene l'indice di AIC

$$\mathrm{AIC} = -2 \log L(\widehat{\boldsymbol{\psi}}) - 2K \ . \tag{3.9}$$

In base al criterio di Akaike si sceglie il modello che minimizza la quantità (3.9), che equivale a massimizzare la (3.8).

Si noti che le assunzioni che portano alla stima asintotica (3.8) non sono del tutto verificate nei modelli mistura, a causa di problemi di identificabilità; pur tuttavia tale criterio viene ampiamente utilizzato. In ogni caso, alcuni studi hanno comunque evidenziato che il criterio di Akaike tende a sovrastimare il numero di componenti.

3.4.2 L'indice BIC/SBC

Un altro criterio di ampio utilizzo per la selezione del modello è il BIC (*Bayesian Information Criterion*) derivato in un contesto Bayesiano, anche se può essere utilizzato in ambiti più ampi, vedi Schwarz (1978), Raftery (1995).

Il fattore di Bayes. Per introdurre il BIC, premettiamo alcune considerazioni sul fattore di Bayes per confrontare due ipotesi alternative, rappresentate due modelli statistici alternativi M_1 e M_2 (aventi vettori dei parametri rispettivamente uguali a $\boldsymbol{\psi}_1$ e $\boldsymbol{\psi}_2$), sulla base di un insieme di dati $\underset{\sim}{\mathbf{X}}$ (realizzazione di un campione, ottenuto dal modello M_1 oppure da M_2). In base al teorema di Bayes si ha:

$$P(M_i|\underset{\sim}{\mathbf{X}}) = \frac{p(\underset{\sim}{\mathbf{X}}|M_i)p(M_i)}{p(\underset{\sim}{\mathbf{X}}|M_1)p(M_1) + p(\underset{\sim}{\mathbf{X}}|M_2)p(M_2)} \qquad i = 1, 2 \tag{3.10}$$

dove $p(M_1)$ e $p(M_2) = 1 - P(M_1)$ sono le probabilità a priori di M_1 e M_2 e $p(\underset{\sim}{\mathbf{X}}|M_1)$ oppure $p(\underset{\sim}{\mathbf{X}}|M_2)$ sono rispettivamente le densità marginali dei dati, condizionatamente al modello M_i ($i = 1, 2$). Nella (3.10) la quantità $p(\underset{\sim}{\mathbf{X}}|M_i)$ è ottenuta integrando rispetto a $\boldsymbol{\theta}_1$

$$p(\underset{\sim}{\mathbf{X}}|M_1) = \int p(\underset{\sim}{\mathbf{X}}|\boldsymbol{\psi}_1, M_1) p(\boldsymbol{\psi}_1|M_1) d\boldsymbol{\psi}_1 = \int (\text{verosimiglianza} \times \text{priori}) \, d\boldsymbol{\psi}_1 \tag{3.11}$$

dove $p(\underset{\sim}{\mathbf{X}}|\boldsymbol{\psi}_1, M_1)$ è la verosimiglianza di $\boldsymbol{\psi}_1$ condizionatamente al modello M_1. La quantità $p(\underset{\sim}{\mathbf{X}}|M_1)$ viene chiamata *verosimiglianza integrata* per il modello M_1, oppure anche verosimiglianza marginale, probabilità marginale dei dati oppure probabilità predittiva dei dati.

Una misura di quanto i dati supportino il modello M_2 rispetto al modello M_1 è fornita dal *posterior odds* di M_2 rispetto a M_1, cioè dal rapporto fra le rispettive probabilità a posteriori. In base all'equazione (3.10), tale misura è data da

$$\frac{P(M_1|\underset{\sim}{\mathbf{X}})}{P(M_2|\underset{\sim}{\mathbf{X}})} = \left[\frac{p(\underset{\sim}{\mathbf{X}}|M_1)}{p(\underset{\sim}{\mathbf{X}}|M_2)}\right]\left[\frac{p(M_1)}{p(M_2)}\right] = B_{12}\left[\frac{p(M_1)}{p(M_2)}\right] . \tag{3.12}$$

Il primo termine a destra dell'equazione (3.12), denotato con B_{12}, è il rapporto delle verosimiglianze integrate dei due modelli e viene chiamato *fattore di Bayes* del modello 1 sul modello 2. Il secondo termine a destra dell'equazione (3.12) è il *prior odds*,e questo spesso è uguale a 1, non indicando alcuna preferenza per l'uno o l'altro modello, cioè $p(M_1) = p(M_2) = 0.5$. Pertanto l'equazione (3.12) può essere scritta

$$\text{posterior odds} = \text{fattore di Bayes} \times \text{prior odds} .$$

Se $B_{12} > 1$, allora i dati favoriscono il modello 1 rispetto al modello 2 e viceversa per $B_{12} < 1$.

L'approssimazione di Laplace. L'indice BIC costituisce un'approssimazione del fattore di Bayes. Il primo punto da affrontare riguarda l'approssimazione della verosimiglianza integrata, ed a tal scopo si considera l'approssimazione di Laplace.

Indichiamo con $h(\boldsymbol{\psi})$ una funzione di densità a priori per $\boldsymbol{\psi}$ e consideriamo la verosimiglianza integrata $p(\mathbf{X})$ in corrispondenza di un campione $\underset{\sim}{\mathbf{X}} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$:

$$p(\underset{\sim}{\mathbf{X}}) = \int p(\boldsymbol{\psi}, \underset{\sim}{\mathbf{X}}) d\boldsymbol{\psi} = \int \exp\left\{\log p(\boldsymbol{\psi}, \underset{\sim}{\mathbf{X}})\right\} d\boldsymbol{\psi} \tag{3.13}$$

dove $p(\boldsymbol{\psi}, \underset{\sim}{\mathbf{X}}) = h(\boldsymbol{\psi})L(\boldsymbol{\psi})$. Ovviamente i risultati sono condizionati rispetto alle densità componenti ed al numero g di componenti. Indichiamo con $\widetilde{\boldsymbol{\psi}}$ la moda di $\log p(\boldsymbol{\psi}, \mathbf{X})$, che soddisfa la relazione

$$g'(\widetilde{\boldsymbol{\psi}}) = \left.\frac{\partial \log p(\boldsymbol{\psi}, \mathbf{X})}{\partial \boldsymbol{\psi}}\right|_{\boldsymbol{\psi}=\widetilde{\boldsymbol{\psi}}} = \mathbf{0} .$$

Consideriamo l'approssimazione di Taylor del secondo ordine di $\log p(\boldsymbol{\psi}, \mathbf{X})$ intorno a $\widetilde{\boldsymbol{\psi}}$:

$$\log p(\boldsymbol{\psi}, \mathbf{X}) \approx \log p(\widetilde{\boldsymbol{\psi}}, \mathbf{X}) + \frac{1}{2}(\boldsymbol{\psi} - \widetilde{\boldsymbol{\psi}})'\mathbf{H}(\widetilde{\boldsymbol{\psi}})(\boldsymbol{\psi} - \widetilde{\boldsymbol{\psi}}) \tag{3.14}$$

dove $\mathbf{H}(\widetilde{\boldsymbol{\psi}})$ denota la matrice Hessiana negativa di $\log(p(\boldsymbol{\psi}, \mathbf{X}))$ calcolata in $\widetilde{\boldsymbol{\psi}}$, e tenendo conto che il termine del primo ordine è nullo in quanto $g'(\widetilde{\boldsymbol{\psi}}) = \mathbf{0}$. Si noti che nella (3.14) l'approssimazione è da considerarsi buona per valori di $\boldsymbol{\psi}$ prossimi a $\widetilde{\boldsymbol{\psi}}$; in ogni caso, per grandi valori di N, la verosimiglianza $p(\underset{\sim}{\mathbf{X}}|\boldsymbol{\psi})$ è concentrata intorno al suo valore massimo e decresce rapidamente all' aumentare della distanza di $\boldsymbol{\psi}$ da $\widetilde{\boldsymbol{\psi}}$, così che solo valori prossimi a $\widetilde{\boldsymbol{\psi}}$ contribuiscono effettivamente nell'integrale (3.13) al fine del calcolo di $\underset{\sim}{\mathbf{X}}$. Sostituendo questa espansione nell'integrale (3.13), si

ottiene:

$$\begin{aligned} p(\mathbf{X}) &= \exp\{\log(\widetilde{\boldsymbol{\psi}}, \mathbf{X})\} \int \exp\left\{-\frac{1}{2}(\boldsymbol{\psi} - \widetilde{\boldsymbol{\psi}})' \mathbf{H}(\widetilde{\boldsymbol{\psi}})(\boldsymbol{\psi} - \widetilde{\boldsymbol{\psi}})\right\} d\boldsymbol{\psi} \\ &\approx p(\widetilde{\boldsymbol{\psi}}, \mathbf{X}) |2\pi \mathbf{H}(\widetilde{\boldsymbol{\psi}})|^{-1/2} \int \frac{1}{|2\pi \mathbf{H}(\widetilde{\boldsymbol{\psi}})|^{-1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{\psi} - \widetilde{\boldsymbol{\psi}})' \mathbf{H}(\widetilde{\boldsymbol{\psi}})(\boldsymbol{\psi} - \widetilde{\boldsymbol{\psi}})\right\} d\boldsymbol{\psi} \\ &= p(\widetilde{\boldsymbol{\psi}}, \mathbf{X})(2\pi)^{q/2} |\mathbf{H}(\widetilde{\boldsymbol{\psi}})|^{-1/2} . \end{aligned} \tag{3.15}$$

Pertanto segue, essendo $p(\boldsymbol{\psi}, \underset{\sim}{\mathbf{X}}) = h(\boldsymbol{\psi}) L(\boldsymbol{\psi})$

$$\log p(\underset{\sim}{\mathbf{X}}) = \log L(\widetilde{\boldsymbol{\psi}}) + \log h(\widetilde{\boldsymbol{\psi}}) - \frac{1}{2} \log |\mathbf{H}(\widetilde{\boldsymbol{\psi}})| + \frac{q}{2} \log(2\pi) + O(N^{-1}) \tag{3.16}$$

che è nota come approssimazione di Laplace o del punto di sella (si noti che l'approssimazione è dell'ordine $O(N^{-1})$, cioè $NO(N^{-1})$ tende ad una qualche costante per $N_{\rightarrow}\infty$). Per grandi campioni, risulta $\widetilde{\boldsymbol{\psi}} \approx \widehat{\boldsymbol{\psi}}$, dove $\widehat{\boldsymbol{\psi}}$ è la stima di massima verosimiglianza e $\mathbf{H}(\widetilde{\boldsymbol{\psi}}) \approx N\mathcal{I}$, dove $\mathcal{I}$ è la matrice informazione attesa di Fisher per una singola osservazione, e quindi $|\mathbf{H}(\widetilde{\boldsymbol{\psi}})| \approx N^q \mathcal{I}$. Tali approsimazioni introducono un errore $O(N^{-1/2})$ nell'equazione (3.16) che diventa:

$$\log p(\mathbf{X}) \approx \log L(\widehat{\boldsymbol{\psi}}) + \log h(\widehat{\boldsymbol{\psi}}) - \frac{1}{2} \log |\mathcal{I}(\widetilde{\boldsymbol{\psi}}; \mathbf{X})| + \frac{q}{2} \log(2\pi) - \frac{q}{2} \log N + O(N^{-1/2}) . \tag{3.17}$$

Si noti che il termine $\log L(\widetilde{\boldsymbol{\psi}})$ è di ordine $O(N)$, il termine $q/2 \log N$ è di ordine $O(\log N)$, mentre gli altri quattro termini sono di ordine $O(1)$. Eliminando i termini di ordine $O(1)$, o di ordine inferiore, nella (3.17), si ottiene una nuova versione dell'approssimazione di Laplace

$$\log p(\mathbf{X}) \approx \log L(\widehat{\boldsymbol{\psi}}) - \frac{q}{2} \log N + O(1) \tag{3.18}$$

che indica che la logverosimiglianza integrata $\log p(\mathbf{X})$ è uguale al massimo della funzione di logverosimiglianza meno un termine di correzione.

La (3.18) conduce all'indice BIC:

$$\mathrm{BIC} = -2 \log L(\widehat{\boldsymbol{\psi}}) + q \log(N) . \tag{3.19}$$

per cui si sceglie il modello in corrispondenza del quale l'indice (3.19) assume valore minimo. Vari studi, suggeriscono l'utilizzo del BIC rispetto all'AIC. Vedi McLachlan & Peel (2000) e bibliografia ivi citata per dettagli.

Capitolo 4

Modelli mistura in Matlab

Nelle versioni di Matlab dalla 7.5.0 in poi la stima dei parametri di un modello mistura gaussiano può essere affrontata utilizzando i file di funzione interni presenti nel toolbox "Statistics". I file di funzione sono simili alle funzioni del linguaggio C, alle subroutine dei linguaggi Fortran e BASIC ed alle procedure del Pascal. Queste funzioni sono le basi per costruire programmi più complessi e possono essere modificati e personalizzati dall'utente.

4.1 I file di funzione: Introduzione

La prima riga in un file di funzione inizia sempre con la definizione della funzione che comprende l'elenco delle variabili di input e di output. La sintassi è la seguente:

`function` [*variabili di output*]= `nomefunzione`(*variabili di input*)

Le variabili di input sono sempre racchiuse tra parentesi tonde, quelle di output tra parentesi quadre. Le parentesi quadre sono facoltative quando c'è una sola variabile di output. Il nome della funzione `nomefunzione` è uguale al nome del file, con estensione *.m*, in cui è salvata la funzione. Ad esempio, la funzione `gmdistribution` è salvata nel file `gmdistribution.m` ed è "chiamata" digitando `gmdistribution` dopo il prompt di Matlab. La parola `function` nella riga di definizione è scritta in lettere minuscole. Le righe di commento iniziano con il simbolo `%` e possono essere poste in qualsiasi punto all'interno della funzione. Se si utilizza il comando `help` per ottenere informazioni sulla funzione, Matlab visualizza tutte le righe dei commenti che si trovano immediatamente dopo la riga di definizione della funzione. Il comando `lookfor` permette, invece, di accedere alla prima riga di commento. E' possibile chiamare le funzioni interne di Matlab anche non specificando esplicitamente le variabili di output. Se viene omesso il punto e virgola (;) alla fine dell'istruzione che chiama la funzione, la prima variabile dell'elenco delle variabili di output viene visualizzata con il nome `ans`.

Le variabili di input e di output specificate nella riga di definizione della funzione sono variabili locali. Questo significa che è possibile utilizzare altri nomi di variabili quando viene chiamata la funzione. Se non sono specificate le variabili di output al momento della chiamata della funzione, queste vengono cancellate quando la funzione è stata eseguita. Tale caratteristica consente di scrivere funzioni di utilità generica usando nomi di variabili scelti dall'utente. Di conseguenza, i file di funzione risultano "portabili" e non devono essere riscritti tutte le volte che vengono utilizzati in un programma diverso.

Le variabili di input ed output possono essere array numerici (matrici, vettori o numeri scalari)

oppure array di strutture. Questa seconda classe di array consente di salvare insieme variabili di natura diversa.

4.1.1 Cenni sugli array di strutture

Per accedere agli elementi delle strutture si usano i nomi dei campi. Questa caratteristica differenzia gli array di strutture dagli array di celle, a cui è possibile accedere utlizzando le normali tecniche di indicizzazione degli array. Per introdurre la terminologia delle strutture, riportiamo come esempio una delle funzioni interne di Matlab, riguardante i modelli mistura, che saranno descritte nel seguito. La funzione `gmdistribution.fit (X,k)` crea un array di strutture contenente le stime di massima verosimiglianza dei parametri di una mistura gaussiana di `k` componenti per la matrice `X`. Tale output dovrà contenere, fra le altre variabili, la dimensione d della normale multivariata, il vettore delle mixing proportions di ogni componente, le specifiche delle matrici di covarianza delle componenti (se diagonali o meno, se uguali o diverse in ogni componente), i vettori delle medie. Ogni tipo di dati (dimensione della distribuzione normale, vettore delle mixing proportions, specifiche delle matrici di covarianza, vettore delle medie, ...) è un *campo* ed il suo nome è il *nome del campo*. Nell'esempio sopra riportato, due campi contengono vettori di elementi numerici, un campo contiene una stringa di testo, un altro uno scalare. Per specificare i campi, sia quando si creano array di strutture, anche con la funzione `struct`, sia per accedere ai campi, una volta creati gli array, si adotta la notazione con il punto (.). Se l'array di strutture definito con la funzione `gmdistribution.fit` è chiamato `obj`:

`obj`=`gmdistribution.fit(X,2)`

un possibile esempio di output potrebbe essere il seguente:

`obj.NDimension` = 3
`obj.PComponents` = $[0.3\ 0.7]$
`obj.CovType:` = 'diagonal'
`obj.SharedCov`='true'
`obj.mu` = [0.2 0.3 2 ; 0.8 1.2 0.6]

Digitando `obj` dopo il prompt di Matlab si ottiene la seguente risposta

`NDimension:` = 3
`PComponents:` = $[0.3\ 0.7]$
`CovType:` = 'diagonal'
`SharedCov:` = 'true'
`mu:` = [0.2 0.3 2 ; 0.8 1.2 0.6]

Per ottenere informazioni sui campi si utilizza il comando `fieldnames`. Ad esempio, digitando `fieldnames(obj)` Matlab visualizza il nome di tutti i campi assegnati all'array `obj`.

Per accedere al contenuto di un determinato campo, si digita il punto (.) fra il nome dell'array di strutture ed il nome del campo. Riprendendo l'esempio precedete, se si digita `obj.mu` Matlab visualizza

0.2 0.3 2

0.8 1.2 0.6

Per accedere agli elementi di un campo, si utilizza la normale indicizzazione. Ad esempio, se si vuole visualizzare il vettore delle medie corrispondente alla prima componente, basta digitare `obj.mu(1,:)`. Se si vogliono salvare nella variabile `M` le medie delle due componenti relative alla prima dimensione della normale multivariata, si può digitare `F=obj.mu(:,1)`. E' possibile assegnare o modificare i valori degli elementi di un campo. Ad esempio, digitando `obj.mu(1,1)=0.4` il

Tabella 4.1: Comandi per gli array di strutture

Comando	**Breve descrizione**
`fieldnames(obj)`	Visualizza i nomi dei campi associati all'array di strutture `obj`
`names=fieldnames(obj)`	Registra nell'array `names` i nomi dei campi associati all'array di strutture `obj`
`F=getfield(obj,'field')`	Registra nell'array `F` il contenuto del campo `field` dell'array di strutture `obj` E' equivalente al comando `F=obj.field`
`isfield(obj, 'field')`	Restituisce 1 (vero) se `field` è il nome di un campo dell'array di strutture `obj` e 0 (falso) in caso contrario
`isstruct(obj)`	Restituisce 1 se `obj` è un array di strutture e 0 se non lo è
`obj=rmfield(obj,'field')`	Elimina il campo `field` dall'array `obj`
`obj=setfield(obj,'field',v)`	Assegna il valore `v` al campo `field` dell'array `obj`
`obj=struct('c1',v1, 'c2', v2, ...)`	Crea l'array di strutture `obj` con i campi `c1`, `c2`, ... che hanno rispettivamente i valori `v1`, `v2`, ...

valore della media della prima dimensione nella prima componente passa da 0.2 a 0.4. L'indicizzazione diretta è la tecnica migliore per creare, modificare o accedere ai valori di un campo. Tuttavia si possono utilizzare anche le funzioni riportate nella tabella 4.1. Il comando `setfield` può essere utlizzato per assegnare i valori di un campo, il comando `getfield` per leggerli. La sintassi

`m11=getfield(obj,'mu',{1,1})` equivale al comando

`m11=obj.mu(1,1)`

La sintassi

`0.4=setfield(obj,'mu',{1,1})` equivale al comando

`obj.mu(1,1)=0.4`

4.2 I file di funzione interni per i modelli mistura

I file di funzione per i modelli mistura presenti in Matlab sono:

`gmdistribution`: Crea un array di strutture per una mistura di componenti gaussiane

`gmdistribution.fit`: Crea un array di strutture con le stime di massima verosimiglianza di un modello mistura.

`cdf(gmdistribution)`: Calcola la funzione di ripartizione per una mistura di componenti gaussiane.

`cluster (gmdistribution)`: Classifica le osservazioni sulla base della mistura, assegnando ogni osservazione alla componente per cui è massima la probablità a posteriori.

`mahal (gmdistribution)`: Calcola la distanza di Mahalanobis di ogni osservazione dalla media di ogni componente.

`pdf(gmdistribution)`: Calcola la funzione di densità di una mistura di gaussiane.

`posterior (gmdistribution)`: Calcola la probabilità a posteriori di ogni componente della mistura.

`random (gmdistribution)`: Genera numeri casuali da una mistura di gaussiane

La sintassi e la descrizione approfondita di ciascuna di queste funzioni è riportata di seguito

I modelli implementati dalle funzioni di Matlab sono quelli di McLachlan & Peel descritti in: McLachlan G. & Peel D. (2000) Finite Mixture Models, John Wiley & Sons, New York

4.2.1 gmdistribution

La sintassi è:

`obj=gmdistribution (mu, sigma, p)`

Descrizione delle variabili di input

`mu` è la matrice di dimensione $k \times d$ che contiene i vettori d-dimensionali delle medie delle k componenti

`sigma` è un array di celle contenete le matrici di covarianza di ciascuna componente. Le dimensioni di `sigma` sono

- $d \times d \times k$ se non vi sono restrizioni sulle matrici. In questo caso, `sigma(:,:,1)` è la matrice di covarianza della prima componente, `sigma(:,:,2)` è la matrice della seconda componente, ...
- $1 \times d \times k$ se le matrici sono vincolate ad essere diagonali, ma possono essere diverse in ogni componente. In questo caso, `sigma(:,:,1)` è un vettore d dimensionale contenente gli elementi sulla diagonale della matrice di covarianza della prima componente.
- $d \times d$ se le matrici di covarianza sono vincolate ad essere uguali in ogni componente. In questo caso, `sigma(:,:)` è la matrice comune a tutte le componenti.
- $1 \times d$ se la matrici di covarianza sono vincolate ad essere le stesse all'interno di ciascuna componente e ad essere diagonali. In questo caso, `sigma(:)` è il vettore d dimensionale degli elementi della diagonale della matrice comune a tutte le componenti.

`p` è il vettore k dimensionale delle mixing proportions. Se la somma degli elementi di `p` non è uguale ad 1, Matlab automaticamente normalizza il vettore. Se `p` viene omessa come variabile di input, i valori di default sono uguali mixing proportions (`p` è vettore di elementi tutti pari a $1/k$).

Descrizione delle variabili di output

I campi dell'array di strutture `obj` sono:

`CovType`: è una stringa di testo uguale a 'diagonal' se le matrici di covarianza sono vincolate ad essere diagonali, a 'full' se le matrici non hanno restrizioni.

`DistName`: è la stringa di testo 'gaussian mixture ditribution'.

`mu` è la matrice contenente i vettori d dimensionali delle k medie.

`NComponents`: contiene lo scalare k, numero di componenti della mistura.

`NDimensions`: contiene lo scalare d, dimensione delle normali nella mistura.

`PComponents`: è il vettore p delle mixing proportions.

`SharedCov`: è una stringa di testo uguale a 'true' se le matrici di covarianza sono vincolate ad essere uguali per ogni componente, uguale a 'false' in caso contrario.

`Sigma`: è lo stesso array di celle delle matrici di covarianza utilizzato come input nella funzione.

Si noti come Matlab distingua i caratteri maiuscoli da quelli minuscoli: digitando, ad esempio, `obj.sigma`, risulta un messaggio di errore. Per visualizzare l'array è necessario digitare `obj.Sigma` oppure solo `sigma`. Nel primo caso si visualizza il contenuto del campo creato nell'array di strutture `obj`, nel secondo caso l'array creato dall'utente come valore di input. I valori dei campi dell'array `obj` possono essere modificati utilizzando i comandi riportati nella tab. 4.1 oppure tramite

l'indicizzazione diretta. Ad esempio, per modificare il nome dell'array `obj`, si possono usare le seguenti sintassi

```
obj.DistName='mistura di gaussiane' oppure
obj=setfield(obj,'DistName','mistura di gaussiane')
```

Esempio

Creiamo un array di strutture contenente i valori di una mistura di tre normali in due dimensioni. Se le componenti sono

$N_1(\boldsymbol{\mu}_1; \boldsymbol{\Sigma}_1)$, $N_2(\boldsymbol{\mu}_2; \boldsymbol{\Sigma}_2)$, $N_3(\boldsymbol{\mu}_3; \boldsymbol{\Sigma}_3)$, con

$\boldsymbol{\mu}_1 = [-2.5\ 2]$, $\boldsymbol{\mu}_2 = [-1\ -1.5]$ e $\boldsymbol{\mu}_3 = [3\ -3]$,

$\boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0.4 \end{pmatrix}$, $\boldsymbol{\Sigma}_2 = \begin{pmatrix} 0.3 & 0 \\ 0 & 0.2 \end{pmatrix}$ e $\boldsymbol{\Sigma}_3 = \begin{pmatrix} 0.4 & 0 \\ 0 & 0.6 \end{pmatrix}$

e le tre componeneti hanno uguale probabilità a priori, i comandi da digitare sono:

```
mu=[-2.5 2; -1 -1.5; 3 -3];
sigma=cat(3, [1 0; 0 0.4],[0.3 0; 0 0.2], [0.4 0; 0 0.6]);
p=ones(1,3)/3;
obj=gmdistribution(mu, sigma,p);
```

Matlab crea il file `obj` e visualizza la seguente risposta

```
obj= Gaussian mixture distribution with 3 components in 2 dimensions
Component 1:  Mixing proportion:  0.333 Mean:  -2.50 2.00
Component 2:  Mixing proportion:  0.333 Mean:  -1.00 -1.50
Component 3:  Mixing proportion:  0.333 Mean:  3 -3
```

Se si desidera visualizzare l'array `sigma` si può digitare `obj.Sigma` dopo il prompt e la risposta è:

```
 ans(:,:,1)=
 1.00  0
 0    0.40
ans(:,:,2)=
 0.30 0
 0  0.20
ans(:,:,3) =
 0.40 0
 0 0.6
```

4.2.2 gmdistribution.fit

La sintassi è:

```
obj=gmdistribution.fit (X, k, par1, val1, par2, val2, ...)
```

Descrizione delle variabili di input

`X` è la matrice dei dati $n \times d$, dove n è il numero di osservazioni e d il numero di variabili. Se nella matrice X vi sono dati mancanti, Matlab adotta l'esclusione listwise (la riga della matrice con almeno un dato mancante viene complemente eliminata al momento dell'adattamento del modello).

