ESERCITAZIONI DI STATISTICA BIOMEDICA

ALCUNE NOTE SU R

Matteo Dell'Omodarme

AGOSTO 2012

Indice

Introduzione e notazione

Queste note, sviluppate per le esercitazioni del corso di Statistica Biomedica presso la Scuola Normale Superiore di Pisa, intendono illustrare alcune delle potenzialità di R, utilizzato come strumento di indagine statistica. L'obiettivo è quello di presentare varie tecniche e imparare ad adoperarle nella pratica.

Nel presentare il materiale si assume una conoscenza di base di R. Per una panoramica generale sul linguaggio o informazioni specifiche sulla sintassi dei comandi di base si rimanda al sito ufficiale della R Foundation [44] accessibile all'indirizzo web:

http://www.r-project.org

che mette a disposizione, oltre al codice sorgente del programma, molta documentazione.

Nel corso del testo, se non diversamente specificato, si fa uso della seguente notazione:

- m: indica la media di un campione, così m_A è la media del campione A.
- s^2: indica la varianza di un campione, così s^2_A è la varianza del campione A.
- Il carattere "$\wedge$" indica la stima sul campione di un parametro teorico, così $\hat{\beta}$ è la stima campionaria del parametro β.
- Il simbolo ">" è utilizzato come primo carattere nelle linee che identificano un input fornito a R. Se il comando è troppo lungo per essere contenuto in una sola riga le righe di continuazione iniziano con il simbolo "+".
- [...]: indica una serie di istruzioni o di testo che non viene mostrata.

Capitolo 1

Statistica descrittiva e funzioni di distribuzione

In questo capitolo vengono introdotte le funzioni statistiche che permettono di estrarre da un set di dati informazioni di riepilogo quali la media, la varianza o i quartili. Vengono poi presentati alcuni metodi grafici (mono e bidimensionali) con cui è possibile ispezionare le caratteristiche del campione in esame. Il capitolo si chiude con una breve panoramica sulle funzioni di distribuzione implementate in R di più comune utilizzo.

1.1 Funzioni statistiche

Dato un campione A, per calcolarne la media, la varianza, la deviazione standard e la mediana si usano le seguenti funzioni:

```
> mean(A)
> var(A)
> sd(A)
> median(A)
```

La taglia di A, il valore massimo e il minimo si ottengono con le chiamate:

```
> length(A)
> max(A)
> min(A)
```

La funzione *summary* genera un riepilogo di sei statistiche calcolate sul campione, ossia il minimo, il primo quartile, la media, la mediana il terzo quartile e il massimo. Ad esempio, sul campione dei primi 20 numeri interi si ottiene il seguente output:

```
> A <- 1:20     # vettore contenente i primi 20 numeri interi
> summary(A)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    5.75   10.50   10.50   15.25   20.00
```

Dati due campioni A e B, la loro covarianza e il coefficiente di correlazione si ottengono con le funzioni:

```
> cov(A, B)
> cor(A, B)
```

Se si hanno dai dati classificati in base a uno o più fattori, la funzione *table* consente di costruire una tabella di classificazione dei conteggi per ogni combinazione dei fattori in gioco. Ad esempio, si consideri il fattore A a 4 livelli (da 0 a 3):

```
> A <- factor( c(0,0,0,0,1,1,2,2,2,2,2,2,2,3,3,3) )
```

Per contare le occorrenze di ognuno dei livelli si usa la chiamata:

```
> table(A)
```

che fornisce in output:

```
A
0 1 2 3
4 2 7 3
```

Se si hanno due fattori, si costruisce la tabella di contingenza dei conteggi come nel caso seguente:

```
> A <- factor( c(0,0,0,0,1,1,2,2,2,2,2,2,2,3,3,3) )
> B <- factor( c(0,1,0,0,1,1,1,1,0,0,1,1,1,1,0,1) )
> table(B, A)

   A
B   0 1 2 3
  0 3 0 2 1
  1 1 2 5 2
```

in cui i livelli del primo fattore vengono disposti per riga e quelli del secondo per colonna.

La funzione *summary* è molto utile anche per riepilogare le informazioni contenute in un data frame con più variabili. Come esempio si consideri un set di dati della libreria standard *MASS*, relativo ai dati sulle eruzioni del gayser "Old Faithful" del parco nazionale di Yellowstone (Wyoming). I dati provengono da un lavoro di Azzalini e Bowman (Applied Statistics 39, 357-365, 1990).

```
> library(MASS)    # carica la libreria MASS
> data(geyser)     # carica il dataset di nome geyser
> geyser           # visualizza i dati
    waiting  duration
1        80 4.0166667
2        71 2.1500000
3        57 4.0000000
[...]
299      79 2.0000000
```

Si tratta di 299 osservazioni di due variabili: *duration* che rappresenta la durata dell'eruzione (in min) e *waiting* il tempo di attesa fino alla successiva eruzione. Un riepilogo rapido dei dati si può avere con la chiamata:

```
> summary(geyser)
    waiting          duration
 Min.   : 43.00   Min.   :0.8333
 1st Qu.: 59.00   1st Qu.:2.0000
 Median : 76.00   Median :4.0000
 Mean   : 72.31   Mean   :3.4608
 3rd Qu.: 83.00   3rd Qu.:4.3833
 Max.   :108.00   Max.   :5.4500
```

1.2 Visualizzazioni grafiche dei dati

Si consideri un campione di 200 numeri casuali estratti da una distribuzione χ^2 a 2 gradi di libertà:

```
> A <- rchisq(200, 2)
```

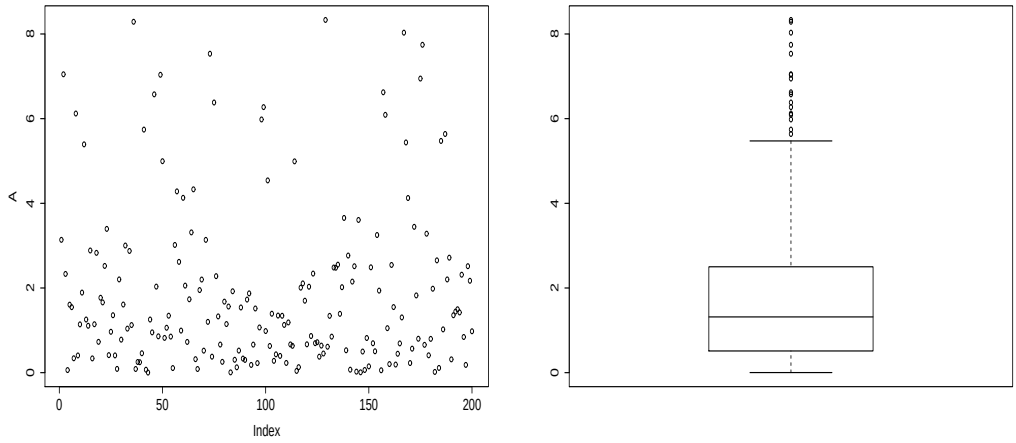

Figura 1.1: Plot $X - Y$ e box-and-whisker plot di 200 numeri a distribuzione $\sim \chi^2(2)$.

Una prima analisi grafica può essere condotta mediante un semplice plot $X - Y$:

```
> plot(A)
```

che produce l'output a sinistra in Fig. 1.1.

Ulteriori informazioni sulla distribuzione dei dati, sulla sua simmetria e sulla presenza di eventuali dati particolarmente distanti dalla media (outliers) possono essere desunte osservando il diagramma a scatola con baffi (box-and-whisker plot):

```
> boxplot(A)
```

Il risultato, a destra in Fig. 1.1, mette in luce l'asimmetria della distribuzione con la presenza della lunga coda destra.

Riprendendo i dati relativi al geyser "Old Faithful" del parco nazionale di Yellowstone, presentati in Sec. 1.1, è possibile evidenziare tendenze nelle distribuzioni della durata e dei tempi di attesa fra due eruzioni esaminando i boxplots delle due variabili. Per far ciò, e per disporre i due grafici fianco a fianco, si può dividere la finestra di output grafico in 2 colonne per 1 riga con la chiamata:

```
> par(mfrow=c(1,2))  # divide la finestra grafica in 1 riga e 2 colonne
```

I due grafici si producono nel modo seguente:

```
> boxplot(geyser$waiting, xlab="waiting")
> boxplot(geyser$duration, xlab="duration")
```

L'output di Fig. 1.2 evidenzia che la distribuzione del tempo di attesa è maggiormente simmetrica di quella della durata dell'eruzione.

1.2.1 Istogrammi

I metodi grafici finora presentati permettono di avere un rapido riepilogo dei dati, ma non di avere un'idea della loro funzione di densità, la quale consente di mettere in luce eventuali comportamenti come bimodalità o simili. Per affrontare problemi di questo genere si può ricorrere alla generazione di istogrammi o di stime di kernel density.

Sia A un campione di 120 numeri casuali di distribuzione normale standard:

```
> A <- rnorm(120)        # 120 num. casuali ~ N(0,1)
```

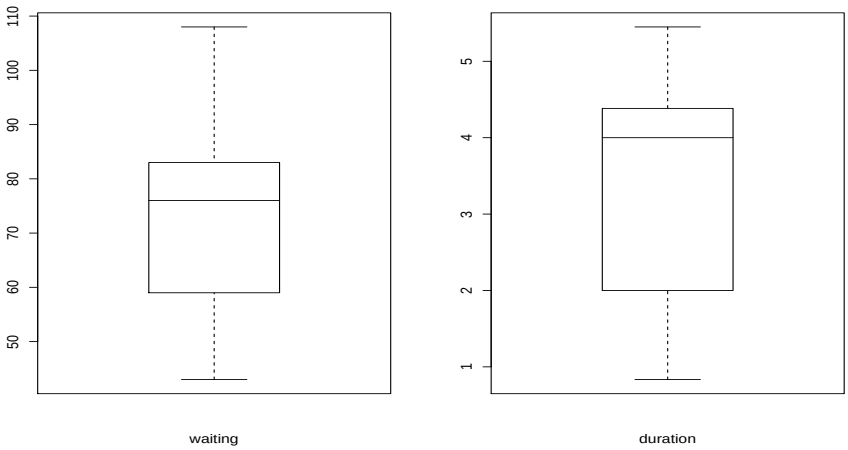

Figura 1.2: Box-and-whisker plot dei tempi di attesa fra due eruzioni del geyser "Old Faithful" del parco nazionale di Yellowstone e delle loro durate (in min).

Per riepilogare i dati in istogramma è necessario calcolare il numero di classi (o bin) statisticamente appropriato. Detto numero si ottiene arrotondando all'intero più vicino la radice quadrata del numero dei dati:

```
> nclassi <- round(sqrt(length(A)))
```

Si costruisce quindi il vettore delle classi in cui suddividere il range di A:

```
> classi <- seq(min(A), max(A), length=nclassi+1)
```

Si è fatto uso della funzione *seq*, che accetta in questo specifico esempio tre argomenti: gli estremi dell'intervallo e il numero di punti con cui campionare regolarmente tale intervallo. Il comando per ottenere l'istogramma è infine:

```
> hist(A, breaks=classi)
```

Oltre a riportare i dati in istogramma presentando le frequenze assolute è possibile presentare l'istogramma delle frequenze relative:

```
[...]
> hist(A, breaks=classi, prob=TRUE)
```

Per variabili continue, una rappresentazione più sofisticata si può ottenere graficando la kernel density. Se si suppone di avere n osservazioni $x_1, \ldots, x_n$ lo stmatore di kernel density è definito come:

$$\hat{f}_h(x) = \frac{1}{nh} K(h^{-1}(x - x_i)))$$

dove K è una funzione detta kernel e h un parametro detto *bandwidth*. La funzione K deve essere una funzione di densità simmetrica e centrata in 0; una scelta classica è la funzione gaussiana. La stima della densità $\hat{f}_h(x)$ in un punto x è dunque la media di n funzioni di densità centrate nei punti osservati x_i; il parametro h regola la dispersione di queste densità. Per valori piccoli di h avranno importanza solo i punti vicini a x_i, mentre per h grande saranno rilevanti anche osservazioni lontane. Un valore intermedio del parametro consente di mettere in luce l'andamento essenziale della funzione di densità da stimare.

In R è possibile far uso della funzione *density*:

```
> lines(density(A))
```

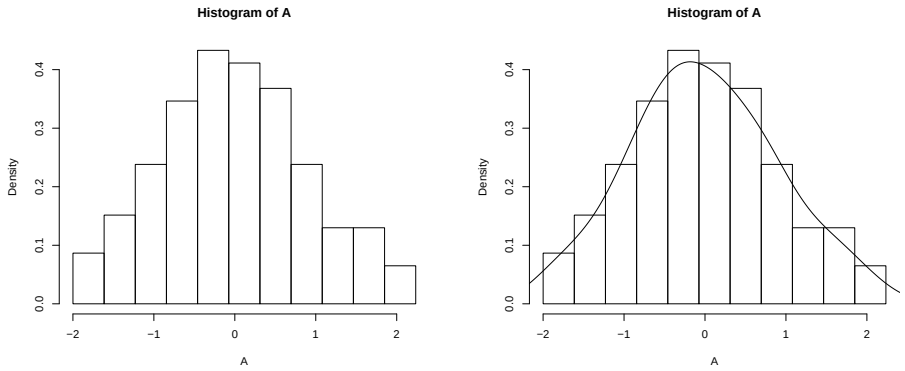

Figura 1.3: Istogramma ottenuto dalla generazione 120 numeri casuali $\sim N(0,1)$ con sovrapposta la kernel density (a destra).

La chiamata alla funzione *lines* sovrappone il grafico di kernel density all'istogramma realizzato in precedenza. Se non si specifica altrimenti, usando l'opzione *bw*, la funzione *density* utilizza un algoritmo di ottimizzazione per determinare la larghezza di banda migliore (per ulteriori dettagli si veda la pagina di manuale relativa alla funzione *density*). Per rappresentare solamente la kernel density si usa il comando *plot* al posto di *lines*. I risultati sono presentati in Fig. 1.3.

Esempio

I metodi presentati possono essere applicati ai dati del geyser "Old Faithful". Come in precedenza si divide la finestra grafica in due parti:

```
> par(mfrow=c(1,2))
```

quindi si generano l'istogramma per i tempi d'attesa fra due eruzioni e la sua kernel density:

```
> hist(geyser$waiting, prob=TRUE, xlab="waiting")
> lines(density(geyser$waiting), col="red")
```

a cui si affianca l'equivalente grafico per la durata delle eruzioni:

```
> hist(geyser$duration, prob=TRUE, xlab="duration")
> lines(density(geyser$duration), col="red")
```

L'output, in Fig. 1.4, evidenzia che le distribuzioni di entrambe le variabili sono bimodali, risultato che non si poteva evincere dall'osservazione di Fig. 1.2.

1.2.2 Rappresentazioni bidimensionali

La generalizzazione al caso di stime kernel density in due dimensioni può essere ottenuta con la funzione *kde2d* della libreria *MASS* come nell'esempio seguente.

Nel caso delle eruzioni del geyser "Old Faithful" la stima kernel density congiunta delle variabili durata e attesa si ottiene con la chiamata:

```
> f1 <- kde2d(duration, waiting, n=50)
```

La funzione accetta, oltre alle variabili da utilizzare, l'opzione n che specifica il numero di punti da usare in ogni direzione per arrivare alla stima della funzione di densità.

La rappresentazione grafica dei punti e della kernel density, date in Fig. 1.5, possono essere ottenute con le chiamate:

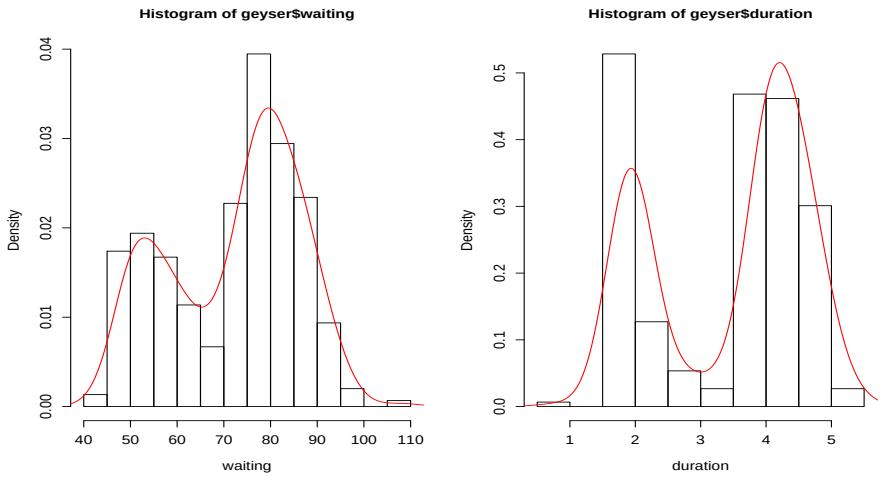

Figura 1.4: Istogrammi e stime kernel density dei tempi di attesa fra due eruzioni del geyser "Old Faithful" del parco nazionale di Yellowstone e delle loro durate (in min).

```
> par(mfrow=c(1,2))
> plot(geyser$duration, geyser$waiting, xlab = "duration", ylab="waiting")
> contour(f1, xlab = "duration", ylab="waiting")
```

La funzione *contour* realizza un grafico detto contour plot. Tale funzione accetta molte opzioni, tra cui *levels* che permette di specificare un vettore con i livelli in corrispondenza dei quali devono essere tracciati i contorni.

1.3 Funzioni di distribuzione

R mette a disposizione numerose funzioni di distribuzione, sia discrete che continue. Tra le principali si ricordano le seguenti.

1.3.1 Distribuzione binomiale

È possibile calcolare la densità di probabilità, la funzione di distribuzione e i quantili di una distribuzione binomiale tramite le tre funzioni:

```
dbinom(x, size, prob, log = FALSE)
pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)
qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)
```

Lo standard che R segue per le varie distribuzioni è quello di identificare con un nome la distribuzione (in questo caso *binom*) e farlo precedere dalle lettere d, p e q per identificare la densità, la distribuzione e i quantili.

I primi tre argomenti delle funzioni devono essere obbligatoriamente specificati, mentre gli altri sono opzionali; se essi non vengono inseriti R assume un valore preimpostato. Ad esempio l'argomento *log* della funzione *dbinom* (che permette di ottenere il logaritmo delle probabilità in luogo delle probabilità stesse) ha di default il valore $FALSE$. Allo stesso scopo serve l'opzione *log.p* (di default $FALSE$). Infine l'opzione *lower.tail* permette di scegliere fra i valori di probabilità P(X <= x) (valore $TRUE$, impostato di default) e P(X > x) (valore $FALSE$).

È possibile fare un plot della densità della binomiale $B(x, 10, 0.65)$ e della sua funzione di distribuzione nel modo seguente:

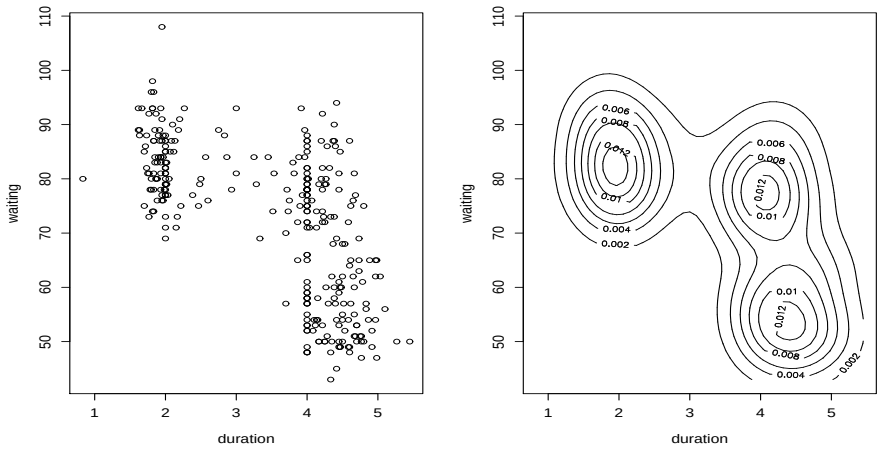

Figura 1.5: Grafico e stima kernel density congiunta dei tempi di attesa fra due eruzioni del geyser "Old Faithful" del parco nazionale di Yellowstone e delle loro durate (in min).

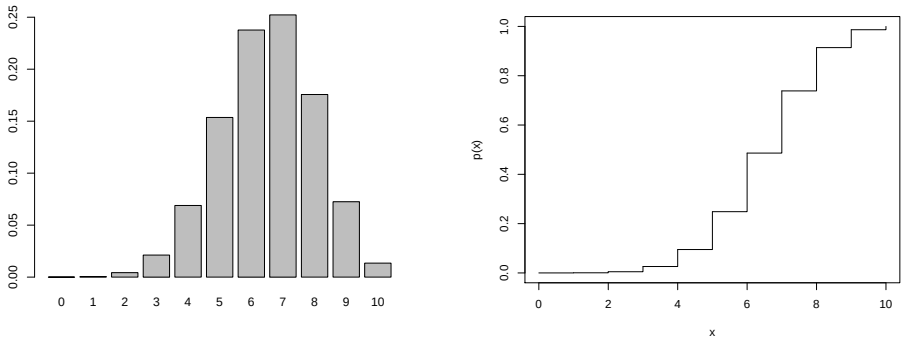

Figura 1.6: Istogramma della densità della distribuzione binomiale $B(x, 10, 0.65)$ e della sua funzione di distribuzione.

```
> barplot(dbinom(0:10,10,0.65), col="grey", names.arg=0:10)
> plot(0:10, pbinom(0:10, 10, 0.65), type="s", xlab="x", ylab="p(x)")
```

La funzione *barplot* accetta vari argomenti fra cui il colore con cui riempire le barre e le etichette (*names.arg*) da porre sotto ogni barra. Per la funzione *plot* l'unico argomento usato nell'esempio è *type* che imposta un grafico a scala. I due plot sono mostrati in Fig. 1.6.

1.3.2 Distribuzione di Poisson

La probabilità che un evento casuale si verifichi x volte quando in media si verifica *lambda* volte è dato dalla distribuzione di Poisson. In R vi si accede con:

```
dpois(x, lambda, log = FALSE)
ppois(q, lambda, lower.tail = TRUE, log.p = FALSE)
qpois(p, lambda, lower.tail = TRUE, log.p = FALSE)
```

dove *lambda* identifica la media della distribuzione. Ad esempio il diagramma della distribuzione di Poisson con media 2 si ottiene con:

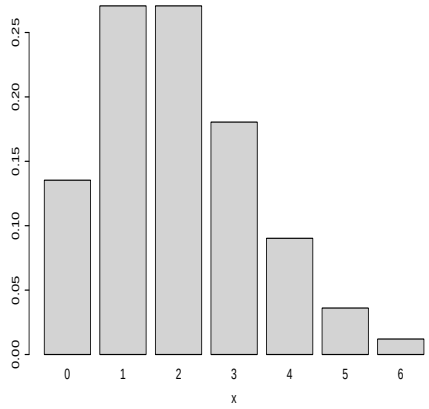

Figura 1.7: Istogramma della densità della distribuzione di Poisson di media 2.

```
> barplot(dpois(0:6,2), col="lightgrey", names.arg=0:6, xlab="x")
```

ed è presentato in Fig. 1.7.

1.3.3 Distribuzione binomiale negativa

Questa distribuzione permette di calcolare la probabilità che un numero di fallimenti x avvenga prima del successo *size* in una sequenza di prove bernoulliane per la quali la probabilità del singolo successo è *prob*.

```
dnbinom(x, size, prob, mu, log = FALSE)
pnbinom(q, size, prob, mu, lower.tail = TRUE, log.p = FALSE)
qnbinom(p, size, prob, mu, lower.tail = TRUE, log.p = FALSE)
```

Ad esempio la probabilità che lanciando una moneta si ottenga la quinta testa prima della seconda croce è data da:

```
> dnbinom(5, 2, 0.5)

[1] 0.046875
```

dato che la probabilità di ottenere croce sul singolo lancio è 0.5.

1.3.4 Distribuzione normale

Fra le distribuzioni continue particolare importanza ha la distribuzione normale.

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

Con ovvia notazione *mean* è la media della distribuzione e *sd* la sua deviazione standard. La distribuzione normale di media 0 e varianza 1 si dice standard e si indica con la notazione $N(0, 1)$.

1.3.5 Distribuzione χ^2

La somma dei quadrati di n variabili casuali indipendenti $\sim N(0, 1)$ è distribuita secondo una distribuzione χ^2 a n gradi di libertà.

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

df è il numero di gradi di libertà. È anche possibile calcolare la distribuzione di χ^2 non centrale, specificando un valore positivo per il parametro di non centralità *ncp*. Per un esempio del suo utilizzo si veda la sezione 5.3.

1.3.6 Distribuzione t

Il rapporto fra una variabile casuale normale standard e la radice di una variabile casuale $\sim \chi^2(n)$ divisa per n segue una distribuzione di t di Student a n gradi di libertà.

```
dt(x, df, ncp=0, log = FALSE)
pt(q, df, ncp=0, lower.tail = TRUE, log.p = FALSE)
qt(p, df,        lower.tail = TRUE, log.p = FALSE)
```

df rappresenta il numero di gradi di libertà. Specificando un valore positivo per *ncp* si può calcolare la distribuzione di t non centrale.

1.3.7 Distribuzione F

Il rapporto di due variabili casuali indipendenti distribuite rispettivamente $\sim \chi^2(df1)$ e $\sim \chi^2(df2)$, ognuna divisa per i rispettivi gradi di libertà, è distribuito secondo la distribuzione F a ($df1$, $df2$) gradi di libertà.

```
df(x, df1, df2, log = FALSE)
pf(q, df1, df2, ncp=0, lower.tail = TRUE, log.p = FALSE)
qf(p, df1, df2,        lower.tail = TRUE, log.p = FALSE)
```

$df1$ e $df2$ sono i gradi di libertà di numeratore e denominatore.

1.3.8 Distribuzione normale multivariata

Per lo studio di campioni su cui sono misurate più variabili è spesso necessario ricorrere alla funzione di distribuzione normale multivariata, generalizzazione della distribuzione normale in più dimensioni.

Si supponga di misurare su un campione p variabili, tutte di distribuzione normale e fra loro indipendenti. Sia $\boldsymbol{\mu}$ il vettore che contiene le medie di dette variabili e Σ la loro matrice di covarianza. La densità:

$$g(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}}|\Sigma|^{-1/2}\exp[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})] \quad , \quad |\Sigma| = \det(\Sigma) \tag{1.1}$$

è detta densità normale multivariata.

L'ipotesi di indipendenza tra le variabili è fondamentale in quanto, se esse risultano tra loro dipendenti, è possibile che una ad una siano normalmente distribuite, ma che nell'insieme non soddisfino l'ipotesi di normalità multivariata. Un classico esempio coinvolge le variabili $X \sim N(0,1)$ e Y così definita:

$$Y = \begin{cases} X & \text{se } |X| \geq 1 \\ -X & \text{se } |X| < 1 \end{cases}$$

in questo caso sia X che Y hanno distribuzione normale, ma la loro distribuzione congiunta non è normale multivariata.

La funzione di distribuzione normale multivariata è accessibile in R dopo l'installazione della libreria aggiuntiva *mvtnorm*, scaricabile dal sito della distribuzione [44]. Tale libreria implementa la funzione *dmvnorm*, che ritorna la densità di probabilità normale multivariata e la funzione *rmvnorm* che permette di generare dati da una distibuzione specificata.

Ad esempio per valutare la densità normale multivariata nel punto $\mathbf{x} = (0, 0)$ nel caso di due variabili di media $\boldsymbol{\mu} = (1, 1)$ con matrice di covarianza:

$$\Sigma = \begin{pmatrix} 1.0 & 0.5 \\ 0.5 & 1.0 \end{pmatrix}$$

si usa la chiamata:

```
> library(mvtnorm)
> Sigma <- matrix(c(1,0.5,0.5,1), nrow=2)  # matrice di covarianza
> Sigma
     [,1] [,2]
[1,]  1.0  0.5
[2,]  0.5  1.0
> dmvnorm(x=c(0,0), mean=c(1,1), sigma=Sigma)
[1] 0.0943539
```

la funzione accetta tre argomenti: il punto in cui valutare la densità, il vettore delle medie delle variabili, la matrice di covarianza delle variabili.

Se si volesse invece simulare un campione di 5 osservazioni provenienti dalla distribuzione con i parametri dati in precedenza si potrebbe usare la chiamata:

```
> rmvnorm(n=5, mean=c(1,1), sigma=Sigma)
           [,1]       [,2]
[1,]  1.7516475  0.3527596
[2,]  1.7580599  1.0073010
[3,]  1.1606733  0.4987357
[4,] -1.3068941 -1.3132264
[5,]  0.6290436  0.8559206
```

in questo caso il primo argomento specifica la dimensione del campione da generare.

Capitolo 2

Statistica classica: test t, test χ^2 e correlazione

2.1 Test t

Si abbiano due campioni A e B e si voglia stabilire se essi possano o meno essere stati estratti da popolazioni di uguale media. A seconda del problema in esame si hanno i seguenti casi:

1. Test t a un solo campione
2. Test t per dati appaiati
3. Test t assumendo eguale varianza
4. Test t a varianza diversa

2.1.1 Test t a un solo campione

Questo test si usa quando si vuole verificare se un campione possa o meno essere stato estratto da una popolazione di media nota μ.

Esempio

In Geochimica il rapporto tra gli isotopi dell'ossigeno O^{18} e O^{16}, entrambi stabili, è utilizzato come tracciatore della temperatura alla quale si è formato un cristallo. Per semplicità tale rapporto è espresso in relazione al rapporto isotopico dell'acqua nella cosidetta notazione delta (δ). Quindi per definizione δO^{18} dell'oceano è 0, mentre è noto che δO^{18} del mantello terrestre è 5.3.

Si vuole testare l'ipotesi che il campione di zirconi A provenga da una popolazione di rapporto isotopico tipico del mantello. Per prima cosa si inseriscono i dati del campione in un vettore:

```
> A <- c(6.0, 4.4, 5.0, 5.3, 5.2, 5.8, 5.6)
```

Quindi si esegue il test:

```
> t.test(A, mu=5.3)

        One Sample t-test

data:  A
t = 0.1406, df = 6, p-value = 0.8928
alternative hypothesis: true mean is not equal to 5.3
95 percent confidence interval:
```

```
 4.831345 5.825798
sample estimates:
mean of x
 5.328571
```

Dal test si conclude che non si può escludere l'ipotesi che il campione provenga da una popolazione di rapporto isotopico $\delta O^{18} = 5.3$. In output vengono forniti dapprima il valore campionario della statistica t, i suoi gdl e il valore P. Vengono poi riportati l'intervallo di confidenza al 95% per la media nella popolazione e, nell'ultima riga, la media campione.

2.1.2 Test t per dati appaiati

Nel caso di dati appaiati il comando:

```
> t.test(A, B, paired=TRUE)
```

testa l'ipotesi che le medie dei due campioni siano significativamente diverse. Il test è a due code. Se si vuole verificare l'ipotesi che la media del primo campione sia significativamente maggiore di quella del secondo, si fa uso dell'opzione *alt*:

```
> t.test(A, B, paired=TRUE, alt="g")
```

Se invece si vuole testare l'ipotesi che la media di A sia significativamente minore di quella di B, la sintassi è:

```
> t.test(A, B, paired=TRUE, alt="l")
```

Esempio

A un gruppo di volontari vengono misurati i battiti cardiaci a riposo (A) e dopo una sessione di ascolto di musica classica (B). Stabilire se vi è evidenza del fatto che l'ascolto di musica produce un abbassamento del numero dei battiti.

```
> A <- c(77,74,81,65,71)
> B <- c(72,72,82,62,69)
> t.test(A, B, paired=TRUE, alt="g")

        Paired t-test

data:  A and B
t = 2.2691, df = 4, p-value = 0.04291
alternative hypothesis: true difference in means is greater than 0
[...]
```

Dal test si conclude che vi è evidenza significativa del fatto che l'ascolto di musica classica produce un abbassamento del ritmo cardiaco.

2.1.3 Test t assumendo eguale varianza

Nel caso di campioni indipendenti, se le varianze di A e B non sono significativamente diverse si usa la sintassi:

```
> t.test(A, B, var.equal=TRUE)
```

che esegue un test t bidirezionale assumendo uguale varianza. Per i test monodirezionali la sintassi è la stessa introdotta al punto precedente.

Per controllare l'uguaglianza della varianze si può ricorrere al test F:

```
> var.test(A, B)
```

se il risultato di questo test è non significativo si può assumere l'uguaglianza delle varianze.

2.1.4 Test t a varianza diversa

Se il test F di cui al punto precedente fornisce risultato significativo, si procede ad un test t assumendo varianze differenti:

```
> t.test(A, B)
```

in cui i gradi di libertà vengono calcolati secondo la formula di Welch:

$$\nu = \frac{(k_A + k_B)^2}{\frac{k_A^2}{n_A - 1} + \frac{k_A^2}{n_A - 1}}$$

con:

$$k_A = \frac{s_A^2}{n_A} \qquad k_B = \frac{s_B^2}{n_B}.$$

Esempio

Due classi di studenti di pari età vengono sottoposti ad un test per valutare il loro QI. Si stabilisca se vi è differenza significativa fra le due classi.

Si inseriscono i valori dei QI nei vettori A e B:

```
> A <- c(95,101,102,102,103,104,105,106,106,110,113,113,114,116,125)
> B <- c(95,97,97,98,99,99,100,100,100,100,101,101,102,103,103,106,106,107,107,108)
```

Come primo passo si controlla l'omogeneità delle varianze:

```
> var.test(A, B)

        F test to compare two variances

data:  A and B
F = 4.0432, num df = 14, denom df = 19, p-value = 0.005492
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
  1.527496 11.566386
```

Dal valore P del test si ha evidenza del fatto che le varianze sono significativamente diverse. Si esegue quindi il confronto fra le medie dei due gruppi con la chiamata:

```
> t.test(A, B)

        Welch Two Sample t-test

data:  A and B
t = 2.945, df = 19.189, p-value = 0.008252
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
  1.801414 10.631919
sample estimates:
mean of x mean of y
 107.6667  101.4500
```

Si evidenzia differenza altamente significativa fra le due classi. Dall'ultima riga in output si vede che il QI degli studenti della classe A (identificata dall'etichetta *mean of x*) risulta superiore.

2.2 Test χ^2

2.2.1 Goodness-of-fit

Si abbia una serie A di frequenze osservate e si voglia verificare se esse seguano o meno una certa legge di distribuzione. Il problema più semplice si ha quando si vuole controllare che la distribuzione sia uniforme. In tal caso la sintassi è:

```
> chisq.test(A)
```

Se invece si vuole testare l'iptesi che le frequenze siano distribuite secondo una particolare distribuzione è necessario costruire il vettore delle frequenze teoriche (che avrà la stessa lunghezza del vettore delle frequenze osservate e sarà costruito a partire dalla funzione di distribuzione teorica ipotizzata). Per una distribuzione poissoniana si può procedere come nell'esempio seguente.

Esempio

Per valutare la bontà di un nuovo apparato industriale si conta il numero di microfratture per pezzo prodotto. Si controllano 45 campioni e risulta che 20 di essi non contengono microfratture, 15 ne contengono 1 e 10 ne contengono 2. Testare l'ipotesi che la distribuzione di microfratture sia casuale (e quindi segua una distribuzione di Poisson).

```
> A <- c(20, 15, 10)          # la serie di frequenze osservate
> fratture <- 0:2             # il numero di difetti corrispondenti trovate
> mu <- sum(A*fratture)/sum(A)     # la media della poissoniana
> prob <- dpois(fratture, mu)      # vettore frequenze teoriche
> prob[3] <- 1 - sum(prob[1:2])    # correzione sull'ultimo valore
```

La correzione sull'ultimo valore è necessaria ai fini della corretta normalizzazione, in modo tale che la somma delle frequenze teoriche sia 1.

```
> teo <- prob*sum(A)              # frequenze teoriche
> chisq <- sum((A - teo)^2/teo)  # valore campionario di chi quadro
> chisq
[1] 0.4676315

> gdl <- 3 - 1 - 1                # perdo 2 gdl, 1 per normalizzazione
                                  # 1 per aver stimato mu sul campione
> p <- 1 - pchisq(chisq, gdl)     # valore p del test
> p

[1] 0.4940788
```

da cui si conclude che la distribuzione teorica si accorda bene ai dati. In Fig. 2.1, ottenuta con la chiamata:

```
> barplot(rbind(A,teo), beside=TRUE, names=0:2, legend=c("Osservate",
+  "Teoriche (Poisson)"))
```

sono rappresentate, per un agevole paragone, le serie di frequenze osservate e teoriche.

Si noti che la chiamata diretta a:

```
> chisq.test(A, p=prob)

        Chi-squared test for given probabilities

data:  A
X-squared = 0.4676, df = 2, p-value = 0.7915
```

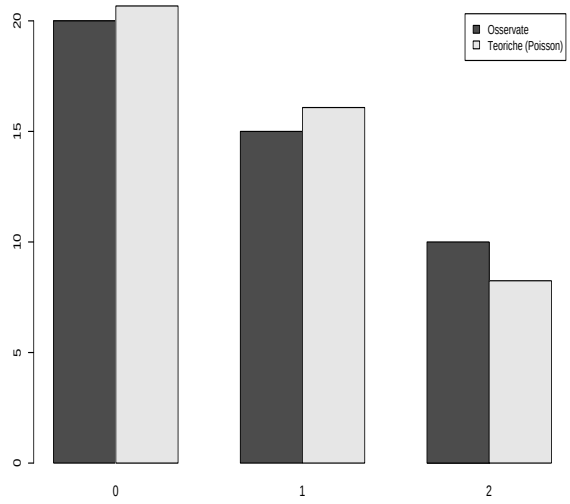

Figura 2.1: Confronto fra frequenze di microfratture osservate e teoriche nell'ipotesi di legge poissoniana.

fornisce il valore corretto di χ^2, ma valuta in maniera scorretta i gradi di libertà dato che non può tener conto del fatto che 1 gdl viene perso a causa del fatto che si è stimato il parametro *mu* sul campione. □

A volte è necessario accorpare più classi di frequenza perché il test χ^2 mantenga la sua validità. Ad esempio, supponendo che il vettore A abbia lunghezza 10 (così come il vettore *teo*) e che si debbano accorpare le classi da 7 a 10 comprese, i comandi saranno i seguenti:

```
> A <- c(A[1:6], sum(A[7:10]))
> teo <- c(teo[1:6], sum(teo[7:10]))
> chisq <- sum((A - teo)^2/teo)
[...]
```

avendo cura di ricalcolare correttamente i gradi di libertà.

2.2.2 Tabelle di contingenza

Se i dati raccolti possono essere classificati secondo due chiavi di classificazione è possibile costruire una tabella di contingenza per verificare l'indipendenza delle due chiavii.

Ad esempio si supponga di classificare 103 pazienti sottoposti a 3 diverse cure A, B e C (prima chiave di classificazione) in base anche a una seconda chiave che rappresenta la risposta al trattamento ("+" se c'è miglioramento; "=" se non c'è nessun miglioramento). Se i conteggi sono quelli riepilogati in Tab. 2.1 l'analisi si può svolgere nel modo seguente. Per prima cosa si inseriscono i dati in una tabella di tre righe e due colonne:

```
> Tab <- matrix(c(10,21, 7,30, 18,17), nrow=3, byrow=TRUE)
> Tab
     [,1] [,2]
[1,]   10   21
[2,]    7   30
[3,]   18   17
```

cura	risposta	
	+	=
A	10	21
B	7	30
C	18	17

Tabella 2.1: Risposta di 103 pazienti a tre trattamenti diversi.

La funzione *matrix* viene usata per costruire una tabella di tre righe (opzione *nrows*) inserendo i valori per riga ($byrow = TRUE$). Il test per mettere in luce l'associazione fra le chiavi di classificazione si esegue con il comando:

```
> chisq.test(Tab)

        Pearson's Chi-squared test

data:  Tab X-squared = 8.5321, df = 2, p-value = 0.01404
```

Per capire da dove trae origine la differenza significativa è possibile visualizzare la tabella di contingenza teorica:

```
> res <- chisq.test(Tab)
> res$expected
         [,1]     [,2]
[1,] 10.53398 20.46602
[2,] 12.57282 24.42718
[3,] 11.89320 23.10680
```

ed eventualmente valutare il contributo al valore di χ^2 campionario portato da ciascuna cella:

```
> res$residuals^2

        [,1]       [,2]
A 0.02706814 0.01393213
B 2.47011283 1.27138160
C 3.13565286 1.61393897
```

2.2.3 Confronto di una frequenza teorica con una sperimentale

Su un campione si misuri la frequenza sperimentale p con cui si verifica un evento di interesse. Sia π la frequenza attesa in base a ipotesi teoriche. Le due frequenze π e p possono essere paragonate con il test esatto, come nell'esempio seguente.

Esempio

100 persone vengono sottoposte a un test di cultura generale e 77 di esse lo superano. Tale proporzione è in accordo con la proporzione attesa $\pi = 0.85$?

Il test si esegue chiamando la funzione *binom.test* che accetta tre opzioni: il numero di successi nel campione, la dimensione capionaria e la proporzione teorica attesa:

```
> binom.test(77, 100, p=0.85)

        Exact binomial test

data:  77 and 100
number of successes = 77, number of trials = 100, p-value = 0.03431
```

```
alternative hypothesis: true probability of success is not equal to 0.85
95 percent confidence interval:
 0.6751413 0.8482684
```

Si conclude che vi è differenza significativa fra la proporzione teorica e quella osservata. Si noti che il test visualizza nell'ultima linea l'intervallo di confidenza della proporzione osservata.

2.2.4 Test di McNemar

Si usa questo test per verificare l'accordo fra due diverse tecniche diagnostiche.

Esempio

Due test diagnostici A e B vengono provati su 500 pazienti. Ogni persona viene analizzata con entrambi i test. Ci si domanda se i due test sono equivalenti.

Si inizia l'analisi inserendo i dati in una matrice:

```
> dati <- matrix(c(95, 30, 20, 355), nrow=2,
        dimnames=list("B" = c("+", "-"),"A" = c("+", "-")))
> dati

   A
B    +   -
  + 95  20
  - 30 355
```

L'opzione *dimnames* serve a specificare le etichette di riga e colonna per comodità di lettura. Si noti che in questo caso la matrice viene costruita per colonne. Le informazioni utili per il test sono ricavate dai due elementi fuori diagonale, che risultano popolati in numero sufficiente per usare il test di McNemar:

```
> mcnemar.test(dati, correct=FALSE)

        McNemar's Chi-squared test

data:  dati
McNemar's chi-squared = 2, df = 1, p-value = 0.1573
```

Si conclude che i due metodi diagnostici sono equivalenti. L'opzione $correct = FALSE$ disabilita la correzione di continuità.

2.3 Test di correlazione

Per verificare la correlazione fra due serie di dati A e B (campione bivariato) si ricorre al test di Pearson, come nell'esempio seguente.

```
> A <- rnorm(10)
> B <- rnorm(10)    # due serie di 10 numeri casuali ~ N(0, 1)
> cor.test(A, B)

        Pearson's product-moment correlation

data:  A and B
t = 0.7966, df = 8, p-value = 0.4487
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
```

```
 -0.4323226  0.7693952
sample estimates:
      cor
0.2710966
```

In output si ottiene oltre alla significatività della correlazione anche il suo intervallo di confidenza (di default al 95%). Il test è bidirezionale, ma è possibile scegliere un test monodirezionale:

```
> cor.test(A, B, alt="g")
> cor.test(A, B, alt="l")
```

eseguono rispettivamente un test per associazione positiva e negativa.