`k` è il numero di componenti del modello mistura che si vuole adattare alla matrice dei dati.

`par1, val1, par2, val2, ...` sono i nomi di parametri, con i rispettivi valori imputati dall'utente, che riguardano l'algoritmo EM. I parametri sono:

- `Start`: è il metodo di inizializzazione dell'algoritmo EM. I valori imputati dall'utente possono essere:
  - `RandSample`: per selezionare casualmente k osservazioni dalla matrice X come valori iniziali delle medie delle k componenti. I valori iniziali delle mixing proportions sono uguali e pari a $1/k$. Le matrici di covarianza delle componenti sono uguali e diagonali, con valori sulla diagonale pari alle varianze calcolate sulla matrice X.
  - `S`: dove `S` è un array di strutture con i campi "mu", "Sigma" e "PComponentes" dei valori iniziali del vettore delle medie, delle matrici di covarianza e del vettore delle mixing proportions scelti dall'utente.
  - `s`: dove `s` è un vettore di lunghezza n contenente la componente iniziale associata a ciascuna osservazione nella matrice X. In base alle osservazioni associate a ciascuna componente, Matlab calcola il vettore delle medie, le matrici di covarianza e le mixing proportions iniziali.

  Se il parametro non viene specificato nell'input (entro le parentesi tonte), il valore di default è `RandSample`.
- `Replicates`: se il metodo di inizializzazione dei parametri è `RandSample`, consente di ripetere la stima dei parametri, partendo da diverse inizializzazioni casuali del vettore delle medie, un numero di volte pari al valore specificato. Il valore deve essere un numero intero. Viene automaticamente scelta la stima dei parametri con il massimo valore della funzione di verosimiglianza. Il valore di default (valido anche per gli altri metodi di inizializzazione) è 1.
- `CovType`: i valori imputati dall'utente possono essere `diagonal` se le matrici di covarianza sono vincolate ad essere diagonali, `full` in caso contrario. Il valore di default è `full`.
- `SharedCov`: può assumere i valori logici `true` se tutte le matrici di covarianza sono vincolate ad essere uguali, `false` se possono essere diverse.
- `Regularize`: è un numero non negativo che viene aggiunto alla diagonale delle matrici di covarianza per renderle definite positive. Il valore di default è 0. Se le matrici non sono definite positive e non viene imputato un parametro diverso da 0, in grado di renderle definite positive, l'algoritmo EM implementato in Matalab non riesce a stimare i parametri e viene visualizzato un messaggio di errore.
- `Options`: permette di variare le opzioni dell'algoritmo EM presenti nell'array di strutture `statset`. Le opzioni in `statset` che riguardano l'algoritmo EM sono:
  - `Display`: permetter di decidere le informazioni che devono essere visualizzate durante la stima dei parametri. Può assumere i valori `off`, per non visualizzare alcuna

informazione, `final` per visualizzare solo i risultati finali, `iter` per visualizzare i risultati relativi ad ogni iterazione dell'algoritmo. Il valore di default è `off`.

- `MaxIter`: è il numero massimo di iterazioni dell'algoritmo. Il valore di default è 100.
- `TolFun`: è il valore di convergenza della funzione obiettivo (funzione di verosimiglianza). Il valore di default è `1e-006`.

L'array di strutture `statset` contiene anche altre opzioni, utlizzate in diversi file di funzione del toolbox "Statistics". Per visulizzare tutte le opzioni e i valori che possono essere imputati ad ognuna di queste, basta digitare `statset` dopo il prompt di Matlab. Digitando `statset('gmdistribution')` si ottiene il seguente output:

```
ans =
Display: 'off'
MaxFunEvals: []
MaxIter: 100
TolBnd: []
TolFun: 1.0000e-006
TolX: []
 GradObj: []
 DerivStep: []
FunValCheck: []
Robust: []
WgtFun: []
Tune: []
```

Accanto alle opzioni che non riguardano il file di funzione `gmdistribution` sono visualizzate parentesi quadre vuote (indicano che non vi è alcun valore associato a questi campi). Accanto alle altre opzioni, riguardanti l'algortmo EM, sono visualizzati i valori di default. I valori di default possono essere cambiati seguendo diverse alternative:

- Tramite il comando `options = statset(...)` si crea un nuovo array di strutture chiamato `options`. Se non si specificano i valori di input, tutti i campi del nuovo array risultano vuoti ([ ]). Digitando `options = statset('fieldname1', val1, 'fieldname2', val2, ...)` si crea un array di strutture in cui i campi specificati (`fieldname`) hanno il valore immesso (`val`). I valori non specificati sono automaticamente considerati campi vuoti. I nomi dei campi sono stringhe di testo e devono essere racchiusi tra virgolette. Anche i valori dei campi che sono stringhe di testo devono essere racchiusi tra virgolette. Se viene immesso un valore non plausibile, il valore è automaticamente cambiato con quello di default.
- Digitando `options = statset(oldopts, fieldname1, val1, fieldname2, val2, ...)` si crea il nuovo array `options` che è una copia dell'array `oldopts`, salvo per i valori dei campi `fieldname` che assumono i nuovi valori `val` specificati nell'input.
- Digitando `options = statset(oldopts, newopts)` si modifica l'array di strutture `oldopts` in base al nuovo array `newopts`. Tutti i parametri in `newopts` che non hanno campi vuoti come valori, vengono riscritti nell'array `oldopts`.

Descrizione delle variabili di output

Oltre ai valori già descritti in `gmdistribution`, il file di funzione `gmdistribution.fit` presenta le seguenti variabili aggiuntive:

`AIC`: E' il valore del Criterio di Informazione di Akaike (AIC) raggiunto dall'algoritmo EM.

$AIC = 2 \times n \log L + 2 \times c$

dove n è il numero di osservazioni (numero di righe nella matrice di input $\mathbf{X}$), L è il valore della funzione di verosimiglianza, c è il numero totale di parametri stimati.

`BIC`: E' il valore del criterio di informazione di Bayes (BIC) raggiunto dall'algoritmo EM.

$BIC = 2 \times n \log L + 2 \times c \log(n)$.

Il numero c di parametri stimati è:

$d + k - 1 + k \times d$ se le matrici di covarianza sono diagonali e uguali

$d \times k + k - 1 + k \times d$ se le matrici di covarianza sono diagonali e diverse nele componenti

$d \times (d+1)/2 + k - 1 + k \times d$ se le matrici di covarianza sono uguali

$k \times (d \times (d+1)/2) + k - 1 + k \times d$ se le matrici di covarianza sono diverse nelle componenti

`Converged`: Assume valore logico `true` se l'algoritmo ha raggiunto uno dei criteri di convergenza, `false` se l'algoritmo non ha raggiunto la convergenza.

`Iters`: E' il numero di iterazioni effettuate dall'algoritmo.

`NlogL`: E' il valore del logaritmo della verosimiglianza moltiplicato per n.

`RegV`: E' il valore del parametro di regolarizzazione immesso dall'utente (è pari a 0, il valore di default, se non è stato specificato alcun valore nell'input della funzione).

Esempio

Generiamo una matrice $\mathbf{X}$ di dimensione (2000×2) da una mistura di due componenti gaussiane. Se le componenti hanno distribuzione $N_1(\mu_1; \sum_1)$, $N_2(\mu_2; \sum \mathbf{2}_2)$, con

$\boldsymbol{\mu}_1 = [-2.5\ 2], \quad \boldsymbol{\mu}_2 = [-1\ -1.5]$

$\boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0.4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 0.3 & 0 \\ 0 & 0.2 \end{pmatrix}$

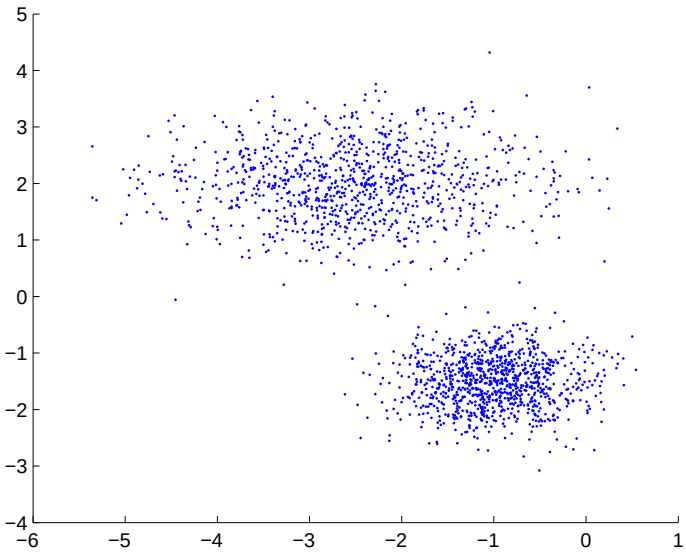
i comandi da digitare sono:

```
mu1=[-2.5 2];
mu2=[-1 -1.5];
sigma1=[1 0; 0 0.4];
sigma2=[0.3 0; 0 0.2];
X=[mvnrnd(mu1, sigma1, 1000); mvnrnd(mu2, sigma2, 1000)];
```

Nella matrice $\mathbf{X}$ le prime 1000 righe sono generate casualmente, tramite la funzione `mvnrnd` dalla distribuzione N_1, le altre 1000 dalla distribuzione N_2. Per visualizzare i valori generati, digitando

```
scatter (X(:,), X(:,2), 10, '.')
```

si ottiene un grafico a dispersione simile al seguente:



Per adattare ai dati $\mathbf{X}$ un modello mistura di due componenti e visualizzare i parametri finali dell'algoritmo EM si può digitare:

```
options=statset('Display','final');
obj=gmdistribution.fit(X,2,'Options', options);
```

Matlab visualizza un output strutturato nella maniera seguente:

```
28 iterations, log-likelihood = -5195.64
```

Il valore del numero di iterazioni e della log-verosimiglianza cambia di volta in volta, a seconda dei valori nella matrice $\mathbf{X}$ generati in modo casuale. Per visualizzare i parametri stimati, digitando `obj.mu` ottieniamo i vettori delle medie:

```
ans =
  -0.9693   -1.4941
  -2.4626    1.9851
```

digitando `obj.Sigma` otteniamo le matrici di covarianza:

```
ans(:,:,1) =
0.3179   -0.0159
-0.0159    0.2140
ans(:,:,2) =
0.9154   -0.0137
-0.0137   0.3869
```

Si noti come i valori delle covarianze non siano esattamente pari a zero, poichè nei valori di input della funzione `gmdistribution.fit` non è stato specificato il valore `'diagonal'` per

il parametro `CovType` ed è stato considerato dall'algoritmo il valore di default `full`. Digitando `obj.PComponents` si ottengono le stime delle probabilità a priori delle due componenti:

```
ans =
0.5001    0.4999
```

L'AIC assume valore minimo nei modelli con due componenti (quelli in cui è correttamente specificata la struttura reale dei dati). Con il seguente programma possiamo calcolare e confrontare l'AIC relativo ai modelli in 1, 2, 3, 4 ,5 e 6 componenti:

```
AIC=zeros(1,6);
cell(1,6);
for k=1:6
obj{k}= gmdistribution.fit(X,k);
AIC(k)=obj{k}.AIC
end
[min, Componenti]= min(AIC)
```

Facendo girare il programma otteniamo il seguente output (poichè l'ultima riga di istruzioni non termina con il punto e virgola (;) il relativo output, oltre ad essere salvato nello spazio di lavoro (workspace), viene anche visualizzato dopo il prompt nella finestra dei comandi (command window)):

```
AIC =
1.0e+004 *
 1.3055    1.0453    1.0457    1.0454    1.0466    1.0466
min =
 1.0453e+004
Ncomponenti =
 2
```

Digitando `obj{2}` si ottengono le stime del modello in due componenti:

```
 ans =
Gaussian mixture distribution with 2 components in 2 dimensions
Component 1:  Mixing proportion:  0.5001 Mean:  -0.9693 -1.4941
Component 2:  Mixing proportion:  0.4999 Mean:  -2.4626 1.9851
```

4.2.3 cdf(gmdistribution)

La sintassi è:

```
y= cdf (obj, X)
```

La funzione `cdf` calcola il valore della funzione di ripartizione della mistura di densità specificata nell'array `obj`, per la matrice `X`. L'array `obj` è creato con la funzione `gmdistribution` oppure `gmdistribution.fit`. Il vettore di output `y` ha lunghezza n, pari al numero di righe della matrice `X`.

Esempio

Con il seguente programma si può visualizzare la funzione di ripartizione della mistura di due gaussiane $N_1(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$, $N_2(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$, con

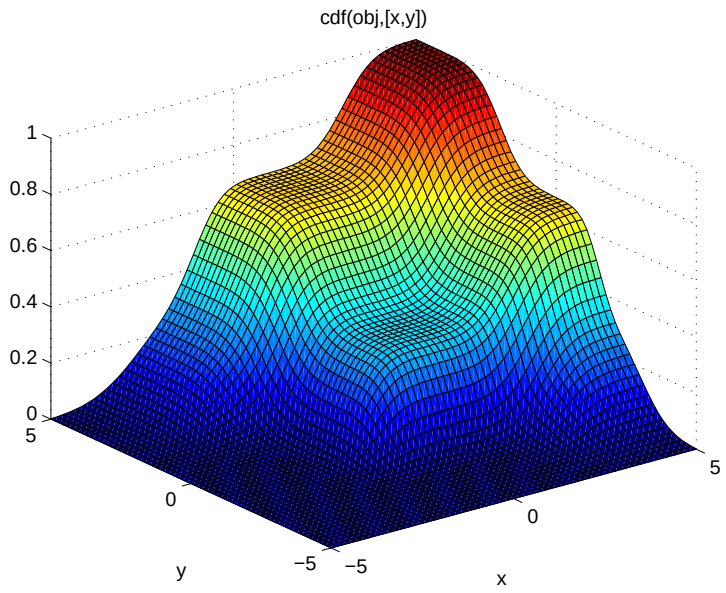
$$\boldsymbol{\mu}_1 = [-2.5\ 2], \quad \boldsymbol{\mu}_2 = [-1\ -1.5],$$

$$\boldsymbol{\Sigma}_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0.4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 0.3 & 0 \\ 0 & 0.2 \end{pmatrix}$$

ed uguali probabilità a priori:

```
mu=[-2.5 2; -1 -1.5];
sigma=cat(3, [1 0; 0 0.4],[0.3 0; 0 0.2]);
p=ones(1,2)/2
obj=gmdistribution(mu,sigma,p);
ezsurf(@(x,y)cdf(obj,[x y]),[-5 5], [-5 5])
```

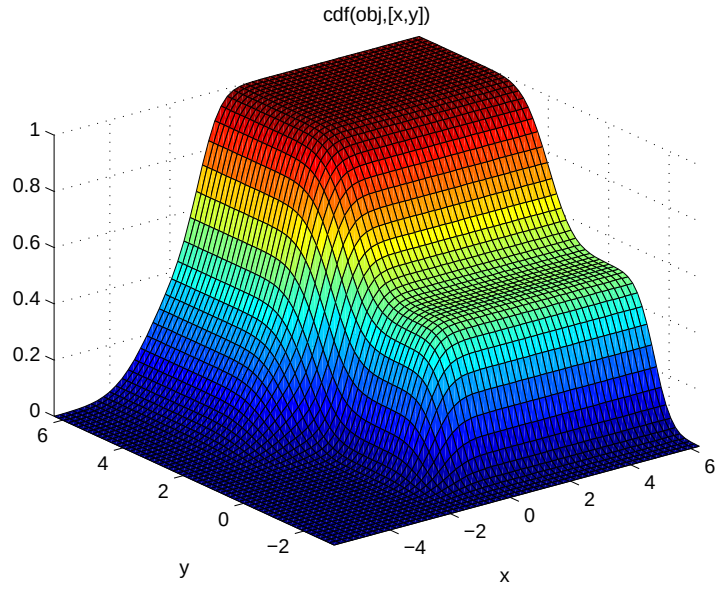
L'output è il seguente grafico:



Per visualizzare, invece, la frequenza relativa cumulata della matrice $\mathbf{X}$ contenente 2000 osservazioni generate casualmente dalla mistura delle due gaussiane, utilizziamo il seguente programma:

```
mu1=[-2.5 2];
mu2=[-1 -1.5];
sigma1=[1 0; 0 0.4];
sigma2=[0.3 0; 0 0.2];
X=[mvnrnd(mu1, sigma1, 1000); mvnrnd(mu2, sigma2, 1000)];
obj=gmdistribution.fit(X,2);
mu=[-2.5 2; -1 -1.5];
sigma=cat(3, [1 0; 0 0.4],[0.3 0; 0 0.2]);
p=ones(1,2)/2;
obj=gmdistribution(mu, sigma,p)
z=cdf(obj, X);
[x,y]=meshgrid(X(:,1),X(:,2));
ezsurf(@(x,y)cdf(obj,[x y]))
```

L'output è il seguente grafico:



4.2.4 cluster(gmdistribution)

La sintassi completa è:

```
[c, nlogl, P, logpdf, M]= cluster (obj, X)
```

La funzione `cluster` classifica le n osservazioni `X` in k gruppi determinati dalle componenti della mistura di gaussiane descritta in `obj`. L'array `obj` è creato con la funzione `gmdistribution` oppure con `gmdistribution.fit`. In generale, i dati `X` da classificare sono gli stessi utilizzati per stimare i parametri della mistura gaussiana descritta in `obj`. La classificazione, in Matlab, è considerata come uno step disgiunto dalla stima della densità. Se i dati `X` da classificare non sono gli stessi utilizzati per la stima della densità mistura, la classificazione ha senso logico solo se i nuovi dati provengono dalla stessa popolazione di quelli utilizzati per la stima. Le osservazioni vengono assegnate alla componente in cui risulta massima la probabilità a posteriori.

Le variabili di output sono:

`c`: è il vettore n dimensionale che contiene il numero della componente a cui afferisce ogni osservazione della matrice `X`.

`nlogl`: è il valore della log verosimiglianza dei dati moltiplicato per n.

`P`: è la matrice delle probabilità a posteriori di ogni componente condizionate alle osservazioni. Ha dimensione $n \times k$ ed il generico elemento $P(i, j)$ è la probabilità a posteriori della componente j data l'osservazione i-esima.

`logpdf`: è un vettore n-dimensionale contenente i logaritmi delle densità di probabilità stimate per ogni osservazione. Le densità sono date dalla somma delle probabilità a posteriori moltiplicate per le probabilità a priori delle componenti:$\sum_{j=1}^{k} p(\mathbf{x}_i|j)p(j)$.

`M`: è una matrice $n \times k$ contenente le ditanze di Mahalanobis dai centroidi delle componenti. Il generico elemento $M(i, j)$ è la distanza di Mahalanobis della osservazione i-esima dal centroide della componente j-esima.

Possono essere omesse tutte le variabili di output, ad eccezione del vettore `c` dei gruppi di apparteneza.

Esempio

Generiamo i dati $\mathbf{X}$ da una mistura di due componenti, $N_1(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$, $N_2(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$), con

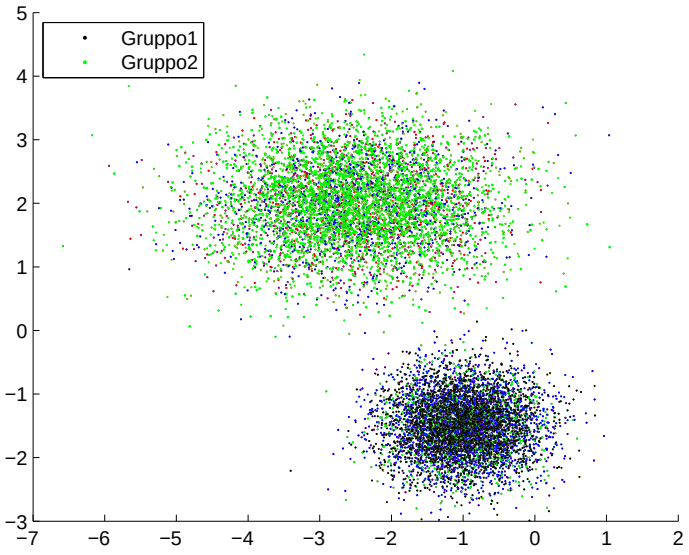
$$\boldsymbol{\mu}_1 = [-2.5\ 2], \quad \boldsymbol{\mu}_2 = [-1\ -1.5],$$

$$\sum_{\mathbf{1}} = \begin{pmatrix} 1 & 0 \\ 0 & 0.4 \end{pmatrix}, \quad \boldsymbol{\Sigma}_2 = \begin{pmatrix} 0.3 & 0 \\ 0 & 0.2 \end{pmatrix}$$

Il programma per generare i dati, stimare i parametri della mistura e classificare i dati in base al modello stimato, è:

```
mu1=[-2.5 2];
mu2=[-1 -1.5];
sigma1=[1 0; 0 0.4];
sigma2=[0.3 0; 0 0.2];
X=[mvnrnd(mu1, sigma1, 1000); mvnrnd(mu2, sigma2, 1000)];
obj=gmdistribution.fit(X,2);
c=cluster(obj,X);
gruppo1=X(c==1,:);
gruppo2=X(c==2,:);
scatter(X(:,1),X(:,2),12,'.');
hold on
h1 = scatter(gruppo1(:,1), gruppo1(:,2),12,'k.')
h2 = scatter(gruppo2(:,1), gruppo2(:,2),12, 'g.')
legend([h1, h2], 'Gruppo1', 'Gruppo2', 'Location', 'NorthWest')
```

L'output è il seguente grafico (se visualizzato a colori, le osservazioni classificate nel primo gruppo appaiono in nero, quelle classificate nel secondo, in verde):



4.2.5 mahal(gmdistribution)

La sintassi è:

```
M = mahal (obj, X)
```

La funzione `mahal` calcola le distanze di Mahalonibis di ogni osservazione nella matrice `X` dai centroidi delle componenti della mistura definita in `obj`. La matrice di output `M` ha dimensione $n \times k$ dove n è il numero di osservazioni (righe nella matrice `X`) e k il numero di componenti della mistura. Il generico elemento $M(i, j)$ è la distanza dell'osservazione j dal centroide della componente k.

Esempio

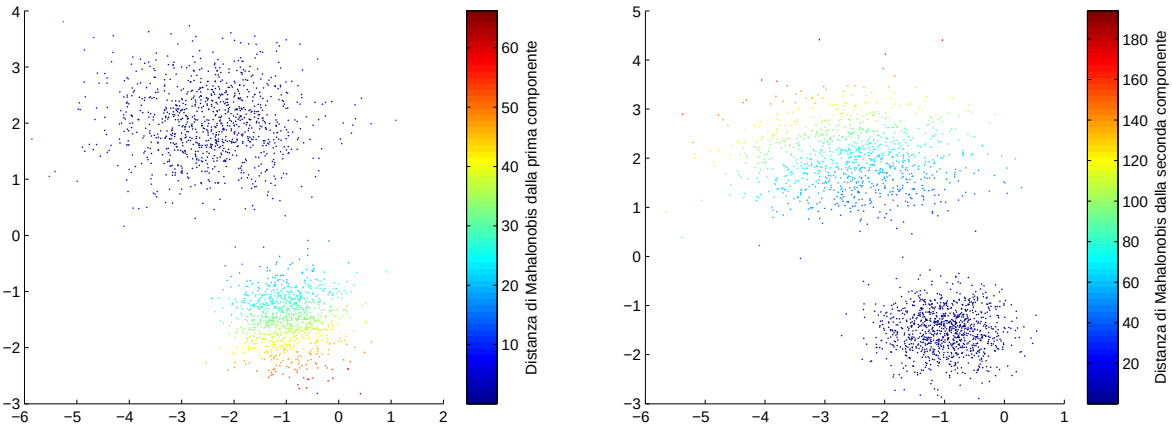
Generiamo una matrice 100×2 casualmente da due distribuzioni normali ed adattiamo ai dati un modello mistura in due componenti. Successivamente calcoliamo e rappresentiamo graficamente le distanze di ogni osservazione dalla prima componente stimata:

```
mu1=[-2.5 2];
mu2=[-1 -1.5];
sigma1=[1 0; 0 0.4];
sigma2=[0.3 0; 0 0.2];
X=[mvnrnd(mu1, sigma1, 1000); mvnrnd(mu2, sigma2, 1000)];
obj=gmdistribution.fit(X,2);
M=mahal(obj, X);
scatter(X(:,1), X(:,2), 12, M(:,1),'.');
hb=colorbar;
ylabel(hb, 'Distanza di Mahalonib dalla prima componente')
```

Per calcolare e rappresentare graficamente le distanze dalla seconda componente, basta sostituire le ultime tre righe con le seguenti:

```
scatter(X(:,1), X(:,2), 12, M(:,2),'.');
hb=colorbar;
ylabel(hb, 'Distanza di Mahalonib dalla seconda componente')
```

I grafici creati sono riportati di seguito (se visualizzati a colori, la barra a destra riporta i colori dal rosso al blu):



4.2.6 pdf(gmdistribution)

La sintassi è:

```
y = pdf (obj, X)
```

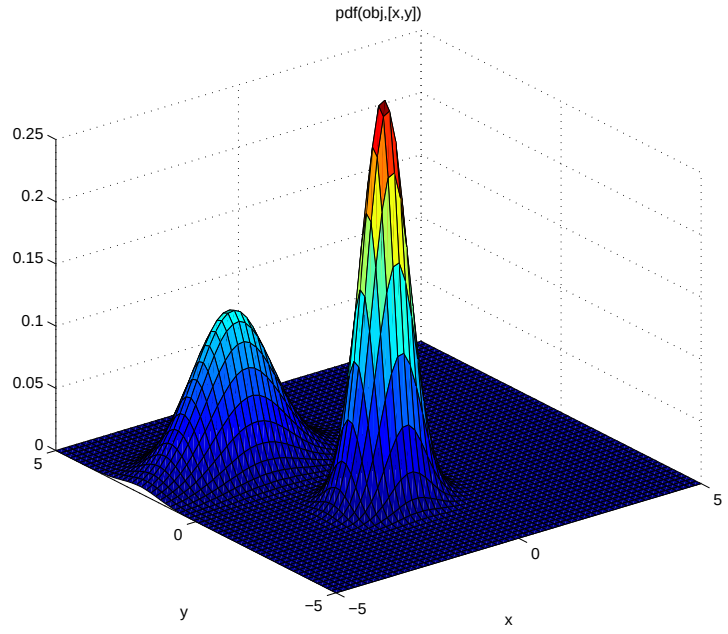
La funzione `pdf` calcola le densità di probabilità delle n osservazioni in `X` in base alla distribuzione specificata in `obj`. L'array "obj" è creato con la funzione `gmdistribution` oppure `gmdistribution.fit`. Il vettore `y` ha lunghezza n e contiene i valori delle n densità.

Esempio

Creiamo, tramite la funzione `gmdistribution`, l'array `obj` contenente i parametri di una mistura di due gaussiane e rappresentiamo graficamente la mistura utilizzando la funzione `pdf`:

```
obj=gmdistribution.fit(X,2);
mu=[-2.5 2; -1 -1.5];
sigma=cat(3, [1 0; 0 0.4],[0.3 0; 0 0.2]);
p=ones(1,2)/2;
obj=gmdistribution(mu, sigma,p);
ezsurf(x,y)pdf(obj,[x y]), [-5, 5], [-5 5]
```

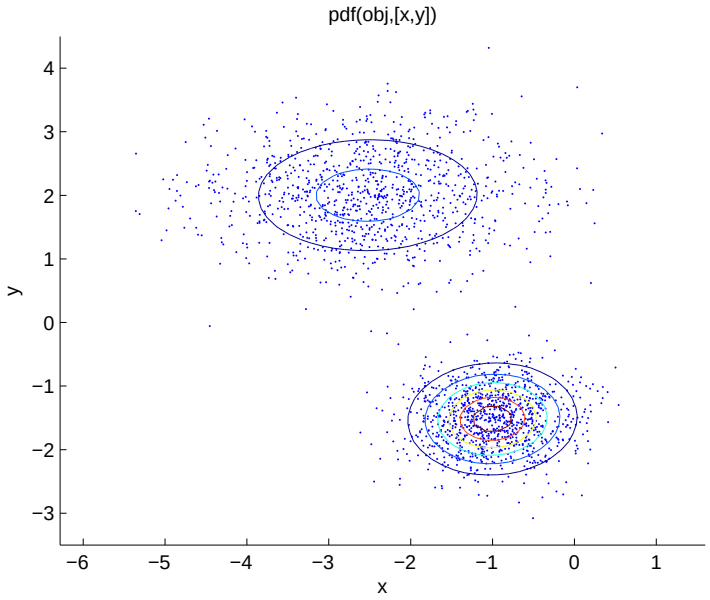
Il grafico di output è:



Per rappresentare graficamente gli isocontorni di probabilità della densità stimata su una matrice di dati **X**:

```
mu1=[-2.5 2];
 mu2=[-1 -1.5];
sigma1=[1 0; 0 0.4];
sigma2=[0.3 0; 0 0.2];
X=[mvnrnd(mu1, sigma1, 1000); mvnrnd(mu2, sigma2, 1000)];
scatter(X(:,1),X(:,2),12,'.')
hold on
obj=gmdistribution.fit(X,2);
h = ezcontour(@(x,y)pdf(obj,[x y]))
```

L'output è:



4.2.7 posterior(gmdistribution)

La sintassi è:

```
[P, nlogl] = posterior (obj, X)
```

La funzione `posterior` calcola le probabilità a posteriori di ciascuna componente nel modello mistura specificato in `obj`, condizionate alle osservazioni `X`. La matrice `X` ha dimensione $n \times d$, dove n è il numero di osservazioni e d il numero di variabili. L'array `obj` è creato con la funzione `gmdistribution` oppure `gmdistribution.fit`. La matrice di output `P` ha dimensione $n \times k$, dove k è il numero di componenti della mistura. Il generico elemento $P(i, j)$ è la probabilità a posteriori della componente j-esima, data l'osservazione i-esima. Lo scalare `nlolg` è il valore della log verosimiglianza.

Esempio

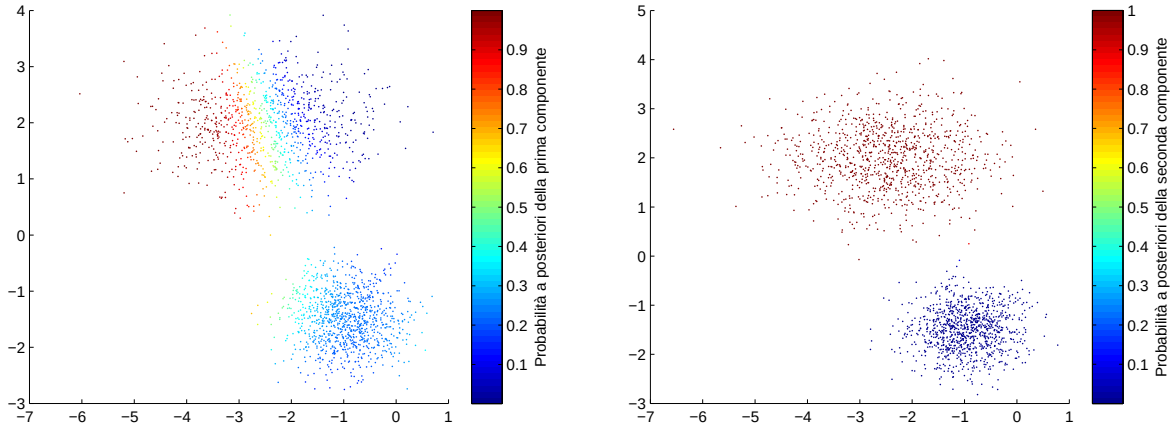
Generiamo i dati in $\mathbf{X}$ casualmente da due distribuzioni normali ed adiattiamo ai dati un modello mistura in due componenti. Successivamente calcoliamo e rappresentiamo graficamente le probabilità a posteriori della prima componente:

```
mu1=[-2.5 2];
mu2=[-1 -1.5];
sigma1=[1 0; 0 0.4];
sigma2=[0.3 0; 0 0.2];
X=[mvnrnd(mu1, sigma1, 1000); mvnrnd(mu2, sigma2, 1000)];
obj=gmdistribution.fit(X,2);
P=posterior(obj, X);
scatter(X(:,1), X(:,2), 12, P(:,1),'.');
hb=colorbar;
ylabel(hb, 'Probabilità a posteriori della prima componente')
```

Per calcolare e rappresentare graficamente le probabilità dalla seconda componente, basta sostituire le ultime tre righe con le seguenti:

```
scatter(X(:,1), X(:,2), 12, M(:,2),'.');
hb=colorbar;
ylabel(hb, 'Probabilità a posteriori della seconda componente'
```

I grafici creati sono riportati di seguito (se visualizzati a colori, la barra a destra riporta i colori dal rosso al blu):



4.2.8 random(gmdistribution)

La sintassi è:

```
[Y id] = random (obj, n)
```

La funzione `random` genera casulamente una matrice `Y` di n osservazioni dalla densità mistura definita in `obj`. Il numero di colonne in `Y` è pari alla dimensione d delle gaussiane nella mistura. Se il valore di `n` non viene specificato nelle variabili di input della funzione, il valore di default è 1 e la funzione genera un vettore d-dimensionale. Il vettore di output `id` indica, per ogni osservazione, la componente utilizzata per generarla.

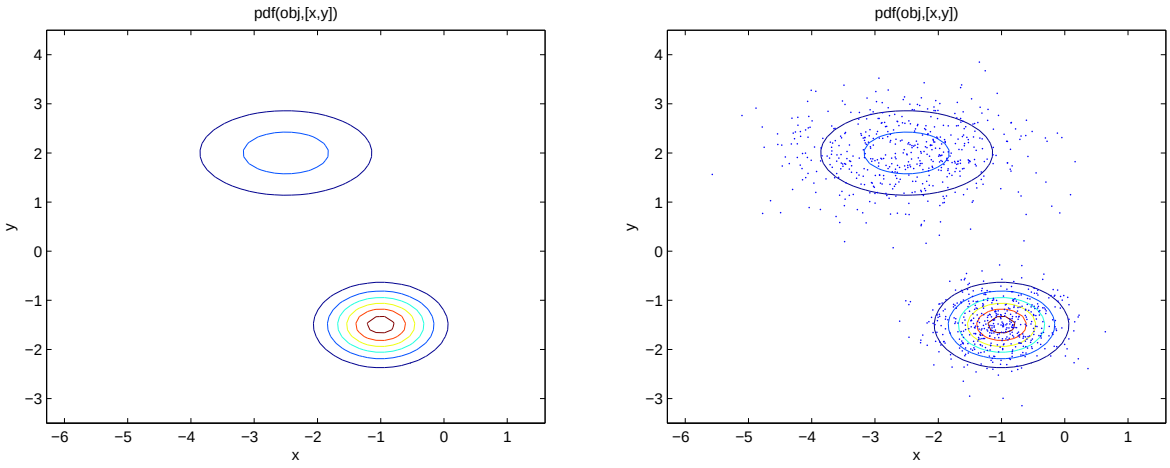
Esempio

Creiamo l'array di strutture `obj` utilizzando la funzione `gmdistribution`. Consideriamo una mistura di due componenti in due dimensioni. Rappresentiamo graficamente la densità e successivamente generiamo da questa 1000 osservazioni. Sovrapponiamo lo scatter plot delle osservazioni generate agli isocontorni della densità mistura:

```
mu=[-2.5 2; -1 -1.5];
sigma=cat(3, [1 0; 0 0.4],[0.3 0; 0 0.2]);
p=ones(1,2)/2;
obj=gmdistribution(mu, sigma,p)
ezcontour(@(x,y)pdf(obj,[x y]))
hold on
```

```
Y=random (obj, 1000);
scatter (Y(:,1),Y(:,2), 10, '.')
```

I grafici creati sono riportati di seguito:



4.3 Alcuni dettagli computazionali

4.3.1 Controllo sulle matrici di covarianza

Per controllare che le matrici Σ immesse come input nella funzione `gmdistribution` siano definite positive, Matlab effettua i seguenti controlli:

Sulle matrici diagonali, calcola il valore massimo degli elementi sulla diagonale moltiplicato per `eps`. Il simbolo `eps` indica il numero più piccolo che, quando viene sommato ad 1 dal computer, crea un numero maggiore di 1. In generale, `eps`=2^{-52}. Se il prodotto è maggiore del minimo degli elementi sulla diagonale, allora Matlab dà il seguente messaggio di errore:

```
error stats:gmdistribution:BadSigma,
Each diagonal covariance matrix must be positive definite
```

Sulle matrici non diagonali, effettua la seguente scomposizione di Cholesky:

```
[T, e]=cholcov(Sigma)
```

Se `Sigma` è quadrata, simmetrica e definita positiva, `T` è una matrice quadrata e triangolare superiore e `T'T=Sigma`. Se `Sigma` non è definita positiva, `T` non risulta necessariamente triangolare o quadrata. In questo caso, `T` contiene gli autovettori associati alla scomposizione in valori singolari della matrice `Sigma`. Gli autovettori associati ad autovalori prossimi a zero (con una piccola tolleranza) vengono omessi. Se vi sono autovalori negativi, allora `T` è una matrice vuota e il numero di autovalori negativi viene memorizzato in `e`. Se `Sigma` non è simmetrica e quadrata, `T` è una matrice vuota ed `e` è un riusultato numerico indefinito (non è un numero). Se `e` non è uguale a zero, Matlab dà il seguente messaggio di errore:

```
error stats:gmdistribution:BadSigma,
Each covariance matrix must be symmetric and positive definite
```

4.3.2 Implementazione dell'algoritmo EM

L'algoritmo EM viene implementato nel file di funzione `gmdistribution.m`. Il file richiama due importanti subroutines: `wdensity.m` ed `estep.m`. Nel seguito vedremo in dettaglio alcune righe di comando riportate in questi files.

La fase E

Nella fase E dell'algoritmo EM si calcola la probabilità di ogni componente condizionata ad ogni osservazione ed alle stime correnti dei parametri (alla prima iterazione, i parametri sono quelli specificati come valori iniziali dell'algoritmo). In termini formali:

$$p(z = j|\mathbf{x}_i, \psi_t) = \frac{p(z = j, \mathbf{x}_i|\psi_t)}{p(\mathbf{x}_i|\psi_t)} = \frac{p(\mathbf{x}_i|z = j, \psi_t)p(z = j|\psi_t)}{\sum_{j=1}^{k} p(\mathbf{x}_i|z = j, \psi_t)p(z = j|\psi_t)} \tag{4.1}$$

dove t indica l'iterazione, ψ il set di parametri stimati (le matrici di covarianza, i vettori delle medie e le mixing proportions), $\mathbf{x}_i$ è il vettore d-dimensionale relativo all'osservazione i-esima e z indica la componente gaussiana della mistura (in problemi di classificazione, z identifica il gruppo di provenienza).

Il primo passo nell'implementazione dell'E-step è il controllo dell'invertibilità delle matrici di covarianza correnti e la stima dei logaritmi delle probabilità $p(\mathbf{x}_i|z = j, \psi_t)p(z = j|\psi_t)$, per ogni $i = 1, \ldots, N$ e per ogni $j = 1, \ldots, k$. Le righe di programma sono:

```
1  for j = 1:k
2  if sharedCov % le matrici di covarianza sono uguali
3  if j == 1 % controllo solo sulla prima matrice
4  if CovTypte == 2 % le matrici non sono diagonali
5  [L,f] = chol(Sigma);
6  diagL = diag(L);
7  if(f ~=0)||any(abs(diagL)<
8  eps(max(abs(diagL)))*size(L,1))
9  error('stats:gmdistribution:wdensity:IllCondCov', ...
10 'Ill-conditioned covariance created.');
11 end
12 logDetSigma = 2*sum(log(diagL));
13 else % le matrici sono diagonali
14 L = sqrt(Sigma);
15 if any(L) < eps( max(L))*d
16 error('stats:gmdistribution:wdensity:IllCondCov', ...
17 'Ill-conditioned covariance created.');
18 end
19 logDetSigma = sum( log(Sigma));
20 end
21 end
22 else % le matrici di covarianza sono diverse
23 if CovType == 2 % le matrici non sono diagonali
24 [L,f] = chol(Sigma(:,:,j) );
25 diagL = diag(L);
26 if (f ~= 0) || any(abs(diagL) <
```

```
27 eps(max(abs(diagL)))*size(L,1))
28 error('stats:gmdistribution:wdensity:IllCondCov', ...
29 'Ill-conditioned covariance created');
30 end
31 logDetSigma = 2*sum(log(diagL));
32 else % le matrici sono diagonali
33 L = sqrt(Sigma(:,:,j));
34 if  any(L) < eps(max(L))*d
35 error('stats:gmdistribution:wdensity:IllCondCov', ...
36 'Ill-conditioned covariance created.')
37 end
38 logDetSigma = sum( log(Sigma(:,:,j)) );
39 end
40 end
41 Xcentered = bsxfun(@minus, X, mu(j,:));
42 if CovType == 2 % le matrici non sono diagonali
43 xRinv = Xcentered /L ;
44 else % le matrici sono diagonali
45 xRinv = bsxfun(@times, Xcentered , (1./ L));
46 end
47 mahalaD(:,j) = sum(xRinv.^2, 2);
48 {log_lh(:,j) = -0.5 * mahalaD(:,j) +(-0.5 *logDetSigma +
49 log_prior(j)) - d*log(2*pi)/2;}
50 end
```

Dalla riga 2 alla 40 viene fatto il controllo sull'invertibilità delle matrici di covarianza e viene calcolato il logaritmo del determinante.

Se le matrici sono diagonali, vengono calcolate le matrici `L` in cui gli elementi sono le radici quadrate degli elementi di `Sigma` (righe 14 e 33). Se qualche valore in `L` è minore della quantità `eps(max(L))*d`, la matrice risulta mal condizionata. `eps(max(L))` è il più piccolo numero che sommato al valore massimo di `L` dà il primo valore maggiore di questo, in virgola mobile e con la stessa precisione (righe 15-17 e 34-36). Il logaritmo del determinante è ottenuto come somma dei logaritmi degli elementi di `Sigma` (righe 19 e 38).

Se le matrici di covarianza non sono diagonali, la matrici `L` sono calcolate con la scomposizione di Cholesky (righe 5 e 24). Le matrici sono mal condizionate se almeno un autovalore è negativo oppure se qualche valore, in termini assoluti, sulla diagonale di `L` è minore della quantità `eps(max(abs(diagL)))*size(L,1))` (righe 7-10 e 26-29). `Size(L,1)` è il numero di righe della matrice `L` ed `eps(max(abs(diagL)))` è il il più piccolo numero che, sommato al valore massimo, in termini assoluti, sulla diagonale di `L`, dà il primo valore maggiore di questo, in virgola mobile e con la stessa precisione. Il logaritrmo del determinante della matrice di covarianza è calcolato come il doppio della somma dei logaritmi degli elementi sulla diagonale di `L` (righe 12 e 31).

Dalla riga 41 alla 49 vengono calcolati i logaritmi delle probabilità $p(\mathbf{x}_i|z = j, \psi_t)p(z = j|\psi_t)$, $\nabla i = 1, \ldots, N$ e $\nabla j = 1, \ldots, k$. Tali quantità possono essere espresse nel modo seguente:

$$-\frac{n}{2}ln(2\pi) - \frac{1}{2}ln|\Sigma_j| - \frac{1}{2}(\mathbf{x}_i - \mu_j)'\Sigma_j^{-}1(\mathbf{x}_i - \mu_j) + lnp(z = j|\psi) \tag{4.2}$$

dove $(\mathbf{x}_i - \mu_j)'\Sigma_j^{-}1(\mathbf{x}_i - \mu_j)$ è la distanza di Mahalanobis fra la media μ_j della componente j-esima ed il vettore $\mathbf{x}_i$ dell'osservazione i-esima. La distanza di Mahalanobis è calcolata nelle righe 41-45.

Le quantità (2) vengono calcolate nelle righe 48 e 49 e salvate nella matrice `log_lh` di dimensione $n \times k$. Le righe di comando relative ai passi successivi dell'E-step sono:

```
maxll=max(log_lh, [],2);
```
% vengono calcolati i massimi di riga della matrice `log_lh`. Tali massimi sono salvati nel vettore `maxll`, di dimensione n.
```
post=exp(bsxfun(@minus, log_lh, maxll));
```
% ad ogni colonna della matrice `log_lh` viene sottratto il vettore dei valori massimi, per evitare problemi di underflow, e viene calcolata la funzione esponenziale.
```
density=sum (post,2);
```
% vengono calcolate le quantità $\sum_{j=1}^{k} p(\mathbf{x}_i|z = j, \psi_t)p(z = j|\psi_t)$, per ogni osservazione $\mathbf{x}_i$. `density` è un vettore n dimensionale
```
post=bsxfun(@rdivide, post, density);
```
% viene calcolata la probabilità di ogni componente condizionata ad ogni osservazioni ed alle stime correnti dei parametri.
```
logpdf= log(density)+ maxll;
ll=sum (logpdf)
```
% viene calcolato il valore della log verosimiglianza.

Il controllo della convergenza avviene con le seguenti righe di comando:

```
llDiff=ll-ll_old;
if llDiff >= 0 && llDiff < options.TolFun *abs(ll)
OptimInfo.Coverdeg=true;
break;
end
ll_old=ll
```

Se l'algoritmo raggiunge la convergenza, nel campo `Converged` dell'array di strutture `OptimInf` viene salvato il valore logico 'true' e le iterazioni vengono interrotte. Se la differenza fra il valore della log verosimiglianza all'iterazione precedente (`ll_old`) e quello corrente è superiore al valore della tolleranza salvato nell'array `options.TolFun`, moltiplicato per il valore corrente della log verosimiglianza, le iterazioni riprendono. Se si è raggiunto il valore massimo delle iterazioni, Matlab manda il seguente messaggio:

```
stats:gmdistribution:FailedToConverge
Failed to converge for gmdistribution with k components
```

La fase M

I vettori μ_j, le matrici Σ_j e le mixing proportions p_j vengono aggiornati (e salvati nell'array di strutture `S`) attraverso i seguenti comandi:

```
S.PComponents = sum(post, 1);
if SharedCov %le matrici di covarianza sono uguali
if CovType == 2 % matrici non diagonali
S.Sigma = zeros(d,d);
  for j = 1:k
   S.mu(j,:) = post(:,j)' * X / S.PComponents(j);
```

```
    Xcentered = bsxfun(@minus, X, S.mu(j,:));
    Xcentered = bsxfun(@times,sqrt(post(:,j)),Xcentered);
    S.Sigma = S.Sigma + Xcentered'*Xcentered;
   end
S.Sigma = S.Sigma/sum(S.PComponents) + regVal *eye(d);
   else %le matrici sono diagonali
    S.Sigma = zeros(1,d);
   for j = 1:k
    S.mu(j,:) = post(:,j)' * X / S.PComponents(j);
    Xcentered = bsxfun(@minus, X, S.mu(j,:));
    S.Sigma = S.Sigma + post(:,j)' *(Xcentered.^2);
   end
  S.Sigma = S.Sigma/sum(S.PComponents)+regVal;
   end
  else % le matrici di covarianza sono diverse
  for j = 1:k
   S.mu(j,:) = post(:,j)' * X / S.PComponents(j);
   Xcentered = bsxfun(@minus, X, S.mu(j,:));
   if CovType == 2 % le matrici non sono diagonali
    Xcentered = bsxfun(@times,sqrt(post(:,j)),Xcentered);
  S.Sigma(:,:,j)=Xcentered'*Xcentered/S.PComponents(j)+regVal*eye(d);
   else % le matrici sono diagonali
   S.Sigma(:,:,j) = post(:,j)'*(Xcentered.^2)/S.PComponents(j) +regVal;
    end
   end
  end
S.PComponents = S.PComponents/sum(S.PComponents);
```

Terminata la fase di stima, per classificare le osservazioni in base alle probabilità stimate, si può utilizzare la seguente riga di comando:

```
[value idx]=max(post, [], 2);
```

dove `value` è il vettore n dimensionale in cui vengono salvate le maggiori probabilità condizionate $maxp(z|\mathbf{x}_i, \psi)$ ed `idx` è il vettore in cui vengono memorizzate le componenti associate a queste probabilità (il gruppo a cui afferisce l'osservazione).