2.3.1 Differenza fra coefficienti di correlazione

Se si vuole stabilire se sia significativa la differenza fra due coefficienti di correlazione $r1$ e $r2$, calcolati su due campioni biavriati indipendenti di taglia $n1$ e $n2$ rispettivamente, si ricorre alla trasformazione di Fisher su $r1$ e $r2$ e si esegue un test z. Il risultato della trasformazione di Fisher:

$$z' = \frac{1}{2} \log \frac{1+r}{1-r}$$

è una variabile normale a media 0 e deviazione standard:

$$\sigma_{z'} = \sqrt{\frac{1}{n-3}}$$

dove n è la taglia del campione bivariato su cui si calcola il coefficiente di correlazione. La trasformazione di Fisher è valida per campioni grandi, ma può essere usata senza grossi errori già per $n > 5$.

Si ha quindi che un intervallo di confidenza bilaterale di livello α per il coefficiente $r1$ è dato da:

$$\tanh(z'_1 \pm z_{1-\alpha/2}\, \sigma_{z'_1})$$

La trasformazione tanh è l'inversa della trasformazione di Fisher e riporta i valori z' sulla scala di r. Come detto in precedenza, in R l'informazione sull'intervallo di confidenza è fornita dalla funzione *cor.test*.

Per testare la differenza fra i due coefficienti $r1$ e $r2$ si può ricorrere al test z:

$$z_{diff} = \frac{z'_1 - z'_2}{\sigma_{z'_1,\, z'_2}} \qquad \sigma_{z'_1,\, z'_2} = \sqrt{\frac{1}{n1-3} + \frac{1}{n2-3}}$$

L'intervallo di confidenza bilaterale di livello α sulla differenza dei coefficienti $r1$ e $r2$ è quindi:

$$\tanh(z'_1 - z'_2 \pm z_{1-\alpha/2}\, \sigma_{z'_1,\, z'_2})$$

Esempio

Si vuole vedere se, per due diverse varietà di grano, la produttività, in tonnellate per ettaro, è correlata allo stesso modo con l'umidità percentuale media registrata nel periodo seguente la semina.

Si inseriscono i dati relativi alla prima varietà nei vettori *acqua*1 e *prod*1, e quelli relativi alla seconda nei vettori *acqua*2 e *prod*2:

```
> acqua1 <- c(47.6,44.9,47.4,56.3,54.3,45.3,49.4,58.4,54.5,50.1,51.0,50.1)
> prod1 <- c(18.3,18.8,21.2,20.4,20.1,20.5,22.0,19.7,23.5,18.4,19.0,19.8)
> acqua2 <- c(51.7,47.6,42.8,46.7,34.5,50.3,43.6,45.7,53.0,38.7,52.9,45.5)
> prod2 <- c(21.3,19.3,23.5,23.4,18.1,18.8,17.9,21.5,22.8,17.7,21.9,21.6)
```

Le correlazioni si esaminano con le chiamate:

```
> cor.test(acqua1, prod1)

        Pearson's product-moment correlation

data:  acqua1 and prod1
t = 0.7714, df = 10, p-value = 0.4583
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.3899620  0.7137993
sample estimates:
      cor
0.2369744

> cor.test(acqua2, prod2)

        Pearson's product-moment correlation

data:  acqua2 and prod2
t = 1.7956, df = 10, p-value = 0.1028
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.1118151  0.8319256
sample estimates:
      cor
0.4937723
```

Nessuna delle due correlazioni è significativa, ma si nota che la correlazione risulta più forte per la seconda varietà. Per vedere se questa differenza raggiunge la significatività si ricorre alla trasformazione di Fisher sui due coefficienti di correlazione e si calcola l'errore standard della differenza delle variabili trasformate:

```
> cor1 <- cor(acqua1, prod1)
> cor2 <- cor(acqua2, prod2)
> z1 <- 1/2*log( (1 + cor1)/(1 - cor1) )
> z2 <- 1/2*log( (1 + cor2)/(1 - cor2) )
> sigma <- sqrt(1/9 + 1/9)
```

Il risultato del test z è:

```
> z <- (z2-z1)/sigma
> z
[1] 0.6352732
```

valore chiaramente non significativo in quanto il percentile di riferimento per la significatività bilaterale è 1.960. Il valore P del test bilaterale è:

```
> 2*(1 - pnorm(z))
[1] 0.5252502
```

Non vi è quindi evidenza che la correlazione fra umidità media del terreno e produttività sia diversa per le due varietà di grano.

2.3.2 Correlazione fra più variabili

Se p variabili vengono misurate su uno stesso campione può essere interessante stabilire se fra coppie di tali variabili esista o meno correlazione significativa. Si calcola cioè quella che viene detta matrice

di correlazione, ossia la matrice $p \times p$ il cui elemento i, j è il coefficiente di correlazione fra le variabili i e j (la matrice è quindi simmetrica, con elementi sulla diagonale pari a 1).

Per stabilire la significatività di queste correlazioni è però necessario introdurre una correzione dovuta al fatto che si conducono più test simultaneamente (confronti multipli). La ragione di questa correzione è chiara se si considera un esperimento a tre variabili. La probabilità di non rifiutare correttamente l'ipotesi nulla che non vi sia correlazione fra le prime due variabili è $1 - \alpha$ (dove α è il livello del test scelto dallo sperimentatore). Analogamente anche la probabilità di non rifiutare correttamente l'ipotesi nulla che non vi sia correlazione fra la prima e la terza variabile è $1 - \alpha$, e così pure per quanto riguarda la seconda e la terza variabile. Se si considerano indipendenti tali confronti la probabilità congiunta di non rifiutare l'ipotesi nulla che non esista correlazione fra nessuna delle coppie di variabili è:

$$(1 - \alpha)^3 < 1 - \alpha.$$

Nel caso in esame scegliendo $\alpha = 0.05$ si ha $(1-\alpha)^3 \sim 0.86$, quindi se ogni test viene condotto a livello del 5% il test globale è a livello del 14%. Si rischia quindi di dichiarare significative più differenze di quante non sia opportuno. Il modo più comune per fronteggiare questo comportamento indesiderato è ricorrere alla correzione di Bonferroni che modifica il livello a cui eseguire ogni singolo confronto in modo tale che il livello del test globale rimanga α. Questa correzione richiede che ogni test singolo sia condotto a livello α/N dove:

$$N = \frac{p(p-1)}{2}$$

è il numero di confronti che si eseguono. Nel caso in esame ogni test parziale deve essere eseguito a livello $\alpha = 0.05/3 = 0.017$, cioè si dichiareranno significative solo le correlazioni con valore P inferiore a 0.017 e altamente significative quelle con valore P minore di $0.01/3 = 0.0033$. La tecnica presentata è equivalente alla correzione dei valori P dei singoli test: si moltiplica per N ogni valore P (con la condizione di porre pari a 1 eventuali valori che, dopo la correzione, superino tale soglia) e si confrontano i valori ottenuti con il livello α.

Altri classici metodi di correzione per confronti multipli sono dovuti a Holm e a Hochberg. Mentre il secondo è applicabile solo nel caso di test indipendenti il primo è del tutto generale. Data la sua superiore potenza rispetto al metodo di Bonferroni è da considerarsi il metodo d'elezione. Il suo funzionamento è il seguente: si supponga di avere svolto k test simultanei e di aver calcolato per ogni test il rispettivo valore P. Si dispongono tali valori in ordine crescente:

$$P_1 \leq P_2 \leq \ldots \leq P_k.$$

Si calcolano quindi i valori P corretti:

$$P'_i = (k - i + 1)\, P_i.$$

Procedendo da sinistra a destra si controlla poi se alcuni di tali valori risultano più piccoli di quelli che compaiono immediatamente alla sinistra. In caso ciò avvenga, si corregge il minore sostituendolo con il valore che compare alla sua sinistra. Si confrontano quindi i valori P' così ottenuti con il livello di significatività α.

Esempio

In un esperimento si conducono 5 test tra loro non indipendenti. I valori P di tali test sono nel vettore seguente:

```
> p <- c(0.001, 0.2, 0.025, 0.011, 0.07)
```

Se si vuole correggere per l'effetto di molteplicità in R si può ricorrere alla funzione *p.adjust* che è in grado di eseguire molti tipi di correzione per confronti multipli. Ad esempio le correzioni di Bonferroni e di Holm si ottengono con le chiamate seguenti:

```
> p.adjust(p, "bonferroni")
[1] 0.005 1.000 0.125 0.055 0.350

> p.adjust(p, "holm")
[1] 0.005 0.200 0.075 0.044 0.140
```

La funzione accetta come input due argomenti: un vettore contenete i valori P da correggere e il nome della tecnica da utilizzare (si rimanda alla pagina del manuale della funzione per ulteriori possibili scelte).

Nel caso di correzione di Bonferroni i valori P sono semplicemente moltiplicati per il numero di test (in questo caso per 5). Si vede che il metodo di Holm è più potente ritornando valori P inferiori o pari a quelli del metodo di Bonferroni.

Capitolo 3

Regressione lineare e non lineare

3.1 Regressione lineare semplice

Uno dei campi in cui le potenzialità di R si esprimono maggiormente è quello della regressione, lineare e non lineare. Una trattazione esaustiva di modelli di regressione lineare e ANOVA in R si trova in [26] (disponibile presso le pagine web del progetto R [44]).

Si definisce il modello lineare:

$$y_i = \alpha + \beta x_i + \varepsilon_i$$

con l'ipotesi che gli errori ε_i siano valori assunti da variabili casuali indipendenti di legge $N(0, \sigma^2)$. Il problema è quello di stimare i parametri α, β e σ^2.

Si supponga ad esempio di voler stabilire se vi sia dipendenza lineare fra l'altezza di un gruppo di piante di pomodoro (in cm) e il peso medio dei frutti raccolti (in g). Si inseriscono i dati nei vettori *peso* e *altezza*:

```
> peso <- c(60,65,72,74,77,81,85,90)
> altezza <- c(160,162,180,175,186,172,177,184)
```

Per fittare il modello lineare si procede nel modo seguente:

```
> mod <- lm(peso ~ altezza)
```

Il modello statistico in R si esprime utilizzando l'operatore "∼" che mette in relazione la variabile dipendente con i regressori. Per visualizzare il risultato dell'analisi si utilizza l'istruzione:

```
> summary(mod)
```

che fornisce un output particolarmente ricco. Innanzitutto presenta alcune informazioni sul modello utilizzato e sui residui:

```
Call:
lm(formula = peso ~ altezza)

Residuals:
   Min     1Q Median     3Q    Max
-7.860 -4.908 -1.244  7.097  7.518
```

Segue la parte riguardante il fit del modello e la significatività dei parametri:

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -62.8299    49.2149  -1.277   0.2489
altezza       0.7927     0.2817   2.814   0.0306 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Infine la parte riguardante i calcoli del coefficiente di correlazione (viene fornito R^2) e quello di σ, seguita dalle informazioni relative all'ANOVA del modello:

```
Residual standard error: 7.081 on 6 degrees of freedom
Multiple R-Squared: 0.569,      Adjusted R-squared: 0.4972
F-statistic: 7.921 on 1 and 6 DF,  p-value: 0.03058
```

In alternativa è possibile, una volta fittato il modello, richiedere la tabella ANOVA:

```
> anova(mod)
```

il cui output, nel caso in esame, è il seguente:

```
Analysis of Variance Table

Response: peso
          Df Sum Sq Mean Sq F value  Pr(>F)
altezza    1 397.15  397.15  7.9207 0.03058 *
Residuals  6 300.85   50.14
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Dall'analisi si conclude che la dipendenza lineare fra le due variabili è significativa.

3.1.1 Analisi dei residui

Per l'analisi dei residui è necessario calcolare i valori di leva, ossia i valori sulla diagonale della hat-matrix H (vedi Sec. 3.2). Nel caso di regressione semplice essi si possono scrivere come:

$$h_i = \frac{(x_i - \bar{x})^2}{d_x^2} + \frac{1}{n}$$

dove n è la taglia dei campioni e d_x^2 la devianza delle x. In R essi si valutano con l'espressione:

```
[...]
> x <- model.matrix(mod)
> lev <- hat(x)              # calcola i valori di leva
```

I residui standardizzati z_i si definiscono a partire dai residui ordinari $\hat{\varepsilon}_i$:

$$z_i = \frac{\hat{\varepsilon}_i}{\hat{\sigma}\sqrt{1 - h_i}}.$$

La funzione *lm* calcola, fra le altre quantità, i residui ordinari $\hat{\varepsilon}_i$ che sono disponibili come:

```
> res <- mod$residuals
```

I residui standardizzati si ottengono con la chiamata:

```
> res.standard <- rstandard(mod)
```

e possono essere plottati contro i valori stimati della variabile dipendente:

```
> plot(mod$fitted.values, res.standard)
> abline(h=0)      # livello di 0 per facilitare la lettura
```

Per verificare che i residui z_i siano distribuiti in modo normale è possibile esaminare un Q-Q plot:

```
> qqnorm(res.standard)
> abline(0, 1)   # retta a 45 gradi per residui standardizzati
```

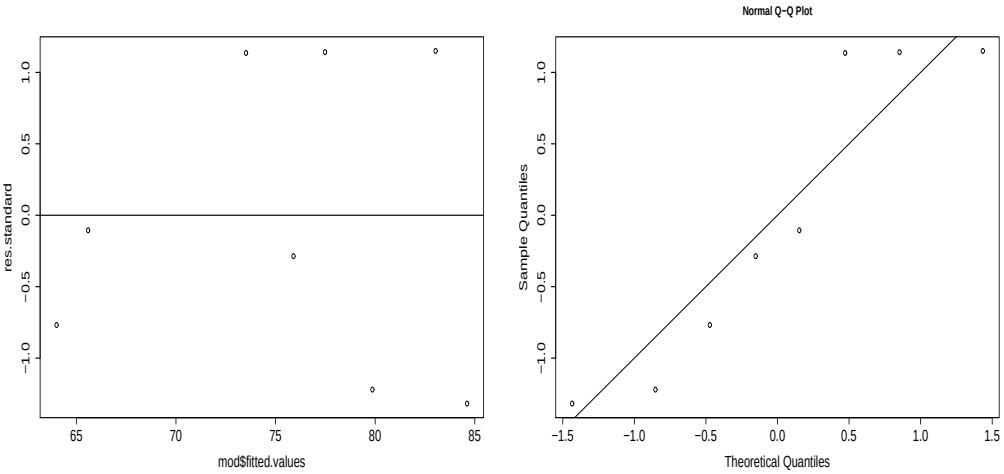

Figura 3.1: Analisi dei residui nel caso dell'esempio. A sinistra il grafico dei residui standardizzati contro i valori fittati dal modello non evidenzia nessun problema di eteroschedasticità o non linearità. A destra il Q-Q plot dei residui standardizzati.

I grafici concernenti l'analisi dei residui si trovano in Fig. 3.1.

Per un'analisi più completa si può eseguire un test di normalità di Shapiro-Wilk:

```
> shapiro.test(res.standard)

        Shapiro-Wilk normality test

data:  res.standard
W = 0.8543, p-value = 0.1053
```

che non mette in evidenza una non normalità significativa.

3.1.2 Intervallo di confidenza della regressione

L'intervallo di confidenza bilaterale della retta di regressione è dato dall'espressione:

$$y = a + bx \pm t_{1-\alpha/2}(n-2)\,\hat{\sigma}\,\sqrt{\frac{1}{n} + \frac{(x-\bar{x})^2}{d_x^2}} \tag{3.1}$$

dove $t_{1-\alpha/2}(n-2)$ è il valore critico (a livello α) della distribuzione t a $n-2$ gradi di libertà. L'espressione sotto radice, calcolata per $x = x_i$ è il valore di leva i-esimo.

Si consideri l'esempio trattato nella Sec. 3.1.

```
> peso <- c(60,65,72,74,77,81,85,90)
> altezza <- c(160,162,180,175,186,172,177,184)
> mod <- lm(peso ~ altezza)
```

Per rappresentare graficamente il modello con il suo intervallo di confidenza si calcolano i valori stimati dalla retta di regressione e quelli che delimitano il suo intervallo di confidenza. Per far ciò si può usare la funzione *predict*. Come primo passo si definisce un grigliato su cui calcolare i valori previsti sia per la retta di regressione sia per i due rami d'iperbole che delimitano l'intervallo di confidenza:

```
> grid <- seq(min(altezza), max(altezza), 2)   # grigliato di passo 2
```

Quindi si calcolano i valori previsti dal modello nei punti di *grid*, e i rispettivi errori standard:

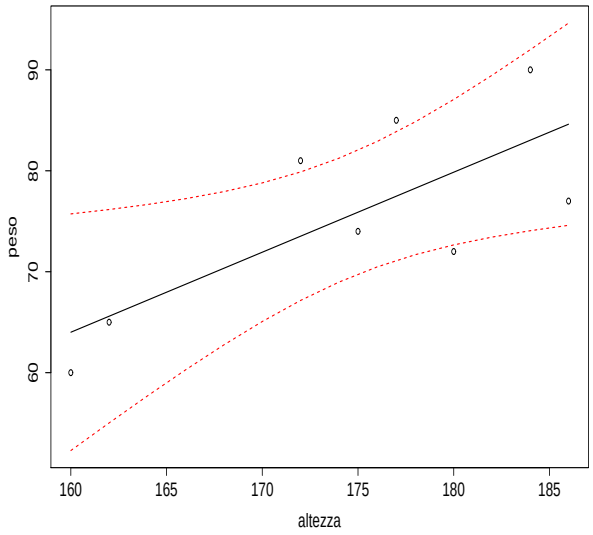

Figura 3.2: Intervallo di confidenza della retta di regressione di Sec. 3.1.2.

```
> yp <- predict(mod, data.frame(altezza=grid), se=TRUE)
```

Si noti che il secondo argomento deve essere necessariamente di tipo *data.frame* e i suoi argomenti devono avere esattamente lo stesso nome dei predittori del modello (in questo caso il data frame contiene la variabile *altezza* i cui valori sono quelli inseriti in *grid*). Il grafico di Fig. 3.2 si realizza quindi con le chiamate:

```
> tc <- qt(0.975, length(altezza) - 2)
> y <- yp$fit
> ysup <- yp$fit + tc*yp$se
> yinf <- yp$fit - tc*yp$se
> matplot(grid, cbind(y, yinf, ysup), lty=c(1,2,2), col=c(1,2,2),
+         type="l", xlab="altezza", ylab="peso")
> points(altezza,peso)
```

In alternativa, se non interessa esplicitamente il valore degli errori standard, è possibile costruire il plot con una procedura più diretta:

```
> yp <- predict(mod, data.frame(altezza=grid), interval="confidence")
> matplot(grid, yp, lty=c(1,2,2), col=c(1,2,2), type="l",
+         xlab="altezza", ylab="peso")
```

Per effetto della espressione sotto radice in Eq. 3.1 l'intervallo di confidenza assume larghezza minima per $x = \bar{x}$, come si verifica dal grafico e dal calcolo della media dei valori del vettore *altezza*:

```
> mean(altezza)

[1] 174.5
```

3.2 Regressione multipla

Si abbiano n osservazioni di una variabile dipendente y associate ai valori dei predittori $x_1, \ldots, x_r$. Si consideri il modello:

$$y_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_r x_{ir} + \varepsilon_i \qquad i = 1, \ldots, n \tag{3.2}$$

dove β_i sono i parametri da stimare e ε il termine d'errore. Particolarmente conveniente è la notazione matriciale:

$$y = X\beta + \varepsilon \tag{3.3}$$

dove $y = (y_1, \ldots, y_n)^T$, $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)^T$, $\beta = (\beta_0, \ldots, \beta_r)^T$ e

$$X = \begin{pmatrix} 1 & x_{11} & \ldots & x_{1r} \\ 1 & x_{21} & \ldots & x_{2r} \\ \ldots & \ldots & \ldots & \ldots \\ 1 & x_{n1} & \ldots & x_{nr} \end{pmatrix}$$

Con il metodo dei minimi quadrati si calcolano i valori di β che minimizzano la somma dei quadrati degli scarti $\varepsilon^T\varepsilon$:

$$\varepsilon^T\varepsilon = (y - X\beta)^T(y - X\beta) \tag{3.4}$$

Espandendo l'espressione, differenziando rispetto a β e uguagliando a zero il risultato si ottengono le cosiddette equazioni normali:

$$X^TX\beta = X^Ty \tag{3.5}$$

e se la matrice X^TX è invertibile si ha:

$$\begin{aligned} \hat{\beta} &= (X^TX)^{-1}X^Ty \\ X\hat{\beta} &= X(X^TX)^{-1}X^Ty \equiv Hy \end{aligned} \tag{3.6}$$

la matrice H è detta hat-matrix. Si hanno le seguenti identità:

$$\begin{aligned} \hat{y} &= X\hat{\beta} = Hy \\ \hat{\varepsilon} &= y - \hat{y} = (I - H)y \\ \hat{\varepsilon}^T\hat{\varepsilon} &= y^T(I - H)y \end{aligned}$$

Nell'ipotesi che sia var $\varepsilon = \sigma^2 I$ si ha anche:

$$\text{var}\,\hat{\beta} = (X^TX)^{-1}X^T\sigma^2 IX(X^TX)^{-1} = (X^TX)^{-1}\sigma^2. \tag{3.7}$$

Esempio

In un esperimento si vuole valutare da quale fonte una certa varietà di pianta tragga il fosforo. Si misura quindi la concentrazione di fosforo y (in ppm = parti per milione) in 17 di tali piante, in associazione con la quantità di fosforo inorganico (x_1) e organico (x_2) nel suolo in cui crescono.

```
> y <- c(64,60,71,61,54,77,81,93,93,51,76,96,77,93,95,54,99)
> x1 <- c(0.4,0.4,3.1,0.6,4.7,1.7,9.4,10.1,11.6,12.6,10.9,23.1,
+         23.1,21.6,23.1,1.9,29.9)
> x2 <- c(53,23,19,34,24,65,44,31,29,58,37,46,50,44,56,36,51)
```

Si esegue il fit del modello lineare di interesse:

```
> mod <- lm(y ~ x1 + x2)
```

Prima di esaminare il risultato del modello è interessante vedere come è possibile richiamare alcuni degli oggetti definiti in precedenza. Le matrici X e $(X^TX)^{-1}$ sono disponibili come:

```
> x <- model.matrix(mod)
> xtx.inv <- summary(mod)$cov.unscaled
```

I valori $\hat{y}_i$, $\hat{\sigma}$ e $\hat{\varepsilon}_i$ stimati dal modello si ottengono con le chiamate:

```
> mod$fitted.values
> summary(mod)$sigma
> mod$residuals
```

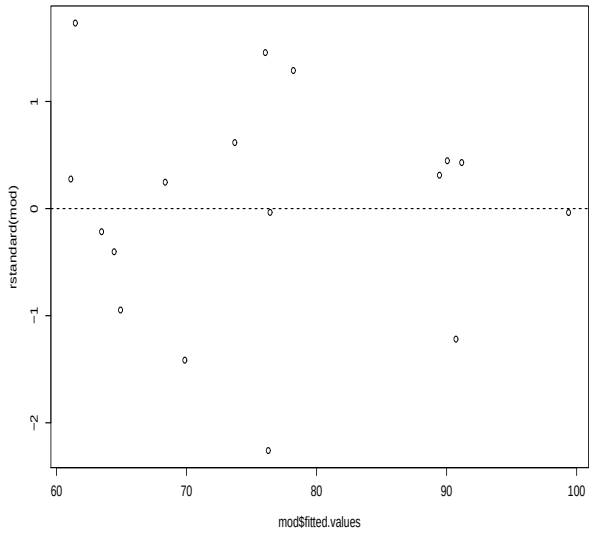

Figura 3.3: Grafico dei residui nel caso dell'esempio 3.2.

Il risultato del fit del modello si esamina con la chiamata:

```
> summary(mod)

[...]
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  66.4654     9.8496   6.748 9.35e-06 ***
x1            1.2902     0.3428   3.764  0.00209 **
x2           -0.1110     0.2486  -0.447  0.66195

Residual standard error: 12.25 on 14 degrees of freedom
Multiple R-Squared: 0.5253,     Adjusted R-squared: 0.4575
F-statistic: 7.746 on 2 and 14 DF,  p-value: 0.005433
```

Si evidenzia quindi che la variabile y dipende in maniera altamente significativa dal predittore $x1$, mentre la dipendenza da $x2$ è non significativa. Prima di concludere è necessario verificare l'adeguatezza del modello esaminando i residui. Il grafico dei residui standardizzati, presentato in Fig. 3.3, si ottiene facilmente:

```
> plot(mod$fitted.values, rstandard(mod))
> abline(h=0, lty=2)
```

L'analisi del grafico non evidenzia nessun particolare problema. Il test di normalità sui residui:

```
> shapiro.test(rstandard(mod))

        Shapiro-Wilk normality test

data:  rstandard(mod)
W = 0.9654, p-value = 0.7331
```

non mette in luce problemi di sorta.

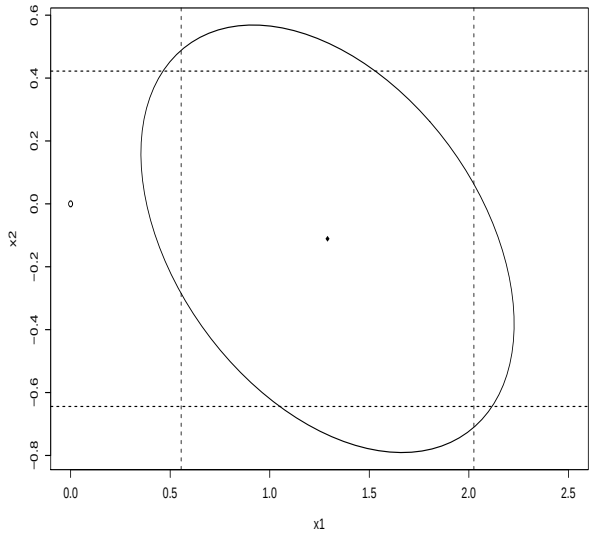

Figura 3.4: Intervallo di confidenza congiunto (ellisse), confrontato con gli intervalli di confidenza individuali (linee tratteggiate) per i predittori $x1$ e $x2$, nel caso dell'esempio 3.2. Sono evidenziati l'origine (cerchietto vuoto) e il punto corrispondente alla miglior stima dei coefficienti di regressione.

Gli intervalli di confidenza che si possono costruire sui parametri di regressione a partire dalle stime dei coefficienti e dei loro errori sono calcolati individualmente. Per avere un intervallo di confidenza congiunto si può usare la libreria *ellipse*, che non fa parte della distribuzione standard e deve essere scaricata a parte. Con la libreria correttamente installata si procede nel modo seguente:

```
> library(ellipse)      # si carica la libreria
> plot(ellipse(mod, c(2, 3)), type="l", xlim=c(0, 2.5))
```

La funzione *ellipse* produce un intervallo di confidenza congiunto per i parametri 2 e 3 del modello *mod*. La regione all'interno dell'ellisse è l'intervallo di confidenza congiunto al 95% per i due parametri. Per valutare quanto esso differisca dalla regione rettangolare individuata dai due intervalli di confidenza individuali si usano le chiamate (Fig. 3.4):

```
> t <- qt(0.975, 14)               # valore critico di t a 14 gdl
> cf <- mod$coeff                  # coefficienti del modello
> er.cf <- summary(mod)$coeff[,2]  # errori sui coefficienti
> lim.l <- cf - t * er.cf
> lim.u <- cf + t * er.cf
> abline(v=c(lim.l[2], lim.u[2]), lty=2)
> abline(h=c(lim.l[3], lim.u[3]), lty=2)
```

È possibile aggiungere il punto (0, 0) e quello che individua la miglior stima dei coefficienti:

```
> points(0,0)
> points(cf[2], cf[3], pch=18)
```

Si osservi che le linee tratteggiate non sono tangenti all'ellisse, dato che le stime degli intervalli di confidenza sono fatte in modi differenti (individuale e congiunta).

Dall'esame del grafico si nots che il cerchietto individuante l'origine si trova all'interno dell'intervallo di confidenza del predittore $x2$, poiché esso non è statisticamente significativo. Dato che le zone individuate dai due intervalli di confidenza non coincidono, è possibile che si verifichi il caso in cui

l'origine sia interna al rettangolo ed esterna all'ellisse. In tale situazione i due test individuali darebbero non significatività dei predittori, mentre il test congiunto giungerebbe alla conclusione opposta. Il secondo risultato è da preferire. Viceversa è possile che l'origine sia interna all'ellisse, ma esterna al rettangolo. Mentre il test congiunto è in questo caso non significativo, i due test individuali lo sono. Anche in questo caso la conclusione del test globale è da preferirsi.

3.2.1 Test per l'eliminazione di un predittore

Per testare la possibilità di semplificare un modello eliminando uno o più predittori che risultino non statisticamente significativi si fitta il modello ridotto e si paragonano le devianze d'errore mediante un test F.

Nel caso dell'Esempio 3.2 il predittore $x2$ risultava non significativo. Per vedere se è possibile eliminarlo dal modello si fitta il modello con il solo $x1$:

```
> mod1 <- lm(y ~ x1)
> summary(mod1)
[...]
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  62.5694     4.4519  14.055 4.85e-10 ***
x1            1.2291     0.3058   4.019  0.00111 *
```

e si testa l'ipotesi nulla che il modello ristretto *mod*1 sia equivalente al modello completo *mod*:

```
> anova(mod1, mod)
Analysis of Variance Table

Model 1: y ~ x1
Model 2: y ~ x1 + x2
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     15 2131.24
2     14 2101.29  1     29.95 0.1995  0.662
```

Dal test risulta che l'ipotesi nulla non può essere rifiutata, quindi il modello ristretto è altrettanto valido di quello completo e la variabile $x2$ può essere eliminata.

3.3 Trasformazione dei dati

Talvolta una trasformazione della variabile dipendente o dei predittori migliora notevolmente il fit di un modello lineare e risolve problemi derivanti da violazioni delle assunzioni del modello.

3.3.1 Trasformazioni della variabile dipendente

Il metodo di Box-Cox è una tecnica per determinare quale sia la trasformazione migliore da applicare alla variabile dipendente y. Il suo funzionamento è ristretto al caso in cui i valori di y siano positivi e calcola la migliore trasformazione $t_\lambda(y)$, indicizzata dal parametro λ, all'interno della famiglia:

$$t_\lambda(y) = \begin{cases} \frac{y^\lambda - 1}{\lambda} & \lambda \neq 0 \\ \log y & \lambda = 0 \end{cases}$$

Quindi $\lambda = 0.5$ corrisponde alla trasformazione $t(y) = \sqrt{y}$, $\lambda = 3$ alla trasformazione $t(y) = y^3$ e $\lambda = 0$ a $t(y) = \log y$. Tramite la tecnica di Box-Cox è possibile calcolare il miglior valore di λ e il suo intervallo di confidenza, come nell'esempio seguente.

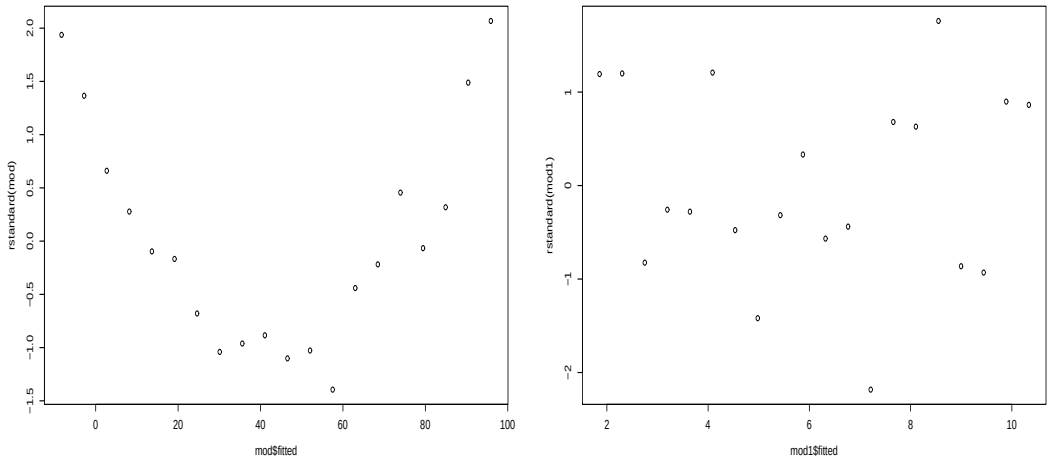

Figura 3.5: Analisi dei residui per l'Esempio 3.3.1. A sinistra il grafico dei residui standardizzati contro i valori fittati dal modello lineare evidenzia forti problemi di non linearità. A destra il corrispondente grafico per il modello trasformato mostra che il problema è stato risolto dalla trasformazione $t(y) = \sqrt{y}$.

Esempio

Un nuovo centro commerciale inizia una campagna pubblicitaria per incrementare il numero dei suoi clienti. Il numero medio di visitatori per unità di tempo viene monitorato per 20 settimane dopo l'inizio della promozione. I dati sono quelli inseriti nei due vettori *tempo* e *visitatori*.

```
> tempo <- 1:20
> visitatori <- c(4,6,7,10,13,18,20,23,29,35,39,45,48,60,67,77,79,87,100,109)
```

Per prima cosa si fitta il modello lineare e si controllano i residui (a sinistra in Fig. 3.5):

```
> mod <- lm(visitatori ~ tempo)
> plot(mod$fitted, rstandard(mod))
```

Dall'analisi del grafico risulta evidente che vi è un forte problema di non linearità. Tramite il metodo di Box-Cox si può tentare di stimare la miglior correzione da apportare ai dati. La chiamata:

```
> library(MASS)     # caricamento della libreria MASS
> boxcox(visitatori ~ tempo, lambda=seq(0.3, 0.6, by=0.01))
```

produce il grafico a sinistra in Fig. 3.6. Sull'asse delle ascisse, il cui range e densità di campionamento possono essere impostati mediante l'opzione *lambda*, si leggono i valori del parametro λ. La miglior trasformazione possibile corrisponde al valore per cui la curva ha il suo massimo. La libreria *MASS*, che mette a disposizione la funzione *boxcox*, fa parte della distribuzione standard di R.

Dal grafico si può leggere la stima della miglior trasformazione da applicare alla variabile y. Questa stima viene fatta tramite il metodo di maximum likelihood, assumendo normalità degli errori. Per ulteriori dettagli sul procedimento si rimanda a [26, 59]. Il risultato migliore è $\lambda \sim 0.45$, ma per ragioni di semplicità di interpretazione è più opportuno scegliere il valore $\lambda = 0.5$ (trasformazione radice quadrata), valore che cade sul margine destro dell'intervallo di confidenza.

Si fitta quindi il modello trasformato e si analizzano i residui (a destra in Fig. 3.5):

```
> mod1 <- lm(sqrt(visitatori) ~ tempo)
> plot(mod1$fitted, rstandard(mod1))
```

Si nota che i problemi del modello precedente vengono risolti dalla trasformazione scelta. A destra in Fig. 3.6 sono riportati i punti sperimentali e i due fit. □

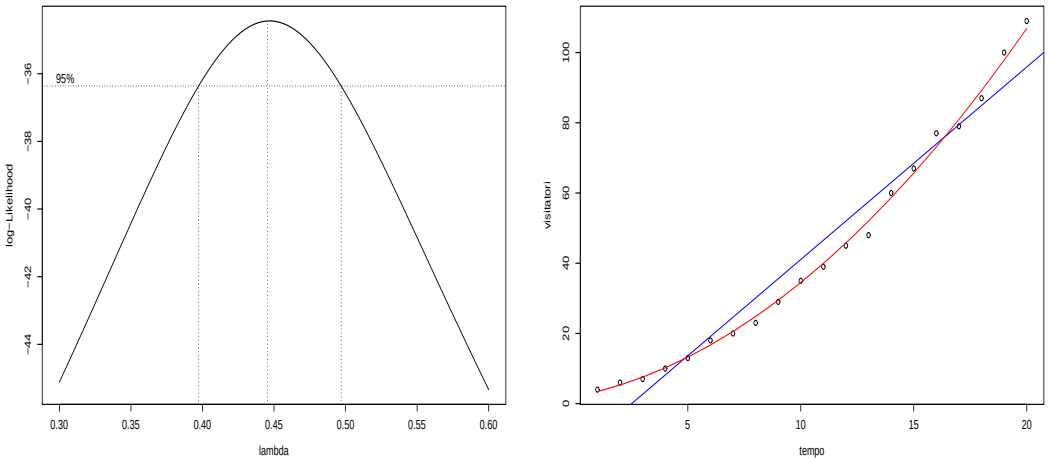

Figura 3.6: A sinistra: grafico della log-likelihood per la trasformazione di Box-Cox nel caso dell'Esempio 3.3.1. Nel grafico di destra sono mostrati i dati sperimentali (cerchi), il fit lineare prima della trasformazione (linea blu) e quello dopo la trasformazione $t(y) = \sqrt{y}$ (linea rossa).

Il metodo di Box-Cox esplora solo una sottofamiglia di trasformazioni possibili sulla variabile dipendente. Altre trasformazioni utili sono la trasformazione logit ($\log \frac{y}{1-y}$), usata quando la variabile dipendente è una proporzione o una percentuale, e la trasformazione di Fisher ($0.5 \log \frac{1+y}{1-y}$), usata quando la variabile dipendente misura una correlazione.

3.4 Minimi quadrati generalizzati

Una delle assunzioni del modello lineare

$$y = X\beta + \varepsilon$$

è che sia var $\varepsilon = \sigma^2 I$. Questa ipotesi può cadere o perché gli errori sono correlati fra loro oppure perché la varianza non è costante. Scrivendo var $\varepsilon = \sigma^2 \Sigma$, la matrice Σ esprime la relazione fra gli errori. Nel caso di varianza non costante ed errori non correlati Σ sarà diagonale, mentre per errori correlati Σ avrà anche elementi fuori diagonale diversi da zero.

Questi modelli lineari vengono trattati con la tecnica dei minimi quadrati generalizzati (Generalized Least Squares, GLS) che minimizzano la quantità:

$$\varepsilon^T \varepsilon = (y - X\beta)^T \Sigma^{-1} (y - X\beta)$$

La soluzione delle equazioni normali è ora:

$$\begin{aligned} \hat{\beta} &= (X^T \Sigma^{-1} X)^{-1} X^T \Sigma^{-1} y \\ \text{var } \hat{\beta} &= (X^T \Sigma^{-1} X)^{-1} \sigma^2 \end{aligned} \tag{3.8}$$

Dato che si può sempre scrivere $\Sigma = SS^T$ con S matrice triangolare (decomposizione di Choleski) si ha:

$$(y - X\beta)^T S^{-T} S^{-1} (y - X\beta) = (S^{-1} y - S^{-1} X\beta)^T (S^{-1} y - S^{-1} X\beta)$$

cioè la regressione GLS è equivalente a regredire $S^{-1}X$ contro $S^{-1}y$. Infatti:

$$S^{-1} y = S^{-1} X\beta + S^{-1} \varepsilon \quad \rightarrow \quad y' = X'\beta + \varepsilon'$$

conduce a una nuova regressione lineare. Esaminando la varianza dei nuovi errori ε' si ottiene:

$$\text{var } \varepsilon' = \text{var } (S^{-1}\varepsilon) = S^{-1} (\text{var } \varepsilon) S^{-T} = S^{-1} \sigma^2 S S^T S^{-T} = \sigma^2 I$$

e quindi, per il modello trasformato, le ipotesi sugli errori sono soddisfatte.

3.4.1 Minimi quadrati pesati

Se gli errori sono scorrelati, ma la varianza non è uniforme (eteroschedasticità), la matrice Σ è diagonale. Se la sua forma è nota o calcolabile dai dati si utilizza la tecnica dei minimi quadrati pesati (Weighted Least Squares, WLS). Si ha quindi $\Sigma = \text{diag}(1/w_1, \ldots, 1/w_n)$ dove gli elementi w_i sono detti pesi.

Esempio

Si vuole valutare se la resa di una coltivazione (in una opportuna unità di misura) dipenda linearmente dalla piovosità mensile (in mm). Ai fini dell'esperimento il campo viene diviso in 5 sottoparcelle; mensilmente si valuta la resa di ognuna di esse. L'esperimento viene portato avanti per 10 mesi.

Si inizia l'analisi inserendo i dati relativi alla resa mensile delle sottoparcelle e delle piovosità mensili:

```
> resa1 <- c(85, 124, 104, 116, 136, 116, 97, 127, 120, 111)
> resa2 <- c(88, 127, 104, 112, 138, 116, 121, 130, 114, 111)
> resa3 <- c(87, 129, 106, 111, 128, 103, 96, 124, 122, 121)
> resa4 <- c(89, 123, 111, 110, 131, 112, 105, 146, 128, 111)
> resa5 <- c(89, 126, 103, 112, 138, 113, 112, 117, 117, 103)
> resa <- c(resa1, resa2, resa3, resa4, resa5)
> pioggia <- c(77, 110, 97, 90, 114, 107, 83, 108, 101, 98)
> p <- rep(pioggia, 5)
> gruppo <- gl(10, 1, 50)   # fattore che distingue i vari mesi
```

La funzione *gl* viene usata per creare una variabile categoriale, o fattore. Essa accetta tre argomenti: il numero n di livelli (o categorie) del fattore, il numero k di repliche di ogni categoria e la lunghezza totale del vettore da creare. Se non altrimenti specificato, il terzo argomento è assunto di default uguale a $n \times k$. Nell'esempio in questione il vettore *gruppo* è di lunghezza 50, con i primi 10 elementi pari alla sequenza $1, 2, \ldots, 10$ (10 livelli con 1 ripetizione l'uno); questa sequenza viene quindi ripetuta per 5 volte fino a riempire il vettore.

Come primo passo si testa il modello lineare:

```
> mod <- lm(resa ~ p)
> plot(mod$fitted, rstandard(mod))
```

Il grafico dei residui (a sinistra in Fig. 3.7) mostra evidenti problemi di eteroschedasticità. Per tentare di risolverli si può fittare il modello WLS, dove i pesi sono gli inversi delle varianze mensili delle rese dei campi.

```
> var.m <- tapply(resa, gruppo, var)    # varianze mensili
> var.m
    1     2     3     4     5     6     7     8     9    10
  2.8   5.7  10.3   5.2  20.2  28.5 110.7 115.7  28.2  40.8
> mod1 <- lm(resa ~ p, weights=rep(var.m, 5)^-1)
> plot(mod1$fitted, rstandard(mod1))
```

L'opzione *weights* permette di specificare il peso da assegnare all'osservazione corrispondente. Si noti che essendo il vettore *var.m* di dimensione 10 è necessario usare la funzione $rep(var.m, 5)$ per renderlo della lunghezza necessaria. Il grafico dei residui (a destra in Fig. 3.7) mostra che il problema di eteroschedasticità è stato risolto. Il suo particolare aspetto, in cui si notano degli andamenti caratteristici, lascia comunque dei dubbi sulla pianificazione dell'esperimento. Probabilmente l'introduzione di qualche altra variabile (ad esempio la temperatura) avrebbe migliorato il modello lineare.

La significatività del modello si controlla con la chiamata usuale:

```
> summary(mod1)
[...]
```

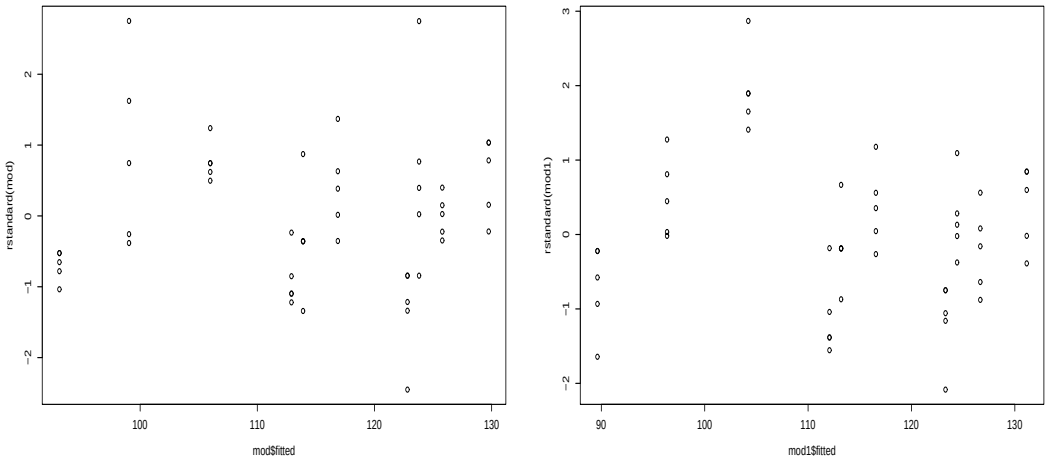

Figura 3.7: Analisi dei residui per l'Esempio 3.4.1. A sinistra il grafico dei residui standardizzati contro i valori fittati dal modello lineare evidenzia forti problemi di eteroschedasticità. A destra il corrispondente grafico per il modello WLS trasformato mostra che questo problema è stato risolto.

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.20830    5.71739   0.561    0.577
p            1.12229    0.06146  18.260   <2e-16 ***
```

Si conclude che la dipendenza della produttività dalla piovosità è molto altamente significativa. □

Nell'esempio appena trattato i pesi da utilizzare possono essere calcolati direttamente dai dati. Se non si hanno repliche delle osservazioni il procedimento è inattuabile e il problema di eteroschedasticità può essere risolto esattamente solo avendo ulteriori informazioni su come sono stati raccolti i dati. Un esempio classico è una regressione di una variabile dipendente y su un predittore x in cui i valori di y_i sono le medie calcolate su un gruppo di numerosità n_i. Se i valori di n_i non sono tutti uguali e si suppone che i gruppi abbiano tutti la stessa varianza σ^2 si ha: var y_i = var $\varepsilon_i = \sigma^2/n_i$. Questo problema di eteroschedasticità si risolve facilmente utilizzando i pesi $w_i = n_i$, ma se l'informazione sui valori di n_i non viene fornita assieme ai valori di x e y la correzione non può essere effettuata.

Talvolta l'andamento di var ε_i non è noto esattamente ma si può supporre che sia legato a uno dei predittori da una legge a potenza:

$$\text{var } \varepsilon \propto x_j^t$$

Nell'esempio seguente è mostrato come trattare un caso del genere in R.

Esempio

Un impianto chimico può funzionare a diverse pressioni fra 5 e 20 atm. Per verificare se la produttività dell'impianto varia in modo lineare aumentando la pressione di esercizio, si raccolgono i dati inseriti nei vettori *atm* e *produzione*:

```
> atm <- seq(5, 20 by=0.25)
> produzione <- c(178.54, 183.44, 181.09, 194.97, 186.33, 191.28,  189.99, 194.46,
+ 201.42, 203.67, 221.62, 206.44, 192.92, 207.17, 202.55, 201.41, 219.50, 230.64,
+ 230.59, 212.51, 219.17, 218.21, 221.04, 225.08, 230.32, 229.03, 227.89, 239.62,
+ 240.09, 237.06, 218.57, 260.34, 262.56, 233.37, 252.79, 247.29, 248.73, 235.98,
+ 268.43, 256.59, 256.60, 271.11, 271.71, 253.46, 239.39, 263.35, 237.68, 292.69,
+ 264.90, 276.33, 269.23, 284.59, 257.04, 238.85, 212.92, 224.00, 295.38, 255.40,
```

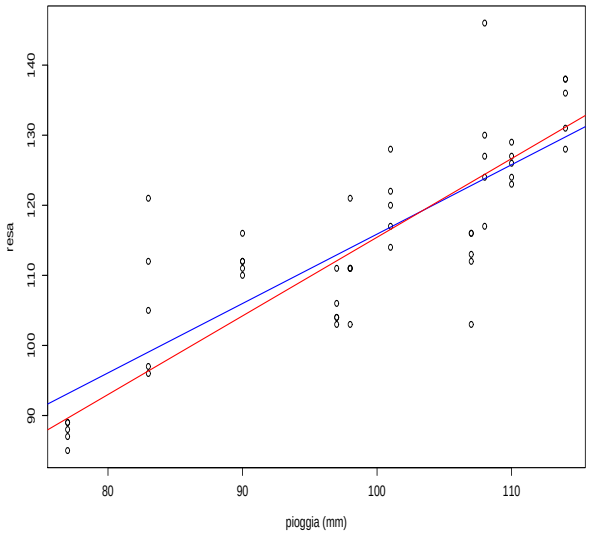

Figura 3.8: Regressione lineare (linea blu) e WLS (linea rossa) nel caso dell'Esempio 3.4.1.

```
+ 271.55, 285.69, 255.92)
```

Dall'osservazione del grafico dei dati in Fig. 3.9 si nota che la produzione tende a variare maggiormente per valori di pressione elevati. Si può quindi supporre che valga una relazione del tipo:

$$\text{var}\,\varepsilon \propto atm^t. \tag{3.9}$$

Si inizia l'analisi con il test del modello lineare standard:

```
> mod <- lm(produzione ~ atm)
> summary(mod)
[...]
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 161.7674     6.2589   25.85   <2e-16 ***
atm           5.7871     0.4723   12.25   <2e-16 ***

Residual standard error: 16.24 on 59 degrees of freedom
Multiple R-Squared: 0.7179,     Adjusted R-squared: 0.7131
F-statistic: 150.1 on 1 and 59 DF,  p-value: < 2.2e-16
```

Il grafico dei residui (a sinistra in Fig. 3.10) mette in luce il problema di eteroschedasticità. Per tentare di risolverlo si può usare la funzione *gls* della libreria *nlme* (parte della distribuzione standard di R):

```
> library(nlme)
> mod1 <- gls(produzione ~ atm, weights=varPower(form= ~ atm))
```

La funzione accetta diversi argomenti: oltre al modello da fittare è possibile specificare, come in questo caso, dei pesi (opzione *weights*) secondo alcuni schemi predefiniti. La funzione $varPower(form = \sim atm)$ richiede che i pesi siano una potenza della variabile *atm*; l'algoritmo si incarica di determinare il miglior valore dell'esponente della legge a potenza. Data la scelta interna dell'algoritmo il numero fornito come stima va moltiplicato per 2 per essere conforme all'Eq.(3.9). Il fit del modello si esamina con la consueta chiamata:

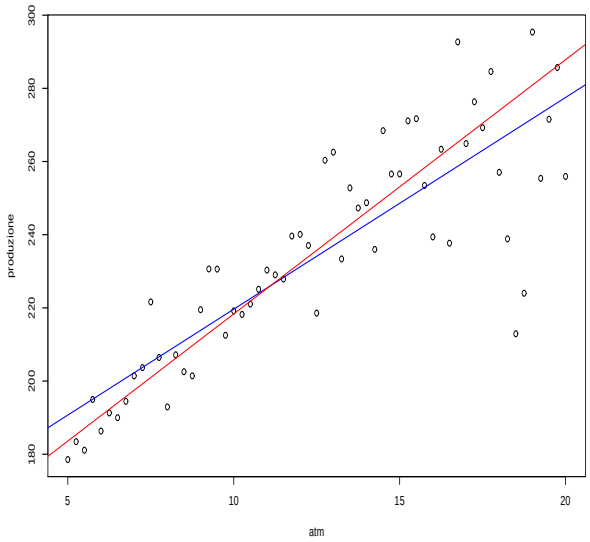

Figura 3.9: Grafico dei dati dell'Esempio 3.4.1. Sono mostrati il fit lineare non pesato (linea blu) e quello pesato (linea rossa).

```
> summary(mod1)
Generalized least squares fit by REML
  Model: produzione ~ atm
  Data: NULL
       AIC      BIC    logLik
  490.5844 498.8945 -241.2922

Variance function:
 Structure: Power of variance covariate
 Formula: ~atm
 Parameter estimates:
   power
1.349463

Coefficients:
               Value Std.Error  t-value p-value
(Intercept) 148.86059  3.273481 45.47471       0
atm           6.94886  0.369461 18.80808       0

[...]
Residual standard error: 0.4758102
Degrees of freedom: 61 total; 59 residual
```

Il ricco output fornisce molte informazioni: dapprima una parte informativa sulla tecnica di fit utilizzata (di default la funzione utilizza la tecnica di "REstricted Maximum Likelihood" o REML), quindi viene data la stima dell'esponente della legge a potenza (in questo caso $2 * 1.349463 \simeq 2.7$). Infine vi è la parte sui coefficienti e la loro significatività. Si nota che il coefficiente del predittore *atm* cambia sensibilmente passando da un modello all'altro. Come test finale, il grafico dei residui per il modello pesato (a destra in Fig. 3.10) evidenzia che il problema di eteroschedasticità del modello non pesato è stato risolto dalla tecnica GLS.

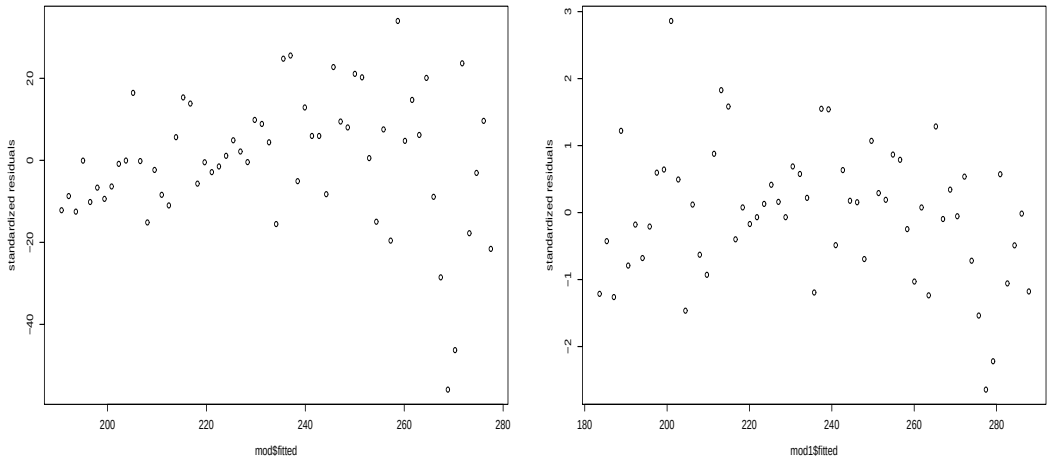

Figura 3.10: Analisi dei residui per l'Esempio 3.4.1. A sinistra il grafico dei residui standardizzati contro i valori fittati dal modello lineare evidenzia problemi di eteroschedasticità. A destra il corrispondente grafico per il modello GLS mostra che il problema può essere risolto da questa tecnica.

3.5 Autocorrelazione e serie temporali

I metodi di regressione discussi in precedenza assumono che vi sia indipendenza fra i residui. Se questa condizione non è soddisfatta il rischio è quello di avere una stima fortemente distorta (per eccesso o per difetto) degli errori sui parametri di regressione. Per correggere questo andamento è possibile ricorrere a tecniche che permettono di avere una stima corretta delle variabili di regressione, specificando un preciso schema di correlazione fra le varie rilevazioni.

Il principale effetto di dipendenza è dovuto a correlazioni spaziali o temporali fra le varie osservazioni. Ad esempio fenomeni climatici o economici di solito presentano dipendenza temporale, nel senso che il presente dipende dal passato, mentre in uno studio agrario è facile che vi sia dipendenza spaziale nel senso che parcelle vicine tendono ad assomigliarsi ai fini dell'esperimento. Nel seguito si tratteranno esclusivamente casi di dipendenza temporale. Per ulteriori dettagli sull'argomento e sulle problematiche connesse si rimanda alla vasta letteratura disponibile, ad esempio [12, 49].

Il modello che si assume è:

$$y = \beta X + \varepsilon$$

dove si suppone che sia:

$$\varepsilon \sim N(\mathbf{0}, \Sigma)$$

con Σ, matrice di covarianza d'errore, simmetrica e semidefinita positiva. Nel caso di regressione ordinaria essa coincide con la matrice identica; nel caso di regressione pesata trattata in Sec. 3.4 ha elementi fuori diagonale nulli e elementi diversi sulla diagonale principale. Nel caso più generale che interessa in questa sezione si suppone che gli elementi fuori diagonale siano a loro volta non nulli. Nel caso in cui la forma di Σ sia nota il problema di determinare le stime $\hat{\beta}$ dei parametri di regressione è facilmente risolto con le tecniche presentate in precedenza (GLS). Ovviamente questo non è quasi mai il caso e si pone il problema di determinare, a partire dai dati, anche gli elementi della matrice di covarianza d'errore. Per ridurre la complessità del problema (gli elementi di Σ da stimare sono ben $n(n-1)/2$) è indispensabile introdurre delle semplificazioni che vincolano la struttura di Σ.

3.5.1 Varie forme di autocorrelazione

La stima GLS può essere effettuata assumendo un modello specifico di dipendenza fra i residui. Il modello più semplice, chiamato modello autoregressivo di ritardo (o lag) 1, o AR(1), presuppone che valgano le seguenti ipotesi:

1. gli n valori della variabile dipendente y_t sono misurati a intervalli costanti;
2. I residui del modello sono legati fra loro dalla semplice regola autoregressiva:

$$\varepsilon_i = \rho\, \varepsilon_{i-1} + \nu_i$$

dove $\nu \sim N(0, \sigma_\nu^2)$ è una componente di "rumore bianco" e ρ è il coefficiente di autoregressione che lega una osservazione alla precedente.

Ovviamente di solito il valore di ρ non è noto. Una sua stima campionaria $\hat{\rho}$ può essere ottenuta nel modo seguente. Sia $\hat{\varepsilon}_i$ il residuo i-esimo ottenuto dal fit del modello lineare atto a descrivere il fenomeno; si ha:

$$\hat{\rho} = \frac{c_1}{c_0}$$

dove c_1 e c_0 sono le due autocovarianze definite come:

$$\begin{aligned} c_0 &= \frac{1}{n-1}\sum_{i=1}^{n} \hat{\varepsilon}_i^2 \\ c_1 &= \frac{1}{n-1}\sum_{i=2}^{n} \hat{\varepsilon}_i \hat{\varepsilon}_{i-1} \end{aligned}$$

Modelli autoregressivi di ordine più alto si ottengono per generalizzazione del modello AR(1). Ad esempio il modello autoregressivo di ritardo 2, o AR(2), è:

$$\varepsilon_i = \rho_1\, \varepsilon_{i-1} + \rho_2\, \varepsilon_{i-2} + \nu_i.$$

In alternativa è possibile avere schemi a media mobile (moving-average o MA). Lo schema MA(1) è definito come:

$$\varepsilon_i = \psi\, \nu_{i-1} + \nu_i$$

cioè i residui di regressione dipendono dagli errori casuali. Schemi MA di ordine più elevato si ottengono con ovvia generalizzazione.

Infine è possibile combinare gli schemi AR e MA in uno schema ARMA. Ad esempio il processo autoregressivo ARMA(1, 1) è definito da:

$$\varepsilon_i = \rho\, \varepsilon_{i-1} + \psi\, \nu_{i-1} + \nu_i.$$

La scelta del modello di autocorrelazione da adottare, oltre che discendere da una analisi dei dati, dovrebbe essere sempre guidata dal giudizio e dalla competenza dello sperimentatore.

3.5.2 Determinare l'esistenza di autocorrelazione

Per verificare se una autocorrelazione sia o meno significativa è possibile ricorrere a vari metodi. Il più semplice di tutti, valido per campioni sufficientemente grandi ($n \simeq 100$) si basa sul fatto che la autocorrelazione in una serie di n variabili indipendenti di eguale varianza è approssimativamente normale con media 0 e deviazione standard $1/\sqrt{n}$. Quindi si considera significativa a livello $\alpha = 0.05$ un'autocorrelazione maggiore in modulo di $1.96/\sqrt{n}$.

Un test più accurato è dovuto a Durbin-Watson, che definisce, per ogni valore di lag h, la statistica:

$$D_h = \frac{\sum_{i=h+1}^{n} (\hat{\varepsilon}_i - \hat{\varepsilon}_{i-h})^2}{\sum_{i=1}^{n} \hat{\varepsilon}_i^2}$$

la cui distribuzione è molto complessa e dipende dalla struttura della matrice X dei predittori. Per grandi campioni si ha $D_h \simeq 2(1 - \rho_h)$ quindi $D = 2$ è indice di assenza di autocorrelazione, $D < 2$ autocorrelazione positiva e $D > 2$ autocorrelazione negativa. Per avere una stima valida della significatività del test è opportuno affidarsi a metodi bootstrap, che permettono di valutare il valore P nel modo piú corretto a seconda della forma di X.

YEAR	TEMP	YEAR	TEMP	YEAR	TEMP	YEAR	TEMP
1880	-0.31	1907	-0.35	1934	-0.02	1961	0.11
1881	-0.27	1908	-0.34	1935	-0.06	1962	0.10
1882	-0.31	1909	-0.25	1936	-0.02	1963	0.11
1883	-0.39	1910	-0.25	1937	0.10	1964	-0.15
1884	-0.48	1911	-0.30	1938	0.14	1965	-0.12
1885	-0.41	1912	-0.21	1939	0.04	1966	-0.02
1886	-0.32	1913	-0.21	1940	0.04	1967	-0.04
1887	-0.43	1914	-0.09	1941	0.07	1968	-0.08
1888	-0.37	1915	0.00	1942	0.01	1969	0.08
1889	-0.22	1916	-0.21	1943	0.00	1970	0.05
1890	-0.45	1917	-0.40	1944	0.15	1971	-0.10
1891	-0.38	1918	-0.29	1945	0.06	1972	0.02
1892	-0.40	1919	-0.16	1946	-0.08	1973	0.15
1893	-0.43	1920	-0.17	1947	-0.05	1974	-0.11
1894	-0.35	1921	-0.12	1948	-0.06	1975	-0.08
1895	-0.29	1922	-0.20	1949	-0.06	1976	-0.21
1896	-0.10	1923	-0.18	1950	-0.13	1977	0.10
1897	-0.08	1924	-0.19	1951	-0.02	1978	0.03
1898	-0.27	1925	-0.12	1952	0.070	1979	0.12
1899	-0.12	1926	0.07	1953	0.111	1980	0.18
1900	0.01	1927	-0.04	1954	-0.132	1981	0.24
1901	-0.08	1928	-0.05	1955	-0.143	1982	0.09
1902	-0.17	1929	-0.22	1956	-0.234	1983	0.31
1903	-0.30	1930	-0.03	1957	0.075	1984	0.10
1904	-0.39	1931	0.05	1958	0.126	1985	0.07
1905	-0.25	1932	-0.01	1959	0.057	1986	0.16
1906	-0.15	1933	-0.11	1960	0.008	1987	0.33

Tabella 3.1: Temperature medie rilevate nell'emisfero nord nel corso degli anni 1880-1987. Le temperature, in gradi centigradi, sono espresse come differenza rispetto alla media sul periodo complessivo di 108 anni (P.D. Jones, Journal of Climatology 1:654–660, 1988).

Esempio

Nel tentativo di verificare l'entità dell'effetto di riscaldamento globale dovuto all'effetto serra, sono stati raccolti i dati di temperatura media annuale dell'emisfero nord. Ci si chiede se i dati in questione supportino la conclusione che la temperatura media sta lentamente aumentando e, in questo caso, quale sia un intervallo di confidenza per tale aumento (i dati sono tratti da Jones, Journal of Climatology 1:654–660, 1988).

Come primo approccio si fitta il modello lineare:

```
> mod <- lm(TEMP ~ YEAR, data=temp)
> summary(mod)
[...]

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -8.7867143  0.6795784  -12.93   <2e-16 ***
YEAR         0.0044936  0.0003514   12.79   <2e-16 ***

Residual standard error: 0.1139 on 106 degrees of freedom
Multiple R-Squared: 0.6067,     Adjusted R-squared: 0.603
F-statistic: 163.5 on 1 and 106 DF,  p-value: < 2.2e-16
```

il quale evidenzia un effetto altamente significativo di aumento medio della temperatura. Un intervallo di confidenza dell'aumento secolare della temperatura media si può ottenere usando la funzione $confint$ nel modo seguente:

```
> confint(mod, 2)*100
         2.5 %    97.5 %
YEAR 0.3796858 0.5190347
```

dove il secondo argomento specifica che si vuole l'intervallo di confidenza solo sul secondo coefficiente di regressione. Si conclude che a livello 95% l'aumento secolare della temperatura media è compreso nell'intervallo [0.38 - 0.52].

È a questo punto necessario verificare le ipotesi del modello lineare, in particolare l'indipendenza dei residui. Un modo veloce per farlo è esaminare il grafico di autocorrelazione parziale (Fig. 3.11):

```
> pacf(mod$residuals, lag=10)
```

da cui si deduce la presenza di un processo AR(1), dato che solo il picco in corrispondenza del lag 1 esce dalla banda che indica la significatività che ha per estremi i valori $\pm 1.96/\sqrt{108}$.

Per verificare precisamente la significatività dei vari coefficienti di correlazione parziale è possibile far uso del test di Durbin-Watson, disponibile nella libreria *car*. Ad esempio per testare la significatività dei primi tre coefficienti si eseguono le chiamate:

```
> library(car)
> durbin.watson(mod, 3)
  lag Autocorrelation D-W Statistic p-value
   1       0.4524824      1.068734   0.000
   2       0.1334321      1.703465   0.144
   3       0.0911412      1.784866   0.290
 Alternative hypothesis: rho[lag] != 0
```

Il calcolo dei valori P dei tre test sono ottenuti tramite simulazione bootstrap. Si ha quindi conferma del fatto che solo il primo coefficiente è significativo, facendo propendere per un modello AR(1).

Per tener conto della struttura di autocorrelazione nel fit del modello lineare si può far uso della funzione gls della libreria *nlme*, come nell'esempio seguente:

```
> library(nlme)
> mod1 <- gls(TEMP ~ YEAR, data=temp, correlation=corAR1(), method="ML")
> summary(mod1)
Generalized least squares fit by maximum likelihood
  Model: TEMP ~ YEAR
  Data: temp
        AIC       BIC   logLik
  -182.1309 -171.4024 95.06546

Correlation Structure: AR(1)
 Formula: ~1
 Parameter estimate(s):
      Phi
0.4606963

Coefficients:
               Value Std.Error   t-value p-value
(Intercept) -8.917303 1.0936560 -8.153663       0
YEAR         0.004562 0.0005656  8.066375       0

[...]
Residual standard error: 0.1129289
Degrees of freedom: 108 total; 106 residual
```

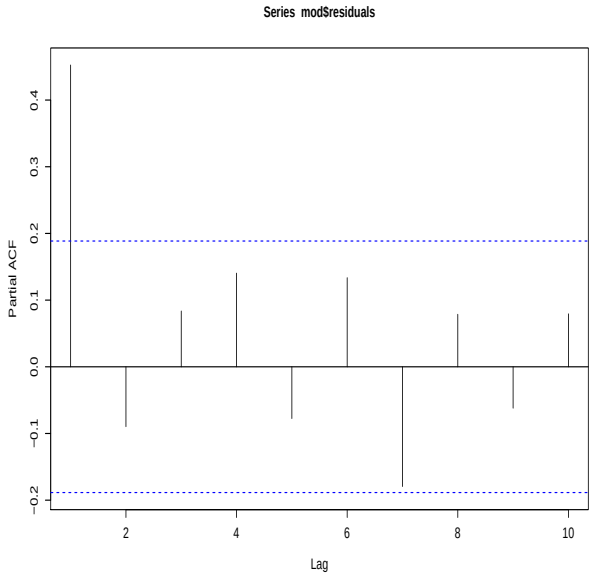

Figura 3.11: Autocorrelazione parziale per i dati relativi al riscaldamento globale. Le linee tratteggiate indicano gli estremi dell'intervallo di significatività posti a $\pm 1.96/\sqrt{108}$. Solo il dato di autocorrelazione corrispondente al primo lag esce dalla banda individuata e risulta quindi significativo.

L'opzione $correlation = corAR1()$ specifica che gli errori non devono essere considerati indipendenti, ma devono adattarsi a un modello autoregressivo con un passo di sfasamento. L'algoritmo valuta quindi, mediante una procedura di maximum likelihood, oltre ai parametri di fit, anche il parametro di correlazione AR(1) che meglio si adatta ai dati. Tale valore, chiamato *Phi*, viene riportato in output: nel caso in questione vale circa 0.46.

Dalla tabella dei parametri di regressione si osserva che il coefficiente di regressione non varia particolarmente (passando da 0.00449 trascurando la correzione per autocorrelazione a 0.00456 considerando gli effetti della correlazione fra i residui), mentre aumenta la stima dell'errore su tale coefficiente (da 0.00035 a 0.00057). Questo andamento è tipico in presenza di autocorrelazione positiva.

L'intervallo di confidenza per l'aumento secolare di temperatura, depurato dall'effetto di autocorrelazione, si ottiene quindi con la chiamata:

```
> intervals(mod1)
Approximate 95% confidence intervals

 Coefficients:
                   lower         est.        upper
(Intercept) -11.085582196 -8.917302721 -6.749023246
YEAR          0.003440738  0.004562015  0.005683292
attr(,"label")
[1] "Coefficients:"

 Correlation structure:
       lower      est.     upper
Phi 0.2766892 0.4606963 0.6121148
attr(,"label")
[1] "Correlation structure:"

 Residual standard error:
    lower       est.      upper
```

```
0.09572005 0.11292887 0.13323155
```

Quindi, corretto l'effetto di correlazione fra i residui, a livello 95% l'aumento secolare della temperatura media è compreso nell'intervallo [0.34 - 0.57].

3.6 Regressione non parametrica

Talvolta non si è in grado di stabilire la relazione funzionale

$$y = f(x)$$

che lega la variabile dipendente y al predittore x. In tali situazioni si può ricorrere a tecniche non parametriche in cui non è necessario specificare la forma funzionale di f. L'idea che sta alla base delle tecniche di regressione non parametriche è quella di sostituire alla retta di regressione o una media locale dei valori della variabile dipendente (kernel smoothing) o un lisciamento di una moltitudine di rette di regressione costruite in corrispondenza di ogni valore di x (LOWESS). In questo secondo caso, in corrispondenza di ogni x_i si considera un intervallo di dimensione fissata e si costruisce la retta di regressione per quei punti. Il risultato finale è il lisciamento di tutte queste rette.

3.6.1 Kernel smoothing

In R è possibile far uso di varie funzioni di regressione non parametrica, la più classica fra esse è *ksmooth*. L'idea di base della tecnica di kernel smoothing è quella di definire uno stimatore di $f(x)$ come una media pesata locale dei valori di y:

$$\hat{f}(x) = \sum_{i=1}^{n} w_i(x) y_i$$

dove $w_1, \ldots, w_n$ sono dei pesi che dipendono dai valori delle x_i. Usualmente si definisce l'andamento dei pesi tramite una funzione di densità (il kernel K), che dipende da un parametro di scala (bandwidth) mediante il quale si controlla la grandezza dei pesi. In altre parole la tecnica dei kernel smoothing può essere vista come un fit locale di una costante alla serie di dati.

Nel caso di regressione semplice si fa uso dello stimatore di Nadaraya-Watson, che definisce la sequenza di pesi come:

$$w_i(x) = \sum_{i=1}^{n} \frac{K(h^{-1}(x - x_i))}{\sum_{j=1}^{n} K(h^{-1}(x - x_j))}$$

dove il parametro h è la bandwidth. Selezionando una bandwidth piccola si da importanza solo ai valori di x vicini a x_i, con l'effetto di evidenziare le irregolarità della serie di punti, mentre con una bandwidth grande si ha un eccessivo lisciamento dello stimatore. Una scelta appropriata di h mette in evidenza l'andamento fondamentale della funzione, senza introdurre troppo rumore.

La funzione *ksmooth*, oltre ai valori di x e y da utilizzare accetta in input le opzioni *kernel*, che permette di scegliere il tipo di kernel di lisciamento e *bandwidth*. Il suo utilizzo è presentato nell'esempio seguente.

Esempio

Il dataset standard *cars* riporta le misurazioni degli spazi di frenata *dist* (in piedi) di 50 vetture in funzione della loro velocità *speed* (in miglia all'ora). È possibile fittare una regressione non parametrica con le chiamate:

```
> data(cars)
> attach(cars)
> plot(speed, dist)
> lines(ksmooth(speed, dist, "normal", bandwidth=2), col=2)
> lines(ksmooth(speed, dist, "normal", bandwidth=5), col=3)
```

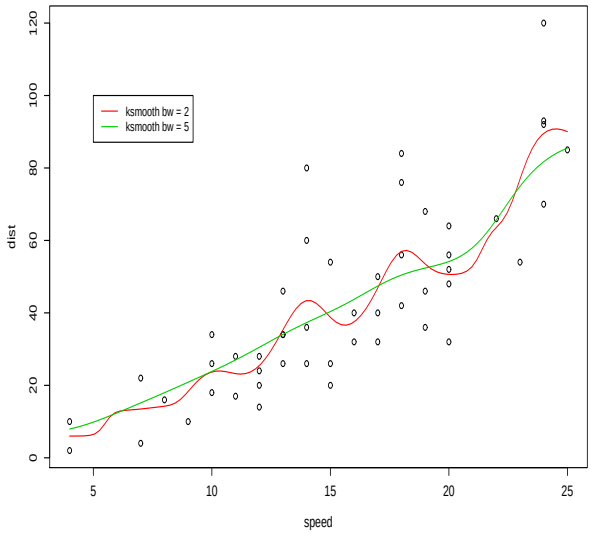

Figura 3.12: Regressioni non parametriche con stimatore kernel smoothing.

Le due curve di Fig. 3.12 sono state realizzate usando lo stesso kernel gaussiano (opzione "normal"), ma con larghezze di banda differenti. La curva verde, relativa alla larghezza di banda maggiore, risulta ovviamente più liscia.

In alternativa alla funzione *ksmooth* è possibile utilizzare le funzioni disponibili nella libreria aggiuntiva *KernSmooth*, le quali consentono fra l'altro di ottenere una stima della bandwidth ottimale.

Nello specifico è possibile avvalersi della funzione *dpill* per la stima della bandwith e della funzione *locpoly* per fittare lo stimatore. Nel caso dell'esempio in questione si ha:

```
> library(KernSmooth)
> h <- dpill(speed, dist)    # bandwidth ottimale
> h
[1] 2.027674
> lines(locpoly(speed, dist, degree=0, bandwidth=h))
```

Come nota conclusiva si può osservare che la tecnica presentata può essere vista come un sottocaso della più generale metodica di fit mediante polinomi locali di grado superiore a 0.

3.6.2 Algoritmo di lisciamento LOWESS

La limitazione della tecnica appena presentata è che non permette di ottenere nessuna inferenza dal modello fittato. Tecniche più robuste e moderne consentono di avere, oltre ai valori fittati dal modello non parametrico, anche l'informazione sull'errore standard da attribuire a tali valori. In particolare in R è disponibile la funzione *loess* [14], discendente della funzione *lowess* che implementava l'algoritmo di lisciamento LOWESS [13]. Questa funzione fa uso di un fit polinomiale locale (con grado dei polinomi fino al secondo), accetta più di un predittore e permette di regolare il livello di lisciamento con l'opzione *span*. Se il valore di *span* è inferiore a 1 tale parametro è interpretabile come la proporzione di punti che viene usata nei fit locali. In mancanza di specificazioni differenti da parte dell'utente esso assume valore di default pari a 0.75.

Si può applicare l'algoritmo *loess* con polinomi di primo e di secondo grado (usando l'opzione *degree*) ai dati relativi agli spazi di frenata delle 50 vetture considerate precedentemente:

```
> mod <- loess(dist ~ speed, data=cars, degree=1)
> mod2 <- loess(dist ~ speed, data=cars, degree=2)
```

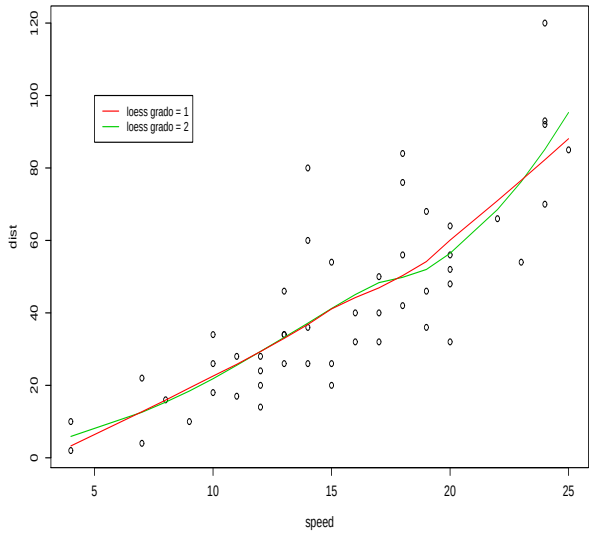

Figura 3.13: Regressioni non parametriche con algoritmo loess.

```
> plot(speed, dist)
> lines(mod$x, mod$fitted, col=2)
> lines(mod2$x, mod2$fitted, col=3)
> legend(5,100, c("loess grado = 1", "loess grado = 2"), col=c(2,3), lty=1)
```

I risultati dei due fit sono in Fig. 3.13. Una volta fittati i modelli, la funzione *predict* può quindi essere usata per ottenere le stime degli errori standard sui valori predetti:

```
> predict(mod2, se = TRUE)
$fit
 [1]  5.893767  5.893767 12.567960 12.567960 15.369183 18.425712 21.828039
[...]

$se.fit
 [1] 9.883804 9.883804 4.976453 4.976453 4.515801 4.316362 4.030120 4.030120
[...]
```

3.6.3 Modelli additivi generali

Si supponga di misurare su un campione di n oggetti una variabile di risposta y e r variabili esplicative $x_1, \ldots, x_r$. Detta X la matrice dei predittori, la classe di modelli additivi generali (sottoclasse dei modelli additivi generalizzati o GAM) è definita da:

$$y_i = \beta_0 + \sum_{j=1}^{r} f_j(X_{ij}) + \varepsilon_i \quad i = 1, \ldots, n \tag{3.10}$$

dove f_j sono funzioni di smoothing univariate e ε_i i termini d'errore con $E[\varepsilon] = 0$ e $Var(\varepsilon) = \sigma^2$. Per effettuare test d'ipotesi è necessario postulare anche la normalità dei termini d'errore: $\varepsilon \sim N(0, \sigma^2)$. Solitamente si assume:

$$E[f_j(x_j)] = 0$$

per evitare di avere costanti libere in ogni funzione.

Per stimare le funzioni f_j si espande ognuna di esse su una base di funzioni di smooth $b_{jk}(x)$ note (tipicamente funzioni spline):

$$y_i = \beta_0 + \sum_{k=1}^{K_1} \beta^*_{1k} b_{1k}(X_{i1}) + \ldots + \sum_{k=1}^{K_r} \beta^*_{rk} b_{rk}(X_{ir}) + \varepsilon_i$$

L'operazione risulta valida a patto che i numeri K_j di nodi utilizzati per le varie espansioni siano sufficientemente grandi. Le uniche incognite in questa formulazione sono solo parametri.

In definitiva si può vedere il modello additivo generale come una generalizzazione di un modello di regressione multipla senza interazione tra i predittori. Tra i vantaggi di questo approccio, oltre a una maggiore flessibilità rispetto al modello lineare, va citato il buon rate di convergenza algoritmico anche per problemi con molte variabili esplicative. Il maggior difetto risiede nella complessità del metodo di stima dei parametri. A tal fine infatti viene solitamente impiegato un agoritmo iterativo di *backfitting*, per i cui dettagli si rimanda ad esempio a [31].

In R è possibile fittare modelli additivi generali facendo uso della funzione *gam* implementata in due versioni – in realtà piuttosto differenti come principi di funzionamento – nelle librerie aggiuntive *gam* e *mgcv*. Nel seguito viene impiegata questa seconda versione – dovuta Simon N. Wood – i cui dettagli di implementazione e le tecniche generali di funzionamento sono ampliamente descritte in [60].

Le funzioni di smooth impiegate nella libreria *mgcv* sono funzioni spline, con la possibilità di specificare differenti basi con diverse caratteristiche di efficienza statistica e algoritmica (si veda la pagina di manuale della funzione *s* per una descrizione delle funzioni di smooth disponibili). Tra le caratteristiche principali delle funzioni implementate vi è la selezione automatica dei parametri di smooth, ossia del grado di smoothing da applicare a ogni termine. Nel caso di spline questo si traduce nella specificazione del numero di gradi di libertà effettivi da attribuire a ogni funzione di smooth.

Le stime dei parametri non sono eseguite mediante backfitting, ma mediante minimizzazione di una funzione di likelihood penalizzata; si somma al logaritmo della funzione di likelihood cambiato di segno una penalità per ogni funzione di smooth, che ne disincentivi le oscillazioni evitando problemi di overfitting. Per controllare il bilancio tra i due addendi in gioco, ogni penalità è moltiplicata per i gradi di libertà attribuiti alla corrispondente funzione di smooth (si veda [61] per una descrizione matematica accurata). La stima dei parametri di smooth avviene o minimizzando il criterio Generalized Cross Validation (GCV):

$$\frac{nD}{(n-f)^2}$$

o minimizzando il criterio Un-Biased Risk Estimator (UBRE):

$$\frac{D}{n} + \frac{2\,\sigma^2\,f}{n} - \sigma^2$$

con D la devianza del modello, f il numero di gradi di libertà effettivi del modello e σ^2 il parametro di scala. Si osservi che il secondo criterio è utilizzabile solamente con σ noto.

Oltre al classico approccio frequentistico, le cui assunzioni in questo ambito sono spesso non rispettate, è disponibile un approccio Bayesiano al calcolo degli intervalli di credibilità, particolarmente utile qualora si vogliano ottenere delle stime dal modello.

Esempio

Per studiare il funzionamento delle funzioni descritte sopra si fa uso del dataset *stackloss* disponibile nella libreria MASS. Al suo interno sono riportati i dati ottenuti da 21 giorni di funzionamento di un impianto che ossida ammoniaca per ottenere acido nitrico. La variabile di risposta *stack.loss* è la percentuale di ammoniaca che sfugge senza venire assorbita; le tre variabili esplicative sono $Air.Flow$ flusso dell'aria di raffreddamento, $Water.Temp$ la temperatura dell'acqua di raffreddamento dell'impianto, *Acid.Conc.* la concentrazione dell'acido circolante nell'impianto.

Si vuole paragonare il modello lineare con un modello additivo. L'analisi inizia caricando il dataset, la libreria *mgcv* e fittando il modello lineare a tre predittori:

```
> library(MASS)
> data(stackloss)
> library(mgcv)

> mod <- lm(stack.loss ~ Air.Flow + Water.Temp + Acid.Conc., data=stackloss)
> summary(mod)
[...]
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -39.9197    11.8960  -3.356  0.00375 **
Air.Flow      0.7156     0.1349   5.307  5.8e-05 ***
Water.Temp    1.2953     0.3680   3.520  0.00263 **
Acid.Conc.   -0.1521     0.1563  -0.973  0.34405

Residual standard error: 3.243 on 17 degrees of freedom
Multiple R-Squared: 0.9136,     Adjusted R-squared: 0.8983
F-statistic:  59.9 on 3 and 17 DF,  p-value: 3.016e-09
```

L'analisi dei residui del modello (non riportata) non mette in luce problemi particolarmente evidenti. Si conclude quindi che il modello ben si adatta alla situazione reale, con un valore di R^2 molto elevato.

Il fit del modello additivo generale si effettua con la seguente chiamata:

```
> mod.gam <- gam(stack.loss ~ s(Air.Flow, k=7) +
+ s(Water.Temp, k=7) + s(Acid.Conc., k=7), data=stackloss)
```

La funzione *gam* ha un funzionamento simile a quello di *lm*; i predittori vengono passati alla funzione *s* e l'opzione *k* serve a specificare il numero di nodi nel calcolo delle spline. Il valore $k-1$ rappresenta il numero massimo di gradi di libertà effettivi attribuibili a ogni spline; va quindi controllato in output che il valore di *k* specificato sia abbastanza maggiore del numero di gradi di libertà attribuiti alla corrispondente spline, altrimenti le stime possono essere parecchio distorte. Nell'esempio in questione per tutte e tre le spline si sceglie un numero di nodi pari a 7. L'output del modello si esamina con la chiamata:

```
> summary(mod.gam)

Family: gaussian
Link function: identity
[...]
Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   17.524      0.613   28.59 8.66e-15 ***

Approximate significance of smooth terms:
               edf Est.rank      F  p-value
s(Air.Flow)   1.000        1 19.073 0.000518 ***
s(Water.Temp) 2.565        6  4.746 0.006308 **
s(Acid.Conc.) 1.000        1  1.097 0.311034

R-sq.(adj) =  0.924   Deviance explained = 94.1%
GCV score = 10.735   Scale est. = 7.8899    n = 21
```

In questo caso si hanno due tabelle di significatività: la prima per i coefficienti parametrici (in questo caso solo l'intercetta), la seconda per i termini di smooth. Da questa seconda tabella si nota che i gradi di libertà effettivi (*edf* in tabella) della prima e della terza variabile sono entrambi 1, il che è una chiara spia del fatto che per questi termini niente si guadagna introducendo un trattamento di smoothing e che ben sono descritti da una relazione lineare. Si può quindi fittare un secondo modello che utilizza un trattamento non parametrico della sola seconda variabile esplicativa:

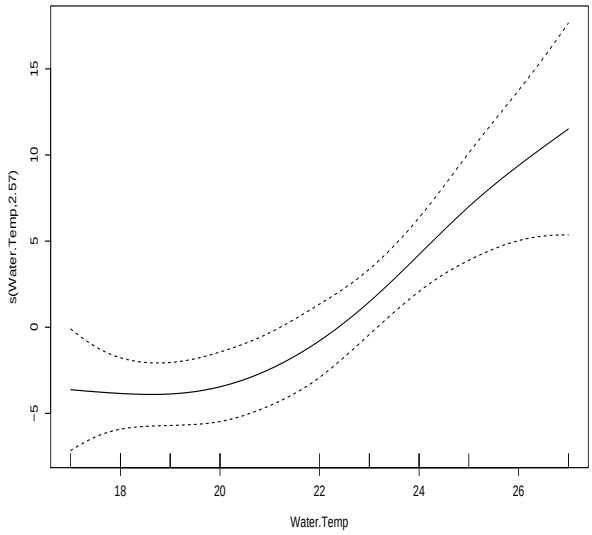

Figura 3.14: La funzione di smooth della variabile $Water.Temp$. Le linee trattegiate delimitano l'intervallo di confidenza della funzione di smooth al 95%.

```
> mod.gam2 <- gam(stack.loss ~ Air.Flow + s(Water.Temp, k=7) + Acid.Conc.,
+ data=stackloss)
> summary(mod.gam2)
[...]
Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -5.3904    11.7485  -0.459 0.652758
Air.Flow      0.5903     0.1352   4.367 0.000518 ***
Acid.Conc.   -0.1478     0.1411  -1.047 0.311034

Approximate significance of smooth terms:
                edf Est.rank     F p-value
s(Water.Temp) 2.565        6 4.746 0.00631 **

R-sq.(adj) =  0.924   Deviance explained = 94.1%
GCV score = 10.735   Scale est. = 7.8899    n = 21
```

Il grafico della spline adattata alla variabile $Water.Temp$ (Fig. 3.14) si ottiene con la chiamata:

```
> plot(mod.gam2)
```

L'andamento che si ottiene non è comunque troppo lontano dalla linearità. Un confronto formale tra il modello lineare e il modello additivo generale si può effettuare mediante funzione *anova*:

```
> anova(mod, mod.gam2)
Analysis of Variance Table

Model 1: stack.loss ~ Air.Flow + Water.Temp + Acid.Conc.
Model 2: stack.loss ~ Air.Flow + s(Water.Temp, k = 7) + Acid.Conc.
   Res.Df     RSS      Df Sum of Sq      F  Pr(>F)
1 17.0000 178.830
2 15.4345 121.777  1.5655    57.053 4.6192 0.03388 *
```

Dall'analisi si conclude che la differenza tra i due modelli è significativa ($P = 0.034$): il modello additivo si adatta meglio al set di dati del modello lineare.