Per calcolare le probabilità $p(\mathbf{x}_i|\psi)$ associate ad ogni osservazione:

```
y=sum(exp(log_lh),2)
```

Per calcolare le probabilità cumulate:

```
y = zeros(size(X,1),1); if obj.SharedCov
    if CovType==1
        for j=1:obj.NComponents
c = normcdf(X(:,1),obj.mu(j,1), sqrt(obj.Sigma(1,1)));
            for t=2:obj.NDimensions
c = c .* normcdf(X(:,t),obj.mu(j,t),sqrt(obj.Sigma(1,t)));
```

```
            end
            y = y + obj.PComponents(j) * c;
        end
    else
        for j=1:obj.NComponents
y = y+obj.PComponents(j) *mvncdf(X,obj.mu(j,:),obj.Sigma);
        end
    end
else
    if CovType==1
        for j=1:obj.NComponents
             c = normcdf(X(:,1),obj.mu(j,1),sqrt(obj.Sigma(1,1,j)));
             for t=2:obj.NDimensions
c=c.* normcdf(X(:,t),obj.mu(j,t), sqrt(obj.Sigma(1,t,j)));
             end
            y = y + obj.PComponents(j) * c;
        end
    else
        for j=1:obj.NComponents
y=y+obj.PComponents(j)*mvncdf(X,obj.mu(j,:),obj.Sigma(:,:,j));
        end
    end
 end
```

Capitolo 5

Classificazione mediante modelli mistura

5.1 Introduzione

L'emergere di nuovi importanti ambiti applicativi, tra i quali il Data Mining, l'analisi di dati sul Web, l'analisi genetica, la classificazione dei tessuti in ambito diagnostico e il riconoscimento delle immagini, hanno suscitato recentemente un crescente interesse intorno al problema della classificazione. Come si è brevemente anticipato nel paragrafo 2.4, l'analisi cluster, o analisi dei gruppi, consiste nell'identificazione di gruppi di osservazioni che siano coesive tra loro e separate dagli altri gruppi. Questo paragrafo si propone di illustrare in maggior dettaglio come i modelli mistura offrano un prezioso ambito metodologico per l'analisi dei gruppi.
Molte procedure di clustering si basano su metodologie intuitivamente ragionevoli, ma per lo più di natura euristica. Un'ampia famiglia di esse fa uso di classificazione agglomerativa gerarchica, mediante la quale, partendo da tanti gruppi quante le unità statistiche, ad ogni passo del processo si fondono due gruppi che ottimizzino un qualche criterio (p.es. la somma dei quadrati 'within group' di Ward 1963, oppure la distanza minima tra i gruppi, come nel single-link method). Una seconda famiglia di metodi di classificazione si basa invece sulla ricollocazione iterativa, in base alla quale si trasferisce una unità da un gruppo all'altro, fino all'ottimizzazione di un criterio globale. Pur rappresentando un campo fertile e ricco di ricerca, queste procedure non offrono un metodo sistematico per affrontare alcune questioni cruciali che emergono nell'analisi cluster, come quella della determinazione del numero dei gruppi, della scelta del metodo di classficazione e infine di come debbano essere trattati gli outliers. Inoltre le proprietà statistiche di questi metodi sono generalmente sconosciute, precludendo così la possibilità di svilupparne l'inferenza.
Alla fine degli anni 1990 (si veda Boch 1996, 1198a, 1998b, per una rassegna ben completa) emerse la possibilità di fondare l'analisi cluster su modelli probabilistici. Questo permise di ricavare, dall'osservazione dei dati, un criterio per la scelta del metodo di classificazione più adeguato, e inoltre generò nuovi metodi di classificazione. Si comprese che molti dei metodi euristici altro non erano che approssimazioni di metodi di stima basati su particolari modelli probabilistici (ad esempio, il k-means clustering con il criterio di Ward equivale alla stima di massima verosimiglianza per un modello multinormale a varianze vincolate).
I modelli mistura, proposti e studiati nel contesto dell'analisi cluster fin dagli anni '60 (Wolfe 1963, 1965, 1967, 1970; Edwards e Cavalli-Sforza 1965; Day 1969; Scott e Symons 1971; Duda e Hart 1973; Binder 1978), hanno dimostrato di poter offrire un approccio statistico completo e potente anche per le questioni rimaste aperte nelle precedenti metodologie di analisi dei gruppi (McLachlan e Basford 1988; Banfield e Raftery 1993; Cheeseman e Stutz 1995; Fraley e Raftery 1998). Se infatti nei modelli mistura ogni distribuzione di probabilità di una componente corrisponde ad un gruppo, il problema di determinare il numero di cluster e di scegliere un metodo di classificazione appropriato

ai dati può essere riformulato come un problema statistico di scelta del modello. In questo modo, modelli che differiscano per numero di componenti o per la scelta della distribuzione adottata in ogni componente possono essere comparate, ovvero valutate nel loro adattamento ai dati. Anche gli outliers possono essere trattati in modo conveniente, aggiungendo al modello una o più componenti per i dati estremi. Un decisivo lavoro di Fraley e Raftery del 2002 ha coniugato felicemente due approcci: l'agglomerazione gerarchica basata sulla verosimiglianza di classificazione (Murtagh e Raftery 1984; Banfield e Raftery 1993) e l'algoritmo EM per la stima di massima verosimiglianza del modelli mistura gaussiani, con covarianze parametrizzate mediante scomposizione spettrale (McLachlan e Basford 1988; Celeux e Govaert 1995). Questi due approcci sono complementari, in quanto il primo tende a produrre ragionevolmente buone partizioni anche in assenza di informazione iniziale sui gruppi, ovviando ad un noto punto critico dell'EM: l'inizializzazione (per via della molteplicità delle mode della curva di verosimiglianza), mentre l'EM tipicamente produce migliori partizioni dei dati quando è avviato in maniera oculata. Inizializzando l'EM con partizioni generate da agglomerazione gerarchica basata su modelli e usando il Criterio di Informazione Bayesiana (BIC) per determinare il numero dei gruppi presenti nei dati, Dasgupta e Raftery nel 1998 furono i primi a ottenere risultati molto incoraggianti in problemi assai difficili, quali il riconoscimento di campi minati e la classificazione di eventi sismici. Nel presente paragrafo ci occuperemo di illustrare la parametrizzazione mediante scomposizione spettrale per i modelli mistura gaussiani, poi vedremo in dettaglio come combinare inizializzazione, EM e BIC in una completa strategia di clustering. Faremo uso delle routines di Mclust in R che ci permetteranno di verificare la capacità di questa metodologia nella classificazione dei dati.

5.1.1 Due approcci per l'analisi di modelli mistura

Molti autori hanno considerato metodi di classificazione basati su modelli mistura gaussiani, in letteratura sono stati proposti due diversi approcci di massima verosimiglianza (CML). A grandi linee, possiamo dire che il 'mixture approach' ha come fine la massimizzazione della verosimiglianza rispetto ai parametri della mistura, mentre il 'classification approach' si pone come obiettivo la massimizzazione della verosimiglianza, sia sui parametri della mistura, che sulle etichette identificatrici delle componenti che originano i dati osservati.

L'approccio basato su modelli mistura

Come si è visto in dettaglio nel Capitolo 2, se ci riferiamo ad un modello mistura di normali, la logverosimiglianza è data da:

$$\mathcal{L}\left(\boldsymbol{\psi}|\mathbf{x}_1,\ldots,\mathbf{x}_N\right) = \sum_{n=1}^{N} \log \sum_{j=1}^{g} \alpha_j \phi_j(\mathbf{x}_n;\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j) \tag{5.1}$$

dove $\mathbf{x}$ rappresenta la matrice dei dati, g è il numero delle componenti nella mistura, α_j è la probabilità che una osservazione appartenga alla j-esima componente ($\alpha_j \geq 0; \sum_{j=1}^{g} \alpha_j = 1$), e

$$\phi_j(\mathbf{x}_n;\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j) = (2\pi)^{-\frac{p}{2}} |\boldsymbol{\Sigma}_j|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x}_n-\boldsymbol{\mu}_j)'\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n-\boldsymbol{\mu}_j)\right). \tag{5.2}$$

La stima di massima verosimiglianza è quel valore del vettore dei parametri che massimizza la verosimiglianza, ed è generalmente ottenuto usando l'algoritmo EM. Una partizione dei dati, in questo contesto, può essere ricavata direttamente dalla stima ML dei parametri, assegnando ogni unità $\mathbf{x}_n$ per $(n = 1,\ldots,N)$ alla componente che presenta maggiore probabilità a posteriori di averla generata.

L'approccio basato sulla classificazione

Nel caso di clustering gerarchico basato su un modello, invece, supponendo che i dati osservati provengano da una densità mistura e che il numero di osservazioni di ogni componente abbia distribuzione multinomiale di parametri N e $p_1, \ldots p_g$, la logverosimiglianza assume l'espressione:

$$\mathcal{L}(\boldsymbol{\psi}, \mathbf{z}_1, \ldots \mathbf{z}_N | \mathbf{x}_1, \ldots, \mathbf{x}_N) = \sum_{j=1}^{g} \sum_{\mathbf{x}_n \in P_j} \log\left(p_j \phi_{z_{nj}}(\mathbf{x}_n; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\right)$$

dove P_j indica il j-esimo cluster. La stima di massima verosimiglianza può essere ottenuta mediante la versione classificativa dell'algoritmo EM (anche detta CEM). A partire da una partizione iniziale P_0, l'algoritmo CEM consiste, al passo k, nel calcolo delle probabilità condizionate $\tau_j(\mathbf{x}_n)$ per $n = 1, \ldots, N$ e $1 \leq j \leq g$ secondo la (2.19) (passo E), poi la partizione viene rivalutata assegnando ciascun $\mathbf{x}_n$ alla componente che presenta il massimo valore $\tau_j(\mathbf{x}_n)$ per $j = 1, \ldots, g$ (passo C) e infine vengono calcolate le stime di massima verosimiglianza $\hat{\alpha_j}$, $\hat{\boldsymbol{\mu}_j}$ e $\hat{\boldsymbol{\Sigma}}_j$, in base ai cluster P_j per $j = 1, \ldots, g$ (passo M).
In entrambi questi modelli le componenti o cluster sono elissoidali e centrate sulle medie $\boldsymbol{\mu}_j$, mentre le covarianze $\boldsymbol{\Sigma}_j$ ne determinano tutte le altre caratteristiche geometriche.

5.1.2 Parametrizzazioni possibili nei modelli gaussiani

Nei modelli mistura per l'analisi cluster, fino agli anni 1990, si prendevano in considerazione solamente modelli con uguali matrici di covarianza nelle diverse componenti, ovvero

- $\boldsymbol{\Sigma}_1 = \ldots = \boldsymbol{\Sigma}_g = \sigma^2 \mathbf{I}$ con σ^2 ignoto
- $\boldsymbol{\Sigma}_1 = \ldots = \boldsymbol{\Sigma}_g = Diag(\sigma_1^2, \ldots, \sigma_g^2)$ con $(\sigma_1^2, \ldots, \sigma_g^2)$ ignote e dove $Diag(\sigma_1^2, \ldots, \sigma_g^2)$ indica la matrice diagonale composta dagli elementi $(\sigma_1^2, \ldots, \sigma_g^2)$ sulla diagonale
- $\boldsymbol{\Sigma}_1 = \ldots = \boldsymbol{\Sigma}_g = \boldsymbol{\Sigma}$ con $\boldsymbol{\Sigma}$ matrice simmetrica definita positiva ignota

oppure il modello, molto dispendioso in termini di parametrizzazione, in cui non si poneva nessuna restrizione sulle matrici di covarianza $\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_g$.
Banfield e Raftery (1993) proposero una nuova impostazione che consente modelli intermedi, nei quali, di volta in volta, solo alcune delle caratteristiche geometriche (orientazione, volume e/o forma) delle distribuzioni della mistura sono mantenute costanti. Come è ben noto, ogni matrice di covarianza può essere parametrizzata mediante scomposizione spettrale (eigenvalue decomposition) nella forma seguente

$$\boldsymbol{\Sigma}_j = \lambda_j \mathbf{D}_j \mathbf{A}_j \mathbf{D}_j^{'} \tag{5.3}$$

laddove $\lambda_j = |\boldsymbol{\Sigma}_j|^{\frac{1}{q}}$, $\mathbf{D}_j$ è la matrice ortogonale degli autovettori di Σ_j e $\mathbf{A}_j$ è una matrice diagonale, tale che $|\mathbf{A}_j| = 1$, con gli autovalori normalizzati di $\boldsymbol{\Sigma}_j$ sulla diagonale, posti in ordine decrescente. Quindi

- il parametro λ_j determina il volume del j-esimo cluster,
- $\mathbf{D}_j$ il suo orientamento, e
- $\mathbf{A}_j$ la sua forma.

L'idea introdotta da Banfield e Raftery fu quella di trattare λ_j, $\mathbf{A}_j$ e $\mathbf{D}_j$ come insiemi indipendenti di parametri. Permettendo che alcune - ma non tutte - queste quantità varino nelle diverse componenti, si ottengono parametrizzazioni parsimoniose e modelli di immediata interpretazione, particolarmente appropriati in molte situazioni di classificazione.
Ad esempio, se il più grande autovalore di $\mathbf{\Sigma}_j$ è assai maggiore degli altri, allora il j-esimo cluster è concentrato lungo una retta nello spazio q-dimensionale, e tale retta indica la componente principale della distribuzione relativa al j-esimo cluster. Analogamente, se i due maggiori autovalori sono dello stesso ordine di grandezza e dominano gli altri, allora il j-esimo cluster è molto prossimo ad un piano nello spazio q-dimensionale. In corrispondenza di autovalori di $\mathbf{\Sigma}_j$ della stessa grandezza, invece, il j-esimo cluster è approssimativamente sferico. Questo approccio non è altro che la acuta generalizzazione di un lavoro di Murtagh e Raftery (1984), nel quale si è presentata la classificazione mediante un modello mistura, con volumi e forme costanti nelle componenti, applicata al riconoscimento di caratteri e ad altre complicate situazioni di classificazione nelle quali è assai indicato l'uso di cluster assottigliati e altamente lineari, possibilmente anche sovrapposti (come il riconoscimento di spigoli nelle immagini, ecc).
E' bene notare, infine, che l'approccio basato sulla scomposizione spettrale comprende i tre modelli con matrici di covarianza uguali e il modello dispendioso senza vincoli sulle matrici come casi particolari.
Volendo ora descrivere i modelli che derivano dalla parametrizzazione (5.3), innanzittutto si osservi che nel caso unidimensionale vi sono solo due possibili casi: denoteremo con la sigla E il caso di componenti con uguale varianza, mentre con V denoteremo il caso di componenti eteroschedastiche. Nel caso multidimensionale, la prima famiglia di modelli si riferisce a componenti di forma sferica, ovvero $\mathbf{A}_j = \mathbf{I}$, dove $\mathbf{I}$ è la matrice identità. In questa parametrizzazione, due modelli sono in competizione: [$\lambda\mathbf{I}$] e [$\lambda_j\mathbf{I}$].
La seconda interessante famiglia di modelli si basa sull'assunzione che le matrici di covarianza $\mathbf{\Sigma}_j$ siano diagonali. Ciò significa, nella parametrizzazione (5.3), che le matrici di orientamento $\mathbf{D}_j$ sono matrici di permutazione. Poichè, in questo caso, le variazioni nell'orientamento non sono significative, si osserva che quest'ipotesi dà luogo a quattro modelli [$\lambda\mathbf{A}$], [$\lambda_j\mathbf{A}$], [$\lambda\mathbf{A}_j$] e [$\lambda_j\mathbf{A}_j$].
Nel caso elissoidale (il più costoso in termini parametrici) si potranno, per esempio, assumere volumi variabili (V), mantenere uguale la forma (E) e supporre ancora variabili gli orientamenti (V) nelle componenti, [da cui la sigla VEV], richiedendo che $\mathbf{A}_j = \mathbf{A}$ e (con $\mathbf{A}$, $\mathbf{D}_j$ e λ_j parametri da stimare a partire dai dati osservati). Analoghe considerazioni portano a descrivere gli altri modelli elissoidali.
La tabella 5.1 riassume le strutture di matrici di covarianze per i modelli ottenuti da questa parametrizzazione. Nelle prime due colonne si specifica la sigla e il tipo di modello, poi si indicano il tipo di distribuzione corrispondente e il vincolo eventualmente posto su volume, forma ed orientamento. Nella settima colonna si riporta il costo del modello in termini del numero di parametri da stimare, nella successiva colonna viene indicata la natura del passo M nelle procedure EM ed ECM per il modello e infine vi è riportata la reperibilità nelle routines di Mclust.

In questa impostazione, i passi E e C dell'algoritmo EM o di ECM rimangono invariati. Invece il passo M deve essere svolto in dettaglio, per entrambi gli algoritmi, per ciascuno dei casi considerati. Per unificare la presentazione, faremo uso di una matrice di classificazione $\mathbf{c} = (c_{nj}, n = 1, \ldots, N$ e $j = 1, \ldots, g)$ con $0 \leq c_{nj} \leq 1$ e $\sum_{j=1}^{g} c_{nj} = 1$, con il vincolo ulteriore che $c_{nj} \in \{0, 1\}$ nel caso in cui $\mathbf{c}$ definisca una partizione, come nell'approccio di classificazione. Con questa convenzione, in entrambe le metodologie (quella basata sui modelli mistura e quella basata sulla classificazione), il passo M consiste nel massimizzare in $\boldsymbol{\psi}$ la funzione:

$$\mathcal{L}\left(\boldsymbol{\psi}|\mathbf{x}_1, \ldots, \mathbf{x}_N\right) = \sum_{j=1}^{g}\sum_{n=1}^{N} c_{nj} \log\left(\alpha_j \phi_{z_{nj}}(\mathbf{x}_n; \boldsymbol{\mu}_j, \mathbf{\Sigma}_j)\right) \tag{5.4}$$

Tabella 5.1: *Parametrizzazione delle matrici di covarianza* $\mathbf{\Sigma}_j$

sigla	Modello	Distribuzione	Volume	Forma	Orientamento	# parametri	passo M	HC	EM
E		univariata	uguale			1	CF	•	•
V		univariata	variabile			g	CF	•	•
EII	$\lambda\mathbf{I}$	sferica	uguale	uguale	NA	$\alpha+1$	CF	•	•
VII	$\lambda_j\mathbf{I}$	sferica	variabile	uguale	NA	$\alpha+q$	CF	•	•
EII	$\lambda\mathbf{A}$	diagonale	uguale	uguale	assi coord	$\alpha+q$	CF		•
VEI	$\lambda_j\mathbf{A}$	diagonale	variabile	uguale	assi coord	$\alpha+q+g-1$	IP		•
EVI	$\lambda\mathbf{A}_j$	diagonale	uguale	variabile	assi coord	$\alpha+qg-g+1$	CF		•
VVI	$\lambda_j\mathbf{A}_j$	diagonale	variabile	variabile	assi coord	$\alpha+qg$	CF		•
EEE	$\lambda\mathbf{DAD}'$	elissoidale	uguale	uguale	uguale	$\alpha+\beta$	CF	•	•
VEE	$\lambda_j\mathbf{DAD}'$	elissoidale	variabile	uguale	uguale	$\alpha+\beta+g-1$	IP		
EVE	$\lambda\mathbf{DA}_j\mathbf{D}'$	elissoidale	uguale	variabile	uguale	$\alpha+\beta+(g-1)(q-1)$	IP		
VVE	$\lambda_j\mathbf{DA}_j\mathbf{D}'$	elissoidale	variabile	variabile	uguale	$\alpha+\beta+(g-1)q$	IP		
EEV	$\lambda\mathbf{D}_j\mathbf{AD}_j'$	elissoidale	uguale	uguale	variabile	$\alpha-g\beta+(g-1)q$	CF		•
VEV	$\lambda_j\mathbf{D}_j\mathbf{AD}_j'$	elissoidale	variabile	uguale	variabile	$\alpha-g\beta+(g-1)(q-1)$	IP		•
EVV	$\lambda\mathbf{D}_j\mathbf{A}_j\mathbf{D}_j'$	elissoidale	uguale	variabile	variabile	$\alpha-g\beta+(g-1)$	CF		
VVV	$\lambda_j\mathbf{D}_j\mathbf{A}_j\mathbf{D}_j'$	elissoidale	variabile	variabile	variabile	$\alpha-g\beta$	CF		•

Nel caso mixture $\alpha = gq+g-1$ *parametri, nel caso classification* $\alpha = gq$*, mentre* $\beta = (q(q+1)/2)$*;*
CF denota un passo M in forma chiusa, mentre IP indica che il passo M necessita di una procedura iterativa.
(• indica la disponibilità in Mclust per clustering gerarchico (HC) e/o EM per dati multidimensionali.)

per una data matrice **c** e assegnati i valori osservati $\mathbf{x}_1, \ldots, \mathbf{x}_N$. Quando si tratta di algoritmo EM, **c** definisce una 'fuzzy classification' e si ha che $c_{nj} = \tau_j(\mathbf{x}_n)$ per $1 \leq n \leq N$ e $1 \leq j \leq g$. Se invece si tratta dell'algoritmo CEM, **c** definisce una partizione e si ha $c_{nj} = 1$ se $\mathbf{x}_n \in P_j$ e 0 altrimenti. Quindi per entrambi gli approcci e per ognuno dei modelli considerati, le formule di aggiornamento delle proporzioni e delle medie delle componenti sono, per $1 \leq j \leq g$

$$p_j = \frac{n_j}{N};$$

$$\boldsymbol{\mu}_j = \bar{\mathbf{x}}_j = \frac{\sum_{n=1}^{N} c_{nj}\mathbf{x}_n}{n_j}$$

dove $n_j = \sum_{n=1}^{N} c_{nj}$. Inoltre, nel caso in cui **c** definisce una partizione, $n_j = \#P_j$. La matrice di scattering all'interno dei cluster, denotata con **S**, è definita da

$$\mathbf{S} = \sum_{j=1}^{g}\sum_{n=1}^{N} c_{nj}(\mathbf{x}_n - \bar{\mathbf{x}}_j)(\mathbf{x}_n - \bar{\mathbf{x}}_j)'$$

mentre la matrice di scattering $\mathbf{S}_j$ del j-esimo cluster (o fuzzy cluster) prende la forma, per $j = 1, \ldots, g$

$$\mathbf{S}_j = \sum_{n=1}^{N} c_{nj}(\mathbf{x}_n - \bar{\mathbf{x}}_j)(\mathbf{x}_n - \bar{\mathbf{x}}_j)'.$$

Le formule di aggiornamento delle matrici dipendono dal modello considerato. Nel seguito verranno presentati due casi, quello relativo al modello $\lambda_j\mathbf{DAD}'$ (di tipo iterativo) e quello relativo al modello

$\lambda \mathbf{D}_j \mathbf{A}_j \mathbf{D}_j'$ con soluzione in forma chiusa, a titolo di esempio (per maggiori dettagli e per l'analisi completa di tuti i casi previsti nella parametrizzazione basata sulla scomposizione spettrale, si rinvia a Celeux e Govaert (1995).
Modello $[\lambda_j \mathbf{DAD}']$.
In questa situazione, è conveniente usare la notazione $\mathbf{\Sigma} = \lambda_j \mathbf{C}$ con $\mathbf{C} = \mathbf{DAD}'$. Massimizzare (5.4) equivale a minimizzare

$$\mathcal{L}_1 (\lambda_1, \ldots, \lambda_g, \mathbf{C}) = \sum_{j=1}^{g} \frac{1}{\lambda_j} \text{tr} \, (\mathbf{S}_j \mathbf{C}^{-1}) + q \sum_{j=1}^{g} n_j \log(\lambda_j).$$

La minimizzazione di $\mathcal{L}_1$ si ottiene per via iterativa.

- Se si mantiene fissa la matrice $\mathbf{C}$, i valori λ_j che minimizzano $\mathcal{L}_1$ sono dati da

$$\lambda_j = \frac{\text{tr} \, (\mathbf{S}_j \mathbf{C}^{-1})}{q n_j}$$

  .

- Se si mantengono fissi i volumi λ_j, la matrice $\mathbf{C}$ che minimizza $\mathcal{L}_1$ minimizza anche tr $(\sum_{j=1}^{g} \frac{1}{\lambda_j} \mathbf{S}_j) \mathbf{C}^{-1}$ e quindi si può dimostrare che è pari a

$$\mathbf{C} = \frac{\sum_{j=1}^{g} \frac{1}{\lambda_j} \mathbf{S}_j}{|\sum_{j=1}^{g} \frac{1}{\lambda_j} \mathbf{S}_j|^{1/q}}$$

Modello$[\lambda \mathbf{D}_j \mathbf{A}_j \mathbf{D}_j']$.
In questo caso conviene usare la notazione $\mathbf{\Sigma}_j = \lambda \mathbf{C}_j$ con $\mathbf{C}_j = \mathbf{D}_j \mathbf{A}_j \mathbf{D}_j'$. Massimizzare l'equazione (5.4) equivale a minimizzare

$$\mathcal{L}_2 (\lambda, \mathbf{C}_1, \ldots, \mathbf{C}_g) = \frac{1}{\lambda} \sum_{j=1}^{g} \text{tr} \, (\mathbf{S}_j \mathbf{C}_j^{-1}) + Nq \log(\lambda).$$

Calcoli diretti mostrano che la soluzione di minimo è data da

$$\mathbf{C}_j = \frac{\mathbf{S}_j}{|\mathbf{S}_j|^{1/q}}$$

e da

$$\lambda = \frac{\sum_{j=1}^{g} |\mathbf{S}_j|^{1/q}}{N}$$

Come già accennato, nel seguito faremo uso di un pacchetto software in R, denominato Mclust, disegnato appositamente per ottenere la stima di modelli mistura di normali e per la classificazione di dati basata su modelli mistura. Esso offre una vasta gamma di funzioni che combinano la classificazione gerarchica basata su modelli, l'impiego dell'algoritmo EM per la stima dei parametri delle misture e il Criterio di Informazione Bayesiana (BIC) all'interno di strategie più complesse per la classificazione, la stima della densità e l'analisi discriminante. Fornisce anche ulteriori funzionalità per ottenere dati simulati dai modelli stessi e per visualizzare e rappresentare i modelli, con i relativi risultati di clustering e classificazione.
Una pagina web con i relativi link (compresa anche la licenza d'uso) può essere trovata all'indirizzo $http://www.stat.washigton.edu/mclust$.

La versione che useremo è la versione 3, disponibile come pacchetto contribuito (mclust) in linguaggio R e può essere ottenuto dal sito del progetto CRAN, ovvero
$http://cran.r-project.org$.
Seguendo le istruzioni per installare i pacchetti di R sulla vostra macchina, si digita poi
`> library(mclust)`
all'interno di una finestra R per utilizzare il software Mclust.

5.2 Analisi cluster mediante modelli mistura

Nella agglomerazione gerarchica, ogni passo di fusione corrisponde ad un certo numero di cluster e ad una specifica partizione dei dati. Una data partizione può essere rappresentata mediante variabili indicatori, che possono essere usate come probabilità condizionate in un passo M dell'algoritmo di EM per la stima dei parametri, inizializzando così l'algoritmo stesso. Questo procedimento, combinato con l'impiego del Fattore di Bayes (approssimato mediante il BIC) per la selezione del modello, permette di fornire una strategia completa di clustering, composta dalle seguenti fasi successive:

- si fissa un numero massimo M di cluster e un insieme di modelli mistura da considerare;
- si svolge l'agglomerazione gerarchica per ottenere una massimizzazione approssimata della verosimiglianza di classificazione per ogni modello, e se ne ottiene la corrispondente classificazione in gruppi (considerando da 1 ad M gruppi);
- si applica l'algoritmo di EM per ogni modello e per ogni numero di cluster da 2 a M, a partire dalla classificazione ottenuta mediante l'agglomerazione gerarchica;
- si calcola il BIC per il caso di un solo cluster per ogni modello scelto e per i modelli mistura a 2,3,...,M clusters con i parametri stimati mediante EM.