3.6.4 Projection pursuit regression (PPR)

La tecnica di projection pursuit regression (PPR) è una forma modificata di modello additivo generale, che consente di trattare – seppur non esplicitamente – le interazioni tra le variabili esplicative. Il modello PPR applica un modello additivo alle variabili proiettate in un sottospazio di dimensione M scelta dall'utente:

$$y_i = \beta_0 + \sum_{j=1}^{M} \beta_{ij} f_j(\alpha_j^T X) + \varepsilon_i \quad i = 1, \ldots, n \tag{3.11}$$

dove i vettori α_j rappresentano le direzioni di proiezione. Le funzioni f_j sono dette *ridge functions*. Oltre alla stima dei coefficienti l'algoritmo di fit dovrà anche calcolare le M direzioni migliori lungo cui proiettare la matrice dei predittori.

In R è possibile fittre un modello PPR mediante la funzione *ppr*, la quale assume vettori α di lunghezza unitaria, fitta un numero $M_{\max}$ di termini, quindi riduce il loro numero a M eliminando il termine meno importante e rifittando il modello. Sia M che $M_{\max}$ devono essere specificati dall'utente. Come nota di avvertimento è bene tener presente che il risultato finale che si ottiene può differire da calcolatore a calcolatore, dato che l'algoritmo è particolarmente sensibile al compilatore utilizzato.

Esempio

Facendo uso del dataset *stackloss* utilizzato nella Sec. 3.6.3 si adatta un modello PPR alla serie di dati. La sintassi di base è la seguente:

```
> mod.ppr <- ppr(stack.loss ~ Air.Flow + Water.Temp + Acid.Conc.,
+ data=stackloss, nterms=2, max.terms=8, sm.method="gcv")
```

le opzioni *nterms* e *max.terms* e servono per specificare rispettivamente i valori di M che $M_{\max}$. Una tecnica standard per scegliere questi valori consiste nello specificare un alto valore per *max.terms* e impostare $nterms = 2$; si controlla quindi dalle tabelle in output quale sia effettivamente il valore di M migliore e si rifitta il modello modificando di conseguenza il valore *nterms*. L'opzione *sm.method* serve per specificare il metodo di smoothing. L'opzione standard *sm.method="supsmt"* implementa il super smoother di Friedman (si veda [59] per i dettagli), mentre le opzioni *sm.method="spline"* e *sm.method="gcvspline"* utilizzano delle funzioni spline con un numero fissato di nodi o con un numero di nodi scelto automaticamente dalla funzione mediante criterio GCV. Nell'esempio in questione si adotta proprio questo criterio.

L'output della funzione è il seguente:

```
> summary(mod.ppr)
[...]

Goodness of fit:
  2 terms   3 terms   4 terms   5 terms   6 terms   7 terms   8 terms
0.7917983 0.8455534 0.8544320 0.8917782 0.8844758 0.9479827 0.0000000

Projection direction vectors:
           term 1      term 2
Air.Flow    0.31346657  0.28793029
Water.Temp  0.94538911 -0.36324154
Acid.Conc. -0.08932044  0.88608788

Coefficients of ridge terms:
  term 1   term 2
```

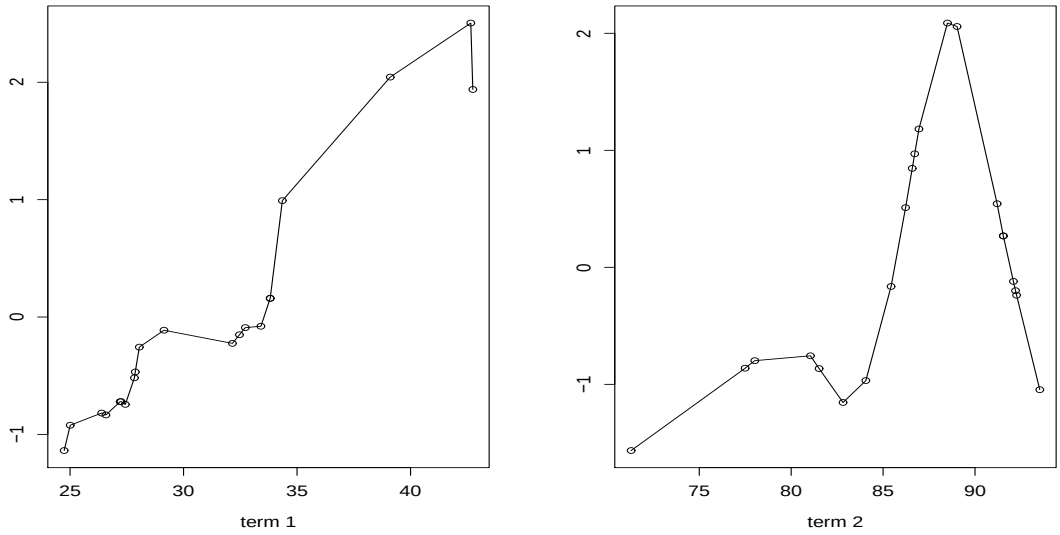

Figura 3.15: Andamento delle funzioni di smooth nelle due direzioni selezionate dall'algoritmo *ppr*.

```
9.736902 1.623251

Equivalent df for ridge terms:
term 1 term 2
 14.39   8.86
```

Nella prima tabella si hanno le statistiche di bontà di adattamento per i modelli da 2 a 8 termini (si tratta della somma dei quadrati dei residui). Si vede che il modello a 2 termini ha la prestazione migliore; non si presenta quindi la necessità di fittare nuovamente il modello modificando l'opzione *nterms*. Nella tabella seguente si hanno le coordinate delle due direzioni di proiezione, mentre nella seguente le stime dei coefficienti delle 2 funzioni di ridge. L'ultima tabella presenta il numero di gradi di libertà effettivi dei due termini.

L'andamento delle due funzioni di smooth (Fig. 3.15) si ottiene con le seguenti chiamate:

```
> par(mfrow=c(1,2))
> plot(mod.ppr)
```

Per confrontare la prestazione del modello appena fittato con quello ottenuto in Sec. 3.6.3 è possibile graficare i valori osservati della variabile dipendente *stack.loss* contro i valori fittati dei due modelli. Come si può osservare in Fig. 3.16 il confronto evidenzia che i valori fittati del modello PPR si adattano molto meglio alla situazione relae di quanto non facciano quelli del modello additivo generale.

3.7 Regressione resistente e robusta

Se gli errori non sono distribuiti normalmente il metodo di fit dei minimi quadrati può fornire un risultato fortemente distorto. Ciò avviene in particolar modo quando la distribuzione degli errori presenta lunghe code. In tali situazioni un possibile rimedio è rimuovere le osservazioni che producono i residui più alti (outliers) e fittare nuovamente i dati rimanenti. Ovviamente, quando i punti che presentano residui elevati sono molti, la perdita di informazione che si ha eliminandoli tutti è troppo grande per essere accettabile. Una tecnica migliore è fare uso di regressione robusta, cioè metodi che sono meno sensibili a dati che si discostino notevolmente dalla media.

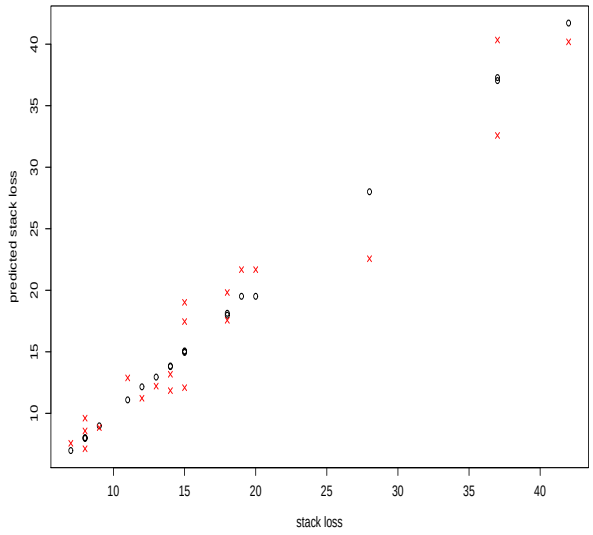

Figura 3.16: Confronto dei valori osservati della variabile dipendete (in ascissa) con quelli predetti dal modello PPR (cerchi neri) e dal modello additivo generali (crocette rosse) di Sec. 3.6.3. Il modello PPR ha una prestazione nettamente migliore.

3.7.1 Regressione robusta

Uno dei metodi di regressione robusta più comuni è detto stima M [37]. Si consideri il modello lineare fittato:

$$\hat{y}_i = X_i \, \hat{\beta} + \hat{\varepsilon}_i$$

dove come di consueto $\hat{\varepsilon}_i$ $(i = 1, \ldots, n)$ sono i residui, X_i è la i-esima riga della matrice $n \times (p+1)$ dei p predittori. Lo stimatore M minimizza la funzione obiettivo:

$$\sum_{i=1}^{n} \rho(\varepsilon_i) = \sum_{i=1}^{n} \rho(y_i - X_i \, \beta) \tag{3.12}$$

dove la funzione ρ specifica il contributo di ogni residuo al totale generale; in questa notazione porre $\rho(\theta) = \theta^2$ corrisponde a utilizzare la tecnica standard dei minimi quadrati.

Derivando la funzione obiettivo rispetto ai coefficienti β e eguagliando a zero i risultati si ottiene un sistema di $p+1$ equazioni che permettono di ricavare le stime dei coefficienti:

$$\sum_{i=1}^{n} \rho'(\hat{y}_i - X_i \, \hat{\beta}) \, X_i = 0 \tag{3.13}$$

Se si definisce la funzione peso w:

$$w_i = w(\hat{\varepsilon}_i) = \frac{\rho'(\hat{\varepsilon}_i)}{\hat{\varepsilon}_i}$$

le equazioni (3.13) possono essere riscritte come:

$$\sum_{i=1}^{n} w_i \, (\hat{y}_i - X_i \, \hat{\beta}) \, X_i = 0$$

Dato che i pesi w dipendono dai residui il fit del modello si esegue con tecniche iterative (IRLS, Iterative Reweighted Least Squares) che assumono una stima iniziale dei coefficienti $\boldsymbol{b}$ e si arrestano quando giungono a convergenza.

Per quanto riguarda le funzioni obiettivo due scelte comuni sono lo stimatore di Huber ρ_H e quello di Tukey ρ_B (detto anche stimatore biquadratico o *bisquare estimator*), definiti come:

$$\rho_H(\varepsilon) = \begin{cases} \frac{1}{2}\varepsilon^2 & \text{per} |\varepsilon| \leq k \\ k|\varepsilon| - \frac{1}{2}k^2 & \text{per} |\varepsilon| > k \end{cases} \qquad \rho_B(\varepsilon) = \begin{cases} \frac{k^2}{6}\left(1 - \left(1 - (\frac{\varepsilon}{k})^2\right)^3\right) & \text{per} |\varepsilon| \leq k \\ \frac{k^2}{6} & \text{per} |\varepsilon| > k \end{cases} \tag{3.14}$$

k è una costante che permette di regolare il funzionamento degli stimatori; bassi valori di k producono stimatori più resistenti alla presenza di outliers ma meno efficienti se gli errori non si discostano troppo dalla normalità. Solitamente si assume $k = 1.345\sigma$ per ρ_H e $k = 4.687\sigma$ per ρ_B, dove σ è la deviazione standard d'errore. Queste scelte portano ad avere una efficienza del 95% rispetto alla tecnica standard dei minimi quadrati quando gli errori sono distribuiti normalmente, ma garantiscono ancora buona protezione dagli outliers. Nella pratica al valore σ si sostituisce una sua stima robusta, che comunemente è $\hat{\sigma} = \text{median}(|\hat{\varepsilon}|)/0.6745$.

Esempio

Per eseguire il fit dei modelli di regressione robusta presentati, in R si utilizza la funzione *rlm* (Robust Linear Model) disponibile nella libreria standard *MASS*. Si consideri ad esempio un impianto chimico che ossida ammoniaca per produrre acido nitrico. Uno dei parametri che regola il funzionamento globale è la temperatura dell'acqua di raffreddamento. Si vuole capire se la perdita di produzione (in percentuale) dipende da tale temperatura in modo lineare. Dato che ci si attende che alcuni valori si discostino notevolmente dalla retta di regressione (fermate parziali dell'impianto) si fitta sia il modello lineare standard sia due modelli di regressione robusta.

L'analisi inizia con l'inserimento dei dati. Un plot dei valori della perdita di produzione contro la temperatura (Fig. 3.17) rivela la presenza di alcuni outliers, a conferma di quanto atteso teoricamente.

```
> temp <- 20:40
> prod <- c(28.6,27.1,2.6,32.1,33.2,34.7,6.6,37.3,39.9,42.2,43.9,43.0,44.0,
+  45.3,47.9,47.9,50.1,50.5,54.2,40.3,55.7)
> plot(temp, prod)
```

Il modello lineare ottenuto con il fit dei minimi quadrati porta a un risultato fortemente distorto, principalmente a causa del punto in corrispondenza del valore $temp = 22$:

```
> mod <- lm(prod ~ temp)
> summary(mod)
[...]
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -12.0991    10.0480  -1.204    0.243
temp          1.6844     0.3283   5.131 5.94e-05 ***

Residual standard error: 9.11 on 19 degrees of freedom
Multiple R-Squared: 0.5808,     Adjusted R-squared: 0.5587
```

Si nota che la stima di σ ottenuta dal fit è piuttosto elevata.

Per tentare di ovviare alla presenza di outliers si fittano due modelli di regressione robusta con stimatori di Huber e biquadratico. Dato che la funzione *rlm* di default implementa lo stimatore di Huber, nel primo caso il fit si esegue con la semplice chiamata:

```
> library(MASS)
> mod.h <- rlm(prod ~ temp)
> summary(mod.h, correl=FALSE)

Call: rlm(formula = prod ~ temp)
Residuals:
```

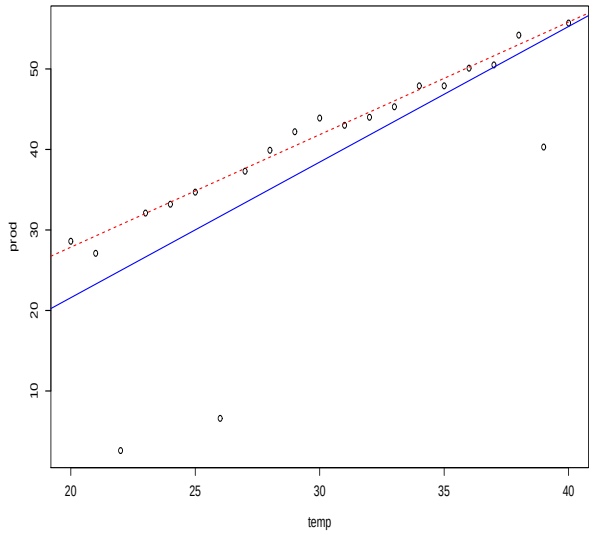

Figura 3.17: Rette di regressione ottenute con il metodo dei minimi quadrati (linea continua) e con un metodo di regressione robusta con stimatore biquadratico (linea tratteggiata). Nel secondo caso l'influenza degli outliers è evidentemente ridotta.

```
      Min        1Q    Median        3Q       Max
-29.30876  -0.69884   0.05111   0.71117   2.35121

Coefficients:
            Value   Std. Error t value
(Intercept) -0.7515  1.6099    -0.4668
temp         1.4100  0.0526    26.8056

Residual standard error: 1.055 on 19 degrees of freedom
```

L'opzione $correl = FALSE$ passata alla funzione *summary* serve a sopprimere l'output della matrice di correlazione fra i coefficienti. Si nota che la stima del coefficiente della temperatura aumenta di circa il 30%, a conferma del fatto che la presenza di outliers distorce il fit standard. Si osservi anche che la stima di σ diminuisce di circa un ordine di grandezza. Nell'output mancano sia la stima di R^2 che di F, oltre che alle inferenze sui coefficienti. Ciò è dovuto al fatto che tali quantità o non sono calcolabili (non nel modo consueto) o non hanno lo stesso significato. Date le complessità teoriche che si devono affrontare per costruire un intervallo di confidenza per i parametri del modello, il metodo che viene più frequentemente utilizzato per raggiungere tale scopo fa uso di simulazioni numeriche di ricampionamento bootstrap (si veda ad esempio [26]).

Come passo successivo si fitta il modello con stimatore biquadratico. L'opzione $method = "MM"$ passata alla funzione *rlm* utilizza tale stimatore dopo aver determinato un "buon" valore iniziale per i parametri (in questo caso infatti avere un buon punto di partenza è determinante per la convergenza del modello):

```
> mod.b <- rlm(prod ~ temp, method="MM")
> summary(mod.b, correl=FALSE)

Call: rlm(formula = prod ~ temp, method = "MM")
Residuals:
     Min       1Q   Median       3Q      Max
```

```
-29.6500  -0.9347  -0.2415   0.4636   2.0568

Coefficients:
            Value   Std. Error t value
(Intercept) -0.1056  1.3002    -0.0812
temp         1.3983  0.0425    32.9150

Residual standard error: 1.212 on 19 degrees of freedom
```

Come si vede i risultati dei due fit ottenuti con la funzione *rlm* sono praticamente identici. In Fig. 3.17 sono confrontati il fit standard e quello con stimatore biquadratico. Le due rette si ottengono con le usuali chiamate:

```
> abline(mod, col="blue")
> abline(mod.b, col="red", lty=2)
```

Come regola generale, quando si sospetta che alcuni punti possano distorcere in maniera importante un modello lineare, si procede al fit sia con le tecniche tradizionali sia usando un metodo robusto. Se i due risultati non sono troppo dissimili allora l'uso del modello standard è giustificato. In caso contrario le stime ottenute con tecniche robuste risultano più affidabili.

3.7.2 Regressione resistente

Un metodo alternativo di regressione detto LTS (Least Trimmed Squares) si basa sulla minimizzazione di una porzione $q < n$ dei quadrati dei residui ordinati in ordine crescente. Una tecnica di tale genere sopporta quindi un certo numero di osservazioni molto distanti dalla media senza fornire stime distorte dei parametri. Si parla in questo caso di regressione resistente.

In R la funzione che esegue una regressione resistente è *ltsreg* disponibile all'interno della libreria *MASS*. Il suo standard è di scegliere il valore di q secondo la seguente formula:

$$q = \lfloor n/2 \rfloor + \lfloor (p+1)/2 \rfloor$$

dove $\lfloor x \rfloor$ indica l'intero più vicino minore o uguale a x. Con i dati dell'esempio precedente si ha:

```
> mod.lts <- ltsreg(prod ~ temp)
> mod.lts
Call:
lqs.formula(formula = prod ~ temp, method = "lts")

Coefficients:
(Intercept)         temp
      2.132        1.309

Scale estimates 0.5667 0.6527
```

Si noti che in questo caso non si ha nessuna stima dell'errore sui parametri. Per ovviare a questa carenza si è costretti a ricorrere a tecniche bootstrap (si veda Sec. 12.3 per il trattamento di questo specifico esempio mediante ricampionamento bootstrap). Nell'ultima riga vengono presentate due stime indipendenti della variabilità degli errori del modello.

Capitolo 4

Analisi della varianza

4.1 ANOVA

La procedura per eseguire una ANOVA passa attraverso il fit del modello lineare corrispondente.

4.2 ANOVA a una via

In questo caso, che è il più semplice, si hanno r serie di dati che si classificano in base a un solo criterio. Il modello lineare si scrive come:

$$x_{ij} = \mu + \alpha_i + \varepsilon_{ij} \qquad i = 1, \ldots, r \quad j = 1, \ldots, n_i$$

con $\varepsilon_{ij} \sim N(0, \sigma^2)$. α_i esprime l'effetto dovuto all'appartenenza al gruppo i-esimo. L'ipotesi nulla H_0 è che per ogni i sia $\alpha_i = 0$, mentre l'ipotesi alternativa H_1 è che almeno un α_i sia diverso da 0.

Nel seguente esempio è riportata la procedura che permette di eseguire il test in R.

Esempio

Siano A e B gli errori di battitura per minuto di due dattilografi. Ci si chiede se ci sia differenza significativa fra le loro abilità.

Si inizia inserendo i dati e unendoli in un unico vettore:

```
> A <- c(1,2,3,4,4,5,7,9)
> B <- c(3,3,5,8)
> Dati <- c(A, B)           # concateno i dati
```

È quindi necessario usare un secondo vettore in cui tenere traccia del gruppo a cui appartengono i valori in *Dati*. Si crea quindi a tale scopo il vettore *gruppo*:

```
> fA <- rep(1, length(A))       # etichetta di gruppo: 1
> fB <- rep(2, length(B))       # etichetta di gruppo: 2
> gruppo <- factor(c(fA, fB))   # concateno i gruppi in un fattore
```

Essendo l'appartenenza ai gruppi un dato categoriale, si usa la funzione *factor* che forza R a trattarli in tal modo e non come dati numerici quantitativi. A tal punto occorre fittare il modello lineare che mette in relazione gli errori di battitura (in *Dati*) con il dattilografo (in *gruppo*):

```
> modello <- aov(Dati ~ gruppo)   # fit del modello lineare
> anova(modello)                  # tabella ANOVA

Analysis of Variance Table
```

```
Response: Dati
          Df Sum Sq Mean Sq F value Pr(>F)
gruppo     1  0.375   0.375   0.058 0.8145
Residuals 10 64.625   6.462
```

da cui si conclude che non vi è differenza significativa fra i due dattilografi.

Per valutare i coefficienti del modello lineare μ, α_1 e α_2 si possono usare le chiamate seguenti:

```
> mean(Dati)
[1] 4.5

> model.tables(modello)
Tables of effects

 gruppo
         1    2
    -0.125 0.25
rep  8.000 4.00
```

da cui si ha: $\mu = 4.5$ (media generale), $\alpha_1 = -0.125$ (effetto dovuto all'appartenenza al primo gruppo) e $\alpha_2 = 0.25$ (effetto del secondo gruppo). La funzione *model.tables* è molto utile per tabelle sui fit di modelli lineari ottenuti con la funzione *aov*, soprattutto in casi complessi (si veda ad esempio la Sec. 4.10). □

Se si vogliono calcolare le medie e varianze dei due campioni un modo rapido per procedere è fare uso della funzione *tapply*:

```
> tapply(Dati, gruppo, mean)     # calcola la media dei vari gruppi
> tapply(Dati, gruppo, var)      # calcola la varianza dei vari gruppi
```

tapply suddivide i dati passati come primo argomento secondo le chiavi di classificazione passate come secondo argomento e ad ognuno di questi sottogruppi applica la funzione specificata come terzo argomento.

4.2.1 Test per l'omogeneità delle varianze

Per verificare che l'ipotesi di omogeneità delle varianze sia soddisfatta è possibile usare il test di Bartlett:

```
> bartlett.test(Dati, gruppo)
```

il cui output evidenzia che non ci sono problemi dovuti a differenza di varianza nei due gruppi:

```
        Bartlett test for homogeneity of variances

data:  Dati and gruppo
Bartlett's K-squared = 0.0373, df = 1, p-value = 0.8468
```

Il test di Bartlett ha il difetto di essere molto sensibile all'ipotesi di normalità dei dati e fornisce troppi risultati significativi se i dati provengono da distribuzioni con lunghe code. Per ovviare a questo inconveniente si può ricorrere al test di Fligner-Killeen, uno dei test per l'omogeneità delle varianze più robusto per scostamenti dalla normalità [17]. Per questo test la sintassi è:

```
> fligner.test(Dati, gruppo)

        Fligner-Killeen test for homogeneity of variances

data:  Dati and gruppo
Fligner-Killeen:med chi-squared = 0.0125, df = 1, p-value = 0.911
```

In questo caso particolare i risultati dei due test coincidono non mettendo in luce nessun problema di non omogeneità delle varianze.

Se l'ipotesi di omogeneità delle varianze non è soddisfatta, è possibile utilizzare il test F con una correzione (dovuta a Satterthwaite) ai gradi di libertà della varianza d'errore. Detta n_i la numerosità di ogni gruppo e s_i^2 la relativa varianza, il numero di gradi di libertà *df* da assegnare alla varianza d'errore si calcola come:

$$df = \frac{\left(\sum \mathrm{v}_i\right)^2}{\sum \mathrm{v}_i^2/(n_i - 1)} \qquad , \qquad \mathrm{v}_i = (n_i - 1)s_i^2.$$

Nel caso dell'esempio in questione si ha:

```
> n <- tapply(Dati, gruppo, length)   # numerosita' dei gruppi
> s2 <- tapply(Dati, gruppo, var)     # varianze dei gruppi
> v <- (n - 1) * s2
> df <- sum(v)^2/sum(v^2/(n - 1))     # gdl della varianza d'errore
> df
[1] 9.921307
```

da confrontarsi con il valore 10 riportato nella tabella ANOVA. Dato che non vi sono particolari problemi dovuti alla disomogeneità delle varianze i due numeri sono quasi identici. La significatività della differenza fra i due gruppi si testa con la chiamata:

```
> 1 - pf(0.058, 1, df)
[1] 0.8145891
```

4.3 Contrasti

Si definisce contrasto fra le medie di r gruppi $\mu_1, \ldots, \mu_r$ una combinazione lineare L

$$L = \sum_i c_i \mu_i, \tag{4.1}$$

dove tutti i c_i sono noti e $\sum_i c_i = 0$. Ad esempio:

- $\mu_1 - \mu_2$ è un contrasto con $c_1 = 1$ e $c_2 = -1$, equivalente al paragone delle medie dei gruppi 1 e 2. Tutte le differenze fra coppie di gruppi sono contrasti.
- $(\mu_1 + \mu_2)/2 - \mu_3$ è un contrasto, che risulta direttamente interpretabile come il confronto fra la media dei gruppi 1 e 2 combinati contro la media del gruppo 3.

Quanto detto vale in un disegno bilanciato in cui la taglia di tutti i gruppi è uguale. Se ciò non è vero si definisce contrasto la combinazione lineare

$$L = \sum_i n_i c_i \mu_i, \tag{4.2}$$

con la condizione

$$\sum_i n_i c_i = 0. \tag{4.3}$$

È facile verificare che nel caso di gruppi di uguale taglia n, l'Eq. (4.2) si riduce all'Eq. (4.1) moltiplicata per n.

In R sono disponibili diversi tipi di contrasti fra gruppi, utili per esplorare disegni sperimentali differenti. La libreria *multcomp* mette a disposizione i test più usati. Si noti che fra di essi non vi sono i test di Duncan e di Newman-Keuls; benché molto diffusi questi test non garantiscono protezione contro errori di tipo I sull'esperimento e sono quindi sconsigliati dai più moderni testi sull'argomento [36].

4.4 Contrasti fra due gruppi: test di Tukey

Il test HSD (Honest Significant Difference) di Tukey è una tecnica tramite quale è possibile confrontare fra loro a due a due le medie dei vari gruppi.

Si abbiano ad esempio tre campioni A, B e C. Come primo passo si esegue il confronto indifferenziato:

```
> A <- c(400, 450, 420, 430, 380, 470, 300)
> B <- c(300, 350, 380, 270, 400, 320, 370, 290)
> C <- c(270, 300, 250, 200, 410)
> na <- length(A)
> nb <- length(B)
> nc <- length(C)
> dati <- c(A, B, C)
> g <- factor(c(rep("A",na), rep("B",nb), rep("C",nc)))
> anova(res <- aov(dati ~ g))

Analysis of Variance Table

Response: dati
          Df Sum Sq Mean Sq F value   Pr(>F)
g          2  45057   22529  6.5286 0.007875 **
Residuals 17  58663    3451
```

Il confronto rivela una differenza altamente significativa. Si può procedere ad analizzare da dove questa differenza tragga origine mediante un test di Tukey:

```
> TukeyHSD(res, "g", ordered=TRUE)
  Tukey multiple comparisons of means
    95% family-wise confidence level
    factor levels have been ordered

Fit: aov(formula = dati ~ g)

$g
         diff        lwr      upr     p adj
B-C  49.00000 -36.910577 134.9106 0.3326095
A-C 121.14286  32.903658 209.3821 0.0069972
A-B  72.14286  -5.850313 150.1360 0.0724100
```

Il test evidenzia che esiste differenza altamente significativa fra i gruppi A e C, mentre le altre differenze non sono significative. L'opzione *ordered* richiede che i gruppi siano ordinati in ordine crescente per media prima di effettuare il test. Se si vuole condurre il test a un differente livello, ad esempio $\alpha = 0.01$ si può effettuare la chiamata:

```
> TukeyHSD(res, "g", ordered=TRUE, conf.level=0.99)
```

4.5 Contrasti fra due gruppi: test di Dunnet

Questo test si impiega per confrontare un gruppo di controllo con diversi gruppi sperimentali. Per poterlo impiegare in maniera semplice e rapida è necessario installare una libreria supplementare, che non fa parte della distribuzione standard di R, ossia la libreria *multcomp* che a sua volta dipende dalla libreria *mvtnorm* (entrambe disponibili presso il sito *www.r-project.org*).

Esempio

Si stabilisca tramite test di Dunnett se qualcuno dei fattori B, C e D differisce significativamente dal controllo A.

```
> A <- c(2, 2, 3, 5, 1, 4, 6)
> B <- c(3, 5, 7)
> C <- c(6, 8, 7)
> D <- c(2, 4, 3)
> dati <- c(A, B, C, D)
> gp <- factor(c(rep("A",7), rep("B",3), rep("C",3), rep("D",3)))
> library(multcomp)
```

La libreria mette a disposizione la funzione *glht*[1], che viene chiamata nel modo seguente:

```
> mod <- aov(dati ~ gp)
> cont <- glht(mod, linfct = mcp(gp = "Dunnett"))
> confint(cont)

         Simultaneous Confidence Intervals for General Linear Hypotheses

Multiple Comparisons of Means: Dunnett Contrasts

Fit: aov(formula = dati ~ gp)

Estimated Quantile = 2.7265

Linear Hypotheses:
           Estimate lwr     upr
B - A == 0  1.7143  -1.3290  4.7575
C - A == 0  3.7143   0.6710  6.7575
D - A == 0 -0.2857  -3.3290  2.7575

95% family-wise confidence level
```

da cui si evidenzia che solo il contrasto A vs. C è significativo. L'opzione $linfct$ serve per specificare il tipo di contrasto; in questo caso si richiede un test di Dunnett sulla variabile *gp* (nella pagina di manuale di *glht* vi sono alcuni utili esempi su come costruire i contrasti). È anche possibile utilizzare la funzione *summary* per calcolare la significatività dei confronti singoli o del test globale. Nel primo caso la chiamata è:

```
> summary(cont)

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Dunnett Contrasts

Fit: aov(formula = dati ~ gp)

Linear Hypotheses:
           Estimate Std. Error t value p value
B - A == 0   1.7143     1.1168   1.535  0.3548
C - A == 0   3.7143     1.1168   3.326  0.0169 *
D - A == 0  -0.2857     1.1168  -0.256  0.9903
---
(Adjusted p values reported)
```

[1]Fino alla vesione 0.4-8 le funzioni di interfaccia erano *simint* e *simtest*.

mentre se si volesse una stima di significatività globale si dovrebbe aggiungere l'opzione *test* alla chiamata precedente (si veda la pagina di manuale di *summary.glht* per le opzioni implementate).

4.6 Contrasti multipli

Sia L un contrasto. La sua varianza può essere stimata, nel caso di disegno bilanciato, come:

$$\text{var } L = \sum_{i=1}^{r} \text{var}(c_i \mu_i) = \sum_{i=1}^{r} c_i^2 \text{var}(\mu_i) = \sigma^2 \sum_{i=1}^{r} \frac{c_i^2}{n_i}. \tag{4.4}$$

Dove σ è la varianza d'errore ottenuta come nel test ANOVA. Per disegni non bilanciati la varianza di L si calcola come:

$$\text{var } L = \sum_{i=1}^{r} \text{var}(n_i c_i \mu_i) = \sum_{i=1}^{r} n_i^2 c_i^2 \text{var}(\mu_i) = \sigma^2 \sum_{i=1}^{r} n_i c_i^2. \tag{4.5}$$

Per il teorema di Scheffé la dimensione dell'intervallo di confidenza del contrasto L è data da:

$$\sqrt{(r-1)F^{\alpha}_{r-1,N-r}}\ \sqrt{\text{var } L}.$$

Il contributo di L alla devianza del fattore è:

$$d_L^2 = \sigma^2\ \frac{L^2}{\text{var} L}.$$

A ciascun contrasto spetta 1 gdl.

Esempio

I vettori a, b e c registrano il numero di difetti orari di tre linee produttive. Ci si chiede se vi sia differenza fra le tre linee e da dove eventualmente tale differenza si origini.

```
> a <- c(1,1,2,3,5,3)
> b <- c(1,2,4,2,2,5)
> c <- c(5,5,6,7,4,6)
> dati <- c(a, b, c)
> gp <- factor(rep(LETTERS[1:3], each=6))
> ni <- as.vector(tapply(gp, gp, length)) # num. dati nei gruppi
```

Il punto di partenza è il confronto indifferenziato:

```
> anova(res <- lm(dati ~ gp))

Analysis of Variance Table

Response: dati
          Df Sum Sq Mean Sq F value   Pr(>F)
gp         2 34.111  17.056  9.0294 0.002667 **
Residuals 15 28.333   1.889
```

che evidenzia differenza altamente significativa nel numero di difetti orari delle tre linee. Si possono esaminare le medie dei tre gruppi:

```
> tapply(dati, gp, mean)

       A        B        C
2.500000 2.666667 5.500000
```

In questo caso l'origine della differenza sembra evidente, dato che la linea C ha in media circa il doppio dei difetti delle altre due. Si può essere tentati dal procedere a un'analisi formale provando i contrasti:

1. A ∪ B vs. C: verifica che la media dei gruppi A e B presi insieme differisca dalla media di C. In questo caso $c_1 = c_2 = 1/2$, $c_3 = -1$.

2. A vs. B: verifica che le medie dei gruppi A e B differiscano. Si ha $c_1 = 1$, $c_2 = -1$, $c_3 = 0$.

È però necessario riflettere sul fatto che questa analisi segue alla ispezione dei dati campionari. In questo caso, come tutte le volte che si testano dei contrasti non pianificati a priori, è necessario attuare una correzione per la molteplicità. Dato che è possibile raggruppare i tre oggetti in due gruppi (di uno e due elementi rispettivamente) in tre modi diversi i valori P ottenuti dalla analisi a posteriori dovranno essere moltiplicati per tre. Supponendo invece che i contrasti fossero stati pianificati in precedenza, lo sperimentatore non dovrà effettuare alcuna correzione. Nel seguito si suppone di ricadere in questo secondo caso e non si attua nessuna correzione sui valori di significatività dei contrasti.

Quando i vettori associati ai due contrasti sono ortogonali fra loro e si parla di *confronti ortogonali*. In generale, dato un disegno bilanciato, due contrasti $\mathbf{c} = (c_1, \ldots, c_r)$ e $\mathbf{d} = (d_1, \ldots, d_r)$ si dicono ortogonali se:

$$\sum_{i=1}^{r} c_i d_i = 0. \tag{4.6}$$

Per un disegno non bilanciato l'equazione corrispondente risulta:

$$\sum_{i=1}^{r} n_i c_i d_i = 0. \tag{4.7}$$

Per il problema in esame si definisce la matrice dei contrasti come

```
contrasti <- cbind(c(1/2,1/2,-1), c(1,-1,0))
```

Per calcolare la stima della varianza associata ai due contrasti e il loro intervallo di confidenza si calcola la varianza d'errore del modello:

```
> sigma <- summary(res)$sigma
```

e si fa uso del teorema di Scheffé:

```
> varcontrasti = sigma^2 * colSums(contrasti^2/ni)
> ngp <- length(levels(gp))  # num. gruppi
> n <- length(dati)          # num. tot. dati
> F <- qf(0.95, ngp-1, n-ngp)
> intconf <- sqrt((ngp-1)*F) * sqrt(varcontrasti)
> intconf

[1] 1.864872 2.153369
```

A partire dalla matrice dei contrasti si calcola il peso per cui moltiplicare ciascun dato:

```
> coeff <- (contrasti/ni)[gp,]
```

e infine si valutano i due contrasti:

```
> cont <- colSums(coeff*dati)
> cont

[1] -2.9166667 -0.1666667
```

Si conclude facilmente che il risultato del primo contrasto è significativo mentre non lo è il secondo.

Il contributo del contrasto alla devianza si calcola come:

```
> cont^2 /varcontrasti * sigma^2

[1] 34.02777778  0.08333333
```

Dato che i due confronti sono ortogonali, nel senso precisato sopra, si verifica che questi due numeri sommano a 34.111, ossia alla devianza del fattore esaminato. □

Lo stesso risultato si può raggiungere molto più velocemente utilizzando la libreria *multcomp*. Il procedimento sarebbe stato il seguente:

```
> library(multcomp)       # carica la libreria necessaria
> contrasti <- rbind(c(1/2,1/2,-1), c(1,-1,0))  # contrasti per riga

> mod <- aov(dati ~ gp)
> rescont <- glht(mod, linfct = mcp(gp = contrasti))
> confint(rescont)
[...]
Linear Hypotheses:
       Estimate lwr     upr
1 == 0 -2.9167  -4.6169 -1.2164
2 == 0 -0.1667  -2.1299  1.7966

95% family-wise confidence level
```

In output si osservano il valore dei contrasti (che risultano identici a quelli calcolati manualmente) e l'intervallo di confidenza simultaneo al 95%. La funzione *glht* appartiene alla classe di contrasti multipli di tipo single-step, basati appunto sulla costruzione di un intervallo di confidenza simultaneo per i contrasti controllando l'errore globale di tipo I o family-wise error rate (FWER). Questi metodi garantiscono che il FWER non ecceda il livello α desiderato. Anche in questo caso si può calcolare il contributo di ciascun contrasto alla varianza dovuta al fattore; il calcolo risulta semplicemente:

```
> sigma <- summary(lm(dati ~ gp))$sigma
> coef(rescont)^2/diag(vcov(rescont)) * sigma^2
          1           2
34.02777778  0.08333333
```

La significatività dei singoli contrasti si valuta con la chiamata:

```
> summary(rescont)
[...]
Linear Hypotheses:
       Estimate Std. Error t value p value
1 == 0  -2.9167     0.6872  -4.244 0.00140 **
2 == 0  -0.1667     0.7935  -0.210 0.97238

(Adjusted p values reported)
```

Sia nel calcolo degli intervalli di confidenza che dei valori P viene fatta una correzione per molteplicità (si hanno due contrasti simultanei). Se non si desidera questa correzione, le chiamate alle funzioni saranno rispettivamente:

```
> confint(rescont, calpha = univariate_calpha())

> summary(rescont, test = adjusted("none"))
```

Nel caso di disegni non bilanciati si procede come nel caso seguente.

Esempio

Si abbiano i tre gruppi:

```
> a <- c(1,1,2,3,5,3,2)
> b <- c(1,2,4,2,2,5)
> c <- c(5,5,6,7,4,6)
```

Si eseguano i contrasti $a \cup b$ vs. c e a vs. b.

La forma generale dei contrasti **c** e **d** è in questo caso:

$$\begin{aligned} \mathbf{c} &= (c_1, c_1, c_3) \\ \mathbf{d} &= (d_1, d_2, 0) \end{aligned} \tag{4.8}$$

Imponendo la condizione data dall'Eq. (4.3) si ottengono le equazioni:

$$\begin{aligned} c_1 &= -\frac{c_3 n_c}{n_a + n_b} \\ d_1 &= -\frac{d_2 n_b}{n_a} \end{aligned} \tag{4.9}$$

che portano ai due contrasti:

$$\begin{aligned} L_1 &= \frac{c_3 n_c}{n_a + n_b}(m_c(n_a + n_b) - m_a n_a - m_b n_b) \\ L_2 &= d_2 n_b (m_b - m_a). \end{aligned} \tag{4.10}$$

Le varianze dei due contrasti risultano rispettivamente:

$$\begin{aligned} \mathrm{var}L_1 &= \sigma^2 \frac{c_3^2 n_c}{n_a + n_b}(n_a + n_b + n_c) \\ \mathrm{var}L_2 &= \sigma^2 \frac{d_2^2 n_b}{n_a}(n_a + n_b) \end{aligned} \tag{4.11}$$

e quindi i contributi dei due contrasti alla devianza del fattore si calcolano come:

$$\begin{aligned} d_{F1}^2 &= \frac{n_c(m_a n_a + m_b n_b - m_c(n_a + n_b))^2}{(n_a + n_b)(n_a + n_b + n_c)} \\ d_{F2}^2 &= \frac{n_a n_b (m_a - m_b)^2}{n_a + n_b}. \end{aligned} \tag{4.12}$$

Con un po' di algebra si dimostra che le due componenti sommano alla devianza complessiva attribuibile al fattore in esame.