Valori massimi del BIC attestano forte evidenza per un modello e un certo numero di cluster.
In molte situazioni applicative, i modelli di misture gaussiane multivariate parametrizzate mediante scomposizione spettrale hanno mostrato di essere un buon insieme di modelli per la classificazione. Eseguendo l'agglomerazione gerarchica per uno solo dei modelli (p.es. quello a covarianze non vincolate) si usano poi le partizioni così individuate come valori di inizializzazione per l'algoritmo di EM per qualsiasi altra parametrizzazione. Cominciando da un semplice esempio (ma successivamente approfondiremo il discorso), si vedrà come Mclust fornisce funzionalità per l'analisi cluster, combinando clustering gerarchico basato su modelli (paragrafo 5.5), algoritmo EM per modelli mistura di normali (paragrafo 5.3.1) e BIC (paragrafo 5.5). Si consideri il dataset bidimensionale `faithful` (incluso nel pacchetto di R e rappresentato in Figura 5.1) che riporta il tempo intercorso tra le eruzioni e la durata delle eruzioni stesse (in minuti) dell'Old Faithful geyser nel Parco di Yellowstone. I seguenti comandi eseguono l'analisi cluster del dataset `faithful` e ne stampano i risultati:

```
> faithfulMclust <- Mclust(faithful)
> faithfulMclust
bestmodel: EEE with 3 components
```

In questo caso il modello migliore, ovvero il più adeguato ai dati e con il numero ottimale di componenti, risulta essere una mistura a 3 componenti omoschedastiche, denotato con il simbolo EEE (Equal volume, Equal shape, Equal orientation). Il risultato della clusterizzazione può essere visualizzato mediante:

```
> plot(faithfulMclust,data=faithful)
```

Questo comando produce quattro grafici che riassumono i risultati della procedura `Mclust` e sono

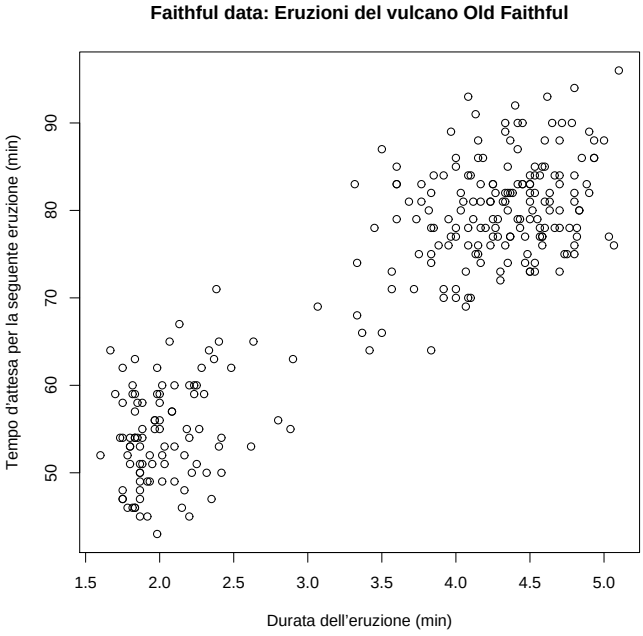


Figura 5.1: *Il dataset bi-dimensionale faithful.*

riportati in Figura 5.2. Il primo grafico mostra i valori del BIC per ciascuno dei modelli esaminati e per ogni scelta del numero di componenti. Il secondo grafico riguarda il processo di classificazione e sovrappone, sui dati classificati, le ellissi delle matrici di covarianza di ciascuna componente. Il terzo grafico riporta le curve di livello della densità della mistura. Il quarto, infine, rappresenta il grado di incertezza nella classificazione dei dati osservati. Possibili parametri di input per `Mclust` sono il numero delle componenti della mistura e la struttura di covarianza adottata. Se non si specifica nulla, Mclust compara i valori BIC ottenuti stimando i parametri per tutte le 10 strutture di covarianze previste in tabella 5.1 e facendo variare da 1 a 9 il numero di componenti. I risultati della procedura comprendono i valori dei parametri del modello di massimo valore BIC, seguiti dai risultati di classificazione e di incertezza. L'oggetto prodotto da Mclust è in realtà una lista, contenente un certo numero di campi che descrivono con molto dettaglio il modello selezionato. I nomi di tali campi possono essere richiesti mediante il comando `names` come segue:

```
> names(faithfulMclust)
 [1] "modelName"      "n"              "d"              "G"
 [5] "BIC"            "bic"            "loglik"         "parameters"
 [9] "z"              "classification" "uncertainty"
```

Una descrizione di ciascuno di essi è rintracciabile mediante `>help(Mclust)`.

5.2.1 mclustBIC e la funzione summary

Nel caso in cui si fosse interessati, per esempio, a esplorare i risultati relativi ad un modello diverso da quello con il massimo valore BIC e, in generale, per poter svolgere ulteriori analisi sullo stesso dataset, `Mclust` può essere eseguito più volte. Per evitare di eseguire calcoli già svolti e non più necessari (soprattutto nel caso di grandi dataset), è possibile separare l'analisi in diverse parti, utilizzando la funzione `mclustBIC`. Per il dataset `faithful`, la seguente sequenza di comandi produce gli stessi risultati di classificazione della chiamata a `Mclust`

```
> faithfulBIC<-mclustBIC(faithful)
> # the summary method allows specification of the models
```

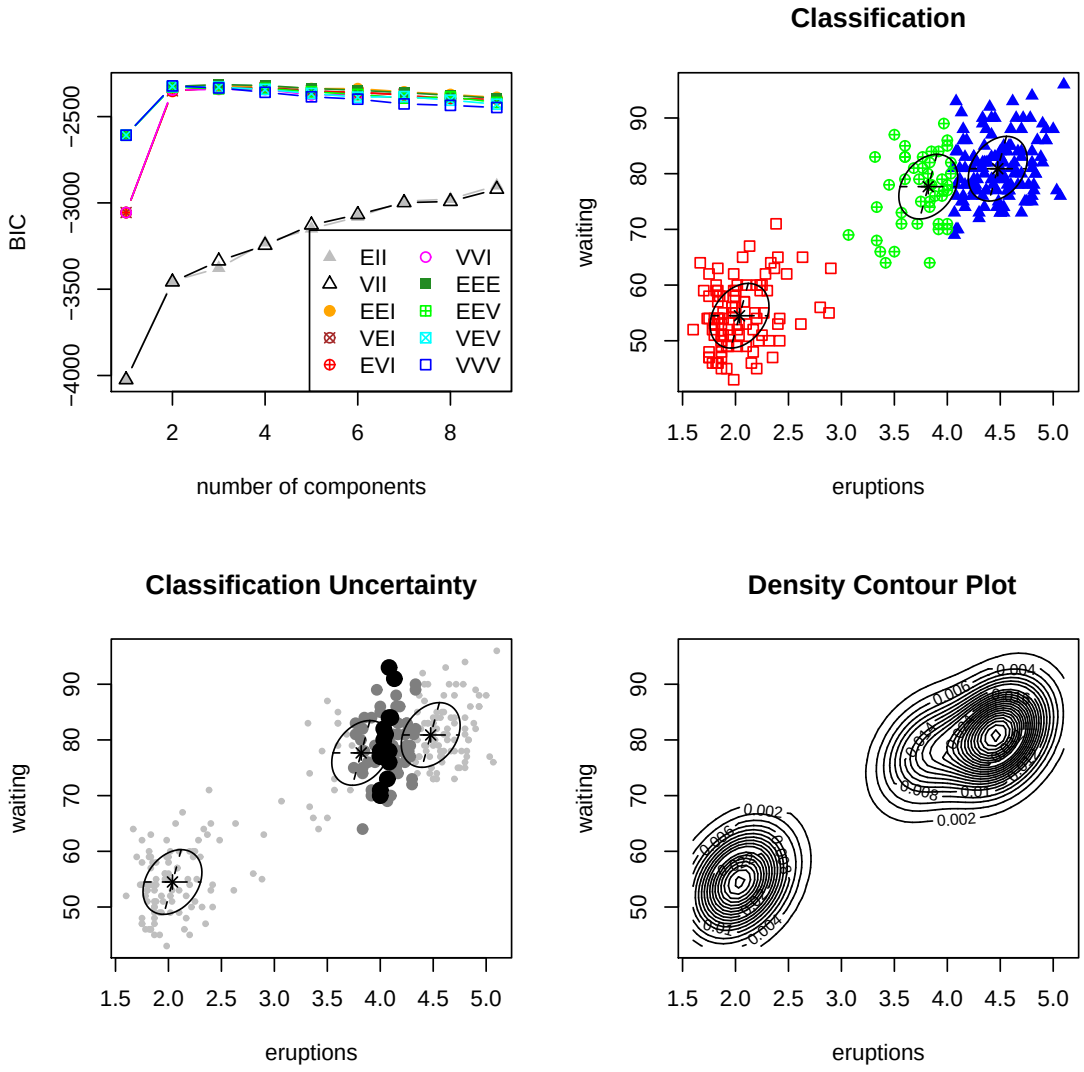


Figura 5.2: *Grafici associati alla funzione Mclust per il faithful dataset: grafico dei valori di BIC, risultato della classificazione, grado di incertezza, curve di livello della densità. Le ellissi sovraimpresse sui grafici corrispondono alle covarianze delle componenti.*

```
> # and number of clusters over which the best model is to be chosen
> # allowing models other than the maximum BIC model to be extracted and analyzed
> faithfulSummary<-summary(faithfulBIC,data=faithful)
> faithfulSummary

classification table:
  1   2   3
130  97  45
best BIC values:
    EEE,3     EEE,4     VVV,2
-2314.386 -2320.207 -2322.192
```

Anche se il metodo con cui vengono mostrati i risultati è diverso, l'oggetto `faithfulSummary` contiene la stessa lista di nomi di `faithfulMclust`, ad eccezione di `BIC`, la tabella dei valori BIC che è possibile visualizzare, per tutti i casi analizzati, mediante:

```
> faithfulBIC
 BIC:
        EII       VII       EEI       VEI       EVI       VVI       EEE
1 -4024.721 -4024.721 -3055.835 -3055.835 -3055.835 -3055.835 -2607.623
2 -3452.998 -3458.300 -2354.601 -2350.607 -2352.618 -2346.065 -2325.220
3 -3377.712 -3336.542 -2323.008 -2332.698 -2332.204 -2342.371 -2314.386
4 -3230.246 -3245.732 -2323.676 -2331.829 -2334.756 -2343.068 -2320.207
5 -3149.395 -3128.214 -2337.696 -2348.300 -2355.891 -2374.307 -2336.975
6 -3081.411 -3067.559 -2338.118 -2363.112 -2357.725 -2372.748 -2347.297
7 -2990.335 -2998.497 -2356.461 -2370.061 -2375.851 -2393.101 -2361.206
8 -2978.090 -2991.851 -2371.809        NA -2395.989        NA -2376.917
9 -2899.779 -2920.951 -2388.629        NA -2399.083        NA -2393.728

        EEV       VEV       VVV
1 -2607.623 -2607.623 -2607.623
2 -2329.116 -2325.416 -2322.192
3 -2338.986 -2329.352 -2333.894
4 -2336.750 -2342.472 -2359.216
5 -2356.225 -2366.193 -2385.288
6 -2371.732 -2387.445 -2398.974
7 -2392.963 -2384.183 -2426.488
8 -2385.815 -2404.950 -2434.990
9 -2418.305 -2428.351 -2447.286
```

I valori NA riguardano modelli per i quali non è stato possibile stimare i parametri (usando le inizializzazioni di default). Per dati multivariati, le inizializzazioni di default per tutti i modelli usano la classificazione che si ottiene mediante clustering gerarchico, basato su un modello senza vincoli. Per dati univariati, l'inizializzazione di default consiste nel dividere i dati in quantili per ricavarne medie e varianze in ogni gruppo. Il metodo `summary` per `mclustBIC` permette la specificazione del modello e del numero delle componenti tra cui cercare il miglior modello, inoltre è possibile anche estrarre e analizzare modelli diversi da quello che realizza il massimo valore di `BIC`. Il metodo `plot` per `mclustBIC` permette la specificazione dei modelli e del numero di componenti, permette anche di fornire argomenti alla funzione `legend` e scegliere i limiti sull'asse verticale del grafico. A titolo di esempio, il comando che segue permette di visualizzare il massimo valore di BIC con

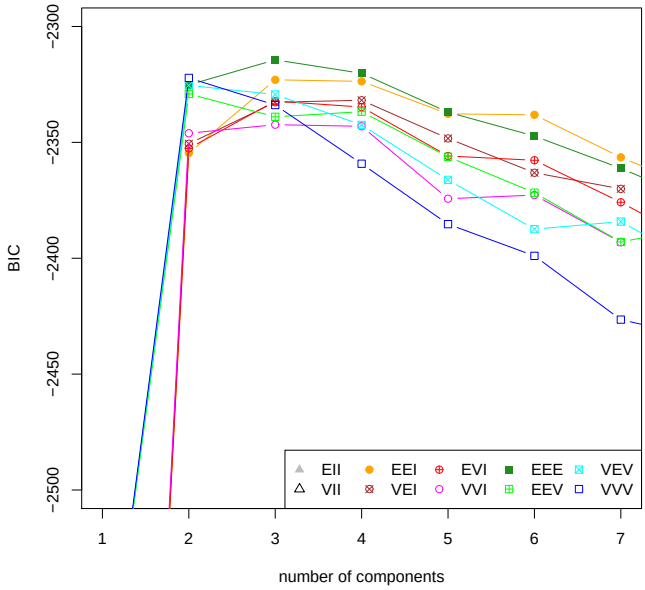


Figura 5.3: *Grafico dei valori BIC per il dataset faithful, con la modifica all'asse verticale che permette di vedere con maggior dettaglio il massimo valore.*

maggiore dettaglio, suggerendo al metodo di plot di scegliere l'intervallo [-2500, -2300] per i valori delle ordinate :

```
> plot(faithfulBIC, G = 1:7, ylim = c(-2500, -2300),
       legendArgs = list(x="bottomright", ncol = 5))
```

Il grafico che si ottiene è riportato in Figura 5.3.

5.2.2 Un esempio di cluster analysis estesa

Si consideri il dataset `wreath`, illustrato in Figura 5.4. Si tratta di 1000 osservazioni bivariate, simulate da un modello mistura con 14 componenti ben separate tra loro e per le quali le matrici di covarianza hanno uguale forma e volume, ma differiscono nell'orientamento. I valori BIC possono essere ottenuti mediante l'esecuzione di `mclustBIC`:

```
> data(wreath)
> wreathBIC<-mclustBIC(wreath)
> plot(wreathBIC, legendArgs = list(x="topleft"))
```

Facendo riferimento al grafico BIC (mostrato a sinistra in Figura 5.5), il massimo valore del BIC è assunto fuori dal range dei valori di default per il numero di componenti in `mclustBIC` (e anche `Mclust`). Più componenti possono essere considerate (per esempio, fino a 20) con un'analisi che non richieda di rivalutare i risultati già ottenuti:

```
> wreathBIC <-mclustBIC(wreath, G = 1:20, x=wreathBIC)
> plot(wreathBIC, G = 10:20, legendArgs = list(x="bottomleft"))
> summary(wreathBIC, wreath)

classification table:
 1  2  3  4  5  6  7  8  9 10 11 12 13 14
74 69 63 74 68 70 71 66 83 77 66 77 61 81
```

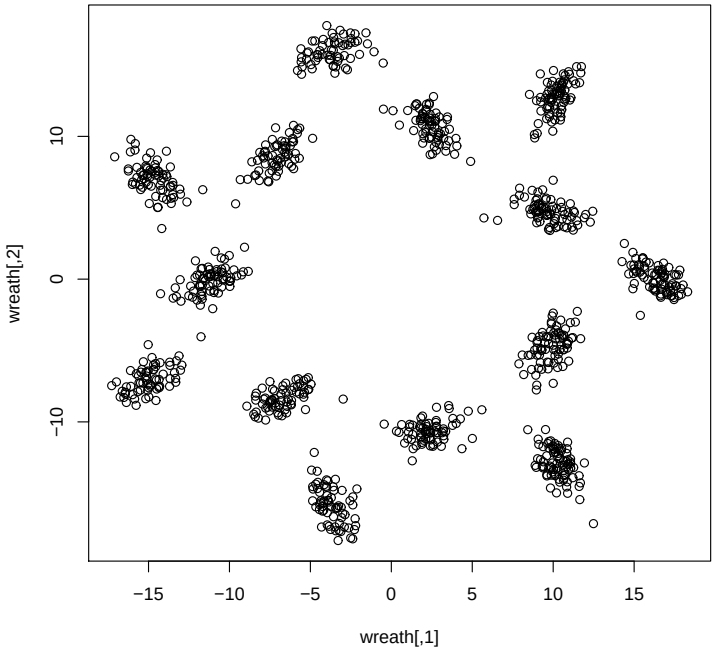


Figura 5.4: *Il dataset bi-dimensionale wreath, generato mediante 1000 osservazioni simulate da una mistura di normali con 14 componenti, tutte con matrici di covarianza con uguale volume e forma, ma orientamento variabile.*

```
best BIC values:
   EEV,14    EEV,15    EEV,16
-10902.77 -10925.54 -10953.10
```

Il nuovo grafico del BIC è riportato nella parte destra della Figura 5.5. Utilizzando la funzione `summary` per ottenere il miglior modello secondo il criterio BIC, si ottiene un modello EEV a 14 componenti, che corrisponde al metodo con cui i dati sono stati generati. In modo analogo a quanto visto prima, si può ottenere una rappresentazione completa dei risultati di classificazione, di incertezza e la densità ottenuta con i comandi:

```
> wreathmclust<-Mclust(wreath, G=1:20)
> plot(wreathmclust, data=wreath)
```

Nel caso in cui si volesse invece restringere l'insieme di modelli e/o il numero dei cluster da considerare, si può usare ancora il metodo summary:

```
> wreathSphericalModel<-summary(wreathBIC,data=wreath,modelNames=c("EEI","VII")
```

In questo caso si è scelto di concentrare l'attenzione solo su modelli a covarianze sferiche e se ne può ottenere il risultato mediante:

```
> wreathSphericalModel

classification table:
 1  2  3  4  5  6  7  8  9 10 11 12 13 14
75 69 63 74 68 70 71 65 83 77 66 77 61 81

best BIC values:
   EEI,14    EEI,15    EEI,16
```

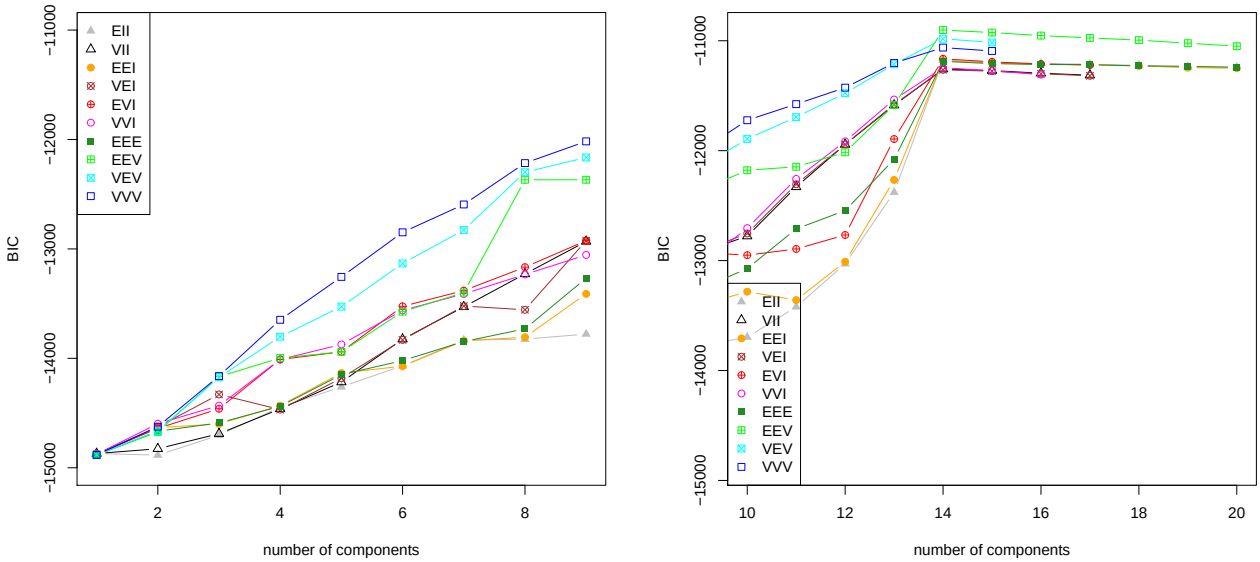


Figura 5.5: *Valori di BIC per il dataset wreath, considerando tutti i tipi di strutture di modelli. A sinistra: relativamente a modelli con fino a 9 componenti (come per default in mclustBIC e mclust). A destra: relativamente a modelli con componenti da 10 a 20. Si nota con evidenza un massimo in corrispondenza di 14 componenti (per tutti i modelli esaminati).*

```
-11181.84 -11202.56 -11213.52
```

5.2.3 Classificazione di datasets con rumore e outliers

La presenza di rumore all'interno di dati rilevati, così come di eventuali osservazioni estreme può essere spesso fronteggiata, in questo contesto metodologico, aggiungendo uno o più termini alla mistura, dedicati a rappresentare i dati 'non conformi'. In numerose applicazioni è stata impiegata con successo una mistura in cui una componente modellizza il rumore come un processo di Poisson omogeneo (Banfield e Raftery 1993; Dasgupta e Raftery 1998; Campbell *et al.* 1997, 1999). Il modello corrispondente è

$$\mathcal{L}_{mix}\left(\boldsymbol{\psi};\tau_0,\tau_1,\ldots,\tau_g|\mathbf{y}\right)=\prod_{n=1}^{N}\left(\frac{\tau_0}{V}+\sum_{j=1}^{g}\tau_j\phi_j(\mathbf{x}_n;\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)\right) \tag{5.5}$$

nel quale V è l'ipervolume della regione dei dati, $\tau_j \geq 0$ e $\sum_{j=1}^{g}\tau_j = 1$. Outliers isolati possono qualche volta essere trattati con il metodo 'iterated sampling' (e.g. Fayyad e Smyth 1996), mediante il quale punti di bassa probabilità sono rimossi dai cluster e il processo di classificazione/rimozione viene ripetuto finchè le rimanenti osservazioni hanno una densità relativamente alta. Un'alternativa assai interessante per il trattamento dei dati con rumore, è quella di utilizzare misture di distribuzioni t (Peel e Machlaclan 2000). Quando i dati contengono una parte importante di rumore, il metodo di classificazione basato sui modelli deve essere così modificato:

- si ottiene una categorizzazione iniziale di ogni osservazione come 'dato' oppure 'rumore'. Il 'Voronoi method' (Allard e Fraley 1997) e il metodo di 'nearest neighbor' (Byers e Raftery 1998) sono metodi di questo tipo;
- si applica il clustering gerarchico ai dati senza rumore;

- si applica EM, basato su modelli gaussiani cui si aggiungono una o più componenti per il rumore, all'intero dataset. I valori iniziali per z_{nj} si formano aggiungendo alle variabili indicatori che si ottengono come risultato della classificazione gerarchica una riga di 0s per ogni osservazione inizialmente classificata come rumore e una colonna di variabili indicatori contenenti il risultato del passo di identificazione del rumore (1 in caso di rumore e 0 altrimenti).