Tornando al problema in esempio, ponendo $c_3 = d_2 = 1$, si ha:

```
> ma <- mean(a)
> mb <- mean(b)
> mc <- mean(c)
> na <- 7
> nb <- 6
> nc <- 6
> L1 <- (-ma*na-mb*nb + mc*(na+nb))*nc/(na+nb)
> L2 <- nb*(mb-ma)
> vL1 <- nc*(na+nb+nc)/(na+nb)
> vL2 <- nb*(na+nb)/na
> L <- c(L1, L2)
> L
```

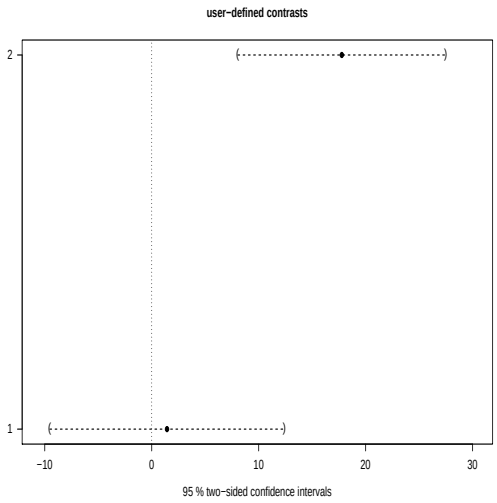

Figura 4.1: Contrasti multipli nel caso dell'esempio 4.6.

```
[1] 17.769231  1.428571
> vL <- c(vL1, vL2)
> sum(L^2/vL)

[1] 36.18922
```

E si verifica facilmente che questo numero coincide con la devianza del fattore *gp*:

```
> dati <- c(a,b,c)
> gp <- factor(c(rep("A",na), rep("B",nb), rep("C", nc)))
> anova(res <- lm(dati ~ gp))

Analysis of Variance Table

Response: dati
          Df Sum Sq Mean Sq F value  Pr(>F)
gp         2 36.189  18.095  10.141 0.00143 **
Residuals 16 28.548   1.784
```

Si noti che la funzione *glht* può essere chiamata anche in questo caso, dopo aver opportunamente impostato la matrice dei contrasti (ponendo anche in questo caso $c_3 = d_2 = 1$). In questo caso la struttura della matrice tiene conto anche della numerosità dei gruppi:

$$\begin{aligned} \mathbf{c} &= (n_a c_1, n_b c_1, n_c c_3) \\ \mathbf{d} &= (n_a d_1, n_b d_2, 0) \end{aligned} \tag{4.13}$$

con c_1 e d_1 dati in Eq. 4.9. Si ha quindi:

$$c_1 = -\frac{1 \cdot 6}{7+6} = -\frac{6}{13} \quad , \quad d_1 = -\frac{1 \cdot 6}{7} = -\frac{6}{7}$$

La matrice dei contrasti è quindi:

```
> contr <- rbind( c(-7*6/13, -6*6/13, 6), c(-6, 6, 0))
```

I test si eseguono con le chiamate:

```
> rescont <- glht(res, linfct = mcp(gp = contr))
> confint(rescont)
[...]
Linear Hypotheses:
       Estimate lwr     upr
1 == 0 17.7692   8.0460 27.4924
2 == 0  1.4286  -9.5318 12.3890

95% family-wise confidence level
```

Si nota che i contrasti risultano uguali al caso precedente (per la scelta appropriata della matrice *contr*). Si può anche verificare che il quadrato dell'errore standard ($res1\$sd^2$) diviso per la varianza d'errore σ^2 risulta identico ai valori calcolati in vL in precedenza. La libreria *multcomp* possiede anche una funzione *plot*:

```
> plot(ct)
```

che produce in output il grafico di Fig. 4.1.

4.7 ANOVA a due vie senza repliche

In questo caso i dati possono essere classificati secondo due chiavi di classificazione e disposti in matrice. La logica del test non cambia, dato che occorrerà semplicemente costruire il modello lineare appropriato e fittarlo. Il modello lineare si scrive come:

$$x_{ij} = \mu + \alpha_i + \beta_j + \varepsilon_{ij} \qquad i = 1, \ldots, r \quad j = 1, \ldots, c$$

con $\varepsilon_{ij} \sim N(0, \sigma^2)$. α_i esprime l'effetto dovuto all'appartenenza all'i-esimo livello del fattore di riga, mentre β_j è l'effetto dovuto al livello j-esimo del fattore di colonna.

Esempio

In un disegno bilanciato a due fattori incrociati un vasto terreno viene diviso in appezzamenti, nei quali vengono seminate tre diverse varietà di granoturco. Gli appezzamenti di terreno sono sottoposti a cinque tipi diversi di preparazione. Entrambi i trattamenti sono assegnati in maniera casuale agli appezzamenti. Lo scopo dell'esperimento è verificare se vi sia differenza significativa in produttività (in quintali) dei vari appezzamenti in relazione ai due fattori.

	preparazione				
varietà	1	2	3	4	5
1	100	120	110	100	90
2	130	120	150	120	140
3	100	100	120	110	110

Tabella 4.1: Produttività di granoturco (in quintali per appezzamento), in relazione ai fattori *varieta* e *preparazione*.

Occorre introdurre innanzitutto i 15 valori di produttività, riportati in Tab. 4.1, in un apposito vettore. Si possono organizzare i dati per riga in modo che i primi 5 siano relativi alla prima varietà di granoturco e alle differenti modalità di preprazione e così via:

```
> prod <- c(10,12,11,10,9, 13,12,15,12,14, 10,10,12,11,11)*10
> varieta <- gl(3, 5)            # fattore a 3 livelli con 5 ripetizioni
> preparazione <- gl(5, 1, 15)   # fattore a 5 livelli e 1 ripetizione
```

varieta e *preparazione* sono i vettori (di dimensione 15) che classificano i dati. A questo punto è possibile fittare il modello lineare e produrre la tabella ANOVA:

```
> av <- aov(prod ~ varieta + preparazione)
> anova(av)

Analysis of Variance Table

Response: prod
             Df  Sum Sq Mean Sq F value   Pr(>F)
varieta       2 2293.33 1146.67 10.1176 0.006445 **
preparazione  4  573.33  143.33  1.2647 0.359093
Residuals     8  906.67  113.33
```

Si conclude che non vi è evidenza di differenza significativa nella produttività dovuta alle differenti preparazioni del terreno. Si nota invece che vi è differenza altamente significativa fra le tre varietà di granoturco. □

Se si volesse detrminare l'origine della differenza si potrebbe ricorrere al test di Tukey:

```
> TukeyHSD(av, "varieta")
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = prod ~ varieta + preparazione)

$varieta
    diff        lwr       upr     p adj
2-1   28   8.760818 47.239182 0.0078992
3-1    4 -15.239182 23.239182 0.8270933
3-2  -24 -43.239182 -4.760818 0.0179607
```

Si evidenzia che vi è differenza significativa fra la varietà 2 e le altre, le quali non differiscono fra loro per quanto riguarda la produttività.

4.7.1 Efficienza del disegno a blocchi randomizzati

Talvolta uno dei fattori di un disegno sperimentale a due vie è utilizzzato come fattore di blocco e si parla di esperimento a blocchi randomizzati (RB). In questo caso non interessa la differenza fra i vari livelli del blocco, il quale viene introdotto per separare parte della variabilità del campione in esame e sottrarala dalla varianza d'errore residua. I fattori di blocco sono utili nel caso in cui le unità sperimentali presentino della variabilità fuori dal controllo dello sperimentatore. Ad esempio una popolazione può essere divisa per classi d'età o per sesso se si sospetta che ai fini dell'esperimento tali fattori possano introdurre una differenza significativa.

Una volta condotto un esperimento con un disegno a blocchi randomizzati, è spesso di interesse stimare quanto efficiente sia stata la scelta dei blocchi nel migliorare la precisione raggiunta. In altre parole si vuole verificare se i blocchi sono stati scelti in maniera appropriata per trattare i dati sperimentali. Per far questo si è soliti confrontare la varianza $\hat{\sigma}^2_{RB}$ stimata dal modello RB con $\hat{\sigma}^2_{CR}$, ossia la varianza che si sarebbe ottenuta dal modello completamente randomizzato (CR), cioè una ANOVA a una via in cui il raggruppamento in blocchi non viene tenuto in conto. Il rapporto $\hat{\sigma}^2_{CR}/\hat{\sigma}^2_{RB}$ è quindi usato per misurare l'efficienza relativa del raggruppamento in blocchi. Detta s^2 la stima della varianza d'errore del modello RB, s^2_b la stima della varianza per il fattore blocchi, $B-1$ i gradi di libertà dei blocchi e $T-1$ i gradi di libertà del trattamento, una stima non distorta di σ^2_{CR} è data da:

$$\hat{\sigma}^2_{CR} = \frac{(B-1)s^2_b + T(B-1)s^2}{TB-1}.$$

	azienda			
clima	1	2	3	4
1	230	220	210	190
2	260	230	260	240
3	250	210	220	210

Tabella 4.2: Tempo di risposta in ms, dal momento dello scatto alla fine dell'elaborazione, di fotocamere digitali prodotte da quattro aziende diverse in tre condizioni climatiche etorogenee.

Esempio

Un fotografo professionista vuole valutare il tempo di risposta allo scatto (in ms) di quattro fotocamere digitali prodotte da diverse aziende. Le fotocamere sono fra loro equivalenti dal punto di vista di ottica e risoluzione in pixel. Il fotografo decide di provare le macchine in tre condizioni climatiche molto diverse (fattore di blocco, *clima*), dato che ritiene che la temperatura possa influire sull'efficienza dei CCD. I tempi di risposta sono dati in Tab. 4.2.

Si inseriscono i dati nei tre vettori *tempi*, *azienda* e *clima*:

```
> tempi <- c(230,220,210,190, 260,230,260,240, 250,210,220,210)
> azienda <- gl(4, 1, 12)
> clima <- gl(3, 4)
```

Il fit del modello produce i seguenti risultati:

```
> av <- aov(tempi ~ azienda + clima)
> anova(av)
Analysis of Variance Table

Response: tempi
          Df  Sum Sq Mean Sq F value  Pr(>F)
azienda    3 1891.67  630.56  5.1591 0.04238 *
clima      2 2600.00 1300.00 10.6364 0.01065 *
Residuals  6  733.33  122.22
```

Si evidenzia differenza significativa sia fra le quattro case produttrici sia fra le condizioni climatiche. In questo caso la scelta del fattore di blocco si è rivelata buona visto che gran parte della devianza viene "spiegata" da tale fattore. L'efficienza di questo disegno rispetto a un disegno completamente randomizzato si ottiene con le chiamate:

```
> s2 <- 122.22
> sb2 <- 1300
> T <- 4
> b <- 3
> ((b-1)*sb2 + T*(b-1)*s2)/(T*b-1)/s2
[1] 2.661192
```

Quindi un disegno che trascurasse il fattore *clima* necessiterebbe di circa 2.7 volte più dati per raggiungere la stessa precisione sperimentale del disegno a blocchi.

4.8 ANOVA a due vie con repliche

In questo caso è possibile stimare oltre agli effetti dovuti ai due criteri di classificazione anche quelli dovuti alla loro interazione. Il modello lineare si scrive come:

$$x_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \varepsilon_{ijk} \qquad i = 1, \ldots, r \quad j = 1, \ldots, c \quad k = 1, \ldots, n_i$$

con $\varepsilon_{ijk} \sim N(0, \sigma^2)$. Il termine γ_{ij} esprime l'interazione fra le due chiavi di classficazione, ed è generalmente l'effetto che interessa testare. L'ipotesi nulla H_0 è che per ogni i sia $\gamma_{ij} = 0$, mentre l'ipotesi alternativa H_1 è che almeno un γ_{ij} sia diverso da 0.

Esempio

Si vuole valutare l'efficacia di cinque diversi concimi chimici su tre varietà di pomodoro. Dato che si sospetta che vi sia interazone tra i fattori il disegno viene replicato due volte per un totale di 30 appezzamenti.

Per prima cosa si costruisce il vettore con le produttvità (in quintali) dei vari appezzamenti e i due vettori relativi alle classificazioni per concime e per varietà:

```
> blocco1 <- c(96,104,105,103,102, 97,105,98,101,93, 99,102,92,101,101)
> blocco2 <- c(96,95,96,106,103, 102,98,101,108,103, 100,93,106,98,103)
> produzione <- c(blocco1, blocco2)
> concime <- gl(5,1,30)
> varieta <- gl(3,5,30)
```

Si fitta quindi il modello lineare includendo il termine di interazione:

```
> av <- aov(produzione ~ varieta * concime)
```

In alternativa la sintassi:

```
> av <- aov(tempi ~ varieta + concime + varieta:concime)
```

produce lo stesso risultato. Il risultato del test evidenzia che non vi è interazione significativa fra i due fattori:

```
> anova(av)
Analysis of Variance Table

Response: produzione
               Df Sum Sq Mean Sq F value Pr(>F)
varieta         2   8.07    4.03  0.1741 0.8419
concime         4  69.53   17.38  0.7504 0.5730
varieta:concime 8  84.27   10.53  0.4547 0.8690
Residuals      15 347.50   23.17
```

Nemmeno le altre variabili risultano statisticamente significative, quindi si conclude che i concimi sono di pari livello e che non vi è differenza nella produttività delle tre varietà.

Se si volesse ottenere una visualizzazione grafica dell'interazione fra i due fattori, si potrebbe richiedere un *interaction plot* come quello mostrato in Fig. 4.2:

```
> interaction.plot(concime, varieta, produzione)
```

dove in assenza assoluta di interazione ci si attende che le linee (relative alle tre diverse varietà) siano parallele.

In caso si trovi una interazione significativa è spesso poco interessate esaminare i livelli dei singoli fattori principali tramite dei contrasti, mentre è possibile procedere a contrasti tra i livelli dell'interazione, ad esempio mediante test di Tukey. Così, se si fosse trovata interazione significativa nell'esempio appena disusso, si sarebbe potuta effettuare la chiamata:

```
> TukeyHSD(av, "varieta:concime")
```

procedendo poi a evidenziare eventuali contrasti interessanti.

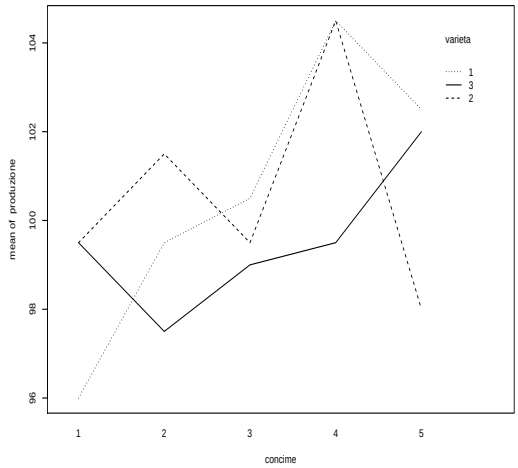

Figura 4.2: Interaction plot per i fattori *concime* e *varieta*. L'apparente presenza di interazione (segmenti non paralleli) non viene evidenziata dal test, probabilmente per l'esiguo numero di repliche.

4.9 Quadrati latini

Questo disegno sperimentale è utile quando i dati possono essere classificati in blocchi secondo non uno ma due fattori. Si supponga ad esempio di voler valutare la qualità del materiale da lavorazione proveniente da tre fornitori A, B, C misurando (su un'opportuna scala) la resistenza dei pezzi prodotti. Si supponga inoltre di utilizzare il meteriale fornito avvalendosi di tre diversi tecnici (detti $t1$, $t2$, $t3$) e di ripetere l'esperimento per tre giorni ($g1$, $g2$, $g3$). In questo tipo di esperimento i fattori "tecnico" e "giorno" rappresentano i blocchi, mentre le differenze nel fattore "fornitore" sono ciò che si intende studiare. In un disegno RB completo, come nelle sezioni precedenti, si dovrebbero avere $3^3 = 27$ misure di resistenza. Il disegno a blocchi incompleto denominato quadrato latino offre la possibilità di utilizzare solamente $3^2 = 9$ misure.

Si usa a tale scopo un disegno del tipo di quello riportato in Tab. 4.3, in cui in ogni riga e in ogni colonna i diversi trattamenti appaiono una e una sola volta. Il modello lineare è in questo caso:

$$x_{ijk} = \mu + \alpha_i + \beta_j + \gamma_k + \varepsilon_{ijk} \qquad i, j, k = 1, \ldots, n$$

con $\varepsilon_{ijk} \sim N(0, \sigma^2)$.

	$g1$	$g2$	$g3$
$t1$	A	B	C
$t2$	B	C	A
$t3$	C	A	B

Tabella 4.3: Quadrato latino 3×3.

Tornando all'esempio citato sopra si supponga che i dati riguardanti la resistenza dei materiali siano inseriti nel vettori *res*, e che i vettori $fornitore$, t e g tengano traccia del fornitore, del tecnico che ha manovrato le macchine e del giorno di lavoro.

```
> res <- c(80,104,97, 107,99,78, 110,77,98)
> fornitore <- factor(c("A","B","C", "B","C","A", "C","A","B"))
> g <- gl(3, 3)    # fattore a 3 livelli con 3 ripetizioni
> g
[1] 1 1 1 2 2 2 3 3 3
```

```
Levels: 1 2 3

> t <- gl(3, 1, 9)
> t
[1] 1 2 3 1 2 3 1 2 3
Levels: 1 2 3
```

I dati soddisfano alle condizioni necessarie per un quadrato latino. Il modello si fitta e si analizza con le chiamate:

```
> mod <- aov(res ~ g + t + fornitore)
> anova(mod)
Analysis of Variance Table

Response: res
          Df  Sum Sq Mean Sq F value  Pr(>F)
g          2    2.89    1.44  0.0718 0.93299
t          2  101.56   50.78  2.5249 0.28370
fornitore  2 1169.56  584.78 29.0773 0.03325 *
Residuals  2   40.22   20.11
```

Si evidenzia differenza significativa fra i fornitori, mentre i fattori che costituiscono i blocchi non raggiungono la significatività. In questo caso è evidente che la separazione dei dati in blocchi non ha raggiunto l'effetto sperato; in particolare l'introduzione del blocco relativo al giorno permette una riduzione di devianza particolarmente piccola.

Avendo concluso che la differenza fra i fornitori è significativa è possibile analizzare l'origine di tale differenza mediante un test di Tukey:

```
> TukeyHSD(mod, "fornitore")
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = res ~ g + t + fornitore)

$fornitore
        diff        lwr      upr
B-A 24.66667   3.097011 46.23632 0.0386909
C-A 23.66667   2.097011 45.23632 0.0418889
C-B -1.00000 -22.569655 20.56966 0.9605144
```

Si conclude quindi che il fornitore A differisce significativamente da B e C, mentre questi ultimi non differiscono fra loro.

4.10 Disegni split-plot

Talvolta è opportuno (o conveniente) pianificare un esperimento per testare un fattore su vasta scala (sulle unità principali) e un secondo fattore su scala sperimentale più ridotta (sulle sottounità), annidata con la precedente. Tali necessità si presentano spesso in agricoltura, come nell'esempio seguente.

Tre fertilizzanti vengono testati per la loro efficacia. Il terreno dedicato all'esperimento viene anche suddiviso in due zone a diverso irrigamento, e in ciascuna delle zone si provano tutti e tre i fertilizzanti. Il fattore "fertilizzante" è in questo caso annidato all'interno del fattore "irrigazione". L'esperimento viene ripetuto in quattro appezzamenti diversi (blocchi) e si misura la produttività mensile del terreno.

		blocco			
irrigazione	fertilizzante	1	2	3	4
irr1	fert1	217	188	162	234
	fert2	158	126	122	159
	fert3	229	160	167	191
irr2	fert1	175	195	213	178
	fert2	152	147	180	137
	fert3	155	161	182	156

Tabella 4.4: Produzione degli appezzamenti, in tonnellate mensili per ettaro, sottoposti a diversi regimi di irrigazione e trattati con diversi fertilizzanti.

Diversamente di un disegno randomizzato, dove tutti i trattamenti vengono assegnati a caso alle unità, in questo caso i diversi livelli del fattore "fertilizzante" vengono assegnati a sottoparti delle stesse unità principali. Quindi ci si aspetta una uniformità maggiore all'interno delle unità principali di quanto non si avrebbe, ad esempio, con un disegno a blocchi randomizzati. Il modo migliore per trattare questo scenario è fare uso di due distinte varianze d'errore con cui pesare gli effetti dei trattamenti sulle unità principali e sulle sottounità.

L'analisi inizia con l'inserimento dei dati di Tab. 4.4:

```
> irr <- gl(2, 12)   # fattore a 2 livelli con 12 ripetizioni l'uno
> fert <- gl(3, 4, 24)
> block <- gl(4, 1, 24)
> prod <- c(217,188,162,234, 158,126,122,159, 229,160,167,191,
+  175,195,213,178, 152,147,180,137, 155,161,182,156)
```

Per specificare un disegno annidato in R si usa la funzione *Error* che permette di stratificare il modello lineare specificando quale termine usare come varianza d'errore. La "regola" da seguire in un disegno split-plot è che per pesare la significatività di un effetto si usa come varianza d'errore l'interazione tra il fattore in studio e tutti quelli nei livelli gerarchici soprastanti. In questo caso si ha il fattore di blocco a livello superiore, annidato al suo interno si vuole studiare l'effetto dell'irrigazione (trascurando il fattore "fertilizzante") e quindi, annidata con i due precedenti, quella dei fertilizzanti. La sintassi per fittare tale modello è:

```
> mod <- aov(prod ~ fert*irr + Error(block/irr/fert))
> summary(mod)
Error: block
          Df  Sum Sq Mean Sq F value Pr(>F)
Residuals  3 1073.67  357.89

Error: block:irr
          Df Sum Sq Mean Sq F value Pr(>F)
irr        1  280.2   280.2  0.1195 0.7524
Residuals  3 7031.5  2343.8

Error: block:irr:fert
          Df Sum Sq Mean Sq F value    Pr(>F)
fert       2 9145.1  4572.5 37.7679 6.637e-06 ***
fert:irr   2 1326.1   663.0  5.4765   0.02042 *
Residuals 12 1452.8   121.1
```

Nel fit del modello si usa l'operatore / che serve per specificare l'annidamento di un fattore con quelli che gli stanno a sinistra nella formula. La sintassi usata definisce un modello annidato a tre livelli con *block* che contiene *irr* che a sua volta contiene $fert$.

La prima tabella in output è relativa al fattore di blocco, la cui significatività non ha alcun interesse pratico. La seconda tabella analizza il fattore "irrigazione" usando come varianza d'errore quella dell'interazione *block* : *irr*; questo fattore risulta non significativo. Si noti che tale tabella può essere ottenuta, come accennato sopra, fittando il modello lineare che non contiene il fattore "fertilizzante" e usando come errore il termine di interazione:

```
> mod2 <- aov(prod ~ irr + block + Error(irr:block))
> summary(mod2)

Error: irr:block
          Df Sum Sq Mean Sq F value Pr(>F)
irr        1  280.2   280.2  0.1195 0.7524
block      3 1073.7   357.9  0.1527 0.9215
Residuals  3 7031.5  2343.8
[...]
```

La terza tabella (identificata come $Error : block : irr : fert$) analizza il fattore "fertilizzante" (annidato con i due fattori *block* e *irr*), che risulta altamente significativo. Anche l'interazione tra fertilizzante impiegato e metodologia di irrigazione risulta significativa. Per chiarire da dove queste differenze originino si possono richiedere le tabelle riassuntive con la chiamata:

```
> model.tables(mod, "means")
```

che produce in output la media totale, le medie raggruppate per fattori e la tabella che consente di analizzare l'interazione:

```
Tables of means
Grand mean

172.6667

 fert
fert
     1      2      3
195.25 147.63 175.13

 irr
irr
     1      2
176.08 169.25

 fert:irr
    irr
fert 1      2
   1 200.25 190.25
   2 141.25 154.00
   3 186.75 163.50
```

L'interazione origina dal fatto che la condizione di irrigamento 2 è sfavorevole per i fertilizzanti 1 e 3, ma risulta vantaggiosa per il fertilizzante 2.

Nell'eseguire eventuali contrasti bisogna prestare attenzione al fatto che si dispone di due diverse varianze d'errore. In conseguenza di ciò alcuni contrasti (all'interno delle unità principali) saranno affetti da errore minore di quelli che coinvolgono livelli delle unità principali. In particolare, in un disegno split-plot fittato con la funzione *aov*, il test di Tukey non può essere utilizzato nel modo consueto. Per paragonare le medie dei vari gruppi è necessario un cambio di approccio teorico e affrontare l'analisi mediante modelli a effetti misti (si veda Sec. 4.13.2).

4.11 Prove ripetute

Talvolta le stesse unità sperimentali sono soggette a più misurazioni, ad esempio per valutare l'efficacia nel corso del tempo di un trattamento. Dato che tali misure sono eseguite sugli stessi soggetti non saranno fra loro indipendenti esse non possono essere considerate repliche, ma si usa il termine di misure ripetute.

Con alcune correzioni il disegno split-plot (o, se non vi sono repliche in ogni cella, quello a blocchi randomizzati), può essere utilizzato per trattare anche il caso di misure ripetute. In questo caso le unità principali sono gli individui e le sottounità i diversi istanti di tempo in cui vengono effettuate le misure (si noti comunque, come prima differenza, che in questo caso le misure sono eseguite sull'intera unità e non su parti di essa). Il problema principale a cui è necessario prestare attenzione è quello della *sfericità*, che altro non è che la generalizzazione dell'omogeneità delle varianze al caso delle misure ripetute. Questo problema può essere efficacemente affrontato con una riduzione dei gradi di libertà nella tabella ANOVA, che si riflette quindi in un test più conservativo.

Esempio

Sei sprinter di una squadra giovanile di atletica leggera vengono sottoposti dal loro allenatore allo stesso programma di allenamento. Il loro tempo di reazione allo starter viene testato in quattro diverse occasioni a intervalli settimanali. I tempi di reazione (in millesimi di secondo) sono dati in Tab. 4.5. L'allenatore vuole verificare se il programma di allenamento migliora la reazione degli atleti allo starter. Oltre alla valutazione della significatività globale nel corso del periodo di quattro settimane, l'allenatore è interessato a confrontare i tempi di reazione alla fine del periodo di allenamento con quelli iniziali.

	settimana			
atleta	1	2	3	4
A1	153	174	143	134
A2	154	198	149	136
A3	187	240	152	151
A4	169	190	163	146
A5	138	167	116	120
A6	136	168	125	112

Tabella 4.5: Tempi di reazione allo start della gara di sei atleti in quattro diverse occasioni durante un periodo di allenamento.

Si inseriscono i dati nel modo consueto:

```
> tempi <- c(153,174,143,134, 154,198,149,136, 187,240,152,151,
+ 169,190,163,146, 138,167,116,120, 136,168,125,112)
> atleta <- gl(6, 4)
> settimana <- gl(4, 1, 24)
> rep <- gl(2, 4, 24)
```

Si fitta il modello lineare multistrato:

```
> mod <- aov(tempi ~ settimana + Error(atleta))
> summary(mod)

Error: atleta
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  5 6857.7  1371.5

Error: Within
          Df  Sum Sq Mean Sq F value    Pr(>F)
```

```
settimana  3 11130.5  3710.2  36.934 3.628e-07 ***
Residuals 15  1506.8   100.5
```

Dall'analisi split-plot si conclude che l'effetto dell'allenamento sui tempi di reazione allo starter è altamente significativo (fattore settimana). Si noti che in questo caso per specificare la stratificazione del modello si è usata la sintassi $Error(atleta)$. Equivalentemente si sarebbe potuto specificare come termine d'errore $Error(atleta/settimana)$ dichiarando espicitamente la stratificazione del modello. Come si può facilmente verificare le due sintassi producono esattamente lo stesso risultato (quello che cambia è esclusivamente il nome che viene assegnato alla seconda tabella in output).

Si noti che questo set di dati può essere analizzato anche con un disegno a blocchi:

```
> mod2 <- aov(tempi ~ settimana + atleta)
> anova(mod2)
Analysis of Variance Table

Response: tempi
          Df  Sum Sq Mean Sq F value    Pr(>F)
settimana  3 11130.5  3710.2  36.934 3.628e-07 ***
atleta     5  6857.7  1371.5  13.654 3.921e-05 ***
Residuals 15  1506.8   100.5
```

Come si vede il risultato per il fattore "settimana" è identico al precedente.

A questo punto rimane da correggere i gradi di libertà della tabella ($Error : Within$) per tener conto del fatto che le misure sono state eseguite sugli stessi soggetti. Il modo più veloce (e conservativo) di farlo è dividere tutti i gdl della tabella per i gdl del fattore secondario (cioè "settimana"), e ricalcolare le significatività. Per quanto riguarda il fattore "settimana" si avrebbe:

```
> 1 - pf(36.934, 1, 5)
[1] 0.001743175
```

da cui si nota che la significatività risulta drasticamente diminuita. Per ridurre i gradi di libertà si è ipotizzato che fra le serie di misure vi sia dipendenza totale e che quindi le serie di repliche possano essere rappresentate da una sola serie; da qui nasce il fatto che al fattore "settimana" si assegna 1 gdl.

Questo è comunque il peggior scenario possibile. È possibile avere un test meno conservativo stimando la correzione da apportare ai gradi di libertà a partire dalla matrice di covarianza delle serie di dati, calcolando il parametro ε di Greenhouse-Geisser. Tale parametro viene poi usato come fattore moltiplicativo per corregere i gdl della tabella $Error : Within$. Sia Σ la matrice di varianza covarianza delle serie settimanali di dati, $nlev$ il numero delle ripetizioni di dati (in questo caso $nlev = 4$) e H la matrice definita come:

$$H = I_{nlev} - \frac{1}{nlev} 1_{nlev \times nlev}$$

con I_{nlev} la matrice identità di dimensioni $nlev \times nlev$, $1_{nlev \times nlev}$ la matrice di dimensioni $nlev \times nlev$ di tutti 1. Il parametro ε è definito come:

$$\varepsilon = \frac{tr(\Sigma H)^2}{(nlev - 1) tr(\Sigma H \Sigma H)},$$

dove $tr()$ è l'operatore traccia.

In R la procedura richiede un po' di lavoro. Per prima cosa bisogna ottenere una matrice in cui i soggetti sono inseriti per riga e le repliche settimanali per colonna:

```
> nlev <- nlevels(settimana)   # livelli del fattore settimana
> M <- matrix(tempi, ncol=nlev, byrow=TRUE)  # matrice dei tempi
                                             # colonne = settimane
> M
     [,1] [,2] [,3] [,4]
```

```
[1,]  153  174  143  134
[2,]  154  198  149  136
[3,]  187  240  152  151
[4,]  169  190  163  146
[5,]  138  167  116  120
[6,]  136  168  125  112
```

Si definisce quindi la funzione GG che riceve in input tale matrice, calcola le matrici Σ e H e restituisce in output il valore di ε:

```
GG <- function(m) {
e <- var(m);         # matrice di varianza covarianza dei dati
nlev <- ncol(m);
H <- diag(nlev) - 1/nlev * matrix(1, nlev, nlev);
num <- sum(diag(e %*% H))^2;
den <- (nlev-1) * sum(diag(e %*% H %*% e %*% H));
epsilon <- num/den;
epsilon
}
```

La funzione GG chiamata sulla matrice dei dati fornisce il seguente risultato:

```
> epsilon <- GG(M)
> epsilon
[1] 0.4315778
```

da cui si vede che la correzione è piuttosto rilevante dato che è abbastanza vicina allo scenario peggiore $\varepsilon = 1/3$.

Tenendo conto della correzione la significatività del fattore "settimana" può essere calcolato come:

```
1 - pf(36.934, 3*epsilon, 15*epsilon)
[1] 0.0004867276
```

Rimane da verificare l'ultimo punto di interesse dell'allenatore: verificato che l'effetto dell'allenamento sui tempi di reazione degli atleti è significativo, è di interesse paragonare i tempi finali con quelli iniziali. Si tratta di un confronto unico che può essere eseguito in maniera appropriata con un test t per dati appaiati:

```
> t.test(tempi[settimana=="1"], tempi[settimana=="4"], paired=TRUE)

        Paired t-test

data:  tempi[settimana == "1"] and tempi[settimana == "4"]
t = 8.2004, df = 5, p-value = 0.0004389
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 15.79014 30.20986
sample estimates:
mean of the differences
                     23
```

Dal risultato l'allenatore ha evidenza altamente significativa che il programma ha migliorato la reazione dei suoi allievi. La media delle differenza prima-dopo è di circa 2/100 di secondo. □

Particolare attenzione deve essere posta nell'analisi di dati in cui si abbiano più misurazioni per ogni cella, come nel caso in cui ogni atleta avesse eseguito in ogni occasione settimanale non una ma due prove di reazione allo starter. La tentazione di considerere queste prove come repliche porta a una

analisi scorretta dato che esse violano il principio fondamentale delle repliche, ossia l'indipendenza. In questo caso si parla più correttamente di *pseudorepliche* e il modo più corretto di utilizzare questa informazione è di sostituire alle pseudorepliche in ogni cella il loro valor medio e ricondursi a un disegno del tipo analizzato in precedenza.

4.12 ANCOVA

Si parla di ANCOVA quando si ha a che fare con un modello lineare contenente sia predittori categoriali che quantitativi.

Esempio

Si vuole valutare la bontà di due differenti tecniche di insegnamento della matematica per le prime classi di scuole medie. A tale scopo, all'inizio dell'anno si sottopongono 10 studenti per ognuno dei due gruppi a una prova d'ingresso e i risultati sono inseriti nel vettore x. Gli stessi studenti vengono valutati anche a fine anno e i risultati inseriti in y. Ci si chiede se vi sia differenza significativa fra le due tecniche di insegnamento.

Per prima cosa, oltre ai dati x e y, è necessaria una variabile categoriale g per classificare gli studenti in base all'insegnamento ricevuto.

```
> x <- c(5,10,12,9,23,21,14,18,6,13,7,12,27,24,18,22,26,21,14,9)
> y <- c(20,23,30,25,34,40,27,38,24,31,19,26,33,35,30,31,34,28,23,22)
> g <- gl(2, 10, labels=0:1)
```

La funzione *gl* viene chiamata con un argomento opzionale, *labels*, che serve per specificare le etichette per i diversi livelli di g. Si sceglie pertanto la classificazione:

- $g = 0$: primo metodo di insegnamento.
- $g = 1$: secondo metodo di insegnamento.

Sono possibili tre modelli lineari differenti da analizzare:

1. La regressione è la stessa per tutti i gruppi, ossia $y \sim x$.
2. La regressione differisce fra i gruppi solo per l'intercetta, ossia $y \sim x + g$. In questo caso il coefficiente di g rappresenta la distanza fra le rette di regressione e quindi l'effetto dei diversi metodi di insegnamento.
3. La regressione differisce fra i gruppi anche per coefficiente di regressione, ossia $y \sim x + g + x : g$ (o brevemente $y \sim x * g$). L'effetto dei diversi metodi di insegnamento dipende in questo caso anche dal livello di partenza degli studenti.

Ovviamente l'interpretazione del modello si semplifica se si riesce a fittare un modello senza termine di interazione.

Come punto di partenza si esegue l'analisi del modello completo e di quello senza interazione:

```
> mod <- lm(y ~ x * g)
> mod1 <- lm(y ~ x + g)
```

Si verifica se è possibile eliminare il termine di interazione confrontando i due modelli:

```
> anova(mod1, mod)

Analysis of Variance Table

Model 1: y ~ x + g
Model 2: y ~ x + g + x:g
```

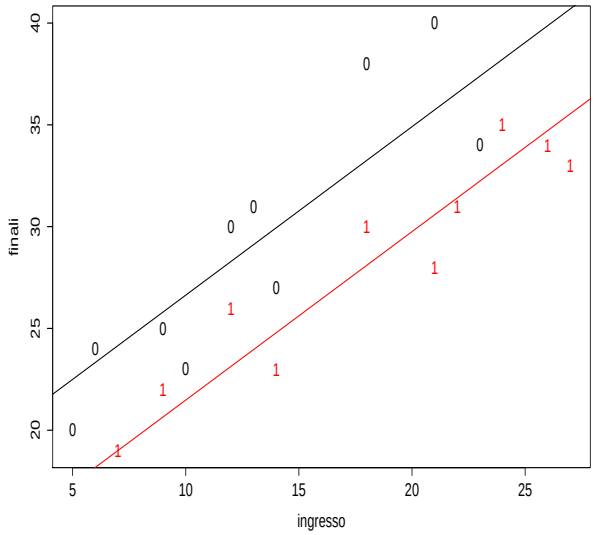

Figura 4.3: Grafico ottenuto dall'ANCOVA che paragona i due diversi metodi di insegnamento dell'esempio 4.12.

```
  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1     17 122.320
2     16 109.983  1    12.337 1.7948 0.1991
```

Si evidenzia che il termine di interazione può essere trascurato. Quindi per semplicità di interpretazione si utilizza il modello $mod1$ nel seguito dell'analisi. I coefficienti del modello sono:

```
> summary(mod1)
[...]
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  18.3600     1.5115  12.147 8.35e-10 ***
x             0.8275     0.0955   8.665 1.21e-07 ***
g1           -5.1547     1.2877  -4.003  0.00092 ***
```

L'effetto dei due metodi di insegnamento è diverso in modo altamente significativo. La differenza fra i due metodi è di circa 5.2 punti. Dal fit del modello si conclude che il metodo $g = 1$ produce risultati peggiori di circa 5.2 punti (il coefficiente del modello ha segno negativo) rispetto al metodo $g = 0$.

Per rappresentare graficamente il risultato dell'analisi si definiscono tre variabili, contenenti le intercette e il coefficiente di regressione comune delle due rette:

```
> int1 <- mod1$coe[1]
> int2 <- mod1$coe[1] +  mod1$coe[3]
> b <- mod1$coe[2]
```

Si crea il grafico e si inseriscono i dati, sostituendo i punti con l'etichetta di gruppo per meglio identificarli:

```
> plot(x, y, type="n", xlab="ingresso", ylab="finali")
> text(x, y, g, col=c("black", "red"))
```

Si inseriscono infine le linee di regressione:

```
> abline(int1, b)
> abline(int2, b, col="red")
```

Il grafico di Fig. 4.3 evidenzia bene la differenza fra i due gruppi.

Alcuni commenti sono opportuni. Se si fossero analizzate esclusivamente le medie dei risultati finali nei due gruppi con un'ANOVA, senza tener conto del livello di partenza degli studenti, si sarebbe trovata una differenza non significativa. L'inserimento nel modello della variabile x cambia drasticamente le conclusioni sui due metodi di insegnamento. Se si analizzano i miglioramenti medi nei due gruppi:

```
> tapply(y - x, g, mean)
   0    1
16.1 10.1
```

si nota che gli studenti del primo gruppo hanno ottenuto un aumento medio di 16.1 punti fra i due test a fronte di un aumento di solo 10.1 punti per gli studenti del secondo gruppo. È da qui che si origina la differenza fra i due metodi di insegnamento.

4.13 Modelli random e modelli misti

I modelli trattati fin qui in questo capitolo vanno sotto il nome di modelli a effetti fissi (*fixed effects models*). Il nome deriva dal fatto che nel modello lineare (scritto per semplicità nel caso di un criterio di classificazione):

$$x_{ij} = \mu + \alpha_i + \varepsilon_{ij} \qquad i = 1, \ldots, r \quad j = 1, \ldots, n_i$$

le quantità α_i vengono considerate costanti, mentre si ha $\varepsilon_{ij} \sim N(0, \sigma^2)$.

Un modello di questo genere è appropriato se si considera il fattore di classificazione come ripetibile, nel senso che è possibile ottenere nuove osservazioni in cui i valori del fattore in esame sono esattamente identici a quelli nello studio. Esempi di questo tipo sono quelle variabili categoriali in cui si ha un numero limitato di classi, come la classificazione di una popolazione per sesso o in fasce d'età. In alcuni esperimenti invece il fattore di classificazione può essere di tipo diverso. Si consideri uno studio clinico condotto su un campione di pazienti sui quali si misura un parametro di interesse. In questo caso la variabile di blocco che identifica i vari pazienti non può essere vista come una quantità riproducibile dato che i pazienti che entrano nello studio sono campionati da una certa popolazione di riferimento. Un nuovo studio selezionerà cioè un campione di pazienti differente dal primo e i livelli della variabile che identifica il paziente non sono quindi ripetibili.

Il modello lineare in questo caso si può scrivere come:

$$x_{ij} = \mu + A_i + \varepsilon_{ij} \qquad i = 1, \ldots, r \quad j = 1, \ldots, n_i$$

dove $A_i \sim N(0, \sigma_a^2)$ e $\varepsilon_{ij} \sim N(0, \sigma^2)$ sono fra loro indipendenti. La modifica al modello è quindi rappresentata dal fatto che si suppone che il fattore abbia a sua volta una distribuzione normale con una certa varianza σ_a^2, che rappresenta la variabilità della popolazione. Modelli di questo tipo sono detti a effetti random (*random effects models*). È importante sottolineare che la finalità di modelli fissi e random è differente. Se in un modello a effetti fissi ci si propone di studiare come cambia il valore di una certa variabile di risposta fra i livelli dei predittori, in un modello a effetti random viceversa quello che interessa è determinare la variabilità della variabile di risposta fra i livelli dei predittori. In questo caso sarà quindi la stima della quantità σ_α^2 a essere di interesse.

Come nota finale è possibile considerare anche modelli in cui compaiano sia fattori fissi sia fattori random. In questo caso si parla di modelli a effetti misti (*mixed effects models*).

In R sono disponibili due librerie tramite cui fittare modelli random e misti. La prima, che fa parte della distribuzione standard, è *nlme* (si veda [42] per una descrizione molto approfondita). La seconda, più recente e versatile, è *lme*4 [4], la quale non è contenuta nell'installazione standard e deve essere scaricata separatamente, assieme alla libreria *Matrix* da cui dipende. Per modelli lineari a effetti misti la maggiore differenza tra le due implementazioni è che le routine della librera *lme*4 sono in grado di trattare efficientemente modelli con effetti random incrociati o parzialmente incrociati e non solo modelli con effetti random annidati, quali quelli discussi nel seguito.

Esempio

In un esperimento, citato in [50], si studia la percentuale di calcio in 4 foglie. Per ogni foglia si eseguono 4 misurazioni indipendenti. Il fattore di blocco $foglia$ è da considerarsi a tutti gli effetti come random.

Si inizia l'analisi inserendo i dati:

```
> calcium <- c(3.28,3.09,3.03,3.03, 3.52,3.48,3.38,3.38,
+  2.88,2.80,2.81,2.76, 3.34,3.38,3.23,3.26)
> foglia <- gl(4, 4)   # fattore foglia
> dati <- data.frame(calcium, foglia)  # creazione di un data frame
```

I dati devono essere inseriti in un data frame per poter condurre l'analisi. Il modello a effetti random si fitta tramite la funzione *lmer* della libreria *lme*4, la quale accetta come argomenti (nella forma minimale) il modello da fittare e il data frame in cui ricercare i dati:

```
> library(lme4)   # si carica la libreria lme4
> mod.r <- lmer(calcium ~ 1 + (1|foglia), dati)
```

si vede che la sintassi della funzione *lmer* richiede che tutte le variabili legate a effetti random vengano specificate tramite l'uso dell'operatore | e delle parentesi tonde (necessarie per garantire la corretta precedenza tra operatori). Il significato della sintassi $(1|foglia)$ è che il modello assume una intercetta random (il valore 1 prima dell'operatore |), e che ogni osservazione che condivide lo stesso valore della variabile di gruppo (ossia $foglia$) avrà lo stesso valore di intercetta. Si noti che è necessario inserire esplicitamente la presenza di un termine costante (il simbolo 1 che segue il carattere tilde) che svolge il ruolo di μ del modello lineare. Questo garantisce che la media della variabile legata all'effetto random possa essere nulla. Il risultato del fit è:

```
> mod.r
Linear mixed-effects model fit by REML
Formula: calcium ~ 1 + (1 | foglia)
   Data: dati
    AIC    BIC logLik MLdeviance REMLdeviance
 -14.55 -13.01  9.277     -20.74       -18.55
Random effects:
 Groups   Name        Variance  Std.Dev.
 foglia   (Intercept) 0.0723791 0.269034
 Residual             0.0066021 0.081253
number of obs: 16, groups: foglia, 4

Fixed effects:
            Estimate Std. Error t value
(Intercept)    3.166      0.136   23.27
```

Nella prima linea dell'output si vede che il modello è stato fittato con tecnica di maximum likelihood ristretta (altre tecniche possono essere specificate con l'opzione *method*, come descritto nella pagina di manuale della funzione *lmer*). Vengono poi presentate alcune statistiche di riepilogo sul fit: indici AIC e BIC (vedi Sec. 9.9.2), log-likelihood ristretta e devianza. Vi è poi la tabella in cui si analizzano gli effetti dei fattori random. In questa tabella vengono riportate le stime della varianza $\hat{\sigma}_a^2 = 0.0724$ e di $\hat{\sigma}^2 = 0.0066$. La colonna etichettata come *Std.Dev.* riporta le radici quadrate delle varianze stimate (e non il loro errore standard come si potrebbe erroneamente pensare). Infine viene presentata una tabella in cui si analizzano gli eventuali fattori a effetto fisso (in questo caso non ve ne sono). Una delle questioni più dibattute è che nella tabella degli effetti fissi non sono riportati i valori P per la significatività dei vari termini[2]. La cosa è dovuta a una precisa scelta da parte del prof. D.