Mclust permette anche la classificazione di dati contenenti rumore, ovvero osservazioni estreme che non appartengono a nessun cluster. Per includere del rumore nel modello, è necessario fornire un suggerimento iniziale sulla natura delle osservazioni da considerare come outliers mediante la componente `noise` del parametro `initialization` per la procedura `Mclust` o `mclustBIC`.
Nell'esempio che segue si aggiunge rumore Poissoniano al dataset `faithful`. Una stima casuale iniziale è usata come rumore, a scopo illustrativo (anche se qui tutto funziona bene, non è raccomandabile come strategia generale, soprattutto per grandi datasets).

```
> # looks for the ranges of the two variables
> b <- apply(faithful,2,range)
> nNoise <-500
> # standard init for random num. generator (0: seed from the internal clock)
> set.seed(0)
> # here we generate the 500 random outliers
> poissonNoise<-apply(b,2,function(x,n) runif(n,min(x)-.1,max=max(x)+.1),n=nNoise)
> # original data joint with Noise data in the new faithfulNdata dataset
> faithfulNdata <- rbind(faithful, poissonNoise)
> set.seed(0)
> faithfulNoiseInit <- sample(c(TRUE,FALSE), size=nrow(faithful)+nNoise,
                                      replace=TRUE,prob=c(3,1))
> faithfulNbic <- mclustBIC(faithfulNdata,
                            initialization=list(noise=faithfulNoiseInit))
> faithfulNsummary <-summary(faithfulNbic, faithfulNdata)
> faithfulNsummary

classification table:
  0   1   2
532 141  99
best BIC values:
    EVI,2     VVI,2     EEI,2
7982.038 -7982.882 -7992.803
```

I risultati sono visualizzati in Figura 5.6. Il grafico di classificazione si ottiene mediante il seguente comando:

```
> mclust2Dplot(faithfulNdata, classification=faithfulNsummary$classification,
               parameters=faithfulNsummary$parameters)
```

mentre il grafico del BIC si genera eseguendo:

```
> plot(faithfulNbic,legendArgs=list(x='bottomleft',horiz=FALSE,ncol=5,cex=0.75)
```

5.2.4 Ulteriori considerazioni per l'analisi cluster

L'esito dell'analisi cluster può essere inficiato dall'inizializzazione di parametri quali il valore di tolleranza per la convergenza, anche se generalmente i valori di default sono adeguati allo scopo. E'

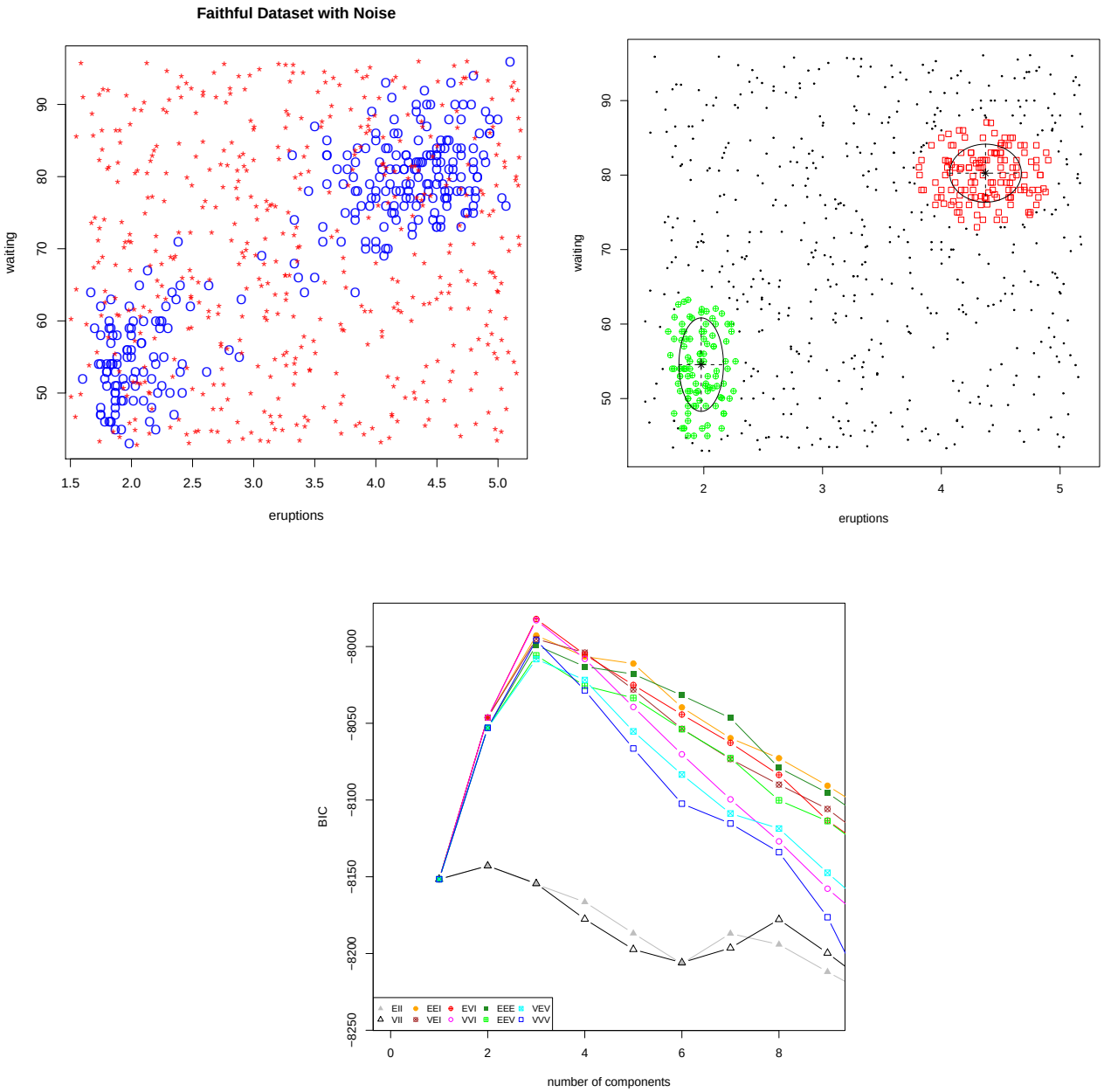


Figura 5.6: *Analisi cluster del Faithful dataset con rumore poissoniano. In alto a sinistra: le 272 osservazioni del Faithful dataset (o) con l'aggiunta di 500 dati di rumore (.). In alto a destra: classificazione ottenuta da Mclust, partendo dalla stima iniziale del rumore. In basso: Valori del BIC.*

anche possibile eseguire la clusterizzazione a partire da una stima iniziale dei parametri, da probabilità condizionate assegnate e da una classificazione diversa da quella generata dalla classificazione gerarchica. Le funzioni di stima delle misture (paragrafo 5.3.1) e di calcolo del BIC (paragrafo 5.5) possono essere utilizzate anche a questo scopo.
Infine, è importante tener conto di problemi numerici nell'analisi cluster. L'algoritmo EM non converge quando la covarianza di una componente diviene mal condizionata (matrice singolare). In queste situazioni può essere utile specificare una densità a priori. In generale l'algoritmo non può procedere se i cluster contengono troppo poche osservazioni, oppure se esse sono prossime ad essere collineari. La valutazione dei parametri può non concludersi anche quando una delle proporzioni delle componenti assume un valore prossimo a zero.. Le funzioni di EM in Mclust verificano ad ogni passo il buon condizionamento delle covarianze e producono una segnalazione di errore (a meno che non si specifichi di disattivare i messaggi di warning) quando le covarianze si avvicinano ad essere singolari, secondo quanto specificato da una soglia il cui valore di default è dato da `emControl()$eps`.

5.3 Algoritmo EM per modelli mistura di normali

Mclust fornisce metodi iterativi di EM (Expectation-Maximization) per la stima di massima verosimiglianza per modelli di misture gaussiane. Nei modelli considerati, una iterazione di EM consiste di un passo E, che valuta la matrice **z** tale che z_{nj} è una stima della probabilità condizionata che l'osservazione n appartenga al gruppo j a partire dalle stime correnti dei parametri, e di un passo M che calcola le nuove stime dei parametri, usando **z**.
Le funzioni Mclust `em` e `me` implementano l'algoritmo EM per misture gaussiane. Oltre ai dati e alla specificazione del tipo di modello, la funzione `em` (che inizia con un passo E) richiede in input i parametri (medie, covarianze e proporzioni delle componenti), mentre la funzione `me` (che inizia con un passo M) richiede le probabilità condizionate z. Entrambe forniscono la stima di massima verosimiglianza dei parametri del modello e di z.

5.3.1 Passi E e M isolati

Le funzioni `estep` e `mstep` realizzano un passo individuale di tipo E o M per il processo iterativo. Le probabilità condizionate **z** e la logverosimiglianza possono essere ottenute come valori di ritorno di `estep`, mentre `mstep` restituisce i valori aggiornati dei parametri ottenuti, a partire dal valore di **z** fornito alla procedura. Nei comandi seguenti si illustra l'applicazione di queste due funzioni relativamente al dataset `iris` (incluso in R). Questo dataset famoso (Fisher's o Anderson's Iris dataset) raccoglie le misure in centimetri di lunghezza e larghezza di sepali e petali, relativamente a 50 fiori di tre diverse specie di fiori: Iris setosa, Iris versicolor, e Iris virginica. La quinta colonna del dataset contiene l'indicazione della specie, e la Figura 5.7 ne rappresenta un subset bidimensionale.

```
> ms<-mstep(modelName="VVV", data= iris[,-5], z=unmap(iris[,5]))
> names(ms)
[1] "modelName"  "prior"      "n"          "d"          "G"
[6] "z"          "parameters"
>
> es<-estep(modelName="VVV", data= iris[,-5], parameters=ms$parameters)
> names(es)
[1] "modelName"  "n"          "d"          "G"          "z"
[6] "parameters" "loglik"
```

In quest'esempio, la stima iniziale di **z** per il passo M è la matrice delle variabili indicatori corrispondenti alla classificazione discreta (iris[,5]). La funzione `unmap` converte una classificazione

Figura 5.7: *Rappresentazione del dataset classificato Iris, solo nelle due componenti relative alle misure di lunghezza e larghezza dei petali.*

espressa in categorie, nella corrispondente classificazione mediante variabili indicatore. Mclust permette la specificazione di una densità a priori, per la quale l'algoritmo EM calcola una valutazione a posteriori.

5.3.2 Incertezza

Sottraendo da 1 la probabilità della componente più verosimile per ogni osservazione, si può ottenere una misura di incertezza nella classificazione, associata alle probabilità condizionate **z**:

```
> meVVViris <- me(modelName="VVV", data=iris[,-5], z=unmap(iris[,5]))
> uncer <- 1 - apply(meVVViris$z,1,max)
```

La funzione di R `quantile`, applicata al vettore di incertezza, sintetizza la qualità della classificazione ottenuta:

```
> quantile(uncer)
          0%          25%          50%          75%         100%
0.000000e+00 0.000000e+00 1.907041e-08 1.392060e-03 3.361880e-01
```

In questo caso l'indicazione che se ne ricava è che la maggioranza delle osservazioni è ben classificata. Si noti però che, quando i gruppi si sovrappongono, nelle regioni di overlapping vi sarà necessariamente un alto grado di incertezza nella classificazione. Quando si conosca la reale classificazione, un grafico che esprima il grado di incertezza delle osservazioni mal classificate può essere ottenuto con la funzione `uncerPlot` (si veda la Figura 5.8):

```
> uncerPlot(z=meVVViris$z, truth=iris[,5])
```

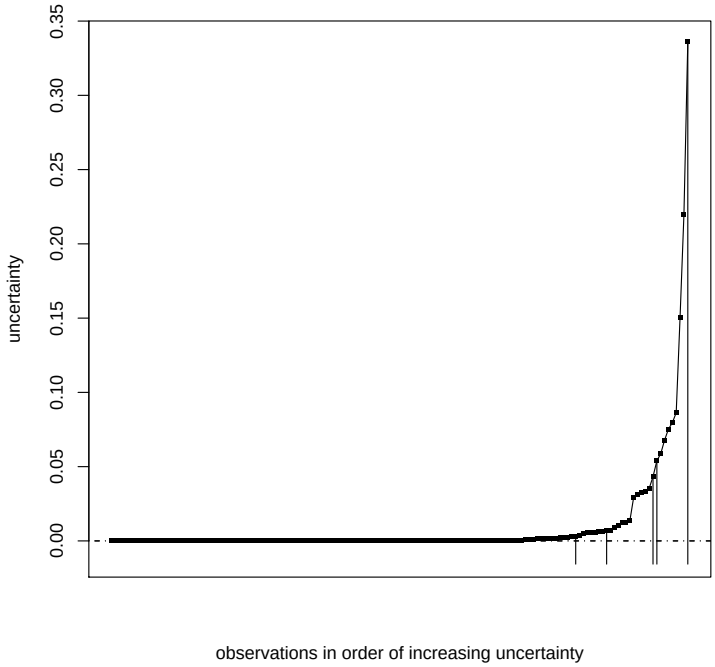


Figura 5.8: *Grafico che rappresenta la misura di incertezza nella classificazione per il modello mistura a 3 componenti gaussiane, per il dataset Iris, ottenuto mediante algoritmo EM senza vincoli. Le linee verticali indicano le osservazioni mal classificate.*

5.3.3 Parametri di controllo

Oltre ai parametri iniziali e alla densità a priori, altri parametri possono influenzare il risultato delle funzioni `em` e `me`. Tra di essi:

- `tol` valore di tolleranza per la convergenza.
  Il valore di default è `emControl()$tol=c(1.e-5,` $\sqrt{\epsilon_M}$, laddove $\sqrt{\epsilon_M}$ è la 'relative machine precision', che è pari a $2.220446e-16$ su macchine IEEE compliant. Il primo valore indica la tolleranza da applicare alla convergenza relativa nella logverosimiglianza dell'algoritmo EM, mentre il secondo valore è il parametro relativo di convergenza da utilizzare per i modelli che prevedono un passo M iterativo (VEI, VEV)

- `eps` valore di tolleranza riguardante il malcondizionamento (ovvero l'approssimarsi alla singolarità delle matrici di covarianza).
  Il valore di default è `emControl()$eps`, posto pari a $\sqrt{\epsilon_M}$.

La funzione `emControl` in Mclust permette di settare questi parametri e di modificarne i valori di default. Anche se questi valori sono in un certo senso schermati, in certi casi possono avere un significativo effetto sui risultati e quindi conviene considerarli attentamente.

5.4 Criterio di Informazione Bayesiana

La funzione `bic` consente l'applicazione del Criterio di Informazione Bayesiana (BIC), dati il valore di massimo della logverosimiglianza per il modello, la dimensione dei dati e il numero di componenti del modello. Come si è visto in dettaglio nel paragrafo 3.4.2, il valore BIC è il valore della massima

verosimiglianza, cui si sottrae un termine di penalizzazione per il numero di parametri nel modello. Esso permette comparazioni fra modelli con differenti parametrizzazioni e/o differente numero di componenti. In generale, a maggiori valori di BIC corrisponde una maggior evidenza della bontà del modello e una indicazione per la scelta del numero di componenti. Per ottenere la valutazione del BIC per il modello a tre componenti senza vincoli sulla varianza, si applica la funzione `bic`:

```
> bic(modelName="VVV",loglik=meVVViris$loglik,n=row(iris),d=ncol(iris[,-5]),G=3)
```

5.5 Clustering gerarchico basato su modelli

L'agglomerazione gerarchica basata su modelli è una metodologia per valutare un massimo approssimato della verosimiglianza di classificazione

$$\mathcal{L}_{cl}\left(\boldsymbol{\psi}, l_1, \ldots, l_N|\mathbf{x}\right) = \prod_{n=1}^{N} \phi_{l_n}(\mathbf{x}_n; \boldsymbol{\mu}_{l_n}, \boldsymbol{\Sigma}_{l_n}) \tag{5.6}$$

nella quale le etichette l_n indicano una unica classificazione per ogni osservazione, ovvero $l_n = j$ se $\mathbf{x}_n$ appartiene al j-esimo cluster. Nella verosimiglianza del modello mistura, ogni componente è ponderata per la probabilità che una osservazione vi appartenga. La presenza di etichette di cluster nella verosimiglianza di classificazione (5.6) introduce un aspetto combinatorio che rende impraticabile la massimizzazione esatta. L'agglomerazione gerarchica basata su un modello procede mediante successiva fusione di coppie di cluster, quando tale fusione corrisponde al massimo incremento nella verosimiglianza di classificazione, comparata alla fusione di tutte le altre possibili coppie. In assenza di informazione iniziale riguardante i gruppi, la procedura inizia trattando ogni osservazione come un cluster atomico. Quando il modello probabilistico in (5.6) è una mistura gaussiana con covarianze sferiche del tipo $\lambda\mathbf{I}$, il criterio di selezione è quello di Ward (1963), basato sulla somma dei quadrati. Altri criteri euristici di clustering, come il 'single link' (nearest neighbor), il 'complete link' (farthest neighbor), e l"average link' non paiono essere associati a nessun modello probabilistico noto, anche se questo è tuttora un ambito di ricerca. Algoritmi numerici efficienti per il clustering gerarchico basato sulla verosimiglianza di classificazione e su modelli mistura gaussiani sono discussi in Fraley (1998)

La funzione `hc` implementa l'agglomerazione gerarchica model-based, mediante metodi veloci basati sulla verosimiglianza di classificazione per normali multivariate. La funzione `hclass` ne determina la risultante classificazione. La Figura 5.9 presenta la matrice di diagrammi a dispersione (scatterplot) dell'Iris dataset, in cui le tre specie sono rappresentate con diversi simboli, ed è generata dal comando:

```
> clPairs(data=iris[,-5], classification=iris[,5])
```

Per applicare all'Iris dataset l'algoritmo di clustering gerarchico senza vincoli sulle covarianze (VVV) basta eseguire:

```
> hcVVViris <-hc(modelName="VVV", data=iris[,-5])
```

mentre la classificazione prodotta da `hc` per diversi valori del numero di cluster può essere ottenuta mediante `hclass`, qui di seguito applicata ad una classificazione per 2 e 3 clusters:

```
> cl <- hclass(hcVVViris, 2:3)
> clPairs(data=iris[,-5], classification = cl[,"2"])
> clPairs(data=iris[,-5], classification = cl[,"3"])
```

Nel prossimo paragrafo si presenteranno altre opzioni per la visualizzazione dei risultati di clustering e classificazione. La classificazione in 3 gruppi può essere comparata con quella reale, relativa alle tre specie da cui provengono i dati e memorizzata nella quinta colonna del dataset Iris, usando la

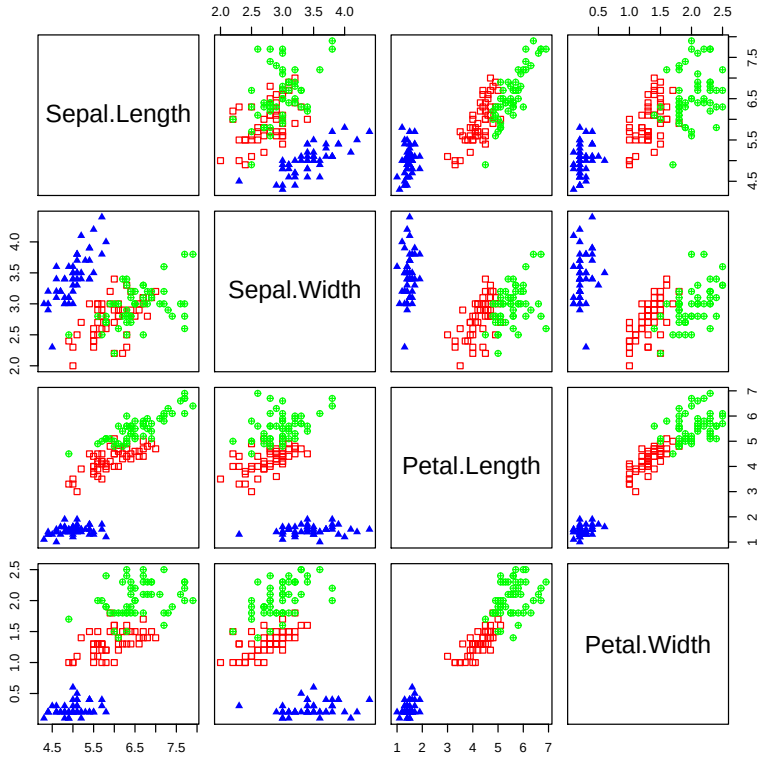


Figura 5.9: *Scatterplot del dataset Iris, con classificazione nelle tre specie.*

funzione `classError`:

```
> classError(cl[,"3"], truth=iris[,5]))
$misclassified)
 [1] 102 107 114 115 120 122 124 127 128 134 139 143 147 150)
$errorRate
[1] 0.09333333)
```

La funzione `hc` inizia per default attribuendo ogni osservazione ad un cluster per sè, e continua il processo finchè le osservazioni sono tutte riunite in un solo cluster. Gli argomenti `partition` e `minclus` possono essere utilizzati per inizializzare il processo con una partizione non banale prescelta, e per terminare l'esecuzione prima di raggiungere lo stadio finale di fusione. La funzione `hc` per clustering gerarchico model based basato sul modello senza vincoli è anche utilizzata dalle funzioni di clustering `McLust` e `mclustBIC` per ottenere i valori iniziali di default per i parametri.

5.6 Rappresentazioni grafiche per dati multidimensionali

Una volta ottenuti i valori dei parametri per un modello mistura, è possibile visualizzare le medie e le deviazioni standard delle componenti nelle proiezioni dei dati. Nel caso bidimensionale, si possono anche raffigurare la densità e le superfici di incertezza.

5.6.1 Grafici per dati bidimensionali

Per visualizzare la classificazione, l'incertezza e/o gli errori di classificazione per i modelli Mclust di dati bidimensionali, può essere usata la funzione `mclust2Dplot`. Nell'esempio seguente, essa è usata per il faithful dataset in Figura 5.1, prima con l'opzione di classificazione, poi con l'opzione di

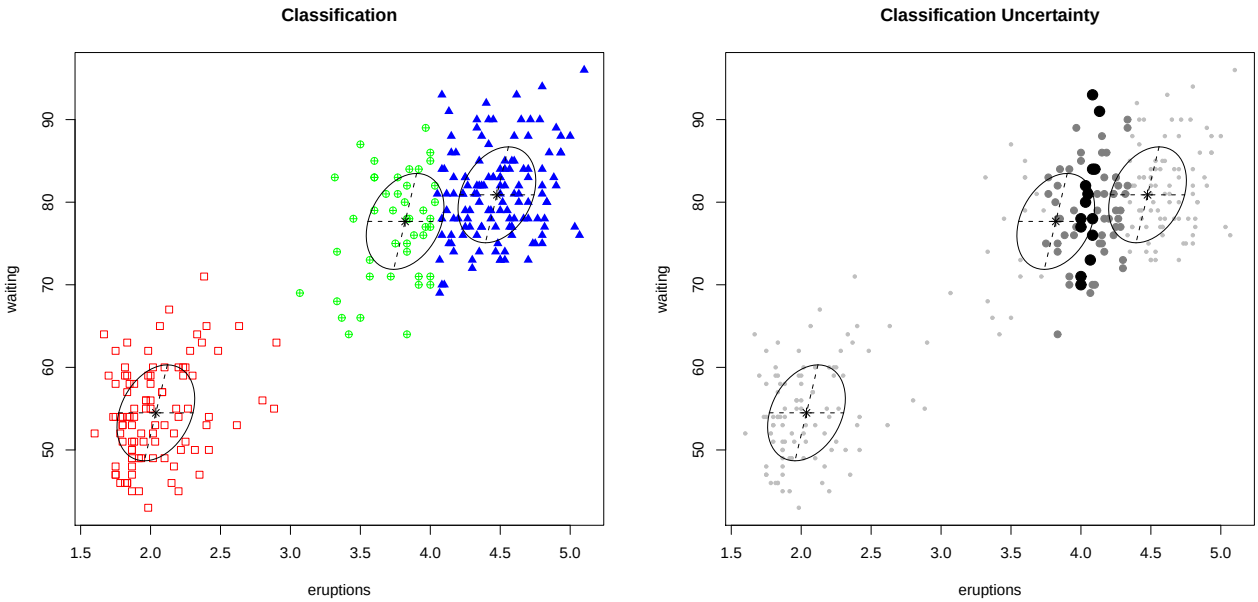


Figura 5.10: *Grafici della classificazione (a sinistra) e dell'incertezza (a destra) creati dalla funzione mclust2Dplot per il moello Mclust del faithful dataset. Nel grafico di classificazione, ogni osservazione è indicata con un simbolo differente, a seconda della componente in cui è stata classificata. Nel grafico di incertezza, i punti pieni in nero corrispondono al 95% quantile di incertezza, quelli di minore ampiezza e di colore grigio corrispondono ai quantili compresi nel range 75-95%, mentre i punti più piccoli sono relativi ai primi tre quartili dei valori di incertezza.*

incertezza:

```
> faithfulMclust <- Mclust(faithful)
> mclust2Dplot(data=faithful, what="classification", identify=TRUE,
                parameters=faithfulMclust$parameters,z=faithfulMclust$z)
> mclust2Dplot(data=faithful, what="uncertainty", identify=TRUE,
                parameters=faithfulMclust$parameters,z=faithfulMclust$z)
```

In Figura 5.10 sono riportati i grafici generati da `mclust2Dplot`. La funzione `surfacePlot` può essere usata per visualizzare la densità e l'incertezza per i modelli Mclust di dati bidimensionali, sempre specificando l'opzione `what`. La vediamo in azione sul Faithful dataset:

```
> faithfulMclust <- Mclust(faithful)
> surfacePlot(data=faithful, what="density", type="contour",
              parameters=faithfulMclust$parameters,transformation="sqrt")
> surfacePlot(data=faithful, what="uncertainty", type="image",
              parameters=faithfulMclust$parameters,transformation="log")
```

mentre la Figura 5.11 ne riporta il risultato.

5.6.2 Grafici per dati multidimensionali

Proiezioni delle coordinate

La funzione `coordProj` permette di visualizzare le proiezioni dei risultati di Mclust su due coordinate specificate come parametri. Si consideri un modello a 3 componenti per l'`iris` dataset e si supponga di essere interessati a proiettare i dati sulla seconda e quarta coordinata, indicata dalla clausola `dimens=c(2,4)`:

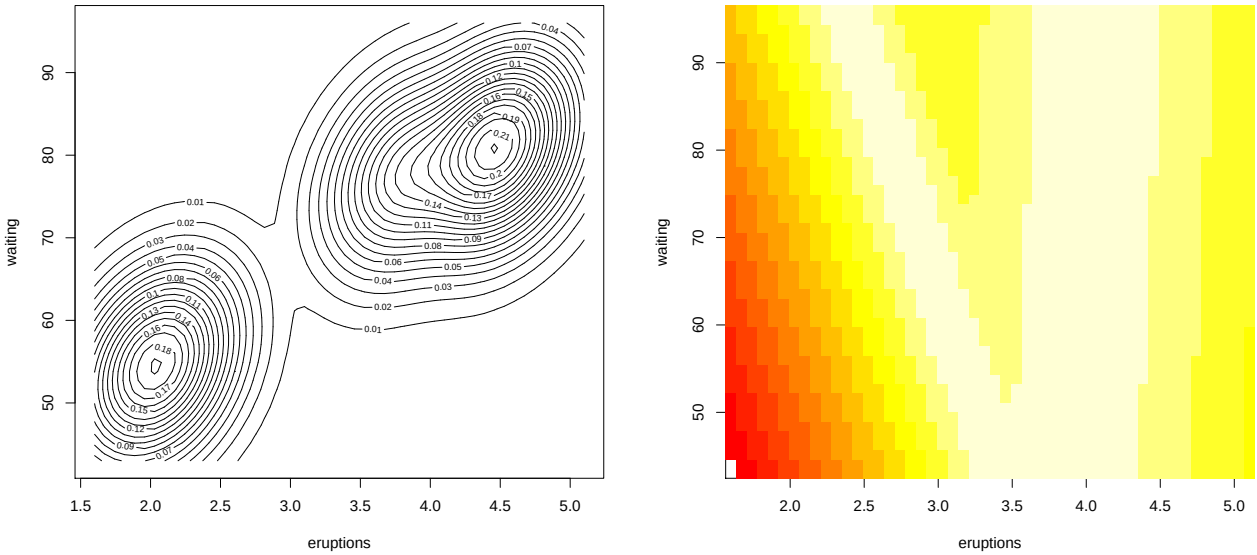


Figura 5.11: *Grafici della densità (per curve di livello, a sinistra) e dell'incertezza (per aree, a destra) creati dalla funzione mclust2Dplot per il modello Mclust del faithful dataset.*

```
> coordProj(data=iris[,-5], dimens=c(2,4), what="classification",
            parameters=irisSummary3$parameters, z=irisSummary3$z)
> coordProj(data=iris[,-5], dimens=c(2,4), what="uncertainty",
            parameters=irisSummary3$parameters, z=irisSummary3$z)
> coordProj(data=iris[,-5], dimens=c(2,4), what="errors",
            parameters=irisSummary3$parameters,z=irisSummary3$z,truth=iris[,5])
```

Il risultato di queste chiamate alla funzione `coordProj` è riportato in Figura 5.12

Proiezioni casuali delle coordinate

La funzione `randProj` permette le medesime funzionalità di `coordProj` su una coppia di coordinate scelte casualmente:

```
> randProj(data=iris[,-5], seed=43, what="classification",
           parameters=irisSummary3$paramaters, z=irisSummary3$z)
> randProj(data=iris[,-5], seed=79, what="classification",
           parameters=irisSummary3$paramaters, z=irisSummary3$z)
> randProj(data=iris[,-5], seed=201, what="classification",
           parameters=irisSummary3$paramaters, z=irisSummary3$z)
```

e il risultato è riportato in Figura 5.13

5.7 Stima di densità

Le capacità di classificazione di Mclust possono anche essere interpretate come una strategia generale per la stima di densità di dati multivariati. Dopo aver applicato le funzioni di clustering allo scopo di adattare il modello ai dati, la funzione `dens` può essere usata per ottenere la densità di un punto relativamente al modello. Come esempio, si userà ancora il dataset `faithful`. Dapprima, per ottenere il modello, si ricorre a `Mclust` (oppure a `mclustBIC` e `summary`) come si era fatto antecedentemente:

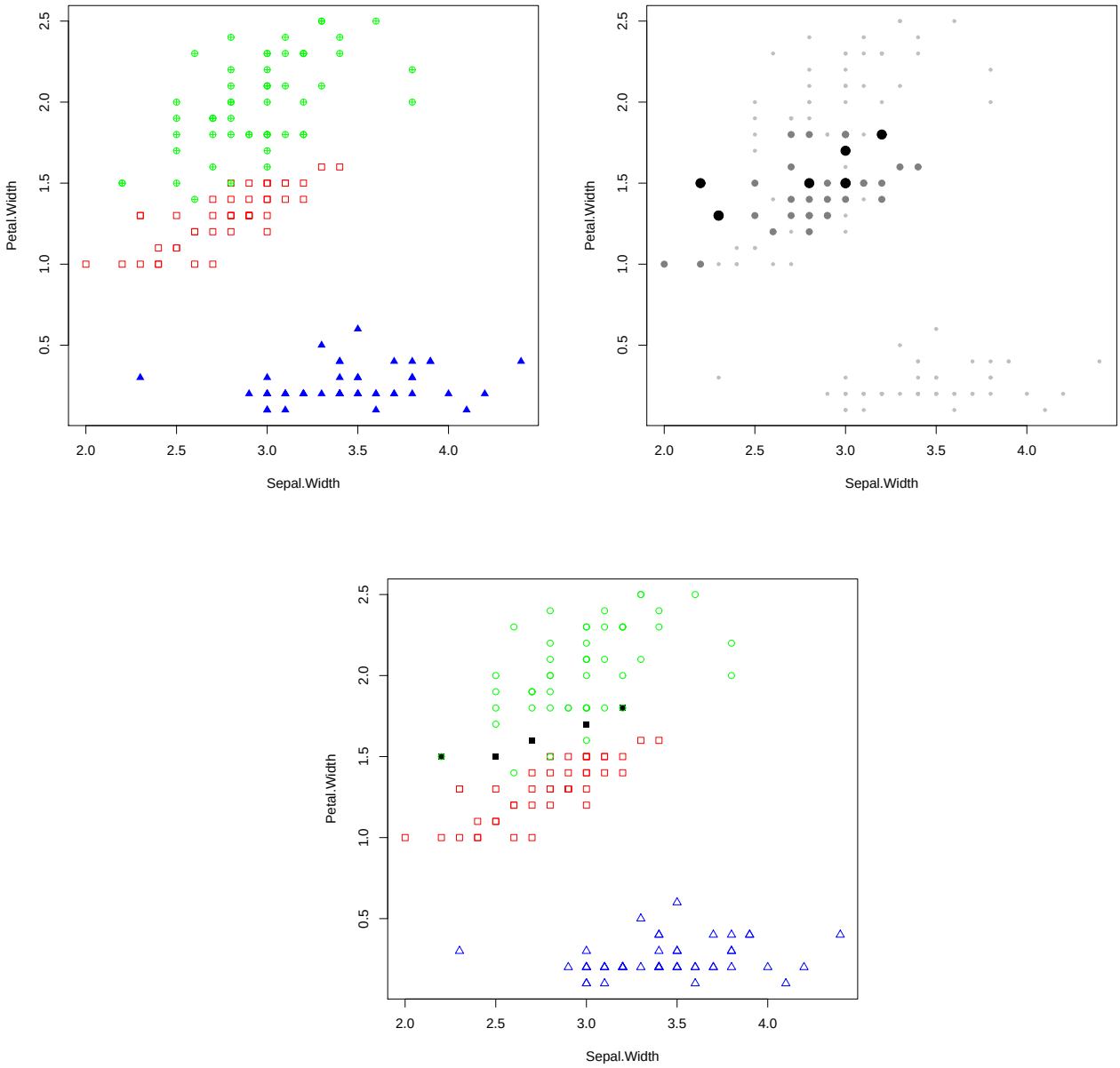


Figura 5.12: *Proiezione dell'iris dataset creata con coordProj, sul piano individuato dalle coordinate "larghezza dei petali" e "larghezza dei sepali". Il primo grafico riporta la classificazione operata dal modello stimato a 3 componenti (sopra) cui sono associati il grafico di incertezza e il grafico degli errori di classificazione (osservazioni in nero).*

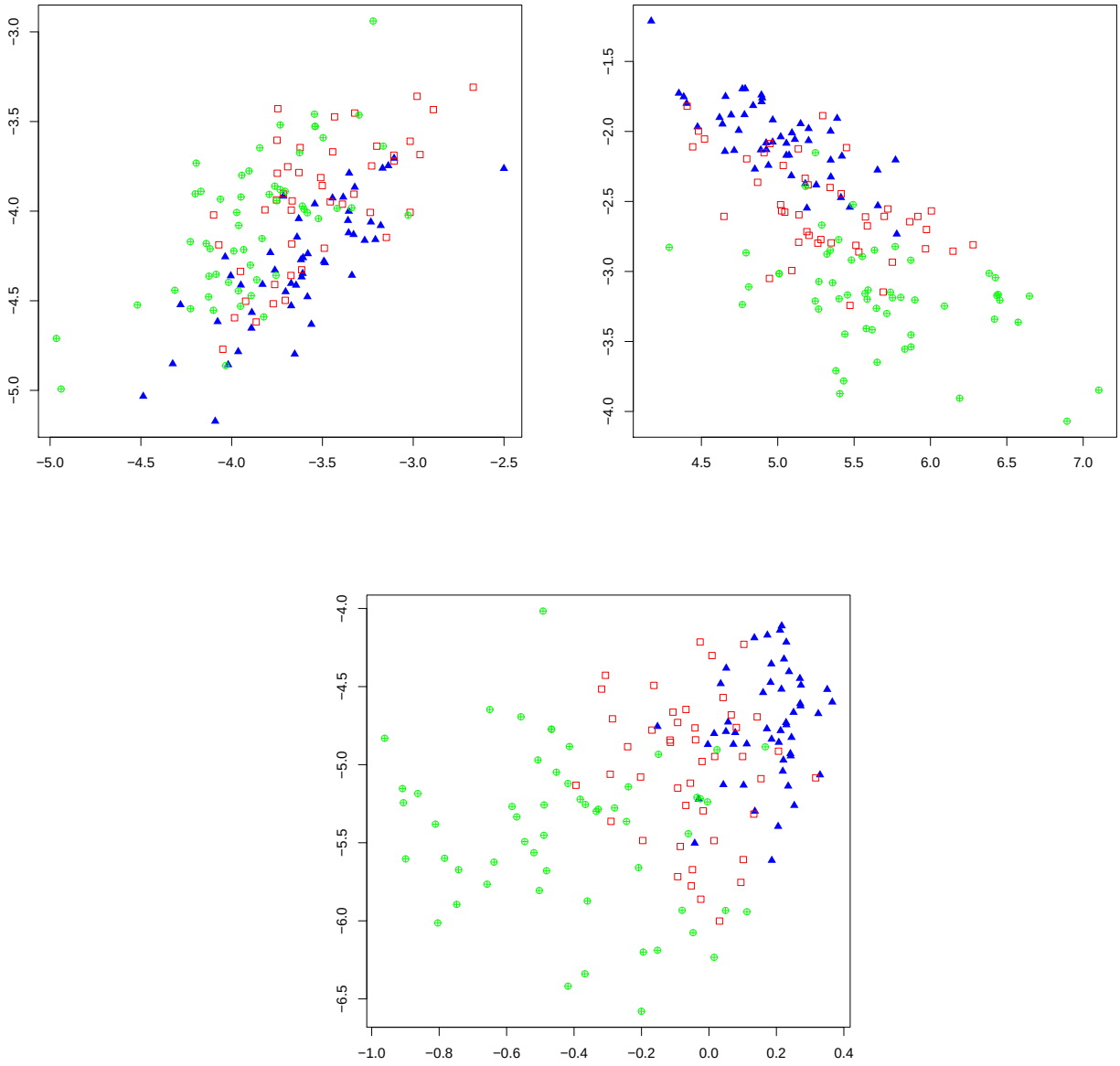


Figura 5.13: *Proiezioni dell'iris dataset create con randProj, sul piano individuato da coordinate scelte randomicamente, con tre diversi scelte del seme. I tre grafici riguardano la classificazione operata dal modello stimato a 3 componenti.*

```
> faithfulMclust <-Mclust(faithful)
```
Conviene disegnare la densità su una griglia bidimensionale di valori, così `grid1` forma una griglia unidimensionale di dimensione specificata nella clusola size sul range di valori indicato nella clusola range; mentre `grid2` forma una griglia bi-dimensionale a partire da due sequenze di valori.
```
> apply(faithful,2,range)
     eruptions waiting
[1,]       1.6      43
[2,]       5.1      96
> x<-grid1(100, range=range(faithful$eruption))
> y<-grid1(100, range=range(faithful$waiting))
> xy <- grid2(x,y)
> xyDens <-dens(modelName=faithfulMclust$modelName, data=xy,
                parameters=faithfulMclust$parameters)
> xyDens <- matrix(xyDens, nrow=length(x), ncol=length(y))
```
Poichè il dataset Faithful è bidimensionale, si possono sfruttare le funzioni `persp`, `contour` e `image` di S-Plus per visualizzare la densità ottenuta:
```
> par(pty="s")
> Z <-log(xyDens)
> persp(x=x, y=y, z=Z, box=FALSE)
> contour(x=x, y=y, z=Z, nlevels=10)
> image(x=x, y=y, z=Z)
```
e il risultato è riportato in Figura 5.14.

5.8 Simulazione da densità mistura

Assegnati i parametri per un modello mistura, è possibile ottenere dati simulati dal modello stesso, per scopi di valutazione e verifica. La funzione `sim` genera dati simulati da un modello mistura ottenuto mediante l'uso delle funzioni di Mclust. Oltre al tipo di modello, `sim` può ricevere un seme allo scopo di garantire la riproducibilità. Come esempio, con le istruzioni che seguono, si simulano due diversi dataset (della stessa ampiezza del dataset faithful), a partire dal modello prodotto da Mclust per lo stesso dataset reale.
```
> faithfulMclust <-Mclust(faithful)
> sim0 <- sim(modelName=faithfulMclust$modelName,
+             parameters=faithfulMclust$parameters,
+             n=nrow(faithful), seed=0)
> sim1 <- sim(modelName=faithfulMclust$modelName,
+             parameters=faithfulMclust$parameters,
+             n=nrow(faithful), seed=1)
```
I risultati possono essere visualizzati nel modo seguente:
```
> xlim<-range(c(faithful[,1], sim0[,2],sim1[,2]))
> ylim<-range(c(faithful[,2], sim0[,3],sim1[,3]))
> mclust2Dplot(data=faithful, parameters=faithfulMclust$parameters,
+              classification=faithfulMclust$classification, xlim=xlim, ylim=ylim)
> mclust2Dplot(data=sim0[,-1], parameters=faithfulMclust$parameters,
+              classification=sim0[,1], xlim=xlim, ylim=ylim)
> mclust2Dplot(data=sim1[,-1], parameters=faithfulMclust$parameters,
+              classification=sim1[,1], xlim=xlim, ylim=ylim)
```
Essi producono i grafici riportati in Figura 5.15.

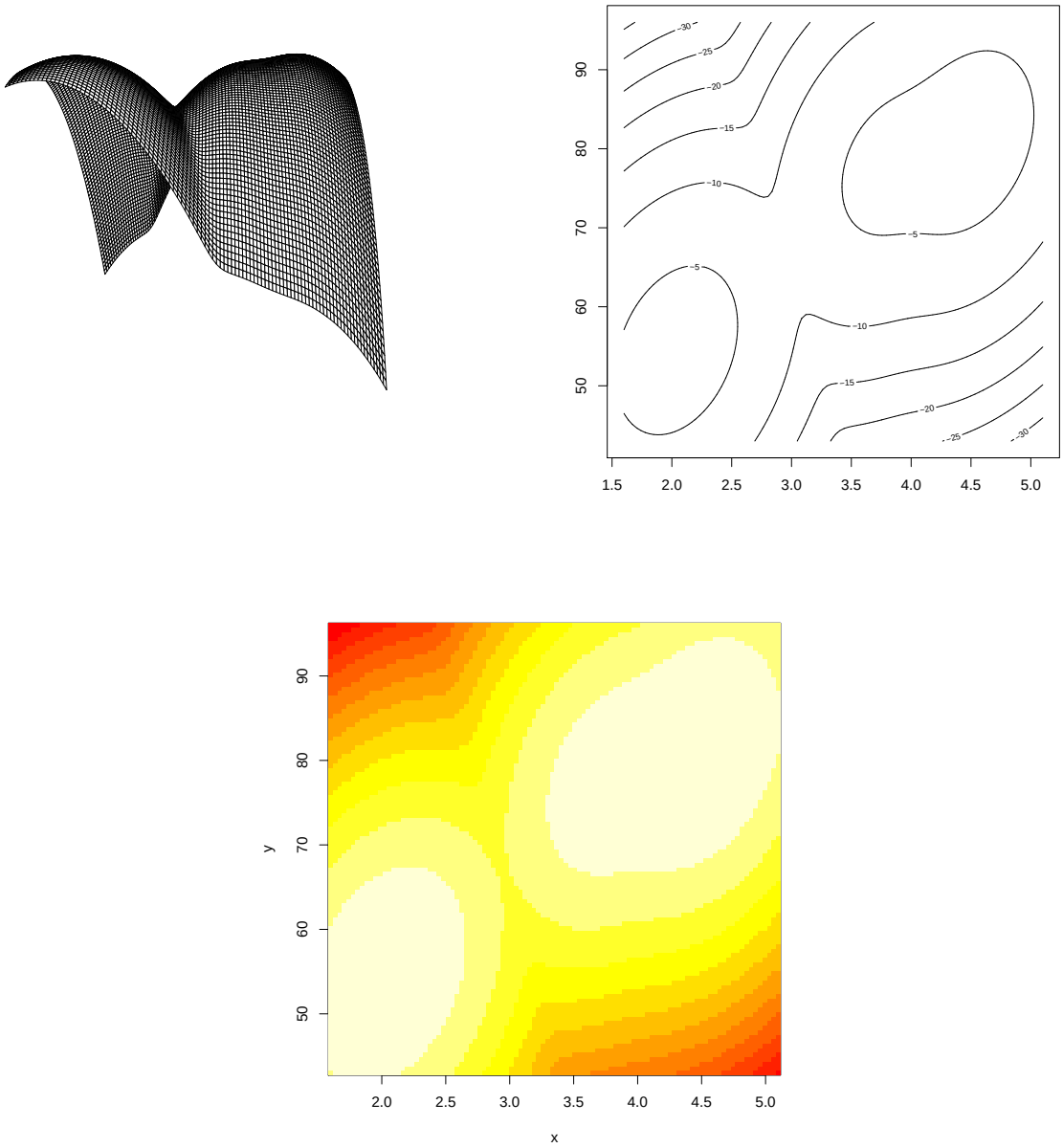


Figura 5.14: *Prospettiva, curve di livello e Image plot della stima di densità per il dataset Faithful ottenuta mediante Mclust.*

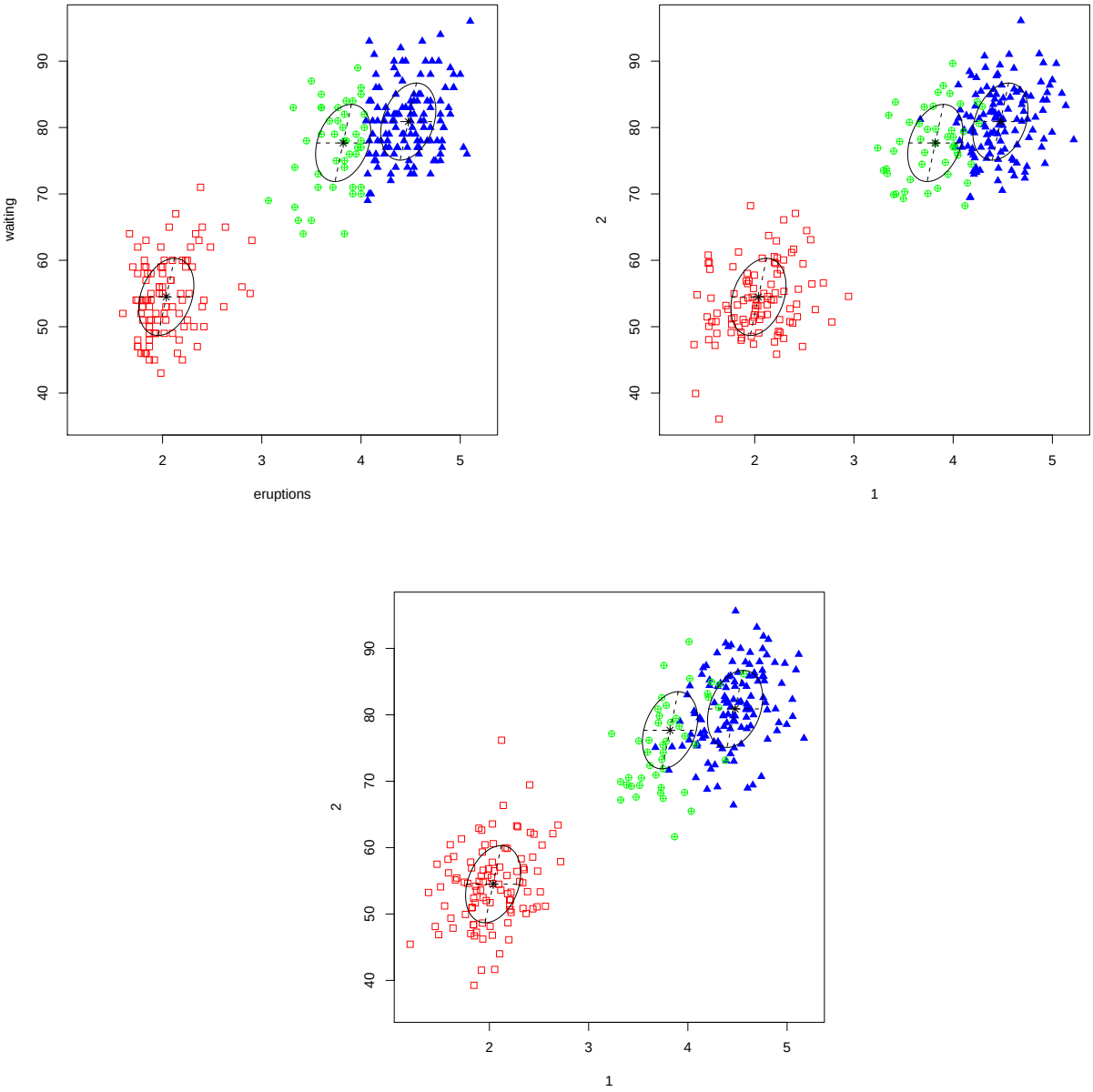


Figura 5.15: *Dataset Faithful: la figura in alto a sinistra è il dataset reale, mentre le figure successive rappresentano datasets della stessa ampiezza, simulati dal modello ottenuto mediante Mclust per il dataset Faithful. Nei grafici compaiono anche le ellissi definite dalle matrici di covarianza del modello.*

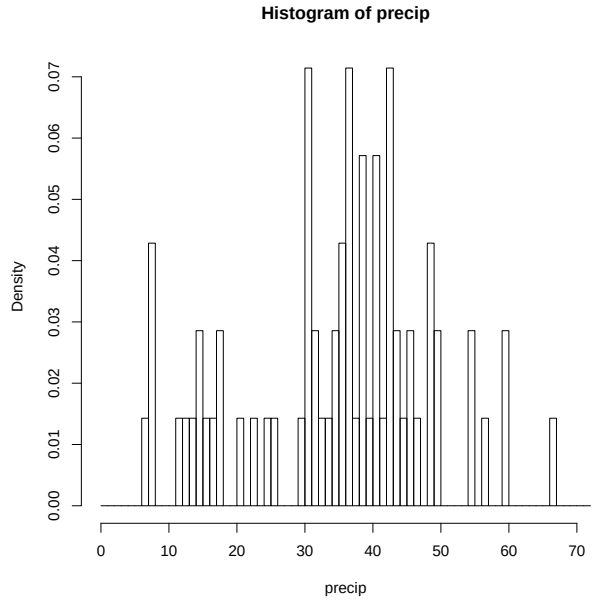


Figura 5.16: *Valori medi delle precipitazioni (in pollici) per ciascuna delle 70 città degli U.S.A. (con Portorico, fonte Statistical Abstracts of the United States, 1975).*

5.9 Dati unidimensionali

Le funzioni Mclust per la clusterizzazione, la stima di densità e l'analisi discriminante possono essere evidentemente applicate anche a dati unidimensionali. L'analisi si semplifica, avendo due soli modelli possibili: uguale varianza (denotato con E) oppure varianza variabile (V).