[2]Questo commento si riferisce alla versione della libreria 0.99875. Altre versioni possono avere comportamenti differenti

Bates, autore della libreria, dato che la specificazione di quanti siano i gradi di libertà da attribuire al denominatore per la statistica F è ancora soggetta ad acceso confronto teorico (si veda anche [4] per una trattazione più approfondita dell'argomento), così come è ancora dibattuto il fatto che la statistica F sia appropriata per testare le ipotesi del modello. Si noti per inciso che la scelta implementata nella libreria *nlme* è differente, dato che essa fornisce delle stime dei valori P. Per l'analisi con questa libreria è necessario un passo preliminare mediante chiamata alla funzione *groupedData* come segue:

```
> dati2 <- groupedData(calcium ~ 1 | foglia, dati)
```

Questa funzione serve per specificare la struttura dei dati. Il fit del modello si ottiene quindi con la chiamata:

```
> mod.r2 <- lme(fixed = calcium ~ 1, random = ~ 1 | foglia, data=dati2)
```

dove si notano i primi due argomenti che specificano effetti fissi e random del modello. Il risultato della chiamata si esamina nel modo seguente:

```
> summary(mod.r2)
Linear mixed-effects model fit by REML
 Data: dati2
        AIC       BIC   logLik
  -12.55464 -10.43049 9.277319

Random effects:
 Formula: ~1 | foglia
        (Intercept)  Residual
StdDev:   0.2690357 0.0812532

Fixed effects: calcium ~ 1
               Value Std.Error DF  t-value p-value
(Intercept) 3.165625 0.1360429 12 23.26931       0

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-0.9697575 -0.6831156 -0.2410296  0.6091412  2.1070443

Number of Observations: 16
Number of Groups: 4
```

Si vede che i risultati di questa funzione coincidono con quelli riportati in precedenza (eccetto ovviamente per il fatto che in questo caso viene restituito un valore P per gli effetti fissi). In alcuni casi è possibile che i risultati delle due funzioni differiscano leggermente, date le differenti implementazioni. In ogni caso nella libreria *lme4* è implementata anche la funzione *lmer2* che fornisce risultati identici a *lme*.

4.13.1 Modello a effetti random: due fattori

È possibile pianificare esperimenti in cui appaia più di un fattore random. Si pensi ad esempio a un disegni gerarchico a due fattori, come nell'esempio seguente.

Esempio

Nel caso della determinazione del calcio all'interno delle foglie si può considerare una modifica sperimentale per cui le foglie provengono da piante selezionate casualmente. Il disegno è quindi di tipo gerarchico a 3 passi:

1. si selezionano casualmente le piante;

pianta	foglia	Ca	
1	1	3.28	3.09
1	2	3.52	3.48
1	3	2.88	2.80
2	1	2.46	2.44
2	2	1.87	1.92
2	3	2.19	2.19
3	1	2.77	2.66
3	2	3.74	3.44
3	3	2.55	2.55
4	1	3.78	3.87
4	2	4.07	4.12
4	3	3.31	3.31

Tabella 4.6: Determinazione della percentuale di calcio in un disegno randomizzato gerarchico.

2. dalle piante selezionate si selezionano casualmente alcune foglie;

3. dalle foglie si traggono dei campioni in zone differenti e si sottopongono ad analisi.

Si supponga di selezionare 4 piante e di prelevare da ognuna di esse 3 foglie e di fare su queste due misurazioni indipendenti del livello di calcio [50]. I dati sono riportati in Tab. 4.6.

Il modello lineare è:

$$x_{ijk} = \mu + A_i + B_j + \varepsilon_{ijk} \qquad i = 1, \ldots, r \quad j = 1, \ldots, c \quad k = 1, \ldots, n_i$$

dove $A_i \sim N(0, \sigma_a^2)$ è l'effetto del fattore random dovuto alla scelta della pianta, $B_j \sim N(0, \sigma_b^2)$ è l'effetto del fattore random dovuto alla scelta della foglia e $\varepsilon_{ijk} \sim N(0, \sigma^2)$ la variabilità all'interno della singola foglia. L'esperimento è teso a verificare l'importanza di queste tre fonti di variabilità.

Supponendo di aver inserito i dati di tabella nelle variabili *ca*, *pianta* e $foglia$ del data frame *dati*, il fit del modello si ottiene con la chiamata:

```
> mod.reff <- lmer(ca ~ (1|pianta) + (1|foglia:pianta), dati)
```

La sintassi utilizzata merita qualche commento. Si osserva che fattore dovuto alla variabilità delle foglie viene inserito con la sintassi $(1|foglia : pianta)$ e non semplicemente $(1|foglia)$. Questo è dovuto semplicemente a come è codificata la variabile $foglia$. Se si osserva la Tab. 4.6 si nota che essa vale 1,2,3 all'interno delle varie piante, anche se ovviamente la foglia identificata dal valore 1 nella prima pianta è diversa dalla foglia 1 della seconda pianta. La classificazione delle foglie è cioè interna alle piante. Per risolvere questo problema di codifica si possono seguire due strade equivalenti. La prima è quella utilizzata nel fit, ossia specificare che la variabile $foglia$ è annidata nella variabile *pianta* e quindi come termine $foglia$ si assume l'interazione $foglia : pianta$. La seconda soluzione porterebbe a definire una nuova variabile $foglia2$ che assume valori diversi su ogni foglia:

```
> foglia2 <- gl(12, 2)
> foglia2
 [1] 1  1  2  2  3  3  4  4  5  5  6  6  7  7  8  8  9  9  10 10 11 11 12 12
Levels: 1 2 3 4 5 6 7 8 9 10 11 12
```

e usare questa variabile nel fit:

```
> mod.reff2 <- lmer(ca ~ (1|pianta) + (1|foglia2), dati)
```

Si può verificare facilmente che i risultati dei due modelli sono identici. L'output del fit è:

```
> summary(mod.reff)
Linear mixed-effects model fit by REML
```

```
Formula: ca ~ (1 | pianta) + (1 | foglia:pianta)
   Data: dati
   AIC   BIC logLik MLdeviance REMLdeviance
 8.175 11.71 -1.088      1.723        2.175
Random effects:
 Groups        Name        Variance  Std.Dev.
 foglia:pianta (Intercept) 0.1587486 0.398433
 pianta        (Intercept) 0.3597618 0.599801
 Residual                  0.0067424 0.082112
number of obs: 24, groups: foglia:pianta, 12; pianta, 4

Fixed effects:
            Estimate Std. Error t value
(Intercept)   3.0121     0.3216   9.365
```

da cui si ottengono le stime $\hat{\sigma}_a^2 = 0.365$, $\hat{\sigma}_b^2 = 0.161$ e $\hat{\sigma}^2 = 0.007$. Si vede che sia la variabilità legata alla scelta della pianta sia quella legata alla selezione delle foglie sono rilevanti.

Per riferimento, l'analisi di questo modello con la funzione *lme* si ottiene nel modo seguente:

```
> dati2 <- groupedData(ca ~ 1|pianta/foglia, dati)

> mod.reff3 <- lme(ca ~ 1, random = ~ 1 | pianta/foglia, data=dati2)
> summary(mod.reff3)
Linear mixed-effects model fit by REML
 Data: dati2
       AIC      BIC    logLik
  10.17294 14.71492 -1.086471

Random effects:
 Formula: ~1 | pianta
        (Intercept)
StdDev:   0.6043356

 Formula: ~1 | foglia %in% pianta
        (Intercept)   Residual
StdDev:   0.4013232 0.08157314

Fixed effects: ca ~ 1
               Value Std.Error DF t-value p-value
(Intercept) 3.012083 0.3240437 12  9.2953       0

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.68241774 -0.24760295 -0.08300936  0.35144488  1.99526345

Number of Observations: 24
Number of Groups:
            pianta foglia %in% pianta
                 4                12
```

Si noti l'uso dell'operatore '/' per stratificare il modello, utilizzato sia nella funzione *groupedData*, sia nella funzione *lme*.

4.13.2 Modello a effetti misti

Talvolta è di interesse un modello che contiene sia effetti fissi che effetti random. Ad esempio il modello contenente due fattori, uno fisso e uno random si scrive:

$$x_{ijk} = \mu + A_i + \beta_j + \varepsilon_{ijk} \qquad i = 1, \ldots, r \quad j = 1, \ldots, c \quad k = 1, \ldots, n_i$$

dove $A_i \sim N(0, \sigma_a^2)$ è l'effetto del fattore random, β_j quello del fattore fisso e $\varepsilon_{ijk} \sim N(0, \sigma^2)$.

Esempio

In una città esistono cinque scuole medie. Un protocollo di valutazione sul grado di preparazione che tali scuole offrono ai loro studenti prevede di selezionare due classi di studenti dell'ultimo anno per ognuna di esse e di selezionare nuovamente a caso per ognuna di queste classi due studenti. Gli studenti prescelti vengono sottoposti a un test al termine del quale ricevono una valutazione (espressa su scala centesimale). Si vuole stabilire se vi è differenza nel livello di preparazione degli alunni delle diverse scuole.

I dati sono riportati in Tab. 4.7. Si tratta di un classico esempio in cui si effettua un campionamento stratificato.

punteggio	scuola	classe
92	1	1
79	1	1
86	1	2
98	1	2
76	2	3
74	2	3
100	2	4
87	2	4
78	3	5
90	3	5
90	3	6
91	3	6
95	4	7
82	4	7
77	4	8
74	4	8
91	5	9
85	5	9
83	5	10
82	5	10

Tabella 4.7: Votazioni di 20 studenti in un processo di valutazione fra 5 scuole.

Supponendo di aver inserito i dati di tabella nel data frame *valutazione*, il modello lineare si fitta con la chiamata:

```
> mod.mix <- lmer(punteggio ~ scuola + (1|classe), valutazione)
```

il cui output è:

```
> mod.mix
Linear mixed-effects model fit by REML
Formula: punteggio ~ scuola + (1 | classe)
   Data: valutazione
   AIC   BIC logLik MLdeviance REMLdeviance
```

```
 123.1 129.1 -55.55      136.7        111.1
Random effects:
 Groups   Name        Variance Std.Dev.
 classe   (Intercept) 41.057   6.4076
 Residual             42.414   6.5126
number of obs: 20, groups: classe, 10

Fixed effects:
            Estimate Std. Error t value
(Intercept)   88.750      5.580  15.906
scuola2       -4.500      7.891  -0.570
scuola3       -1.500      7.891  -0.190
scuola4       -6.750      7.891  -0.855
scuola5       -3.500      7.891  -0.444

Correlation of Fixed Effects:
        (Intr) scuol2 scuol3 scuol4
scuola2 -0.707
scuola3 -0.707  0.500
scuola4 -0.707  0.500  0.500
scuola5 -0.707  0.500  0.500  0.500
```

Si ottengono le due stime stime $\hat{\sigma}^2 = 42.4$ e $\hat{\sigma}_a^2 = 41.0$. Nella tabella finale si ha la valutazione dei livelli del fattore a effetto fisso, da cui si evidenzia che i valori delle variabili t sono piuttosto bassi, tali da far pensare che nessuna delle scuole differisca per la prestazione dei suoi studenti dalla scuola 1 (categoria di riferimento).

L'analisi con la funzione *lme* è la seguente:

```
> valutazione2 <- groupedData(punteggio ~ 1 | classe, valutazione)
> mod.mix3 <- lme(punteggio ~ scuola, random = ~ 1|classe, valutazione2)
> summary(mod.mix3)
Linear mixed-effects model fit by REML
 Data: valutazione2
       AIC      BIC    logLik
  125.0971 130.0534 -55.54853

Random effects:
 Formula: ~1 | classe
        (Intercept) Residual
StdDev:    6.438149  6.50385

Fixed effects: punteggio ~ scuola
            Value Std.Error DF   t-value p-value
(Intercept) 88.75  5.594631 10 15.863423  0.0000
scuola2     -4.50  7.912003  5 -0.568756  0.5941
scuola3     -1.50  7.912003  5 -0.189585  0.8571
scuola4     -6.75  7.912003  5 -0.853134  0.4325
scuola5     -3.50  7.912003  5 -0.442366  0.6767
 Correlation:
        (Intr) scuol2 scuol3 scuol4
scuola2 -0.707
scuola3 -0.707  0.500
scuola4 -0.707  0.500  0.500
scuola5 -0.707  0.500  0.500  0.500
```

```
Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.1682385 -0.5847884 -0.1633813  0.6415164  1.4799257

Number of Observations: 20
Number of Groups: 10
```

La significatività del fattore scuola si può valutare con la chiamata:

```
> anova(mod.mix3)
            numDF denDF   F-value p-value
(Intercept)     1    10 1167.7754  <.0001
scuola          4     5    0.2196  0.9165
```

Questo modello multistrato bilanciato può essere trattato in maniera appropriata anche con un approccio di tipo split-plot, in cui il fattore *classe* viene assunto come termine d'errore:

```
> mod.split <- aov(punteggio ~ scuola + Error(classe), valutazione)
> summary(mod.split)

Error: classe
          Df Sum Sq Mean Sq F value Pr(>F)
scuola     4  110.0    27.5  0.2196 0.9165
Residuals  5  626.0   125.2

Error: Within
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals 10  423.0    42.3
```

Si noti che si ha una stima della varianza d'errore nella seconda tabella: $\hat{\sigma}^2 = 42.3$ e che la stima di $\hat{\sigma}_a^2$ si può ottenere dalla prima tenendo conto del fatto che la varianza che si legge in tabella è una stima non distorta di $\sigma^2 + n_{rep}\sigma_a^2$, dove n_{rep} è il numero di repliche per classe. Ricordando che per ogni classe si selezionano 2 studenti si ha $\hat{\sigma}_a^2 = (125.2 - \hat{\sigma}^2)/2 = 41.45$. Per quanto riguarda la differenza tra le scuole, si vede che il valore campionario, $F = 0.2196$, coincide con quanto trovato precedentemente nel modello fittato con la funzione *lme*.

Infine, per confronto, se si fosse scelto di adottare un modello a effetti fissi si sarebbe ottenuto:

```
> mod.fix <- aov(punteggio ~ scuola + classe, valutazione)
> anova(mod.fix)
Analysis of Variance Table

Response: punteggio
          Df Sum Sq Mean Sq F value  Pr(>F)
scuola     4  110.0    27.5  0.6501 0.63969
classe     5  626.0   125.2  2.9598 0.06785 .
Residuals 10  423.0    42.3
```

Si vede che in questo caso la significatività del fattore *scuola* è aumentata, dato che viene pesato dalla sola varianza d'errore.

Esempio

In uno studio di Winter e collaboratori (Winter, Snell & Stray-Gundersen, *Effects of 100% oxygen on performance of professional soccer players*, JAMA 262:227-229, 1989), si investiga l'effetto dell'inalazione di ossigeno puro sul recupero muscolare dopo uno sforzo. Viene misurato il livello di lattato (in mmol/l) nel sangue in 12 calciatori professionisti in due momenti: immediatamente dopo uno sforzo fisco e dopo 4 minuti di recupero. Il test viene ripetuto due volte da ogni calciatore: in una

soggetto	lact	air	time	soggetto	lact	air	time
1	6.8	normal	A	1	8.5	O	A
2	8.0	normal	A	2	7.3	O	A
3	9.6	normal	A	3	12.3	O	A
4	6.7	normal	A	4	6.3	O	A
5	10.9	normal	A	5	7.6	O	A
6	5.9	normal	A	6	5.9	O	A
7	6.5	normal	A	7	9.0	O	A
8	6.2	normal	A	8	5.6	O	A
9	7.9	normal	A	9	8.2	O	A
10	8.0	normal	A	10	8.1	O	A
11	7.7	normal	A	11	7.5	O	A
12	6.9	normal	A	12	7.5	O	A
1	9.3	normal	B	1	9.3	O	B
2	8.1	normal	B	2	8.4	O	B
3	8.3	normal	B	3	11.1	O	B
4	5.3	normal	B	4	7.0	O	B
5	11.4	normal	B	5	7.0	O	B
6	5.0	normal	B	6	5.7	O	B
7	8.8	normal	B	7	8.5	O	B
8	6.6	normal	B	8	6.0	O	B
9	8.9	normal	B	9	9.7	O	B
10	10.8	normal	B	10	10.5	O	B
11	7.9	normal	B	11	10.2	O	B
12	9.0	normal	B	12	10.2	O	B

Tabella 4.8: Livello di lattato nel sangue di 12 atleti professionisti immediatamente dopo una prova di sforzo ($time = A$) e dopo un tempo di recupero di 4 minuti ($time = B$). Il test viene ripetuto in due occasioni da ogni soggetto; durante un recupero l'atleta inala aria normale ($air = normal$), durante l'altra ossigeno puro ($air = O$).

occasione durante il recupero l'atleta respira aria ordinaria, nell'altra ossigeno puro. I dati registrati sono riportati in Tab. 4.8.

Il disegno utilizzato presenta due fattori di trattamento incrociati: *air* a due livelli (inalazione di aria o di ossigeno) e *time* a due livelli (A = immediatamente dopo lo sforzo, B = dopo il periodo di recupero). Quello che interessa è verificare se l'ossigeno altera i livelli di lattato dopo il periodo di recupero, quindi l'effetto di interesse è quello dell'interazione $air : time$.

L'esperimento è stratificato a tre livelli: i soggetti sono le unità di blocco all'interno di cui è annidata la variabile *air* e all'interno di questa è annidata *time* (ogni soggetto viene sottoposto a due prove e dopo ogni prova vengono fatte due misurazioni di lattato). I dati possono essere analizzati in due modi tenendo conto della statificazione e quindi della presenza di diverse fonti d'errore. Il primo è far uso di un disegno split-plot. Si supponga che i dati siano stati inseriti nel data frame *atleti*. Il modello si fitta con la chiamata:

```
> mod <- aov(lact ~ air * time + Error(soggetto/air/time), atleti)
```

in cui si nota chiaramente la definizione della stratificazione individuata. Il risultato è:

```
> summary(mod)

Error: soggetto
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals 11 86.781   7.889

Error: soggetto:air
```

```
          Df  Sum Sq Mean Sq F value Pr(>F)
air        1  0.9919  0.9919  0.4118 0.5342
Residuals 11 26.4956  2.4087

Error: soggetto:air:time
          Df  Sum Sq Mean Sq F value  Pr(>F)
time       1  6.8252  6.8252  6.9603 0.01501 *
air:time   1  0.0469  0.0469  0.0478 0.82895
Residuals 22 21.5729  0.9806
```

Il fattore di blocco è isoltato nela prima tabella, mentre nell'ultima si legge la significatività del fattore in studio, ossia $air : time$ che risulta non significativa ($P = 0.83$). Si conclude che non si ha evidenza che respirare ossigeno cambi i livelli di lattato nel sangue dopo il periodo di recupero in modo differente dal respirare aria. Le tre componenti d'errore sono legate a tre effetti diversi: nella tabella inferiore essa rappresenta l'ineliminabile variabilità legata agli errori presenti su ogni singola misura di lattato; nella seconda tabella rappresenta la variabilità fra differenti prove di uno stesso soggetto (interazione $soggetto : air$); infine nella tabella superiore essa misura la variabilità che esiste fra i soggetti.

I dati in esame possono essere studiati anche con un approccio differente, fittando un modello a effetti misti. Da quanto appena detto i due fattori random sono $(1|soggetto)$ e $(1|soggetto : air)$. Il fit del modello si ottiene quindi con la chiamata:

```
> mod.mix <- lmer(lact ~ air * time + (1|soggetto) + (1|soggetto:air), atleti)

> summary(mod.mix)
Linear mixed-effects model fit by REML
Formula: lact ~ air * time + (1 | soggetto) + (1 | soggetto:air)
   Data: atleti
   AIC BIC logLik MLdeviance REMLdeviance
 178.8 190 -83.38      166.9        166.8
Random effects:
 Groups       Name        Variance Std.Dev.
 soggetto:air (Intercept) 0.71204  0.84383
 soggetto     (Intercept) 1.37591  1.17299
 Residual                 0.98030  0.99010
number of obs: 48, groups: soggetto:air, 24; soggetto, 12

Fixed effects:
            Estimate Std. Error t value
(Intercept)   7.5917     0.5057  15.014
airO          0.2250     0.5311   0.424
timeB         0.6917     0.4042   1.711
airO:timeB    0.1250     0.5716   0.219

Correlation of Fixed Effects:
           (Intr) airO   timeB
airO       -0.525
timeB      -0.400  0.381
airO:timeB  0.283 -0.538 -0.707
```

Usando questo approccio si ha direttamente una stima delle tre componenti della varianza elencate sopra. Si vede che il maggior contributo alla varianza totale è dato dalla variabilità che esiste fra i soggetti (risultato prevedibile), che vale circa 1.38.

Infine, l'analisi mediante funzione *lme* conferma quanto trovato dal modello split-plot:

```
> atleti2 <- groupedData(lact ~ 1 | soggetto/air, atleti)
> mod.mix3 <- lme(lact ~ air * time, random = ~ 1|soggetto/air, atleti2)

> summary(mod.mix3)
Linear mixed-effects model fit by REML
 Data: atleti2
       AIC      BIC    logLik
  180.7652 193.2546 -83.38262

Random effects:
 Formula: ~1 | soggetto
        (Intercept)
StdDev:    1.170507

 Formula: ~1 | air %in% soggetto
        (Intercept)  Residual
StdDev:   0.8450312 0.9902434

Fixed effects: lact ~ air * time
               Value Std.Error DF   t-value p-value
(Intercept) 7.591667 0.5053668 22 15.022093  0.0000
airO        0.225000 0.5314539 11  0.423367  0.6802
timeB       0.691667 0.4042652 22  1.710923  0.1012
airO:timeB  0.125000 0.5717173 22  0.218640  0.8289
 Correlation:
           (Intr) airO   timeB
airO       -0.526
timeB      -0.400  0.380
airO:timeB  0.283 -0.538 -0.707

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.42222106 -0.66109214 -0.03931057  0.63390621  1.80607790

Number of Observations: 48
Number of Groups:
         soggetto air %in% soggetto
               12                24

> anova(mod.mix3)
            numDF denDF  F-value p-value
(Intercept)     1    22 397.3485  <.0001
air             1    11   0.4118  0.5342
time            1    22   6.9604  0.0150
air:time        1    22   0.0478  0.8289
```

4.13.3 Simulazione Monte Carlo per il calcolo dei valori p

Come ampiamente discusso nelle sezioni precedenti, le analisi condotte con la libreria *lme*4 non forniscono la valutazione delle significatività associate agli effetti fissi. Una interessante possibilità per ricavare tali valori è fare uso di una simulazione basata su catene di Markov Monte Carlo. Una classica implementazione è il campionatore di Gibbs, che suddivide l'insieme di parametri da campionare in sottoinsiemi disgiunti, e quindi itera una procedura di generazione dei valori dei parametri, condizionati ai valori correnti degli altri parametri. Per far ciò viene assunta una distribuzione a priori

dei parametri, dalla quale essi vengono campionati. Si procede poi in maniera ciclica, correggendo la distribuzione a priori dei parametri condizionandola ai valori già generati. Dopo un certo numero di cicli, necessari per portare l'algoritmo a convergenza, i valori estratti possono essere assunti come campionati dalla distribuzione a posteriori dei parametri. Nel caso di modelli misti i parametri si suddividono in tre sottoinsiemi: la varianza σ^2; i coefficienti della matrice di varianza covarianza degli effetti random; i valori degli effetti fissi e di quelli random.

La generazione viene fatta campionando la varianza da una distribuzione χ^2, condizionata al valore dei residui. Quindi si campionano i valori della matrice di varianza covarianza a partire da una distribuzione Wishart. Infine, assumendo i valori generati per i primi due sottoinsiemi, si campionano i valori degli effetti fissi e random a partire da una distribuzione normale multivariata.

Questa procedura è disponibile in R utilizzando la funzione *mcmcsamp* della libreria *lme*4.

Con i dati dell'esempio relativo alla valutazione delle scuole medie trattato precedentemente si può stimare la significatività del fattore fisso *scuola* nel modo seguente. Si parte dal modello misto fittato sul campione:

```
> mod.mix <- lmer(punteggio ~ scuola + (1|classe), valutazione)
```

Il campionamento a catena di Markov Monte Carlo si realizza con la chiamata:

```
> samp <- mcmcsamp(mod.mix, 50000)
```

La funzione accetta come argomenti il modello fittato e il numero di campioni di vettori dei parametri da generare a partire dalla loro distribuzione a posteriori. L'oggetto in output contiene molte informazioni; in particolare è possibile visualizzare i valori dei parametri del modello:

```
> as.matrix(samp)[1:2,]   # visualizza solo le prime due righe
     (Intercept)   scuola2   scuola3   scuola4   scuola5       ST1    sigma
[1,]    88.75000  -4.50000 -1.500000  -6.75000 -3.500000 0.9898394 6.503935
[2,]    92.30692 -10.06616 -7.722368 -11.95868  1.500344 0.8359510 4.596266
```

A parte i valori riportati nella penultima colonna (che riguardano la fattorizzazione della matrice delle varianze dei fattori random), nelle altre colonne si hanno i valori degli effetti fissi e della varianza d'errore del modello (sigma). I valori p associati agli effetti fissi si ottengono mediante la seguente funzione, suggerita dal Prof. Bates:

```
mcmcpvalue <- function(samp)
{
   ## elementary version that creates an empirical p-value for the
   ## hypothesis that the columns of samp have mean zero versus a
   ## general multivariate distribution with elliptical contours.

   ## differences from the mean standardized by the observed
   ## variance-covariance factor
   std <- backsolve(chol(var(samp)),
                    cbind(0, t(samp)) - colMeans(samp),
                    transpose = TRUE)
   sqdist <- colSums(std * std)
   sum(sqdist[-1] > sqdist[1])/nrow(samp)
}
```

Per valutare la significatività del fattore *scuola* (colonne da 2 a 5 nella matrice dei parametri campionati), si procede con la chiamata:

```
> mcmcpvalue(as.matrix(samp)[,2:5])
[1] 0.83518
```

da cui si conclude che non vi è differenza tra le cinque scuole considerate nello studio.

Per inciso, una tecnica alternativa abbastanza diffusa per valutare la significatività di un fattore a effetto fisso è basata sul confronto della log likelihood. In questo caso si fittano due modelli: un primo modello contenente il fattore fisso e un secondo senza di esso. Si calcola poi la differenza delle log likelihood (moltiplicate per -2) e la si confronta con il valore critico della distribuzione χ^2 con gradi di libertà pari al numero di parametri eliminati dal modello (il confronto in R si effettua direttamente con la funzione *anova*, chiamata sui due modelli da paragonare). Nell'utilizzare tale approccio bisogna prestare attenzione a due fatti. In primo luogo il fit del modello deve essere eseguito con tecnica di maximum likelihood e non con maximum likelihood ristretta. In secondo luogo il valore p ottenuto con questo test tende ad essere anti conservativo. Il metodo basato sulla simulazione Monte Carlo è attualmente quello consigliato nella valutazione della significatività di effetti fissi.

Il codice per eseguire le analisi di likelihood è il seguente:

```
> mod.mix <- lmer(punteggio ~ scuola + (1|classe), valutazione, REML=FALSE)
> mod.mix.0 <- lmer(punteggio ~ 1 + (1|classe), valutazione, REML=FALSE)

> anova(mod.mix, mod.mix.0)
Data: valutazione
Models:
mod.mix.0: punteggio ~ 1 + (1 | classe)
mod.mix: punteggio ~ scuola + (1 | classe)
          Df     AIC     BIC  logLik  Chisq Chi Df Pr(>Chisq)
mod.mix.0  3 143.192 146.179 -68.596
mod.mix    7 149.573 156.543 -67.787 1.6188      4     0.8054
```

4.13.4 Confronti multipli: test di Tukey e modelli mixed-effect

Come accennato in Sec. 4.10, nel caso di modelli split-plot non è possibile utilizzare le funzioni per confrontare tra loro, ad esempio mediante test di Tukey, le medie dei vari livelli di un dato fattore. Questo problema, derivante dalle implementazioni delle funzioni *aov* e $TukeyHSD$, può essere aggirato analizzando i disegni split-plot come modelli mixed-effect, e usando su di essi la funzione *glht* della libreria *multcomp*, descritta in Sec. 4.5. Questo approccio permette, tra l'altro, di estendere le analisi a modelli non bilanciati.

Con i dati relativi al confronto delle cinque scuole medie è possibile condurre l'analisi mediante test di Tukey con le seguenti chiamate:

```
> mod.mix <- lmer(punteggio ~ scuola + (1|classe), valutazione)

> library(multcomp)
> cont <- glht(mod.mix, linfct = mcp(scuola = "Tukey"))
```

L'opzione $linfct$, che serve per specificare il tipo di contrasto, richiede di eseguire un test di Tukey sulla variabile *scuola*. Il risultato dell'analisi è:

```
> summary(cont)

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts


Fit: lmer(formula = punteggio ~ scuola + (1 | classe), data = valutazione)

Linear Hypotheses:
           Estimate Std. Error z value p value
```

```
2 - 1 == 0   -4.500      7.912  -0.569   0.980
3 - 1 == 0   -1.500      7.912  -0.190   1.000
4 - 1 == 0   -6.750      7.912  -0.853   0.914
5 - 1 == 0   -3.500      7.912  -0.442   0.992
3 - 2 == 0    3.000      7.912   0.379   0.996
4 - 2 == 0   -2.250      7.912  -0.284   0.999
5 - 2 == 0    1.000      7.912   0.126   1.000
4 - 3 == 0   -5.250      7.912  -0.664   0.964
5 - 3 == 0   -2.000      7.912  -0.253   0.999
5 - 4 == 0    3.250      7.912   0.411   0.994
(Adjusted p values reported -- single-step method)
```

Come prevedibile, nessun contrasto raggiunge la significatività.

4.14 MANOVA

In un disegno sperimentale trattabile mediante ANOVA si ha a che fare con una sola variabile dipendente in relazione a uno o più fattori. Talvolta è invece di interesse la misura contemporanea di più variabili dipendenti; in questo caso il problema può essere analizzato mediante ANOVA multivariata (MANOVA).

Questa tecnica consente infatti di trattare contemporaneamente più variabili dipendenti, con il vantaggio di offrire una maggior protezione contro errori di tipo I rispetto all'analisi separata delle singole variabili dipendenti mediante ANOVA. In aggiunta, l'analisi mediante MANOVA può rivelare una differenza significativa tra i gruppi anche se nessuna delle analisi univariate raggiunge la significatività.

Le ipotesi teoriche che stanno alla base del test MANOVA sono:

- In ciascun gruppo il numero di osservazioni deve essere maggiore del numero di variabili.
- Normalità della distribuzione campionaria delle medie delle variabili dipendenti in ciascun gruppo.
- Omogeneità delle matrici di covarianza in tutti i gruppi (estensione multivariata dell'ipotesi univariata di omogeneità delle varianze).
- Correlazioni non elevate tra le variabili dipendenti, in modo da evitare il problema della collinearità.

Qualora l'ultimo punto non sia soddisfatto, e le variabili dipendenti siano fortemente correlate tra loro, si ha ben poco beneficio a introdurne più di una nell'analisi a discapito di una diminuzione dei gradi di libertà. In questi casi è consigliabile l'analisi del disegno sperimentale mediante ANOVA usando una sola delle varibili dipendenti.

4.14.1 Analisi mediante MANOVA: il procedimento

Si supponga di aver misurato p variabili per ciascuna delle n unità sperimentali. Si considera la matrice $\boldsymbol{T}$ della somma dei quadrati e dei prodotti crociati (sum of squares and cross-products, SSCP):

$$\boldsymbol{T} = (n-1)\boldsymbol{S},$$

dove $\boldsymbol{S}$ la matrice di covarianza totale del campione. L'analisi multivariata della varianza è basata sulla scomposizione della matrice $\boldsymbol{T}$. Nel caso di una MANOVA a un fattore di classificazione la matrice di covarianza viene scomposta in due termini: il primo, $\boldsymbol{H}$, è una misura della dispersione tra i gruppi; il secondo, $\boldsymbol{E}$ che tiene conto della variabilità all'interno di ciascun gruppo (covarianza d'errore). Nel caso di più fattori di classificazione, a ogni fattore (e alle eventuali interazioni) si associa una parte $\boldsymbol{H}_i$ della covarianza totale, analogamante al caso di ANOVA.

Per sottoporre a verifica le ipotesi nulla della MANOVA vengono utilizzate varie statistiche. La più popolare è il Λ di Wilks:

$$\Lambda = \frac{|\boldsymbol{E}|}{|\boldsymbol{E}+\boldsymbol{H}|}.$$

In questo caso si repinge l'ipotesi nulla se Λ è troppo piccolo. La quantità $1-\Lambda$ è spesso interpretata come la parte di variabilità delle variabili dipendenti spiegata dal modello. Si noti comunque che questa stima è affetta da distorsione, specialmente nel caso di piccoli campioni.

La seconda statistica d'uso comune è la traccia di Hotelling-Lawley:

$$\text{Hostelling} = trace(\boldsymbol{H}\boldsymbol{E}^{-1}).$$

La terza statistica è la traccia di Pillai:

$$\text{Pillai} = trace[\boldsymbol{H}(\boldsymbol{H}+\boldsymbol{E})^{-1}].$$

La quarta e ultima è il massimo autovalore di Roy:

$$\text{Roy} = max(\lambda_i)$$

dove λ_i sono gli autovalori della matrice $\boldsymbol{H}\boldsymbol{E}^{-1}$. Questa statistica porta a un limite inferiore per il valore P del test.

Per calcolare la significatività dei test MANOVA si ricorre solitamente a delle approssimazione delle distribuzioni delle quattro statistiche presentate in termini della distribuzione F. In alcuni casi i risultati ottenuti con i quattro metodi coincidono, mentre in altri ciò non è vero. In queste occasioni la scelta migliore è basarsi sul criterio di Pillai (default in R), che è più robusto degli altri tre alle violazioni dell'ipotesi di omogeneità delle varianze e covarianze. Nel caso di piccoli campioni il criterio di Pillai costituisce senza dubbio la scelta migliore.

Se il test MANOVA risulta significativo è poi solitamente di interesse chiedesi quali variabili dipendenti siano responsabili di questa differenza e esaminare quali gruppi si differenzino per i valori medi delle variabili dipendenti. Per risolvere il primo problema si possono esaminare i test ANOVA delle singole variabili dipendenti per esplorare come esse contribuiscano alla significatività del test globale. Per affrontare il secondo si possono eseguire dei confronti multivariati tra vettori di medie tra i vari gruppi o eseguire dei test di Tukey sui risultati dei singoli test ANOVA.

Esempio

In un disegno sperimentale a un fattore di classificazione si vuole verificare la differenza nella composizione chimica di manufatti archeologici, attribuiti alla stessa civiltà, e rinvenuti in quattro differenti siti. Vengono misurate contemporanemante le abbondanze (in una opportuna unità di misura) di vari elementi chimici, tra cui si considerano il potassio (K) e il sodio (Na). Si ha un disegno bilanciato con 5 repliche per ogni sito. I dati sono riportati in Tab. 4.9.

Si supponga di importare i dati e di inserirli nel datatset *manufatti*. L'analisi del disegno si esegue mediante funzione *manova*:

```
> mod <- manova(cbind(K, Na) ~ factor(loc), data=manufatti)
```

A primo membro del modello si deve specificare la matrice in cui, per colonna, si inseriscono i valori delle variabili dipendenti. Il risultato del test si esamina mediante la funzione *summary*:

```
> summary(mod, test="Wilks")
            Df   Wilks approx F num Df den Df Pr(>F)
factor(loc)  3 0.60033  1.45319      6     30 0.2278
Residuals   16
```

L'argomento *test="Wilks"* serve per valutare la significatività mediante la statistica di Wilks. In alternativa è possibile selezionare le altre statistiche specificando come argomento *test* i valori "Hotelling-Lawley" o "Roy". Se non si specifica nessun argomento la funzione *summary* seleziona di default la statistica di Pillai:

K	Na	loc
5.4	4.9	1
4.5	6.7	1
3.9	6.4	1
5.3	4.4	1
6.2	2.7	1
4.0	3.6	2
6.9	5.1	2
4.5	2.8	2
6.1	6.6	2
4.5	6.1	2
4.8	5.2	3
5.6	5.7	3
5.1	6.5	3
9.3	7.1	3
7.4	5.4	3
2.8	2.3	4
6.1	3.0	4
3.9	3.7	4
5.0	5.7	4
6.6	3.4	4

Tabella 4.9: Valori di abbondanza di K e Na rilevati in 20 differenti manufatti rinvenuti in quattro siti archeologici.

```
> summary(mod)
            Df  Pillai approx F num Df den Df Pr(>F)
factor(loc)  3 0.41726  1.40601      6     32 0.2427
Residuals   16
```

Le significatività ottenute con i due metodi non sono equivalenti. In entrambi i casi comunque si conclude che non si ha differenza significativa nelle composizioni chimiche dei manufatti dei quattro siti archeologici.

Per valutare come le due variabili K e Na contribuiscono alla significatività globale si possono eseguire i vari test univariati, che possono essere esaminati rapidamente con la chiamata alla funzione *summary.aov*:

```
> summary.aov(mod)
 Response K :
            Df Sum Sq Mean Sq F value Pr(>F)
factor(loc)  3  7.538   2.513  1.2075 0.3390
Residuals   16 33.292   2.081

 Response Na :
            Df  Sum Sq Mean Sq F value  Pr(>F)
factor(loc)  3 14.0895  4.6965  2.5035 0.09623 .
Residuals   16 30.0160  1.8760
```

Anche nelle analisi univariate non si evidenziano differenze significative fra le quettro località campionate.

Le matrici $\boldsymbol{H}$ e $\boldsymbol{E}$ si possono ottenere con la chiamata:

```
> summary(mod)$SS
$"factor(loc)"
```

```
        k      Na
k  7.5375  8.7965
Na 8.7965 14.0895

$Residuals
        k     Na
k  33.292  3.930
Na  3.930 30.016
```

La prima matrice è $\boldsymbol{H}$, mentre la seconda è $\boldsymbol{E}$. Come verifica, le statistiche Λ di Wilks e traccia di Pillai si possono ottenere nel modo seguente:

```
> E <- summary(mod)$SS[[2]]
> H <- summary(mod)$SS[[1]]

> det(E)/det(H + E)                  # Lambda di Wilks
[1] 0.6003312

> sum( diag(H %*% solve(H + E)) )    # traccia di Pillai
[1] 0.4172551
```

Si vede che i valori coincidono con quanto riportato nelle tabelle del test MANOVA date precedentemente.

Capitolo 5

Metodi non parametrici e potenza dei test statistici

5.1 Test di Kolmogorov-Smirnov

5.1.1 Test di Kolmogorov-Smirnov a un solo campione

Se si vuole verificare l'adattamento di una serie di dati a una distribuzione teorica è possibile utilizzare il test di Kolmogorov-Smirnov, come nel seguente esempio.

Esempio

Nel vettore *masse* sono elencate le masse (in masse solari) di 13 stelle situate nelle vicinanze del Sole. Si verifichi se la distribuzione di massa è uniforme nell'intervallo [1, 2.5] masse solari.

Il comando che consente di svolgere tale test è:

```
> masse <- c(1.03,1.65,1.02,1.23,2.15,1.30,1.34,1.53,2.04,1.35,1.19,1.01,1.33)
> ks.test(masse, "punif", min=1, max=2.5)

        One-sample Kolmogorov-Smirnov test

data:  masse
D = 0.459, p-value = 0.008363
alternative hypothesis: two.sided
```

La funzione *ks.test* accetta come argomenti la serie di dati osservati, la distribuzione teorica da utilizzare e gli argomenti da inoltrare a questa funzione. In questo caso si richiede una distribuzione uniforme (*punif*) nell'intervallo [1, 2.5]. Il test evidenzia che i dati sono distribuiti in maniera altamente non uniforme nell'intervallo richiesto.

Per rappresentare graficamente la differenza fra le due distribuzioni cumulative, sperimentale e teorica, si deve come primo passo caricare la libreria *stepfun*:

```
> library(stepfun)
> plot(ecdf(masse), do.points=FALSE, verticals=TRUE, xlim=c(1, 2.5))
> lines(c(1,2.5), c(0,1), col="red")
```

La funzione *ecdf* calcola la distribuzione cumulativa dei dati osservati, mentre l'istruzione successiva disegna il grafico della funzione cumulativa della distribuzione uniforme in [1, 2.5]. Il risultato è mostrato in Fig. 5.1. □

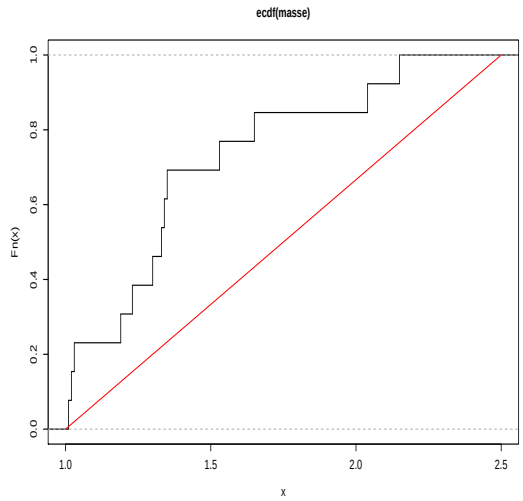

Figura 5.1: Test di Kolmogorov-Smirnov per l'uniformità.

5.1.2 Test di Kolmogorov-Smirnov per due campioni

Se si vogliono confrontare due campioni A e B per stabilire se possano provenire da una stessa distribuzione si usa il test di Kolmogorov-Smirnov nel modo seguente:

```
> ks.test(A, B)
```

Esempio

Si confrontino le distribuzioni di dei valori contenuti nei vettori A e B.

```
> A <- rnorm(30)
> B <- rnorm(20, m=0, sd=1.2)
> ks.test(A, B)

        Two-sample Kolmogorov-Smirnov test

data:  A and B
D = 0.4333, p-value = 0.01693
alternative hypothesis: two.sided
```

Si conclude che le due distribuzioni differiscono in maniera significativa. Per confrontarle graficamente si usano i comandi:

```
> plot(ecdf(B), do.points=FALSE, verticals=TRUE)
> lines(ecdf(A), do.points=FALSE, verticals=TRUE, col.vert="red", col.hor="red")
```

che producono il grafico in Fig. 5.2.

5.2 Metodi non parametrici

R dispone di numerosi test non parametrici da utilizzare quando le ipotesi che sono alla base dei test parametrici classici sono violate in modo grave.

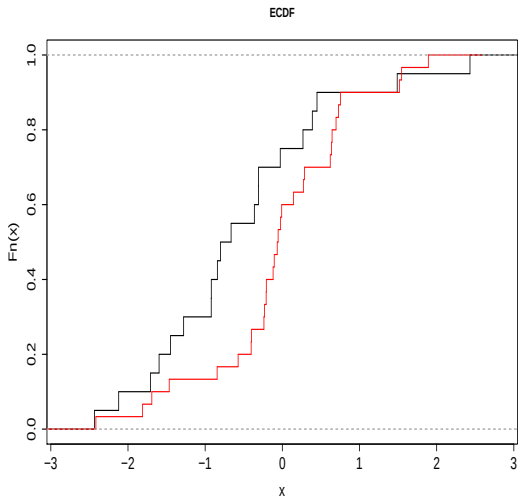

Figura 5.2: Test di Kolmogorov-Smirnov per il confronto di due serie osservate dell'esempio 5.1.2.