5.9.1 Clustering

L'analisi cluster per dati unidimensionali può essere svolta nello stesso modo che nel caso di due o più dimensioni. A titolo di esempio, si userà il dataset `precip` (incluso in R), che contiene il valore medio delle precipitazioni (in pollici) per ciascuna delle 70 città degli U.S.A. (con Portorico, fonte Statistical Abstracts of the United States, 1975). La Figura 5.16 riporta il grafico della funzione di ripartizione empirica del dataset:

Si applichi dunque la funzione di cluster sul dataset `precip`:

```
> precipMclust <- Mclust(precip)
```

e si generino i grafici dei risultati, ottenuti mediante:

```
> plot(precipMclust,precip, legendArgs=list(x="bottomleft"))
```

e visibili in Figura 5.17. L'analisi può essere altresì divisa in due parti: la valutazione del BIC per mezzo di `mclustBIC` e la stima del modello per mezzo di `summary`, come si mostra qui di seguito, relativamente al datest `rivers` (incluso in R). Questo dataset contiene le lunghezze (in miglia) dei 141 maggiori fiumi del Nord America (fonte: US Geological Survey).

```
> riversBIC <- mclustBIC(rivers)
> plot(riversBIC)
> riversModel <- summary(riversBIC, rivers)
> riversModel
```

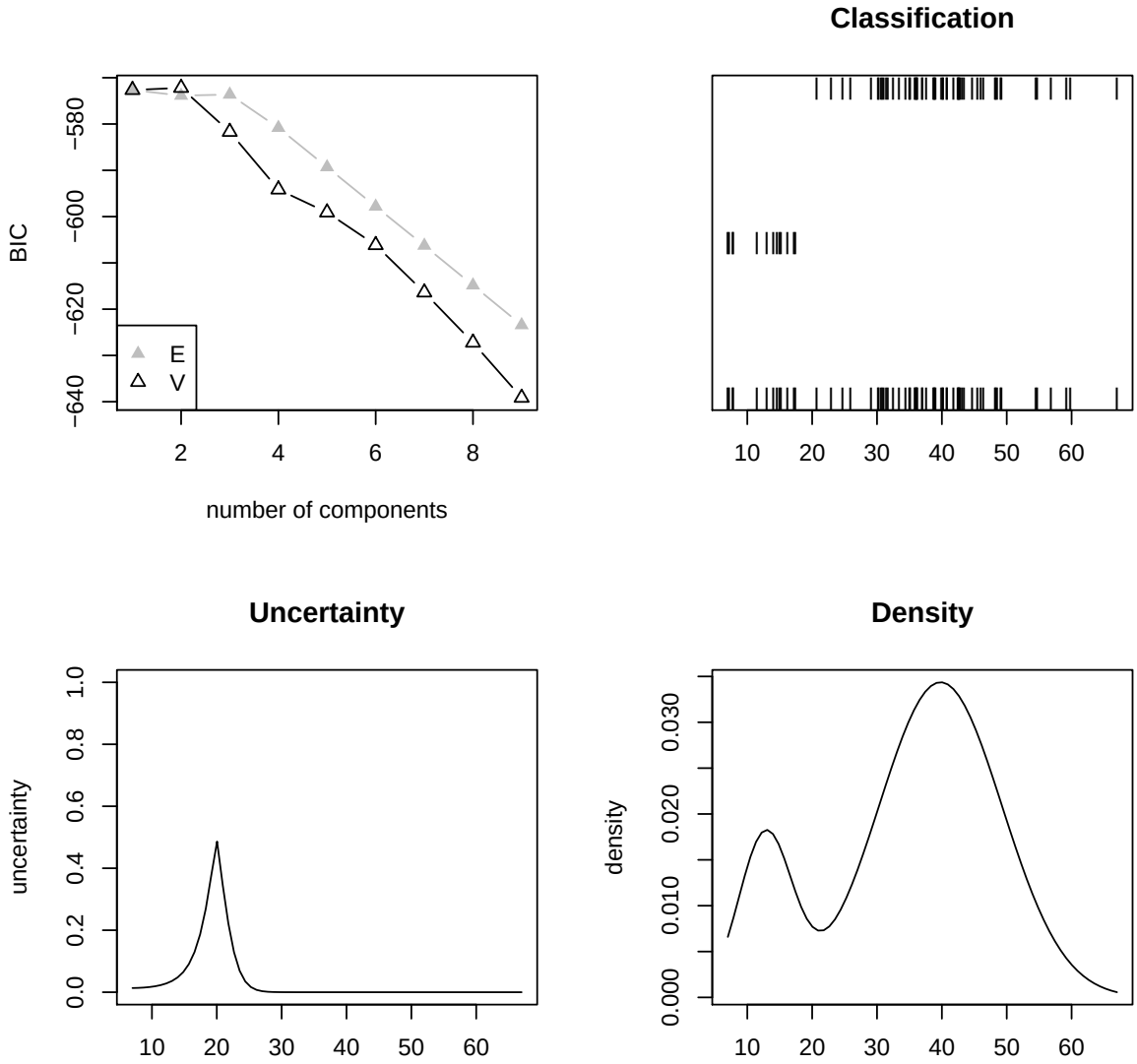


Figura 5.17: *Model-based clustering del dataset unidimensionale precip. In alto a sinistra: valori del BIC per numero di componenti il modello e per tipo di modello. In alto a destra: classificazione prodotta dal modello di maggior BIC (l'intero set dei dati è raffigurato nella fascia bassa del grafico, mentre le classi, separatamente, sono visualizzate su due diversi livelli nello stesso grafico). In basso a sinistra: grafico dell'incertezza nella classificazione. In basso a destra: grafico della densità del modello proposto da Mclust.*

Capitolo 6

Alcuni algoritmi in linguaggio R

R is a free software environment for statistical computing and graphics. It can be downloaded at `http://www.r-project.org/`.

Algoritmo 6.1 (Densità mistura gaussiana univariata con due componenti.) I valori di ingresso sono il valore `x`, il peso `alpha` della prima componente, il vettore delle medie `mu` ed il vettore delle deviazioni standard `sd`. Il valore di uscita è la funzione densità `p` calcolata in `x`.

```
mixt2<-function(x,alpha,mu,sd){
   f1<-dnorm(x,mu[1],sd[1])
   f2<-dnorm(x,mu[2],sd[2])
   p=alpha*f1+(1-alpha)*f2
 p
}
```

Algoritmo 6.2 (Densità mistura gaussiana univariata con tre componenti.) I valori di ingresso sono il valore `x`, il vettore dei pesi `alpha`, il vettore delle medie `mu` ed il vettore delle deviazioni standard `sd`. Il valore di uscita è la funzione densità `p` calcolata in `x`.

```
mixt3<-function(x,alpha,mu,sd){
   f1<-dnorm(x,mu[1],sd[1])
   f2<-dnorm(x,mu[2],sd[2])
   f3<-dnorm(x,mu[3],sd[3])
   p=alpha[1]*f1+alpha[2]*f2+alpha[3]*f3
   p
}
```

Algoritmo 6.3 (Densità mistura gaussiana univariata con g componenti.) I valori di ingresso sono il valore `x`, il vettore dei pesi `alpha`, il vettore delle medie `mu`, il vettore delle deviazioni standard `sd`, il numero di componenti `g`. Il valore di uscita è la funzione densità `p` calcolata in `x`.

```
mixtg<-function(x,alpha,mu,var,g){
 sd<-sqrt(var)
 n<-length(x)
 f<-matrix(0,n,g)
 for (j in 1:g)
    f[,j]<-dnorm(x,mu[j],sd[j])
     p=f%*%alpha
```

```
 p
}
```

Algoritmo 6.4 (Densità mistura gaussiana delle densità in Tabella 1.1 (1-4).) Utilizza le routine `mixt2.r` e `mixt3.r`.

```
plotmixt1<-function(){

 par(mfrow=c(2,2))
 x<-seq(-5,5,0.1)
#
# Gaussian
#
 title<-"Gaussian Density"
 p<-dnorm(x,0,1)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))
#
# Skewed unimodal
#
 title<-"Skewed Unimodal Density"
 alpha<-c(1/5,1/5,3/5)
 mu<-c(0,1/2,13/15)
 sd<-c(1,2/3,5/9)
 p<-mixt3(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))
#
# Strongly skewed
#
 title<-"Strongly Skewed Density"
 p<-dnorm(x,0,1)
 for (i in 1:7)
 p<-p+dnorm(x,3*((2/3)^i-1),(2/3)^i)
 p<-p/8
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))
#
# Kurtotic unimodal
#
 title<-"Kurtotic Unimodal Density"
 alpha<-2/3
 mu<-c(0,0)
     sd<-c(1,0.1)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

 par(mfrow=c(1,1))
}
```

Algoritmo 6.5 (Densità mistura gaussiana delle densità in Tabella 1.1 (5-8).) Utilizza le routine `mixt2.r` e `mixt3.r`.

```
plotmixt2<-function(){

 par(mfrow=c(2,2))
 x<-seq(-5,5,0.1)
#
# Outlier
#
 title<-"Outlier Density"
 alpha<-0.1
 mu<-c(0,0)
     sd<-c(1,0.1)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))
#
#  Bimodal
#
 title<-"Bimodal Density"
 alpha<-0.5
 mu<-c(-1,1)
     sd<-c(2/3,2/3)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

#
#  Separate Bimodal
#
 title<-"Separate Bimodal Density"
 alpha<-0.5
 mu<-c(-1.5,1.5)
     sd<-c(0.5,0.5)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))
#
#  Skewed Bimodal
#
 title<-"Asymmetric Bimodal Density"
 alpha<-3/4
 mu<-c(0,1.5)
     sd<-c(1,1/3)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

 par(mfrow=c(1,1))
}
```

Algoritmo 6.6 (Densità mistura gaussiana delle densità in Tabella 1.1 (9-12).) Utilizza le routine `mixt2.r` e `mixt3.r`.

```
plotmixt3<-function(){

 par(mfrow=c(2,2))
 x<-seq(-5,5,0.1)
#
# Trimodal
#
 title<-"Trimodal Density"
 alpha<-c(9/20,9/20,1/10)
 mu<-c(-6/5,6/5,0)
 sd<-c(3/5, 3/5,1/4)
 p<-mixt3(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))
#
# Claw
#
 title<-"Claw Density"
 p<-dnorm(x,0,1)/2
 for (i in 0:4)
 p<-p+dnorm(x,i/2-1,1/10)/10
 p<-p/8
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

#
# Double Claw
#
 title<-"Double Claw Density"
 p<-dnorm(x,-1,2/3)*49/100+dnorm(x,1,2/3)*49/100
 for (i in 0:6)
 p<-p+dnorm(x,(i-3)/2,1/100)/350
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

#
# Asymmetric Claw
#
 title<-"Asymmetric Claw Density"
 p<-dnorm(x,0,1)/2
 for (i in -2:2)
 p<-p+dnorm(x,(i+1/2),2^(-i)/10)*(2^(1-i)/31)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

 par(mfrow=c(1,1))
}
```

Algoritmo 6.7 (Densità mistura gaussiana delle densità in Tabella 1.1 (13-16).)

```
plotmixt4<-function(){

 par(mfrow=c(2,2))
 x<-seq(-5,5,0.1)
#
# Asymmetric Double Claw
#
 title<-"Asymmetric Double Claw Density"
 p<-dnorm(x,-1,2/3)*46/100+dnorm(x,1,2/3)*46/100
 for (i in 1:3)
 p<-p+dnorm(x,-i/2,1/100)/300
 for (i in 1:3)
 p<-p+dnorm(x,i/2,7/100)*(7/300)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

#
# Smooth Comb
#
 title<-"Smooth Comb Density"
 i<-0
 p<-dnorm(x,(65-96*(1/2)^i)/21,(32/63)/2^i)*(2^(5-i)/63)
 for (i in 1:5)
 p<-p+dnorm(x,(65-96*(1/2)^i)/21,(32/63)/2^i)*(2^(5-i)/63)
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

#
# Discrete Comb
#
 title<-"Smooth Comb Density"
 i<-0
 p<-dnorm(x,(12*i-15)/7,2/7)*(2/7)
 for (i in 1:2)
 p<-p+dnorm(x,(12*i-15)/7,2/7)*(2/7)
 for (i in 8:10)
 p<-p+dnorm(x,2*i/7,1/21)/21
 plot(x,p,type="l",xlab="x",ylab="", main=title,xlim=c(-3.2,3.2))

 par(mfrow=c(1,1))
}
```

Algoritmo 6.8 (Generazione dei grafici delle funzioni di densità mistura) La seguente routine genera i grafici delle funzioni di densità mistura con due componenti (1.8). Richiede in ingresso il peso α del primo miscuglio. Utilizza la routine `mixt2.r`.

```
plotmixt<-function(alpha){

 par(mfrow=c(2,2))
```

```
 x<-seq(-5,10,0.1)
 sd<-c(1,1)

#
# Delta = 1
#
 title<-"Delta = 1"
 mu<-c(0,1)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,
        xlim=c(-3.2,7),ylim=c(0,0.4))

#
# Delta = 2
#
 title<-"Delta = 2"
 mu<-c(0,2)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,
        xlim=c(-3.2,7),ylim=c(0,0.4))

#
# Delta = 3
#
 title<-"Delta = 3"
 mu<-c(0,3)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,
        xlim=c(-3.2,7),ylim=c(0,0.4))

#
# Delta = 4
#
 title<-"Delta = 4"
 mu<-c(0,4)
 p<-mixt2(x,alpha,mu,sd)
 plot(x,p,type="l",xlab="x",ylab="", main=title,
        xlim=c(-3.2,7),ylim=c(0,0.4))

 par(mfrow=c(1,1))
}
```

Algoritmo 6.9 (Generazione di campioni estratti con distribuzione mistura) La funzione `genmixt.r` genera un campione di dimensione n da una mistura gaussiana a due componenti. Richiede in ingresso il peso α ed i parametri delle componenti (μ_1, σ_1), $(\mu_2, \sigma_)$

```
genmixt<-function(n,alpha,mu,sd){
```

```
  x<-numeric(n)
  for (i in 1:n){
     dum<-runif(1,0,1)
     if (dum < alpha)
         x[i]<-rnorm(1,mu[1],sd[1])
      else
         x[i]<-rnorm(1,mu[2],sd[2])
  }
  hist(x)
  x
}
```

Algoritmo 6.10 (Stima dei parametri di una mistura gaussiana univariata mediante algoritmo EM)
La procedura

```
EMuniv<-function(x,g,br,vareq){

 n<-length(x)
 p<-numeric(n)
 f<-matrix(0,n,g)
 w<-matrix(0,n,g)
  mu0<-numeric(g)
  var0<-numeric(g)
  alpha0<-numeric(g)
  eps<-0.01
 itermax<-100
#
# initialize parameter
#

 for (j in 1:(g-1))
    alpha0[j]<-runif(1,0,1/g)
     alpha0[g]<-1-sum(alpha0[1:(g-1)])

  for (j in 1:g){
    mu0[j]<-runif(1,min(x),max(x))
    var0[j]<-var(x[1:round(n/g)])
 }
 if (vareq==T) var0<-rep(mean(var0),g)
 sd0<-sqrt(var0)
#
# first iteration
#
 iter<-0
 for (h in 1:n) {
    for (j in 1:g){
       f[h,j]<-dnorm(x[h],mu0[j],sd0[j])
         }
      }
```

```
 p<-f%*%alpha0
#
# initial likelihood
#
      L0<-sum(log(p))
 cat(iter,L0,"\t",alpha0,"\t",mu0,var0,"\n")
# cat(file="mixtdata.dat",iter,L0,"\t",alpha0,"\t",mu0,"\t",var0,"\n")
 L1<-L0

 alpha<-alpha0
 mu<-mu0
 var<-var0
 for (iter in 1:itermax){
#
# E-step
#
 for (h in 1:n)
    w[h,]<-f[h,]*alpha/p[h]
 alpha<-apply(w,2,mean)

#
# M-step
#
      for (j in 1:g){
    mu[j]<-sum(w[,j]*x)/sum(w[,j])
    var[j]<-sum(w[,j]*((x-mu[j])^2))/sum(w[,j])
 }
 if (vareq==T) var<-rep(mean(var),g)
 sd<-sqrt(var)
 for (h in 1:n) {
    for (j in 1:g){
       f[h,j]<-dnorm(x[h],mu[j],sd[j])
         }
      }
 p<-f%*%alpha
      L<-sum(log(p))
 cat(file="mixtdata.dat",iter,L,"\t",alpha,"\t",mu, "\t",var,"\n",append=T
 if (abs(L1-L)<eps) break
 L1<-L
}
 cat("\n")
# cat("n. iterazioni:  ", iter,"\n")
 cat("logverosimiglianza finale:  ", L,"\n")
# cat("pesi:     ", alpha,"\n")
# cat("medie:    ",mu,"\n")
#  cat("varianze: ", var, "\n\n")

      dens<-mixtg(x,alpha,mu,var,g)
```

```
 hist(x,breaks=br,plot=F)->ht
 xlinf<-min(x)/1.01
 xlsup<-max(x)/0.99
 ylsup=max(dens,ht$dens)
 hist(x,freq=FALSE,xlim=c(xlinf,xlsup),ylim=c(0,ylsup),breaks=br,
       main="",xlab="thickness")
 curve(mixtg(x,alpha,mu,var,g),xlinf,xlsup,add=TRUE,col="red",lwd=2)

 numpar<-3*g-1
 AIC<--2*L+2*numpar
 BIC<--2*L+numpar*log(n)
 cat("AIC: ",AIC,"\n")
 cat("BIC: ",BIC,"\n")
 cat("\n")

}
```

Bibliografia

[1] Allard D. & Fraley C. (1997), Nonparametric maximum likelihood estimation of features in spatial point processes using Voronoi tessellation, *Journal of the American Statistical Association*, **92**, 1485-1493.

[2] Akaike H. (1973), Information theory and an extension of the maximum likelihood principle, in *Second International Symposium on Information Theory*, B.N. Petrov and F.Csaki (Eds.), Akadémiai Kiadó, Budapest, 267-281.

[3] Akaike H. (1974), A new look at the statistical model identification, *IEEE Transactions on Automatic Control*, **19**, 716-723.

[4] Banfield J. D. & Raftery, A. E. (1992), Ice floe identification in Satellite Images using mathematical morphology and clustering about principle curves, *Journal of the American Statistical Association*, **87**, 7-16.

[5] Banfield J. D. & Raftery, A. E. (1993), Model-based Gaussian and non-Gaussian clustering, *Biometrics*, **49**, 803-821.

[6] Bickel P.J. & Doksum K.A. (1977), *Mathematical Statistics*, Holden Day, Oakland.

[7] Biernacki C. (2004), An asymptotic upper bound of the likelihood to prevent Gaussian mixture from degenerating, *Technical Report, Universitè de Franche-Comté*.

[8] Binder D. A. (1978), Bayesian Cluster Analysis, *Biometrika*, **65**, 31-38.

[9] Bock H. H. (1996), Probabilistic Models in Cluster Analysis, *Computational Statistics and Data Analysis*,**23**, 5-28.

[10] Bock H. H. (1998), Probabilistic Approaches in Cluster Analysis, *Bulletin of the International Statistical Institute*,**57**, 603-606.

[11] Bock H. H. (1998), Probabilistic Aspects in Classification, in *Data Science, Classification and Related Methods*, eds. Hayashi C., Yajima K., Bock H.H., Oshumi N., Tanaka Y. and Baba Y. New York: Springer Verlag, 3-21.

[12] Burnham K.O. & Anderson D.R. (2004), Multimodel inference: understanding AIC and BIC in model selection, *http://www2.fmg.uva.nl/modelselection/presentations/AWMS2004-Burnham-paper.pdf*

[13] Byers S.D. & Raftery A.E. (1998), Nearest neighbor clutter removal for estimating features in spatial point processes, *Journal of the American Statistical Association*, **93**, 577-584.

[14] Campbell J.G., Fraley C., Murtagh F. & Raftery A.E. (1997), Linear flow detection in Woven Textiles using Model Based Clustering, *Pattern Recognition Letters*, **18**, 1539-1548.

[15] Campbell J.G., Fraley C., Stanford D., Murtagh F. & Raftery A.E. (1999), Model-Based Methods for Real-Time Textile fault detection, *International Journal of Imaging systems and Technology*, **10**, 339-346.

[16] Campbell N.A. & Mahon R.J. (1974), A multivariate study of variation in two species of rock crab of genus *Letpograspus*, *Australian Journal of Zoology*, **22**, 417-455.

[17] Cassie R.M. (1954), Some uses of probability paper in the analysis of size frequency distributions, *Australian Journal of Marine and Freshwater Research*, **5**, 513-522.

[18] Celeux G. & Govaert G. (1995), Comparison of the mixture and the classification maximum likelihood in cluster analysis, *Journal Statist. Comput. Simul.*, **47**, 127-146.

[19] Celeux G. & Govaert G. (1995), Gaussian parsimonious clustering models, *Pattern Recognition*, **28**, 781-793.

[20] Cheeseman P. & Stutz J. (1995), Bayesian Classification (AutoClass): Theory and Results, in *Advances in Knowledge Discovery and Data Mining*,eds. Fayyad U., Piatesky-Shapiro G., Smyth P. and Uthurusamy R., AAAI Press, 153-180.

[21] Crawford S.L. (1994), An application of the Laplace method to finite mixture distributions, *Journal of the American Statistical Association*, **89**, 259-267.

[22] Dasgupta A. & Raftery A.E. (1998), Detecting features in spatial point processes wuth clutter via Model-Based clustering, *Journal of the American Statistical Association*, **93**, 294-302.

[23] Day N.E. A. (1969), Estimating the components of a mixture of normal distributions, *Biometrika*, **56**, 463-474.

[24] Dempster A.P., Laird N.M. & Rubin D.B. (1977), Maximum likelihood from incomplete data via the EM algorithm (with discussion), *Journal of the Royal Statistical Society B*, **39**, 1-38.

[25] Donoho, D.L. (1988), One-sided inference about functionals of a density, *Annals of Statistics*, **16**, 1390-1420.

[26] Duda R.O. & Hart P.E. (1973), *Pattern Classification and Scene Analysis*, New York: Wiley.

[27] Edwards A.W.F. & Cavalli-Sforza L.L. (1965), A method for Cluster Analysis, *Biometrics*, **21**, 362-375.

[28] Fayyad U. & Smyth P. (1996), From massive Data sets to Science Catalogs: Applications and challenges, in *Statistics and massive data sets: Report to the Committee on Applied and Theoretical Statistics*, eds. Kettenring J. and Pregibon D., National Research Council.

[29] Frayley C. E. (1998), Algorithms for model-based gaussian hierarchical clustering, *SIAM Journal on Scientific Computing*, **20**, 270-281.

[30] Frayley C. & Raftery A. E. (1998), How many clusters? Which clustering method? Answers via model-based cluster analysis, *The Computer Journal*, **41**, 578-588.

[31] Frayley C. & Raftery A. E. (2002), Model-based clustering, discriminant analysis and density estimation, *Journal of the American Statistical Association*, **97**, 611-631.

[32] Frühwirth-Schnatter S. (2006), *Finite Mixture and Markov Switching Models*, Springer, New York.

[33] Greselin F. & Ingrassia S. (2008), Constrained monotone EM algorithms for mixtures of multivariate t-distributions. *Rapporti di Ricerca del Dipartimento di Metodi Quantitativi per le Scienze Economiche ed Aziendali - Università degli Studi di Milano Bicocca, n.142*, 2008.

[34] Hathaway, R.J. (1986), A constrained formulation of maximum-likelihood estimation for normal mixture distributions. *The Annals of Statistics*, **13**, 795-800.

[35] Ingrassia S. (2004), A Likelihood-Based Constrained Algorithm for Multivariate Normal Mixture Models, *Statistical Methods & Applications*, **13** (2004), n.2, 151-166.

[36] Ingrassia S. & Rocci R. (2007), Constrained monotone EM algorithms for finite mixture of multivariate Gaussians, *Computational Statistics & Data Analysis*, **51**, 5339-5351.

[37] Izenman A.J. & Sommer C.J. (1988), Philatelic mixtures and multimodal densities, *Journal of the American Statistical Association*, **83**, 941-953.

[38] Kadane H. B. & Lazar N. A. (2004). Methods and criteria for model selection, *Journal of the American Statistical Association*, **99**, 465, 279-290.

[39] Kotz S. & Nadarajah S. (2004), *Multivariate t Distributions and Their Applications*, Cambridge University Press, New York.

[40] Lange K.L., Little R.J.A. & Taylor G.M.G. (1989), Robust statistical modeling using the t distribution, *Journal of the American Statistical Association*, **84**, 881-896.

[41] Liu C. & Rubin D.B. (1994), The ECME algorithm: a simple extension of EM and ECM with faster monotone convergence, *Biometrika*, **81**, 633-648.

[42] Lo Y., Mendel N.R. & Rubin D.B. (2001), Testing the number of components in a normal mixture, *Biometrika*, **88**, 3, 767-778.

[43] McLachlan G.J. (1992), Discriminant Analyisis and Statistical Pattern Recognition, Wiley, New York.

[44] McLachlan G.J. & Basford K.E. (1988), *Mixture Models: Inference and applications to clustering*, Dekker, New York.

[45] McLachlan G.J. & Krishnan T. (2008), *The EM algorithm and Extensions*, 2nd Edition, Wiley, New York.

[46] McLachlan G.J. & Peel D. (2000), *Finite Mixture Models*, Wiley, New York.

[47] Mardia K.V., Kent J.T. & Bibby J.M. (1992), *Multivariate Analysis*, Academic Press, London.

[48] Meng X.-L. & van Dyk D.A. (1997), The EM algorithm – an old folk song sung to a fast new tune, *Journal of the Royal Statistical Society B*, **59**, 511-567.

[49] Meng X.L. and Rubin D.B. (1993), Maximum likelihood estimation via the ECM algorithm: a general framework, *Biometrika*, **80**, 267–278.

[50] Murtagh F. & Raftery A. E. (1984), Fitting Straight lines to point patterns, *Pattern Recognition*, **17**, 479-483.

[51] Nadarajah S. and Kotz S. (2005), Mathematical properties of the multivariate t distribution, *Acta Applicandae Mathematicae*, **89**, 53-84.

[52] Pearson K. (1894), Contribution to the theory of mathematical evolution, *Philosophical Transaction of the Royal Society of London A*, **185**, 71-110.

[53] Peel D. & McLachlan G.J. (2000), Robust mixture modelling using the t distribution, *Statistics and Computing*, **10**, 339-348.

[54] Raftery A.E. (1995), Bayesian model selection in social research (with Discussion), *Sociological Methodology*, **25**, 111-196.

[55] Schwarz G. (1978), Estimating the dimension of a model, *The Annals of Statistics*,**6**, 461-464.

[56] Scott A.J. & Symons M.J.(1971), Clustering Methods based on likelihood ratio criteria, *Biometrics*, **27**, 387-397.

[57] Shoham S. (2002), Robust clustering by deterministic agglomeration EM of mixtures of multivariate t-distributions, *Pattern Recognition*, **35**, 1127-1142.

[58] Titterington D.M., Smith A.F.M. & Makov U.E. (1985), *Statistical Analysis of Finite Mixture Distributions*, Wiley, Chichester

[59] Weldon W.F.R. (1892), Certain correlated variations in *Crangon vulgaris*, *Proceedings of the Royal Society of London*, **51**, 2-21.

[60] Weldon W.F.R. (1893), On certain correlated variations in *Carcinus moenas*, *Proceedings of the Royal Society of London*, **54**, 318-329.

[61] Wolfe J.H. (1963), Object cluster analysis of social areas, *Master's thesis*, University of California, Berkeley.

[62] Wolfe J.H. (1965), A computer program for the maximum-likelihood analysis of types, *USNPRA technical Bulletin 65-15*, U.S. Naval Personnel Research Activity, San Diego.

[63] Wolfe J.H. (1967), NORMIX: computational methods for estimating the parameters of multivariate normal mixture distributions, *Technical Bulletin USNPRA SRM 68-2*, U.S. Naval Personnel Research Activity, San Diego.

[64] Wolfe J.H. (1970), Pattern clustering by multivariate mixture analysis, *Multivariate behavioral Research* , **5**, 329-350.