5.2.1 Test di Wilcoxon

Questo test è disponibile nelle due varianti di test di Wilcoxon dei ranghi con segno (equivalente al test t per dati appaiati) e test di Wilcoxon della somma dei ranghi (equivalente al test t per campioni indipendenti).

Si considerino per cominciare due campioni A e B appaiati. Se non è possibile utilizzare il test t per verificare se vi sia una differenza significativa fra le loro medie si ricorre al test di Wilcoxon dei ranghi con segno.

Esempio

Si vuole stabilire l'efficacia di un farmaco. Su un campione di 8 pazienti si misura su una scala da 1 a 5 la gravità delle lesioni prima della cura (A). Gli stessi pazienti vengono valutati a trattamento concluso (B). I dati suggeriscono che la cura ha avuto un effetto significativo?

Il test di Wilcoxon, appropriato per una situazione di questo genere si esegue con le chiamate:

```
> A <- c(2,2,3,4,4,2,3,5)
> B <- c(3,1,3,5,4,3,4,4)
> wilcox.test(A, B, paired=TRUE, correct=FALSE)

        Wilcoxon signed rank test

data:  A and B
V = 7, p-value = 0.4142
alternative hypothesis: true mu is not equal to 0
```

Si conclude che non vi è evidenza di effetto significativo del farmaco. L'opzione $paired = TRUE$ specifica che i campioni vanno trattati come dati appaiati, mentre $correct = FALSE$ specifica di non utilizzare la correzione di continuità nell'approssimazione normale alla distribuzione di Wilcoxon, usata per calcolare il valore p. Per quanto riguarda questo calcolo, per campioni di taglia inferiore a 50 e in assenza di ties il valore p viene calcolato in modo esatto, altrimenti R ricorre all'approssimazione normale. □

Nel caso di campioni indipendenti si ricorre al test di Wilcoxon della somma dei ranghi (equivalente al test U di Mann-Whitney). Usando i dati dell'esempio precedente la chiamata sarà semplicemente:

```
[...]
> wilcox.test(A, B, correct=FALSE)

 Wilcoxon rank sum test

data:  A and B
W = 26.5, p-value = 0.55
alternative hypothesis: true mu is not equal to 0
```

5.2.2 Test di Kruskal-Wallis

Nel caso di più di due serie di dati indipendenti, se non è possibile ricorrere ad una ANOVA, si utilizza il test di Kruskal-Wallis, che nel caso di due sole serie di dati fornisce un risultato analogo a quello del test di Wilcoxon della somma dei ranghi. In questo caso è necessario unire tutti i dati in un unico vettore e utilizzare un vettore complementare in cui tenere traccia del gruppo di provenienza:

```
> A <- c(2,2,3,4,4,2,3,5)
> B <- c(3,1,3,5,4,3,4,4)
> dati <- c(A, B)
> g <- rep(1:2, each=8)
```

o in alternativa riunirli in una lista:

```
> dati2 <- list(g1=A, g2=B)
```

Quest'ultima chiamata crea un oggetto di tipo lista (l'equivalente di una struttura C) che contiene due variabili: la prima, chiamata "g1", contenente il vettore A e la seconda, di nome "g2", contenente il vettore B.

Si procede quindi con il test:

```
> kruskal.test(dati, g)       # usando i vettori
> kruskal.test(dati2)         # usando la lista

       Kruskal-Wallis rank sum test

data:  dati2
Kruskal-Wallis chi-squared = 0.3573, df = 1, p-value = 0.55
```

Come si vede il valore p è identico a quello ottenuto nel caso di test di Wilcoxon della somma dei ranghi.

5.2.3 Test di Friedman

Questo test è l'analogo non parametrico dell'analisi della varianza a due fattori a blocchi randomizzati.

Esempio

A 6 psicologi vengono richieste le valutazioni (con un punteggio da 0 a 5 in ordine crescente di validità) di 4 terapie. Si stabilisca se vi sono delle differenze di valutazione fra le 4 diverse terapie.

Il modo più rapido di procedere è inserire i dati relativi alle valutazioni dei 6 medici e unirli in una matrice:

```
> c1 <- c(4, 3, 0, 2)
> c2 <- c(4, 2, 2, 2)
> c3 <- c(3, 2, 2, 1)
> c4 <- c(5, 1, 2, 2)
> c5 <- c(3, 4, 1, 2)
```

```
> c6 <- c(5, 4, 3, 3)
> mat <- rbind(c1, c2, c3, c4, c5, c6)
```

La funzione *rbind* unisce i vettori in una matrice disponendoli per riga. Il test si esegue semplicemente con la chiamata:

```
> friedman.test(mat)

        Friedman rank sum test

data:  mat
Friedman chi-squared = 11.3774, df = 3, p-value = 0.00985
```

Si evidenzia una differenza altamente significativa fra i 4 tipi terapia. □

In alternativa è sempre possibile trasformare i dati all'interno di ciascun blocco in ranghi ed eseguire una ANOVA a due vie sui dati trasformati. Il processo risulta più lungo e macchinoso:

```
> dati <- c(c1, c2, c3, c4, c5, c6)
> gruppo <- factor(rep(1:4, 6))
> blocco <- factor(rep(1:6, each=4))
> rg <- tapply(dati, blocco, rank)    # calcolo i ranghi nei blocchi
> ranghi <- stack(rg)[[1]]   # concateno la lista ed estraggo la prima colonna
> anova(aov(ranghi ~ gruppo + blocco))

Analysis of Variance Table

Response: ranghi
          Df    Sum Sq   Mean Sq   F value   Pr(>F)
gruppo     3   16.7500    5.5833    8.5897 0.001474 **
blocco     5 5.017e-30 1.003e-30 1.544e-30 1.000000
Residuals 15    9.7500    0.6500
```

Si noti che il valore di significatività è diverso dalla tecnica precedente.

5.2.4 Correlazione non parametrica

Per valutare l'associazione fra due serie di dati, in R sono disponibili due misure non parametriche: ρ di Spearman e τ di Kendall. Entrambe sono accessibili dalla funzione *cor.test* (vedi Sezione 2.3) specificando l'opzione *method* appropriata.

Esempio

Si studia la correlazione non parametrica tra i valori degli indici di soddisfazione IS degli abitanti di sette città (espressi in una scala da 1 a 10) e l'area media (in km^2) di zone verdi disponibili nel centro urbano.

Si inseriscono i dati:

```
> IS <- c(1, 2, 3, 4, 5, 6, 8, 10)
> ZV <- c(0.7, 0.4, 1.0, 1.0, 1.1, 1.2, 1.2, 1.2)
```

Per eseguire il test di Spearman si usa la chiamata:

```
> cor.test(IS, ZV, method="spearman")

        Spearman's rank correlation rho
```

```
data:  IS and ZV S = 4, p-value = 0.002171 alternative
hypothesis: true rho is not equal to 0 sample estimates:
    rho
0.94523
```

Per il test di Kendall la sintassi è:

```
> cor.test(IS, ZV, method="kendall")

        Kendall's rank correlation tau

data:  IS and ZV
z.tau = 2.9399, p-value = 0.003283
alternative hypothesis: true tau is not equal to 0
sample estimates:
      tau
0.8486684
```

La conclusione in entrambi i casi è che esiste evidenza di associazione altamente significativa fra i due indici considerati.

5.3 Potenza dei test statistici

Data la possibilità di accedere facilmente alle distribuzioni non centrali (vedi sezione 1.3) in R è particolarmente semplice valutare la potenza di un test disponendo di informazioni sufficienti a valutare il parametro di non centralità.

5.3.1 Potenza di un test t

Si abbia un campione di taglia n su cui si misura un certo parametro. Si supponga di voler discriminare fra le due ipotesi:

- H_0: il campione in esame proviene da una popolazione di media del parametro μ_0.
- H_1: il campione in esame proviene da una popolazione di media del parametro μ_1.

Si supponga anche di conoscere o di poter valutare il parametro ϕ, definito come:

$$\phi = \frac{\mu_1 - \mu_0}{\sigma}$$

dove σ^2 è la varianza della popolazione (uguale nelle due ipotesi). A partire da questo valore si calcola il parametro di non centralità:

$$\lambda = \sqrt{n}\,\phi \tag{5.1}$$

Si vuole calcolare con quale potenza è possibile discriminare fra queste due ipotesi data la taglia del campione.

Sia t_c il valore critico a due code della distribuzione t a $\nu = n - 1$ gradi di libertà. La potenza del test (indicata come $1 - \beta$) è data da:

$$1 - \beta = \int_{t_c}^{\infty} t(x, \nu, \lambda)\, dx \tag{5.2}$$

dove $t(x, \nu, \lambda)$ è la distribuzione di t non centrale a ν gradi di libertà e con parametro di non centralità λ. Usando R è immediato calcolare la potenza di un test t, come nell'esempio seguente.

Esempio

Si abbia un campione di taglia $n = 10$. Posto $\alpha = 0.05$, con quale potenza è possibile discriminare fra due ipotesi per cui vale $\phi = 1$?

Si inizia con il calcolo del valore critico t_c per il test bilaterale a 9 gradi di libertà:

```
> tc <- qt(0.975, 9)
```

La potenza del test si calcola quindi come:

```
> 1 - pt(tc, ncp=sqrt(10)*1, 9)

[1] 0.8030962
```

È anche possibile risolvere il problema inverso, utile in fase di pianificazione di un esperimento, ossia trovare la taglia minima del campione necessaria per raggiungere una data potenza. La procedura richiede di definire una funzione:

```
> f <- function(n, nc, b)
+   { 1 - pt(qt(0.975, n-1), ncp=sqrt(n)*nc, n-1) - b }
```

Se b è la potenza da raggiungere la funzione così costruita ha uno zero per $1 - \beta = b$. Il problema si riduce quindi nel trovare il valore di n per cui f ha uno zero. Ad esempio se $\phi = 0.5$, quale è la taglia minima del campione che consente di avere una potenza pari a 0.85? Per risolvere il problema si fa uso della funzione *uniroot* che permette di trovare lo zero di una funzione dato l'intervallo della variabile indipendente in cui tale zero deve essere cercato:

```
> uniroot(f, c(2,50), nc=0.5, b=0.85)
```

Il secondo argomento passato a *uniroot* è il range di variabilità di n, mentre i seguenti sono gli argomenti da inoltrare alla funzione f. Il risultato è:

```
$root
[1] 37.88252
[...]
```

da cui si conclude che è necessario un campione di almeno 38 elementi per raggiungere la potenza richiesta. □

Quanto detto è facilmente generalizzabile al caso di test t per dati appaiati e per test t a campioni indipendenti. Le uniche modifiche al procedimento, oltre al corretto calcolo dei gradi di libertà, sono nella valutazione del parametro di non centralità. Nel primo caso il valore di λ dato in Eq. (5.1) va diviso per il valore campionario del coefficiente di correlazione fra le due serie di dati in analisi, mentre nel secondo questo risulta definito come:

$$\lambda = \sqrt{\frac{n_A n_B}{n_A + n_B}}\,\phi \tag{5.3}$$

che per gruppi di eguale taglia n' si semplifica in $\lambda = \sqrt{\frac{n'}{2}}\,\phi$.

5.3.2 Potenza di un test χ^2

Si osservi la distribuzione di frequenze con cui avviene un fenomeno. Si supponga di dover discriminare fra due ipotesi:

- H_0: il fenomeno è descritto da una distribuzione teorica per cui le frequenze relative sono $p_{01}, \ldots, p_{0r}$.

- H_1: il fenomeno è descritto da una distribuzione teorica per cui le frequenze relative sono $p_{11}, \ldots, p_{1r}$.

Ci si chiede con che potenza è possibile eseguire la discriminazione se si esamina un campione di taglia n.

Posto:

$$w = \sqrt{\sum_{i=1}^{r} \frac{(p_{1i} - p_{0i})^2}{p_{0i}}}$$

si definisce il parametro di non centralità λ:

$$\lambda = n\, w^2 \tag{5.4}$$

dove n è la taglia del campione in esame. Per il calcolo della potenza del test si procede come nel caso precedente, integrando la distribuzione di χ^2 non centrale.

Esempio

Si osservi un dato fenomeno e si voglia discriminare fra le due ipotesi:

- H_0: il fenomeno è descritto da una distribuzione teorica per cui le frequenze relative sono (0.3, 0.7).
- H_1: il fenomeno è descritto da una distribuzione teorica per cui le frequenze relative sono (0.4, 0.6).

Che potenza si raggiunge usando un campione di taglia $n = 40$?

Per prima cosa si calcola λ:

```
> p0 <- c(0.3, 0.7)
> p1 <- c(0.4, 0.6)
> w <- sqrt(sum( (p0-p1)^2/p0 ))
> lambda <- w^2 * 40
> lambda

[1]  1.904762
```

Quindi si calcola la potenza per un test a un grado di libertà come il caso in esame:

```
> cc <- qchisq(0.95, 1)
> 1 - pchisq(cc, ncp=lambda, 1)

[1] 0.2814324
```

da cui si conclude che la potenza del test non raggiunge il 30%.

Viceversa se interessa sapere quale taglia del campione permetterebbe di raggiungere una potenza del 70%, si deve definire la funzione:

```
> f <- function(n, nc, b, gdl)
+    { 1 - pchisq(qchisq(0.95, gdl), ncp=n*nc^2, gdl) - b }
```

e quindi chiamare la funzione *uniroot*:

```
> uniroot(f, c(2,250), nc=w, b=0.7, gdl=1)

$root
[1] 129.6121
```

quindi la taglia del campione dovrebbe essere di 130.

5.3.3 Potenza dell'ANOVA

Siano dati r gruppi di dati, ognuno di taglia n'. L'ipotesi H_0 prevede che tutti i gruppi provengano da popolazioni di uguale media μ, mentre l'ipotesi H_1 specifica le differenti medie $\mu_1, \ldots, \mu_r$ per i vari gruppi. Posto:

$$f = \sqrt{\frac{\sum_{i=1}^{r}(\mu_i - \mu)^2}{r\ \sigma^2}}$$

dove σ^2 è la varianza comune per le r popolazioni, si definisce il parametro di non centralità λ:

$$\lambda = n\ f^2 \tag{5.5}$$

dove $n = rn'$ è il numero totale di dati a disposizione dello sperimentatore. Per il calcolo della potenza del test si procede come nei casi precedenti, integrando la distribuzione di F non centrale.

Esempio

Si supponga di voler discriminare fra due ipotesi per cui è noto $f = 0.2$ in un disegno sperimentale composto da 4 gruppi. Se si hanno a disposizione 20 individui per gruppo quale potenza si può raggiungere?

```
> n1 <- 20
> r <- 4
> n <- n1*r       # dati complessivi
> df1 <- r-1      # gdl numeratore
> df2 <- n-r      # gdl denominatore
> fc <- qf(0.95, df1, df2)
> 1 - pf(fc, ncp=n*0.2^2, df1, df2)

[1] 0.2780544
```

La potenza che si raggiunge con un esperimento di tal genere è bassa, pertanto è molto elevata la probabilità di incorrere in un errore di tipo II.

Se interessa il calcolo del minimo n' tale per cui si raggiunge una data potenza il procedimento passa per la definizione della funzione di cui trovare lo zero:

```
> f <- function(n1, nc, r, b)
+ { df1 <- r - 1;
+   df2 <- (n1*r) - r;
+   1 - pf(qf(0.95, df1, df2), ncp=(n1*r)*nc^2, df1, df2) - b }
```

Ad esempio, la taglia n' di ogni gruppo necessaria a raggiungere una potenza $1 - \beta = 0.8$ risulta:

```
> uniroot(f, c(20,200), r=4, nc=0.2, b=0.8)

$root
[1] 69.12567
[...]
```

quindi sarebbero necessari almeno 70 individui per gruppo, per un totale di ben 280 individui.

Capitolo 6

Modelli lineari generalizzati (GLM)

6.1 Regressione logistica

In questa sezione viene presentata una panoramica veloce e necessariamente incompleta delle problematiche connesse alla regressione logistica. Per una trattazione approfondita dell'argomento si rimanda a [35, 23].

Il modello di regressione logistica semplice è impiegato quando la variabile dipendente Y è dicotoma o binaria e si ha un unico predittore X. Senza perdita di generalità è possibile codificare i valori assunti da Y con 0 e 1.

Il modello specifico più utilizzato per legare la media dei valori della variabile dipendente dato il valore del predittore, indicata con $E(Y|x)$, è il modello logistico:

$$\pi(x) \equiv E(Y|x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}} \tag{6.1}$$

Introducendo la trasformazione *logit*:

$$g(x) = \ln\left(\frac{\pi(x)}{1-\pi(x)}\right) = \beta_0 + \beta_1 x \tag{6.2}$$

si ottiene un modello lineare che lega $g(x)$ a x.

R mette a disposizione la funzione *glm* che permette di fittare un modello lineare generalizzato una volta specificata la funzione di link desiderata (in questo caso un link logit). Le tecniche utilizzate per la stima dei parametri del modello sono illustrate nel corso del seguente esempio.

Esempio

Si fa uso di un set di dati da [35], disponibile in rete all'indirizzo:
http://www.umass.edu/statdata/statdata/data/chdage.dat

Il file presenta i dati 100 pazienti in cui l'eta (variabile AGE) viene messa in relazione con la presenza o l'assenza di significativi disturbi coronarici (CHD). La variabile CHD è codificata nel modo seguente:

- CHD = 1: disturbo presente.
- CHD = 0: disturbo assente.

Dopo aver acquisito il file di dati lo si importa in R:

```
> chd <- read.table("chdage.dat", head=TRUE)
> chd

     ID AGE CHD
1     1  20   0
2     2  23   0
3     3  24   0
4     4  25   0
[..]
100 100  69   1
```

Il comando *read.table* legge la tabella di dati passata come primo argomento. L'opzione $head = TRUE$ specifica che nella prima riga sono contenuti i nomi dei campi.

Come primo passo si esamina il plot dei dati (i punti sono individuati da cerchi in Fig. 6.1):

```
> attach(chd)
> plot(AGE, CHD)
```

La funzione *attach* rende disponibili le variabili contenute nella tabella *chd* (si può quindi usare la variabile AGE invece di dover scrivere ogni volta *chd*$AGE). Per meglio comprendere la natura della relazione è opportuno suddividere i pazienti in classi d'età e calcolare la media della variabile dipendente in ciascuna classe. Inseriti nel vettore x i limiti delle classi d'età che si vogliono creare, se ne calcolano i punti di mezzo per uso futuro:

```
> x <- c(19.999,29,34,39,44,49,54,59,70)
> mid <- c((x[2:9]+x[1:8])/2)
```

Per valutare il valor medio di CHD nelle classi si costruisce un vettore $GRAGE$, che classifica i dati per classi d'età, usando la funzione *cut*:

```
> GRAGE <- cut(AGE, breaks=x)
> y <- tapply(CHD, GRAGE, mean)
> y

  (20,29]   (29,34]   (34,39]   (39,44]   (44,49]   (49,54]   (54,59]   (59,70]
0.1000000 0.1333333 0.2500000 0.3333333 0.4615385 0.6250000 0.7647059 0.8000000
```

Si sovrappongono i valori in y al grafico precedente:

```
> points(mid, y, col="red", pch=3)    # percentuali nelle classi
```

In Fig. 6.1 questi punti sono identificati dalle crocette rosse. Dall'osservazione del grafico si comincia a capire la natura della relazione.

Se le n osservazioni della variabile dipendente y sono codificate (0,1), come nel caso in questione, si ha che $\pi(x) = E(Y|x)$ è la probabilità condizionata che Y sia uguale a 1 dato x, cioè $P(Y = 1|x)$. Analogamente $1 - \pi(x)$ è la probabilità che Y sia uguale a 0 dato x, cioè $P(Y = 0|x)$. Ne segue che la funzione di likelihood è:

$$l(\boldsymbol{\beta}) = \prod_{i=1}^{n} \pi(x_i)^{y_i} \left[1 - \pi(x_i)\right]^{1-y_i} \quad , \quad y_i = 0, 1 \tag{6.3}$$

da cui si deriva l'espressione della log-likelihood:

$$L(\boldsymbol{\beta}) = \ln l(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left\{ y_i \ln \pi(x_i) + (1 - y_i) \ln \left[1 - \pi(x_i)\right] \right\}. \tag{6.4}$$

La stima dei parametri $\hat{\boldsymbol{\beta}}$ si ottiene massimizzando $L(\boldsymbol{\beta})$ rispetto a $\boldsymbol{\beta}$. Derivando l'Eq. (6.17) rispetto a $\boldsymbol{\beta}$ e uguagliando a zero il risultato si ottengono le equazioni di likelihood:

$$\sum [y_i \pi(x_i)] = 0 \tag{6.5}$$

$$\sum x_i [y_i \pi(x_i)] = 0 \tag{6.6}$$

Tali equazioni, non lineari nei parametri, vengono risolte con tecniche iterative.

In R la soluzione di queste equazioni si ottiene mediante la funzione:

```
> mod <- glm(CHD ~ AGE, family=binomial(link=logit))
> summary(mod)

[...]
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.30945    1.13263  -4.688 2.76e-06 ***
AGE          0.11092    0.02404   4.614 3.95e-06 ***
[...]
    Null deviance: 136.66  on 99  degrees of freedom
Residual deviance: 107.35  on 98  degrees of freedom
AIC: 111.35
```

Il modello stimato tramite maximum likelihood è quindi:

$$\hat{g}(x) = -5.30945 + 0.11092 \times AGE \tag{6.7}$$

La likelihood si ottiene con la chiamata:

```
> logLik(mod)

'log Lik.' -53.67655 (df=2)
```

I valori di $\hat{g}$ si ottengono come:

```
> mod$linear.predictors
```

mentre per $\hat{\pi}(x)$ si usa la chiamata:

```
> mod$fitted.values
```

Il grafico delle predizioni del modello, mostrato con la linea continua in Fig. 6.1 è dato da:

```
> lines(AGE, mod$fitted)   # fit del modello
```

Per interpretare le ultime linee date in output dal fit del modello è necessario introdurre il concetto di devianza:

$$D(\beta) = -2 \ln l(\boldsymbol{\beta}) \tag{6.8}$$

Si noti che la funzione D, calcolata per un modello di regressione lineare semplice, coincide con la devianza d'errore d_E^2. La statistica D è di particolare importanza nello stabilire l'adeguatezza di un modello; per stabilire l'adeguatezza del modello che include una data variabile si calcola l'espressione:

$$G = D(\text{modello con la variabile}) - D(\text{modello senza la variabile}) \tag{6.9}$$

Se la variabile in questione è continua G è distibuito $\sim \chi^2(1)$, mentre se la variabile è categoriale G risulterà distribuito come $\sim \chi^2(p-1)$ dove p sono i livelli assunti dalla variabile in esame. In R la procedura per condurre tale indagine è molto semplice:

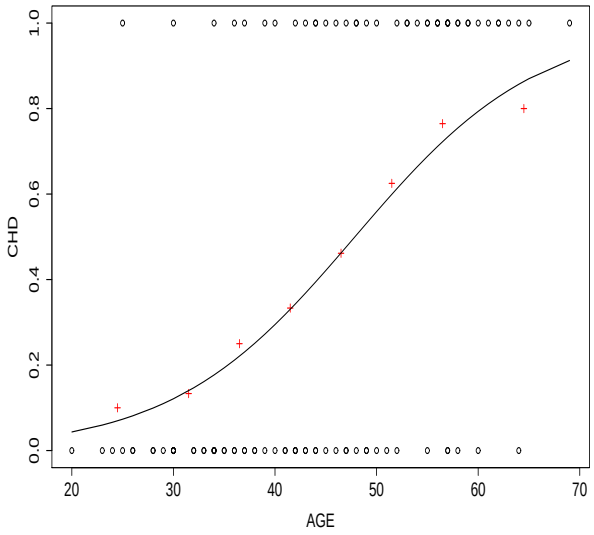

Figura 6.1: Modello logistico. I dati originali sono rappresentati da cerchi. Le crocette si riferiscono ai dati raggruppati in classi d'età e la linea continua è il fit logistico.

```
> anova(mod, test="Chisq")

Analysis of Deviance Table
[...]
     Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                    99     136.66
AGE   1    29.31        98     107.35 6.168e-08
```

Quindi si ha evidenza molto altamente significativa che la variabile AGE sia un buon predittore per CHD. I due valori di devianza (136.66 e 107.35) che appaiono nella tabella ANOVA sono gli stessi che si trovano riassunti nell'output del fit del modello.

Un seconda tecnica per valutare la significatività di un parametro è quella del test di Wald. Si calcola il valore campionario della variabile W:

$$w = \frac{\hat{\boldsymbol{\beta}}}{\hat{SE}(\hat{\boldsymbol{\beta}})}$$

la quale è distribuita $\sim N(0,1)$. Si testa quindi la significatività del valore campionario con un test z. R riporta questo test nelle linee in cui fornisce i parametri e il loro errore standard. I risultati sono nelle colonne "z value" e "Pr($>|$z$|$)". A causa della sua limitata potenza al test di Wald è preferibile l'indagine sul rapporto di likelihood di cui sopra.

6.1.1 Interpretazione dei coefficienti

Il motivo fondamentale per cui la tecnica di regressione logistica è largamente diffusa è che i coefficienti del modello hanno una naturale interpretazione in termini di odds ratio (nel seguito OR).

Per semplificare il discorso si analizza la situazione in cui si ha un predittore x dicotomo a livelli 0 e 1. In questo caso l'odd che sia $y = 1$ fra gli individui con $x = 0$ è $\pi(0)/(1-\pi(0))$. Analogamente per i soggetti con $x = 1$ l'odd che sia $y = 1$ è $\pi(1)/(1-\pi(1))$. L'OR è definito come il rapporto degli odds per $x = 1$ e $x = 0$:

$$OR = \frac{\pi(1)/(1-\pi(1))}{\pi(0)/(1-\pi(0))}.$$

Dato che:

$$\pi(1) = \frac{e^{\beta_0+\beta_1}}{1+e^{\beta_0+\beta_1}} \qquad , \qquad \pi(0) = \frac{e^{\beta_0}}{1+e^{\beta_0}}$$

con alcuni semplici passaggi algebrici si ottiene:

$$OR = e^{\beta_1},$$

che ha per stima campionaria:

$$\hat{OR} = e^{\hat{\beta_1}}.$$

Un intervallo di confidenza bilaterale di livello $1-\alpha$ per OR è dato da:

$$e^{\hat{\beta_1} \pm z_{1-\alpha/2}\ \hat{SE}(\hat{\beta_1})}$$

La generalizzazione al caso in cui la variabile x sia discreta a k livelli è immediata. In questo caso infatti il fit del modello fa uso di $k-1$ variabili dummy (o design variables) e gli OR vengono calcolati rispetto al livello di riferimento della variabile x. Nel seguito si vedrà un esempio di questo procedimento.

Se la variabile x è continua allora il coefficiente $\hat{\beta_1}$ è una stima del logaritmo di OR per l'aumento di una unità in x. Spesso è più appropriato o sensato esprimere l'OR per un aumento in x di c unità; in tal caso la stima di OR è semplicemente:

$$\hat{OR} = e^{c\ \hat{\beta}_1}.$$

e il suo intervallo di confidenza bilaterale di livello $1-\alpha$ è:

$$e^{c\ \hat{\beta_1} \pm z_{1-\alpha/2}\ c\ \hat{SE}(\hat{\beta_1})}$$

Esempio

Nel caso dell'Esempio trattato precedentemente (CHD e AGE) si aveva che il coefficiente della variabile continua AGE valeva 0.111. Quindi l'OR per un incremento di 10 anni d'età è $e^{10\times 0.111} = 3.03$. Cioè per ogni incremento di 10 anni d'età il rischio di disturbo coronarico aumenta di 3 volte circa. Ovviamente il modello sottointende che il logit sia lineare nella variabile età, ossia che l'OR fra persone di 20 contro 30 anni d'età sia lo stesso che fra individui di 40 contro 50 anni. Se una tale modellizzazione non è appropriata un procedimento migliore fa uso, in luogo della variabile continua, di una variabile discreta a più livelli che identifichi dei gruppi d'età fra cui ha senso distinguere.

Per concludere, l'intervallo di confidenza al 95% per OR è:

$$\exp(10 \times 0.111 \pm 1.96 \times 10 \times 0.024) = (1.90, 4.86).$$

6.1.2 Intervallo di confidenza della regressione logistica

L'intervallo di confidenza bilaterale di livello $1-\alpha$ per i parametri $\hat{\boldsymbol{\beta}}$ è dato da:

$$\hat{\boldsymbol{\beta}} \pm z_{1-\alpha/2}\ \hat{SE}(\hat{\boldsymbol{\beta}}). \tag{6.10}$$

Essendo la stima del logit

$$\hat{g}(x) = \hat{\beta}_0 + \hat{\beta}_1 x \tag{6.11}$$

si ha:

$$\hat{var}[\hat{g}(x)] = \hat{var}[\hat{\beta}_0] + x^2\ \hat{var}[\hat{\beta}_1] + 2\ x\ \hat{cov}[\hat{\beta}_0, \hat{\beta}_1] \tag{6.12}$$

L'intervallo di confidenza per il logit a livello $1-\alpha$ è quindi:

$$\hat{g}(x) \pm z_{1-\alpha/2}\ \hat{SE}[\hat{g}(x)] \tag{6.13}$$

dove $\hat{SE}[\hat{g}(x)]$ è la radice dell'espressione in Eq. (6.12).

Per costruire in R l'intervallo di confidenza del logit si può partire dal calcolo della matrice di covarianza dei parametri $\hat{\boldsymbol{\beta}}$:

```
> V <- vcov(mod)
> V

             (Intercept)           AGE
(Intercept)  1.28284816 -0.0266308043
AGE         -0.02663080  0.0005779444
```

Per il calcolo dell'intervallo di confidenza in corrispondenza di un valore di x (ad esempio $x = 50$) si usa la chiamata:

```
> x <- 50
> sqrt(V[1,1] + x^2 * V[2,2] + 2*x*V[1,2])

[1] 0.2542218
```

e si moltiplica questo valore per il valore critico di z al livello desiderato.

Si noti che è possibile ottenere il valore di $\hat{SE}[\hat{g}(x)]$ in maniera più semplice, facendo uso della funzione *predict*:

```
> predict(mod, data.frame(AGE=50), se=TRUE)

$fit
[1] 0.2366039

$se.fit
[1] 0.2542218
```

Per rappresentare graficamente l'intervallo di confidenza (al 95%) della regressione si può adottare la procedura seguente:

```
> grid <-(20:69)     # x in cui valutare la regressione
> se <- predict(mod, data.frame(AGE=grid), se=TRUE)
> gl <- binomial(link=logit)     # funzione di link utilizzata
> plot(mid, y, col="red", pch=3, ylim=c(0,1), ylab="CHD", xlab="AGE")
> lines(grid, gl$linkinv(se$fit))
> lines(grid, gl$linkinv(se$fit-1.96*se$se), col="red", lty=2)
> lines(grid, gl$linkinv(se$fit+1.96*se$se), col="red", lty=2)
```

il cui risultato è mostrato in Fig. 6.2. La funzione *gl$linkinv* permette di ottenere il valore di $\pi(x)$ dato $g(x)$.

6.1.3 Goodness-of-fit

Varie tecniche sono state sviluppate e confrontate per stabilire la bontà del fit di una regressione logistica. Il problema che ci si trova a fronteggiare se si vuole ricorrere a un simile test è che tali tecniche soffrono di una limitata potenza (tipicamente non superiore al 50%) per campioni di taglia $n < 400$ (si veda [34]).

Se la variabile indipendente è categoriale si può paragonare il valore di D per il modello fittato con il valore critico per una distribuzione $\chi^2(n-p)$ (essendo p il numero di parametri del modello). Se D è maggiore del valore critico si rifiuta l'ipotesi nulla che il modello sia un buon fit.

Se la variabile indipendente è continua (come nell'Esempio in esame) la procedura precedente perde di validità e i valori P che si ottengono non sono corretti. L'alternativa che R fornisce, tramite installazione di due librerie supplementari (*Design* e *Hmisc*), è quella dovuta a Osius e Rojek (si veda [35]). L'installazione delle librerie suddette mette a disposizione le funzioni *lrm* e *residuals* per calcolare tale statistica, come nell'esempio seguente:

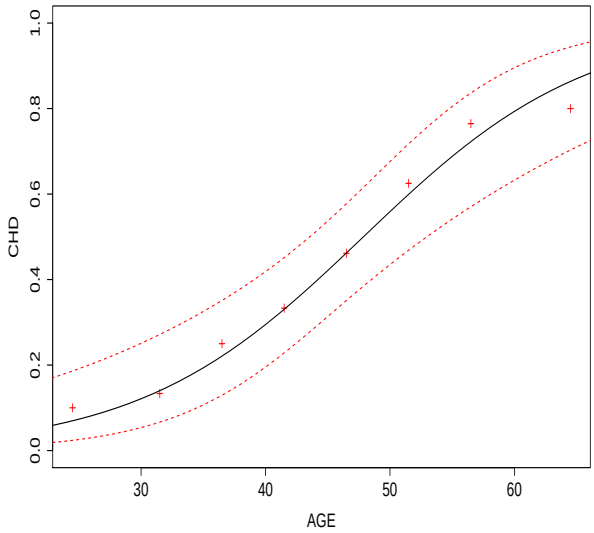

Figura 6.2: Intervallo di confidenza al 95% per la regressione logistica.

```
> library(Design)
> mod2 <- lrm(CHD ~ AGE, x=TRUE, y=TRUE)
> mod2

[...]
Frequencies of Responses
 0  1
57 43

       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
       100      7e-06      29.31          1          0        0.8        0.6
     Gamma      Tau-a         R2      Brier
     0.612      0.297      0.341      0.178

          Coef    S.E.    Wald Z P
Intercept -5.3095 1.13365 -4.68  0
AGE        0.1109 0.02406  4.61  0
```

Come si vede la funzione *lrm* è una procedura alternativa per fittare una regressione logistica. I risultati del modello coincidono con quelli ottenuti in precedenza, e in aggiunta vengono calcolate alcune statistiche che servono per stabilire la bontà del modello. Il test di goodness-of-fit di Osius e Rojek si esegue con la chiamata:

```
> residuals(mod2, "gof")

Sum of squared errors     Expected value|H0                    SD
           17.8301075           17.9263123            0.2204658
                    Z                    P
           -0.4363705            0.6625679
```

dal valore di Z (e del valore P associato) si conclude che l'ipotesi H_0 che il modello sia un buon fit non può essere rifiutata.

6.1.4 Analisi dei residui

Per stabilire la bontà di un modello è sempre necessaria la fase di analisi dei residui. Nel caso della regressione logistica questa procedura necessita dell'introduzione del concetto di *covariate pattern*. Si usa il termine *covariate pattern* per indicare un particolare set di valori delle variabili indipendenti. Ad esempio se in un modello si ha solo una variabile indipendente categoriale a due livelli si avranno solo due possibili *covariate pattern*, se si hanno due variabili a tre livelli ciascuna si avranno $3 \times 3 = 9$ possibili *covariate pattern*, mentre se la variabile indipendente è continua si avrà un numero di *covariate pattern* $\simeq n$.

Si indichi con M il numero di *covariate pattern* del modello. Sia n_j con $j = 1, \ldots, M$ il numero di soggetti che cadono nel *covariate pattern* j-esimo, e Y_j il numero di soggetti con risposta $y = 1$ in ognuno degli M gruppi di taglia n_j. Si definiscono i residui di Pearson:

$$r_j = \frac{Y_j - n_j\hat{\pi}_j}{\sqrt{n_j\hat{\pi}_j(1-\hat{\pi}_j)}} \tag{6.14}$$

Si hanno quindi M residui totali. I residui di Pearson standardizzati sono definiti come:

$$z_j = \frac{r_j}{\sqrt{1-h_j}} \tag{6.15}$$

dove $h_j \equiv H_{jj}$ sono i valori di leva e H è la hat-matrix del modello. Definita la matrice diagonale V $M \times M$ che ha per elementi diagonali $v_j = n_j\hat{\pi}_j(1-\hat{\pi}_j)$ e X la matrice del modello (come in sezione 3.2) la hat-matrix si calcola come:

$$H = V^{1/2}X(X^TVX)^{-1}X^TV^{1/2} \tag{6.16}$$

e come nel caso della regressione lineare si ha:

$$\sum h_j = p$$

dove p è il numero di parametri nel modello. Come diagnostica i residui standardizzati possono essere plottati contro i valori della variabile indipendente, come in Fig. 6.3. In aggiunta, se i valori di n_j non sono troppo piccoli ci si attende che $z_j \sim N(0,1)$ e si può verificare questa ipotesi con un test di normalità.

La valutazione delle quantità sopra elencate tramite R non è immediata e risulta comodo creare una funzione alquanto complessa, di nome *covariate*, la quale si occupa di calcolare tutte le diagnostiche necessarie. La definizione della funzione è la seguente:

```
covariate <- function(mod) {
 X.full <- as.data.frame(model.matrix(mod))
 col <- ncol(X.full)

 AG <- pattern.discover(X.full)
 n <- AG[[2]]$n
 lev <- AG[[1]]
 nY <- tapply(mod$y, lev, sum)
 np <- tapply(mod$fitted, lev, sum)

 pi <- np/n
 rj <- (nY - np)/sqrt(np*(1-pi))

 X <- as.matrix(AG[[2]][,-1])
 V <- diag(np*(1-pi))
 xvx <- solve(t(X) %*% V %*% X)
 sV <- sqrt(V)
```

```
 H <- sV %*% X %*% xvx %*% t(X) * sV
 h <- diag(H)

 zj <- rj/sqrt(1-h)
 X1 <- as.matrix(X[,-1])
 Xnames <- colnames(X)
 colnames(X1) <- Xnames[-1]
 cov <- data.frame(n=n, Y=nY, X1, fitted=pi, h=h, rj=rj, zj=zj)

 return(cov)
}

pattern.discover <- function (x)  {
    x <- data.frame(x)
    col <- ncol(x)
    w <- unlist(x[1])
    for (j in 2:col) {
        w <- paste(w, x[, j], sep = "")
    }
    w <- factor(w, unique(w))
    levels(w) <- 1:length(unique(w))
    select <- match(1:length(w), w)[1:nlevels(w)]
    x <- x[select, ]

    return(list(lev=w, data.frame(n=as.vector(table(w)), X=x )))
}
```

Tale funzione accetta in input un modello fittato tramite la funzione *glm* e produce in output un data frame contenente diverse variabili, che – per ogni covariate pattern – rappresentano: il numero di individui presenti nel covariate pattern, il numero di soggetti con $Y = 1$, i valori dei predittori (una colonna per ogni predittore), i valori di leva, i residui di Pearson e i residui standardizzati. Nel caso in esame si ha:

```
> cp <- covariate(mod)
> cp
   n Y X.AGE     fitted          h          rj          zj
1  1 0    20 0.04347876 0.01869988 -0.21320199 -0.21522381
2  1 0    23 0.05962145 0.02042260 -0.25179663 -0.25440786
3  1 0    24 0.06615278 0.02088718 -0.26615592 -0.26897986
[...]
43 1 1    69 0.91246455 0.02873601  0.30973050  0.31427898
```

Si nota che i 100 soggetti si raggruppano in 43 covariate pattern. L'unico predittore presente nel modello viene indicato con la variabile X nell'output della funzione *covariate*. Per la diagnostica del modello si può generere il grafico dei residui standardizzati e studiarne la normalità:

```
> plot(cp$X, cp$zj, xlab="AGE", ylab="zj")
> shapiro.test(cp$zj)    # test di normalita' dei residui standardizzati

        Shapiro-Wilk normality test

data:  cp$zj
W = 0.9653, p-value = 0.2165
```

Questo ultimo risultato va considerato solo come esempio della metodologia in quanto i valori di n_j sono troppo piccoli perché il test di normalità abbia giustificazione teorica, come si può desumere dalla seguente chiamata:

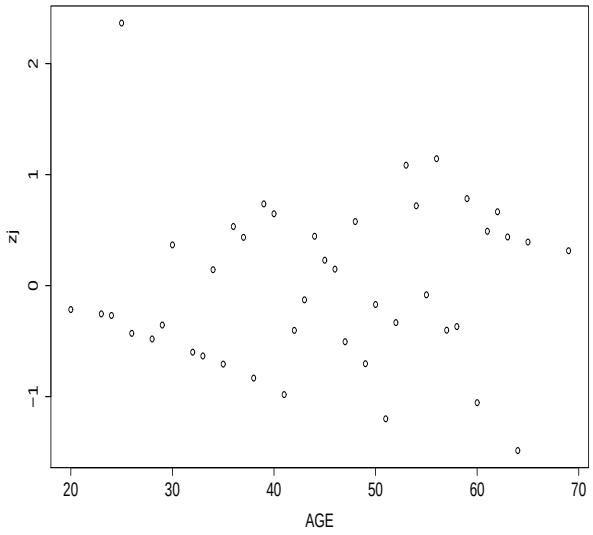

Figura 6.3: Analisi dei residui per regressione logistica.

```
> summary(cp$n)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   2.000   2.326   3.000   6.000
```

Solitamente si assume che tutti gli n_j siano maggiori di 5 perché il test di normalità sia applicabile.

6.2 Regressione logistica multipla

Se ci sono r predittori (con $r > 1$) si parla di regressione logistica multipla. Sia $\boldsymbol{X}$ la matrice dei predittori:

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & \dots & x_{1r} \\ 1 & x_{21} & \dots & x_{2r} \\ \dots & \dots & \dots & \dots \\ 1 & x_{n1} & \dots & x_{nr} \end{pmatrix}$$

dove n è il numero di osservazioni. Il vettore dei parametri da stimare è $\boldsymbol{\beta} = (\beta_0, \dots, \beta_r)^T$ Detta $\boldsymbol{X}_i$ la riga i-esiam della matrice $\boldsymbol{X}$, il modello logistico si scrive:

$$\pi(\boldsymbol{X}_i) = \frac{\mathrm{e}^{\boldsymbol{X}_i\boldsymbol{\beta}}}{1 + \mathrm{e}^{\boldsymbol{X}_i\boldsymbol{\beta}}}.$$

L'espressione della log-likelihood è:

$$\ln l(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left\{ y_i \ln \pi(\boldsymbol{X}_i) + (1 - y_i) \ln \left[1 - \pi(\boldsymbol{X}_i)\right] \right\}. \tag{6.17}$$

Da cui, differenziando rispetto ai parametri e ponendo a zero i risultati, si ottengono le r equazioni di likelihood:

$$\sum_i x_{ij} \left(y_i - \pi(\boldsymbol{X}_i)\right) = 0$$

che vengono risolte con metodi iterativi.

Descrizione e codificazione	Variabile
Low Birth Weight (0 = Birth Weight >= 2500g, 1 = Birth Weight < 2500g)	low
Age of the Mother in Years	age
Weight in Pounds at the Last Menstrual Period	lwt
Race (1 = White, 2 = Black, 3 = Other)	race
Smoking Status During Pregnancy (1 = Yes, 0 = No)	smoke
History of Premature Labor (0 = None, 1 = One, etc.)	ptl
History of Hypertension (1 = Yes, 0 = No)	ht
Presence of Uterine Irritability (1 = Yes, 0 = No)	ui
Number of Physician Visits During the First Trimester (0 = None, 1 = One, 2 = Two, etc.)	ftv
Birth Weight in Grams	bwt

Tabella 6.1: Lista delle variabili contenute nel dataset *birthwt* e loro codificazione.

Per illustrare un problema di questo tipo si fa uso del dataset *birthwt*, disponibile nella libreria *MASS*. I dati si riferiscono a uno studio, condotto al Baystate Medical Center (Springfield, Massachusetts) durante il 1986 condotto per identificare i fattori di rischio associati con il partorire bambini di peso inferiore ai 2500 grammi (low birth weight). I dati si riferiscono a 189 donne. Le variabili contenute nel file sono presentate in Tab. 6.1.

L'analisi si inizia importando i dati:

```
> library(MASS)
> data(birthwt)
> attach(birthwt)
```

La variabile dipendente è in questo caso *low*. Delle altre variabili contenute nel file si usano – nel corso dell'esempio – *age*, *lwt*, *ftv* (contine) e *race* (discreta a 3 livelli). Il modello di interesse si fitta facilmente:

```
> race <- factor(race)     # tratto la variabile RACE come categoriale
> mod.low <- glm(low ~ lwt + race + age + ftv, family=binomial(link=logit))
> summary(mod.low)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.295366   1.071203   1.209   0.2266
lwt         -0.014245   0.006539  -2.178   0.0294 *
race2        1.003897   0.497798   2.017   0.0437 *
race3        0.433108   0.362165   1.196   0.2317
age         -0.023823   0.033722  -0.706   0.4799
ftv         -0.049308   0.167201  -0.295   0.7681
[...]
    Null deviance: 234.67  on 188  degrees of freedom
Residual deviance: 222.57  on 183  degrees of freedom
```

Come per il caso di regressione logistica semplice la bontà del modello può essere valutata ricorrendo al test di devianza. La statistica G, che in questo caso ci si attende distribuita $\sim \chi^2(5)$, può essere calcolata direttamente o ottenuta fittando il modello nullo e sfruttando la funzione *anova*:

```
> mod.low0 <- glm(low ~ 1, family=binomial(link=logit))
> anova(mod.low, mod.low0, test="Chisq")
Analysis of Deviance Table

Model 1: low ~ lwt + race + age + ftv
Model 2: low ~ 1
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       183     222.57
2       188     234.67 -5  -12.099  0.03345 *
```

Si conclude che il modello contiene almeno una variabile che predice in modo adeguato il valore di *low*. Dall'analisi delle significatività dei singoli predittori come dai relativi test di Wald, si conclude che è possibile fittare un modello più "parsimonioso", contenente la sola variabile indipendente *lwt*. Tuttavia, come nel caso di regressione lineare multipla, l'inclusione di una variabile nel modello può avvenire per motivi differenti, ad esempio in questo caso la variabile *race* è considerata in letteratura come importante nel predire l'effetto in questione, quindi la si include nel modello ristretto. Nella Sec. 9.9 verrà descritta una procedura rigorosa per selezionare un sottomodello ottimale, dato un modello sovrabbondante di predittori.

Si può verificare se il modello contenente solamente *lwt* e *race* sia altrettanto buono del modello completo:

```
> mod.low2 <- glm(low ~ lwt + race, family=binomial(link=logit))
> anova(mod.low2, mod.low, test="Chisq")

Analysis of Deviance Table

Model 1: low ~ lwt + race
Model 2: low ~ lwt + race + age + ftv
  Resid. Df Resid. Dev  Df Deviance P(>|Chi|)
1       185    223.259
2       183    222.573   2    0.686     0.710
```

Il modello ristretto si comporta altrettanto bene del modello completo, e se lo scopo della ricerca è costruire il modello più semplice possibile sarà quello adottato. Il risultato finale è quindi:

```
> summary(mod.low2)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.805754   0.844982   0.954   0.3403
lwt         -0.015223   0.006438  -2.365   0.0180 *
race2        1.081066   0.487994   2.215   0.0267 *
race3        0.480603   0.356593   1.348   0.1777
[...]
    Null deviance: 234.67  on 188  degrees of freedom
Residual deviance: 223.26  on 185  degrees of freedom
```

Per quanto riguarda la valutazione degli OR, il predittore *race* è discreto a 3 livelli. In questo caso il livello 1 ($race = \text{White}$) viene assunto come categoria di riferimento. Si ha:

$$\begin{aligned} OR(race2 = \text{Black}, race1 = \text{White}) &= \exp(1.081) = 2.95 \\ OR(race3 = \text{Other}, race1 = \text{White}) &= \exp(0.4806) = 1.62 \end{aligned}$$

6.2.1 Tabelle di classificazione

Un modo spesso utilizzato per presentare i risultati di un modello di regressione logistica sono le tabelle di classificazione. In queste tabelle i dati vengono classificati secondo due chiavi: il valore della

variabile dipendente dicotoma y e il valore di una variabile dicotoma y_{mod} derivato dalla stima della probabilità ottenuta dal modello.

I valori di questa variabile si ottengono confrontando il valore della probabilità con un cut-off point $c \in [0, 1]$; se il valore della probabilità stimata dal modello supera c a y_{mod} si assegna il valore 1, altrimenti il valore 0. Molto spesso si sceglie per c il valore 0.5. Nel caso del modello a due predittori fittato nel paragrafo precedente si ha:

```
> tab <- table(low, mod.low2$fitted > 0.5)
> tab

    FALSE TRUE
  0   124    6
  1    53    6
```

dove il fattore di colonna è la classificazione operata dal modello confrontata con quella reale (fattore di riga). Il numero di casi classificati correttamente si ottiene con la chiamata:

```
> sum(diag(tab))/sum(tab)
[1] 0.6878307
```

quindi circa il 70% dei casi sono classificati in modo corretto; i casi restanti si dividono in 6 "falsi positivi" – ossia predetti "negativi" (a basso rischio) dal modello quando in realtaà sono "positivi" (ad alto rischio) – e 53 falsi negativi.

Questo approccio apparentemente semplice e comodo risente tuttavia del fatto che è frequente il caso in cui a modelli che ben si adattano a descrivere i dati corrispondono tabelle che predicono assai male l'appartenenza ai gruppi $y = 0, 1$ (si veda [35] pag. 156 e seguenti per una spiegazione dettagliata del fenomeno). Inoltre risulta che la corrispondenza è sempre migliore nel gruppo più numeroso, fenomeno che si evidenzia anche nell'esempio appena esaminato. Per questi motivi tale tecnica è da sconsigliarsi come unico metodo per testare l'adeguatezza del modello.

In alternativa, è possibile considerare altre metodiche, come ad esempio la costruzione della curva ROC – con il parametro c variato con continuità nel suo dominio – e la valutazione dell'area sotto di essa. Questa e altre tecniche vengono brevemente presentate nel paragrafo seguente.

6.2.2 Tecniche di valutazione della bontà di un modello logistico

Per valutare la bontà di un modello logistico è possibile costruire la curva ROC (Receiver Operator curve), procedendo nel modo seguente. Al variare del valore di cut-off $c \in [0, 1]$, si costruisce una tabella di classificazione come detto nella sezione precedente. A partire da tale tabella si calcolano il rate di falsi positivi e quello di veri positivi. Tali valori vengono usati per individuare un punto della curva. L'insieme dei punti ottenuto al variare di c in tutto il suo dominio costituisce la curva ROC. L'area sotto la curva è una valutazione della bontà del modello: più tale valore è vicino a 1, più valido è il modello; un valore di 0.5 indica un modello che non descrive il fenomeno meglio di quanto farebbe il puro caso.

Per costruire una curva ROC in R è necessario installare la libreria aggiuntiva *ROCR*. Essa mette a disposizione le funzioni *prediction* , che trasforma i dati di input in una forma richiesta dalle altre funzioni della libreria, e *performance* che valuta gli indici richiesti alla costruzione della curva.

Nel caso dell'esempio trattato precedentemente si procede con le chiamate:

```
> library(ROCR)
> pred <- prediction(mod.low2$fitted, low)
> perf <- performance(pred, "tpr", "fpr")
```

La funzione *prediction* accetta un minimo di due argomenti: il fit del modello logistico e la variabile dicotomica che specifica la classe di appartenenza del soggetto. La funzione *performance* lavora sull'oggetto elaborato dalla funzione precedente e richiede la specificazione di due etichette che identificano le quantità da utilizzare nella costruzione della curva ROC, in questo caso *tpr* (true positive

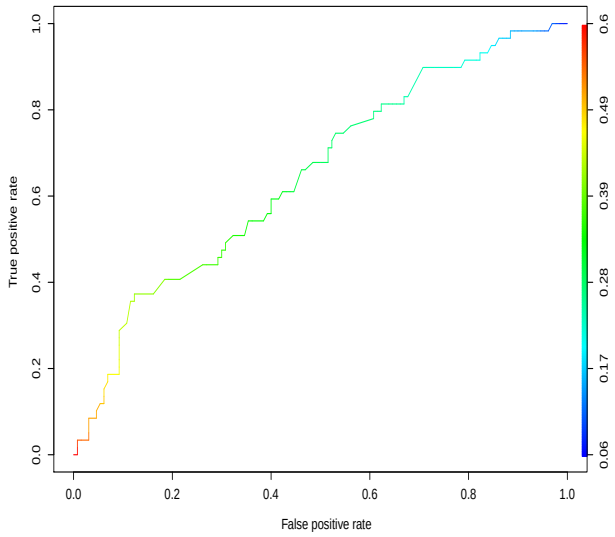

Figura 6.4: Curva ROC costruita per valutare la bontà del modello logistico multivariato. I colori identificano i valori del parametro di cut-off c.

rate) e fpr (false positive rate). Per altre possibili specificazioni si rimanda alla pagina di manuale della funzione.

La curva ROC si visualizza con la chiamata:

```
> plot(perf, colorize=TRUE)
```

dove l'opzione $colorize = TRUE$ serve per inserire, mediante scala di colori riportata a destra, le informazioni relative ai valori di c che hanno concorso alla costruzione della curva. Infine, la valutazione dell'area sotto la curva ROC si ottiene con la chiamata:

```
> performance(pred, "auc")@y.values
[[1]]
[1] 0.6473272
```

dove si è fatto uso dell'operatore @, necesario per accedere allo slot $y.values$ dell'oggetto di classe S4 podotto dalla chiamata a $performance$.

Altri indici valutativi delle performance di un modello logistico prevedono di calcolare il numero di coppie concordanti e discordanti secondo il modello nel modo seguente. Una coppia di soggetti in classi diverse della variabile di risposta dicotomica Y è detta concordante se la probabilità prevista dal modello di essere in classe $Y = 1$ è maggiore nel soggetto di classe $Y = 1$ rispetto a quello di classe $Y = 0$, è detta discordante se avviene il contrario ed è detta *tie* se non ricade in nessuno dei due casi precedenti. Detto n_c il numero di coppie concordanti, n_d il numero di coppie discordanti, n il numero di coppie possibili e N il numero di osservazioni si definiscono i seguenti indici:

- indice c (stima dell'area sotto la curva ROC):

$$c = (n_c + 0.5(t - n_c - n_d))/t$$

- D di Somer:

$$D = (n_c - n_d)/t$$

- τ_a di Kendall:

$$\tau_a = (n_c - n_d)/(0.5N(N-1))$$

- γ di Goodman-Kruskal:

$$\gamma = (n_c - n_d)/(n_c + n_d)$$

Tutti questi indici sono valutati automaticamente dalla funzione *lrm*, come si mostra nell'esempio seguente:

```
> mod3 <- lrm(low ~ lwt + race)
> mod3
[...]

       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
       189      2e-07      11.41          3     0.0097      0.647      0.293
     Gamma      Tau-a         R2      Brier
     0.296      0.127      0.082      0.202

          Coef     S.E.    Wald Z P
Intercept  0.80575 0.84517  0.95  0.3404
lwt       -0.01522 0.00644 -2.36  0.0181
race=2     1.08107 0.48805  2.22  0.0268
race=3     0.48060 0.35667  1.35  0.1778
```

Per l'identificazione degli indici, Dxy indica l'indice D, mentre gli altri sono di immediato riconoscimento. Oltre alle quantità menzionate sopra, la funzione calcola anche gli indici R^2 di Nagelkerke e lo score di Brier.

6.2.3 Calcolo dei residui

Il calcolo dei residui richiede sempre di classificare i dati per *covariate pattern*. A differenza di quanto visto precedentemente, la procedura coinvolge ora più di una variabile. La funzione *covariate* è disegnata per trattare problemi di questo genere e può quindi essere impiegata anche nel caso multivariato. La chiamata:

```
> cp <- covariate(mod.low2)
> cp
    n Y X.lwt X.race2 X.race3     fitted          h           rj           zj
1   1 0   182       1       0 0.29239405 0.04922044 -0.642818744 -0.659247686
2   1 0   155       0       1 0.25479043 0.02369590 -0.584725912 -0.591779318
3   2 1   105       0       0 0.31159517 0.03285479  0.575293840  0.584983857
[...]
109 1 1   142       1       0 0.43171415 0.04155113  1.147321893  1.171927662
```

produce in output un data frame contenente tre colonne *X.lwt*, *X.race*2 e *X.race*3, relative alle tre variabili che entrano nel modello (tenendo conto che *race* è discreto a tre livelli e quindi ha 2 gradi di libertà a cui sono associate due variabili dummy).

Per verificare quali *covariate pattern* forniscono i residui maggiori è spesso usato il plot di z_j^2 contro i valori di $\hat{\pi}_j$ (in Fig. 6.5), che si può ottenere con le chiamate:

```
> plot(cp$fitted, cp$zj^2, type="n", xlab="pi", ylab="zj^2")
> text(cp$fitted, cp$zj^2)
```

dove al posto dei punti è sostituito il numero di *covariate pattern* per meglio identificarlo.

Dall'analisi del grafico risulta che i *covariate pattern* 88 e 85 si adattano piuttosto male al modello. Per verificare a quali casi si riferiscano si può usare la chiamata:

```
> cp[c(85,88), ]
   n Y X.lwt X.race2 X.race3    fitted          h       rj       zj
85 2 2   187       1       0 0.2768981 0.10243189 2.285361 2.412243
88 1 1   165       0       0 0.1536757 0.01595530 2.346745 2.365693
```

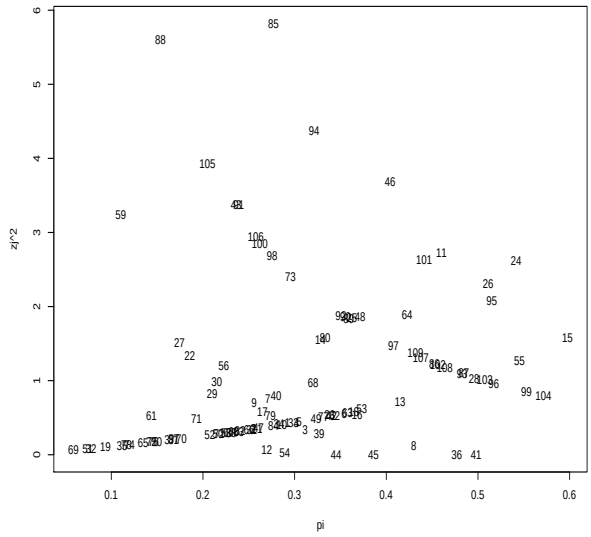

Figura 6.5: Analisi dei quadrati dei residui del modello di regressione logistica a due predittori, uno continuo e uno categoriale.

6.3 Polinomi frazionari e predittori continui

Se in un modello di regressione vi sono predittori continui può sorgere il sospetto che la relazione lineare tra predittori e variabile dipendente sia inappropriata. Nel caso di regressione lineare si hanno indicazioni in tal senso dall'analisi del grafico dei residui standardizzati. Per quanto riguarda il modello logistico questa analisi risulta assai ardua ed è quindi opportuno disporre di tecniche atte a mettere in luce eventuali problemi di non linearità.

Un metodo consolidato è la sostituzione del predittore continuo con uno categoriale, ottenuto dal precedente usando i suoi quartili per definire i livelli di separazione tra le categorie. Se le stime degli odds ratio nei gruppi sono significativamente differenti si ha una indicazione di non linearità e il modello con il predittore categoriale è da preferirsi.

Questa semplice procedura è però minata dalla dipendenza dalla scelta dei punti di separazione e spesso soffre di limitata potenza. Se si vuole preservare la natura continua del predittore in esame è possibile far uso della tecnica dei polinomi frazionari sviluppata nel 1994 da Royston e Altman [47]. Nel caso di regressione logistica semplice il logit $g(x)$ viene modellizzato come:

$$g(x) = \beta_0 + \sum_{j=1}^{J} \beta_j F_j(x) \tag{6.18}$$

dove le funzioni $F_j(x)$ sono definite come:

$$F_1(x) = x^{p_1} \quad , \quad F_j(x) = \begin{cases} x^{p_j} & p_j \neq p_{j-1} \\ F_{j-1}(x)\ln(x) & p_j = p_{j-1} \end{cases} \qquad j = 2, \ldots, J \tag{6.19}$$

In principio le potenze p_j possono assumere qualunque valore, ma, seguendo Royston e Altman, è d'uso restringersi ai valori della lista P = {-2,-1,-0.5,0,0.5,1,2,3}, con la convenzione che $p_j = 0$ denoti il logaritmo della variabile. Ad esempio se si sceglie $J = 2$, $p_1 = 2$, $p_2 = 0$ il logit si scrive:

$$g(x) = \beta_0 + \beta_1 x^2 + \beta_2 \ln(x).$$

Per quanto riguarda il valore di J, solitamente si pone $J = 1$ o $J = 2$.

La tecnica sviluppata da Royston e Altman richiede quindi di fittare tutti i possibili modelli con $J = 1$ e scegliere il migliore (quello con minore log-likelihood). La stessa procedura viene seguita per $J = 2$. Si paragonano quindi i due modelli tra loro e con il modello lineare, tenendo conto che ogni polinomio frazionario contribuisce con 2 gradi di libertà (uno per il coefficiente e uno per l'esponente).

Le ipotesi da verificare sono quindi:

- *Non linearità*: si confronta il modello lineare con il modello con $J = 2$. Per far questo si considerano le log-likelihood del modello lineare $L(1)$ e del modello a polinomi frazionari $L(p_1, p_2)$ e si costruisce la variabile:
$$G = -2(L(1) - L(p_1, p_2))$$
la quale, nell'ipotesi nulla di equivalenza dei due modelli, è il valore campionario di una statistica χ^2 a 3 gradi di libertà. Se il test risulta non significativo non si ha evidenza di non linearità, altrimenti si procede con il punto seguente.

- *Semplificazione*: Si testa il modello con $J = 2$ contro il modello con $J = 1$. Come sopra si costruisce la funzione G a cui spettano in questo caso 2 gdl. Se il test risulta significativo si preferirà il modello $J = 2$.

In generale i predittori continui possono essere più d'uno. In tal caso si procede costruendo indipendentemente le funzioni a polinomi frazionari per ognuna di tali variabili. La procedura viene quindi iterata una seconda volta ricercando nuovamente per ognuno dei predittori la forma funzionale migliore, scegliendo però per gli altri predittori continui le forme funzionali selezionate al passo precedente. L'algoritmo si arresta quando nessuna delle forme funzionali varia rispetto al passo precedente (solitamente due cicli bastano per arrivare a convergenza). Questa procedura risulta utilizzabile non solo nell'ambito della regressione logistica, ma anche per la usuale regressione lineare o per i modelli di Cox (si veda Sec. 7.3). Per ulteriori dettagli si rimanda a [47, 48].

In R questa tecnica è accessibile mediante le funzioni mfp e fp della libreria mfp. La funzione fp accetta come argomenti il nome della variabile da trasformare e il numero dei gradi di libertà da usare (pari a $2J$ come detto sopra). La funzione mfp esegue il confronto fra i vari modelli e propone quello che meglio si adatta ai dati.

Esempio

Nel caso dell'Esempio di Sec. 6.1 si vuole verificare che il predittore *AGE* entri linearmente nel modello o se meglio si adatti alla situazione in studio un modello non lineare.

Come primo passo si carica la libreria mfp:

```
> library(mfp)
```

Il test sulla linearità si conduce quindi con la chiamata:

```
> mod <- mfp(CHD ~ fp(AGE, df=4), data=chd, family=binomial(logit), verbose=TRUE)

        Variable        Deviance        Power(s)
-------------------------------------------------
Cycle 1
        AGE             136.663
                        107.353         1
                        107.353         1
                        107.212         3 3


Tansformation
    shift scale
AGE     0    10
```

```
Fractional polynomials
    df.initial select alpha df.final power1 power2
AGE          4      1  0.05        1      1      .


Null model: 136.6630
Linear model: 107.3531
Final model: 107.3531
```

La funzione $fp(AGE, df = 4)$ richiede di usare per la variabile AGE un modello con $J = 2$, mentre l'opzione $verbose = TRUE$ fa in modo che venga presentata la tabella iniziale in cui si vedono i test eseguiti. Da tale tabella si vede che l'algoritmo converge in un passo (cosa ovvia quando si ha un solo predittore); si leggono quindi 4 devianze, relative al modello che non contiene il predittore in studio, a quello che lo contiene linearmente, a quello a polinomi frazionari con $J = 1$ e infine a quello con $J = 2$. Nell'ultima colonna sono anche elencati i valori di p_j per i modelli migliori in ognuno degli ultimi due casi. Dai confronti delle devianze si vede che in questo caso il modello lineare è perfettamente adeguato.

Il modello selezionato dall'algoritmo è riepilogato nella tabella *Fractional polynomials* riportata poche righe dopo.

La tabella *Tansformation* riepiloga le trasformazioni che vengono fatte sulle variabili in fase di calcolo dall'algoritmo per motivi di stabilità numerica. In particolare se i predittori hanno valori negativi o particolarmente grandi il calcolo non viene condotto sulla variabile x ma sulla trasformata $x' = (x + shift)/scale$.

6.4 Regressione logistica multinomiale

Una generalizzazione delle tecniche di regressione logistica rende possibile trattare il caso in cui la variabile dipendente è categoriale a più di due livelli. Si parla in questo caso di regressione logistica multinomiale o policotomica. Per approfondimendti sull'argomento si rimanda a [35, 23].

Una prima distinzione da operare è fra regressione logistica nominale e ordinale. Si parla di regressione logistica nominale quando non vi è un ordine naturale fra le categorie della variabile dipendente, come possono essere la scelta fra tre candidati politici o fra alcuni colori. Quando invece è possibile classificare i livelli della variabile dipendente in una scala ordinata si parla di regressione logistica ordinale.

6.4.1 Regressione logistica ordinale

Si assuma che la variabile dipendente Y abbia $K + 1$ livelli codificati $k = 0, 1, \ldots, K$. Si indichi con $P(Y = k|\mathbf{x}) = \pi_k(\mathbf{x})$ la probabilità che la variabile dipendente sia di livello k condizionata al vettore $\mathbf{x}$ dei p predittori.

Nel caso di regressione logistica ordinale è possibile scegliere fra vari modelli. I più diffusi sono: il modello adjacent-category, il modello continuation-ratio e il modello proportional odds. Nel seguito verrà trattato l'ultimo di questi.

Il modello proportional odds lega la probabilità che sia $Y \leq k$ con quella che risulti $Y > k$ basandosi sull'ipotesi che l'effetto dei predittori sia lo stesso in tutte le categorie di Y. Si ha quindi:

$$\ln\left[\frac{P(Y \leq k|\mathbf{x})}{P(Y > k|\mathbf{x})}\right] = \ln\left[\frac{\pi_0(\mathbf{x}) + \ldots + \pi_k(\mathbf{x})}{\pi_{k+1}(\mathbf{x}) + \ldots + \pi_K(\mathbf{x})}\right] = \tau_k + \mathbf{x}^T\boldsymbol{\beta} \tag{6.20}$$

per $k = 0, 1, \ldots, K - 1$. Come detto, mentre le intercette τ_k dipendono dalla categoria k, i predittori $\boldsymbol{\beta}$ ne sono indipendenti. Si noti anche che nel caso di $K = 1$ questo modello si riduce al caso canonico di regressione logistica multipla.

Esempio

Con i dati di Sec. 6.2 si studia il caso in cui la variabile dipendente $levBWT$ sia categoriale a 4 livelli, come nello schema seguente:

- $levBWT = 0$ se $BWT > 3500$
- $levBWT = 1$ se $3000 < BWT \leq 3500$
- $levBWT = 2$ se $2500 < BWT \leq 3000$
- $levBWT = 3$ se $BWT \leq 2500$

Si è scelta la convenzione che più basso è il peso alla nascita del bambino più alto è il valore della variabile dipendente. Questa scelta rispecchia quanto fatto in Sec. 6.2 dove la variabile dipendente *low* (a 2 livelli) assumeva valore 1 se il peso del bambino alla nascita era inferiore a 2500 g, 0 altrimenti.

Si inizia l'analisi importando il file di dati e creando la variabile dipendente categoriale secondo lo schema dato sopra:

```
> data(birthwt)
> levBWT <- cut(birthwt$bwt, c(0,2500, 3000, 3500, 10000), right=TRUE, label=FALSE)
> levBWT <- abs(levBWT - 4)
```

l'ultima riga è necessaria per codificare i livelli di $levBWT$ nell'ordine desiderato (0 = peso maggiore, 3 = peso minore).

Si vuole studiare la dipendenza di $levBWT$ dalle variabili *lwt* e *smoke*. I modelli possono essere fittati con la funzione *lrm* della libreria *Design*:

```
> library(Design)
> mod <- lrm(levBWT ~ lwt, data=birthwt)
> mod
[...]
Frequencies of Responses
 0  1  2  3
47 45 38 59

       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
       189      3e-06       8.66          1     0.0033      0.604      0.208
     Gamma      Tau-a         R2      Brier
     0.212      0.155      0.048      0.184

     Coef     S.E.     Wald Z P
y>=1  2.76636 0.601014  4.60  0.0000
y>=2  1.67347 0.577705  2.90  0.0038
y>=3  0.79841 0.568087  1.41  0.1599
lwt  -0.01247 0.004308 -2.89  0.0038
```

In output si ha dapprima il calcolo delle frequenze osservate delle K classi di Y, poi la zona in cui vengono presentate le varie statistiche sul modello e infine la tabella dei i coefficienti e le loro significatività. I primi tre coefficienti che compaiono ($y >= 1$, $y >= 2$ e $y >= 3$) sono le tre intercette dei modelli. Il coefficiente di *lwt* è molto simile a quanto trovato nel caso di variabile dipendente dicotomica di Sec. 6.2. L'effetto del predittore risulta altamente significativo.

Anche in caso di regressione multinomiale rimane valida l'interpretazione dei coefficienti in termini di odds; in questo caso l'odds ratio di partorire bambini più leggeri (alto $levBWT$) contro bambini pesanti (basso $levBWT$) per un incremento di 10 libbre della variabile *lwt* è:

$$exp(-0.01247 \times 10) = 0.88$$

cioè un incremento di 10 libbre in *lwt* riduce l'odds di una nascita di un bambino di basso peso di circa il 12%.

Si passa quindi all'analisi del modello che contiene la variabile *smoke*:

```
> mod2 <- lrm(levBWT ~ smoke, data=birthwt)
> mod2
[...]
Frequencies of Responses
 0  1  2  3
47 45 38 59

       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
       189      3e-07       8.18          1     0.0042      0.576      0.152
     Gamma      Tau-a         R2      Brier
     0.314      0.113      0.045      0.181

      Coef    S.E.   Wald Z P
y>=1   0.8333 0.1929  4.32  0.0000
y>=2  -0.2528 0.1822 -1.39  0.1652
y>=3  -1.1215 0.1987 -5.64  0.0000
smoke  0.7708 0.2718  2.84  0.0046
```

Anche in questo caso la variabile ha effetto altamente significativo. L'odds ratio di partorire bambini di basso peso contro bambini di alto peso per una madre fumatrice rispetto a una non fumatrice è:

$$exp(0.7608) = 2.14$$

6.4.2 Regressione logistica ordinale multipla

Se i predittori sono più di uno si procede come nel caso di regressione logistica multipla dicotomica. Ad esempio il modello che contiene sia la variabile *lwt* che *smoke* si può fittare nel modo seguente:

```
> mod3 <- lrm(levBWT ~ smoke + lwt, data=birthwt)
[...]
Frequencies of Responses
 0  1  2  3
47 45 38 59

       Obs  Max Deriv Model L.R.       d.f.          P          C        Dxy
       189      9e-13      15.88          2      4e-04      0.629      0.259
     Gamma      Tau-a         R2      Brier
     0.262      0.193      0.086      0.178

      Coef     S.E.    Wald Z P
y>=1   2.43540 0.61496  3.96  0.0001
y>=2   1.31302 0.59499  2.21  0.0273
y>=3   0.41447 0.58787  0.71  0.4808
smoke  0.72924 0.27316  2.67  0.0076
lwt   -0.01191 0.00433 -2.75  0.0060
```

Si nota che entrambi i predittori sono altamente significativi al test di Wald e che la stima dei coefficienti non cambia molto rispetto ai due modelli univariati fittati in precedenza.

Il test per l'eliminazione di un predittore va condotto come di consueto confrontando le devianze dei modelli senza e con il predittore in studio. Ad esempio si può verificare se l'inserimeto del predittore *smoke* (a cui spetta 1 gdl) nel modello con solo *lwt* risulta significativo:

```
> mod$dev - mod3$dev
[1] 0.000000 7.042398
> 1 - pchisq(7.042398, 1)
[1] 0.007960238
```

L'ipotesi nulla è che i modelli siano equivalenti. Visto il valore ottenuto ($P = 0.008$) essa viene respinta e il modello con entrambe le variabili è da preferirsi.

6.5 Regressione di Poisson e modelli log-lineari

Una forma molto comune di dato è il numero di volte in cui un certo evento si verifica, o equivalentemente il tasso con cui esso si presenta in un numero variabile di osservazioni. Se gli eventi si verificano indipendentemente e allo stesso tasso allora, nel tentativo di legare fra loro la variabile dipendente Y (numero di conteggi) e la matrice dei p predittori X, si può usare la distribuzione di Poisson:

$$Y \sim Poisson(\beta X).$$

Questo tipo di modellizzazione è inappropriato se gli eventi tendono ad avvenire in modo clusterizzato o se la dipersione del numero dei conteggi non è ben descritta da una distribuzione di Poisson (che, si ricorda, ha varianza pari al valor medio). In particolare, se la dispersione dei conteggi è significativamente più elevata della loro media, si ha il caso di iperdispersione, che è associabile a una distribuzione binomiale negativa delle variabili Y.

In Sec. 2.2.2 si è visto un esempio di classificazione di conteggi in base a due fattori in una tabella di contingenza. L'analisi che viene introdotta ora si basa su un approccio molto più versatile che passa attraverso la definizione e il fit di un modello lineare generalizzato. Data la particolare funzione di link che si usa si parla di modello log-lineare. Le tecniche matematiche che sono alla base di queste modellizzazioni sono ampiamente descritte in [23]. Senza scendere nei dettagli, sono riportati di seguito gli aspetti principali della problematica.

6.5.1 Modelli log-lineari e tabelle di contingenza

Nel caso di tabella di contingenza a due (o eventualmente più) chiavi di classificazione, si può modellizzare il valore di aspettazione di variabili Y, i cui valori campionari y sono il numero di conteggi nelle celle della tabella (o iper-tabella), come un prodotto di variabili (vedi [23]). La funzione di link logaritmica produce un modello log-lineare:

$$\log Y = \beta X$$

dove X è la matrice dei predittori e β il vettore dei parametri da stimare. In R l'indagine si condurrà fittando il modello lineare generalizzato, facendo nuovamente uso della funzione glm. Dato che il disegno sperimentale pone spesso dei vincoli sui valori dei conteggi disposti in tabella (ad esempio il totale generale è fissato dallo sperimentatore, e così possono esserlo i totali marginali), tali informazioni devono essere incorporate nel modello.

La bontà di un modello si può stabilire esaminando i residui standardizzati o residui di Pearson, definiti come:

$$r_i = \frac{y_i - \hat{y}_i}{\sqrt{\hat{y}_i}}$$

dove y_i sono le frequenze osservate nelle celle della tabella e $\hat{y}_i$ quelle teoriche previste dal modello. La somma dei quadrati dei residui è legata alla distribuzione χ^2 dato che:

$$x^2 = \sum_i r_i^2$$

è il valore campionario di una variabile χ^2 a $n - p$ gradi di libertà, dove n è il numero totale di celle della tabella e p il numero di parametri stimati dal modello (compreso il termine b_0).

Per stabilire se una o più variabili possano essere, dal punto di vista statistico, rimosse dal modello senza produrre peggioramenti significativi è possibile confrontare, come nel caso di regressione logistica, le devianze dei due modelli. Nel caso di modelli log-lineari la devianza è definita come:

$$D = 2 \sum_i \left[y_i \log \frac{y_i}{\hat{y}_i} - (y_i - \hat{y}_i) \right]$$

Dato che questo tipo di studio è solitamente finalizzato a verificare (o confutare) l'indipendenza fra i fattori di classificazione, saranno proprio le interazioni fra i fattori a dover essere analizzate. Per un esempio pratico si riprendono i dati di Sec. 2.2.2, relativi alla valutazione di tre diverse cure. In questo studio i pazienti venivano classificati a seconda del fatto che, in seguito alla cura seguita, fossero o no migliorati. Per condurre l'analisi con la funzione *glm* si inserisce il numero di pazienti nel vettore *conteggi*, accompagnato dai fattori *stato* e *cura* che tengono traccia del miglioramento/non miglioramento del paziente e della cura somministrata:

```
> conteggi <- c(10,7,18, 21,30,17)
> stato <- factor(c(rep("migliorato", 3), rep("non migliorato", 3)))
> cura <- factor(rep(c("A","B","C"), 2))
```

Il test χ^2 sulla tabella di contingenza era teso a verificare l'ipotesi di indipendenza fra i due fattori o, in altre parole, che la loro interazione non fosse significativa. Il calcolo svolto in Sec. 2.2.2 mostrava che tale interazione è in questo caso al limite della alta significatività, portando a concludere che le tre cure non sono equivalenti.

Per riprodurre questa analisi fittando un modello log-lineare si usa la funzione *glm* con link di tipo *poisson*. I modelli senza e con interazione si fittano con le chiamate:

```
> mod <- glm(conteggi ~ stato + cura, family=poisson())
> mod1 <- glm(conteggi ~ stato * cura, family=poisson())
```

Per valutare la significatività dell'interazione si ricorre alla funzione *anova* per paragonare le devianze dei due modelli:

```
> anova(mod, mod1, test="Chisq")
Analysis of Deviance Table

Model 1: conteggi ~ stato + cura
Model 2: conteggi ~ stato * cura
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         2     8.6561
2         0 -2.665e-15  2   8.6561    0.0132
```

Si conclude che, in accordo a quanto trovato precedentemente, l'interazione fra i due fattori è quasi altamente significativa.

È molto interessante notare che i valori dei conteggi previsti dal modello adittivo coincidono esattamente con quelli della tabella teorica calcolata durante il test χ^2. Infatti questi ultimi sono:

```
> chisq.test(matrix(conteggi, nc=2))$expected
         [,1]     [,2]
[1,] 10.53398 20.46602
[2,] 12.57282 24.42718
[3,] 11.89320 23.10680
```

mentre per il modello log-lineare si possono ottenere con la chiamata:

conteggi	ulcera	stato	aspirina
62	G	controllo	NU
6	G	controllo	U
39	G	caso	NU
25	G	caso	U
53	D	controllo	NU
8	D	controllo	U
49	D	caso	NU
8	D	caso	U

Tabella 6.2: Studio caso-controllo per stabilire se l'aspirina è un fattore di rischio per l'ulcera.

```
> e <- mod$fitted       # valori previsti dal modello
> matrix(e, nc=2)
         [,1]     [,2]
[1,] 10.53398 20.46602
[2,] 12.57282 24.42718
[3,] 11.89320 23.10680
```

Il vantaggio dei modelli log-lineari è che sono utilizzabili in situazioni più complesse, quando i fattori in gioco sono più di due, come nell'esempio seguente.

Esempio

In uno studio retrospettivo caso-controllo (analizzato in [23]), alcuni pazienti che soffrono di ulcera sono appaiati a pazienti simili che non ne soffrono. I pazienti che soffrono di ulcera sono classificati a seconda del sito dell'ulcera: gastrica (G) o duodenale (D). Si accerta quindi l'uso di aspirina fra i vari pazienti (U = utilizzatori di aspirina, NU = non utilizzatori). I risultati sono riportati in Tab. 6.2.

Si vuole stabilire se l'ulcera è associata all'uso di aspirina e se l'effetto dell'aspirina è diverso a seconda del sito dell'ulcera.

Si inizia l'analisi inserendo i dati:

```
> conteggi <- c(62,6, 39,25, 53,8, 49,8)
> ulcera <- factor(c(rep("G",4), rep("D",4)))
> stato <- factor(rep(c("cont","cont","caso","caso"),2))
> aspirina <- factor(rep(c("NU","U"),4))
```

Per stabilire se l'aspirina è un fattore di rischio per l'ulcera, si deve accertare la significatività dell'interazione fra le variabili *aspirina* e *stato* dopo avere corretto per gli effetti delle altre variabili. I modelli da confrontare sono quindi quello che contiene le variabili *stato* e *ulcera*, la loro interazione e la variabile *aspirina* con quello che contiene anche l'interazione fra le variabili *aspirina* e *stato*. I due modelli si fittano con le chiamate:

```
> mod <- glm(conteggi ~ ulcera * stato + aspirina, family=poisson())
> mod2 <- glm(conteggi ~ ulcera * stato + aspirina * stato, family=poisson())
```

Il modo migliore per stabilire la significatività dell'interazione è paragonare le devianze dei due modelli:

```
> anova(mod, mod2, test="Chisq")
Analysis of Deviance Table

Model 1: conteggi ~ ulcera * stato + aspirina
Model 2: conteggi ~ ulcera * stato + aspirina * stato
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         3    21.7893
2         2    10.5384  1  11.2508    0.0008
```

La differenza fra i modelli è altamente significativa. Si conclude quindi che l'aspirina può essere considerata un fattore di rischio per l'ulcera. Un modo alternativo per verificare la significatività del termine di interazione è quello di analizzare il test di Wald che viene presentato dalla chiamata:

```
> summary(mod2)
```

tuttavia, data la sua bassa potenza, per stabilire la significatività di un qualunque coefficiente è preferibile eseguire il test sulle devianze.

Per stabilire se l'aspirina è associata in modo differete a i siti di ulcera, si fitta il modello che comprende anche l'interazione fra le variabili *aspirina* e *ulcera*:

```
> mod3 <- glm(conteggi ~ ulcera*stato*aspirina - ulcera:stato:aspirina,
+   family=poisson())
> anova(mod2, mod3, test="Chisq")
Analysis of Deviance Table

Model 1: conteggi ~ ulcera * stato + aspirina * stato
Model 2: conteggi ~ ulcera * stato * aspirina - ulcera:stato:aspirina
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1         2    10.5384
2         1     6.2830  1   4.2555    0.0391
```

Il confronto di questo modello con il precedente raggiunge la significatività. Per interpretare questo risultato si possono esaminare i coefficienti del modello fittato:

```
> summary(mod3)
[...]
Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)          3.81846    0.14515  26.307  < 2e-16 ***
ulceraG             -0.06977    0.20415  -0.342  0.73254
statocont            0.21517    0.19172   1.122  0.26174
aspirinaU           -1.37910    0.29514  -4.673 2.97e-06 ***
ulceraG:statocont    0.10574    0.26147   0.404  0.68590
ulceraG:aspirinaU    0.70005    0.34603   2.023  0.04306 *
statocont:aspirinaU -1.14288    0.35207  -3.246  0.00117 **

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 127.749  on 7  degrees of freedom
Residual deviance:   6.283  on 1  degrees of freedom
AIC: 59.898
```

Dal fatto che il penultimo coefficiente del modello sia positivo si conclude che l'uso di aspirina è un fattore di rischio maggiore per l'ulcera gastrica rispetto a quella duodenale.

Per quanto riguarda la bontà di adattamento del modello finale, i residui di Pearson si ottengono con la chiamata:

```
> residuals(mod3,"pearson")
         1          2          3          4          5          6          7
 0.4529949 -1.1264726 -0.5318374  0.7468503 -0.4612175  1.6275938  0.5136050
         8
-1.0235214
```

da cui si ha la statistica x^2:

```
> sum(residuals(mod3,"pearson")^2)
[1] 6.48795
```

Analizzando l'output della funzione *summary* dato in precedenza si nota che vi è un solo grado di libertà residuo, quindi il valore P per il test di bontà d'adattamento è:

```
> 1 - pchisq(6.48795, 1)
[1] 0.01086082
```

da cui si conclude che il modello non fitta particolarmente bene i dati pur impiegando 7 parametri per descrivere 8 rilevazioni.

Capitolo 7

Analisi della sopravvivenza

In alcuni esperimenti si misurano i tempi a partire da un ben definito istante iniziale fino al verificarsi di un particolare evento (che viene normalmente detto "failure" o "decesso"). Ad esempio, in campo medico rientrano in tale categoria studi longitudinali su pazienti a cui è stata diagnosticata una malattia fino al momento del loro decesso, mentre in campo ingegneristico si può pensare ai tempi di vita di un particolare componente meccanico o elettronico. Dati di questo genere sono caratterizzati solitamente da una distribuzione fortemente asimmetrica con una lunga coda destra. Dato che alcuni soggetti "sopravvivono" oltre il tempo di osservazione si ha il problema di non conoscere in quale particolare istante futuro essi andranno incontro al "decesso". In questo caso si parla di dati troncati o *censored*. Il problema generale è quello di valutare la probabilità di sopravvivenza in funzione del tempo, eventualmente in dipendenza da altre variabili in studio.

7.1 Funzioni di sopravvivenza e di rischio

Sia T una variabile casuale che descrive i tempi di sopravvivenza dei pazienti e $f(t)$ la sua funzione di densità di probabilità. La probabilità di un decesso prima del tempo t è data dalla funzione di distribuzione cumulativa:

$$F(t) = Pr(T < t) = \int_0^t f(t')\, dt'$$

La funzione di sopravvivenza che dà la probabilità di sopravvivenza oltre il tempo t è quindi:

$$S(t) = 1 - F(t).$$

La funzione di rischio (hazard function) $h(t)$ è definita come la probabilità di decesso nell'intervallo di tempo infinitesimo fra t e $t + \delta t$, data la sopravvivenza fino al tempo t. In formula:

$$h(t) = \frac{f(t)}{S(t)}$$

Questa relazione si può scrivere anche nella forma:

$$h(t) = -\frac{d}{dt} \ln S(t)$$

e quindi:

$$S(t) = \exp(-H(t)) \tag{7.1}$$

dove si è posto:

$$H(t) = \int_0^t h(t')\, dt'.$$

La funzione $H(t)$ viene comunemente detta funzione di rischio cumulativa.

7.2 Stime campionarie

Per affrontare il problema di dare una stima della funzione di sopravvivenza $S(t)$ si consideri un gruppo di n pazienti. Sia $Y_i(t)$ con $i, 1, \ldots, n$ un indicatore del fatto che il paziente i-esimo sia vivo e a rischio al tempo t. $Y_i(t)$ assume valore 1 in caso che la condizione sia soddisfatta e 0 altrimenti (paziente deceduto o censored). Si ha quindi che $Y(t) = \sum_i Y_i(t)$ è il numero di pazienti a rischio al tempo t. Si definisce anche $d(t)$ il numero di decessi che avvengono al tempo t.

La stima campionaria di sopravvivenza più comune è quella di Kaplan-Meier (KM), che è un prodotto di probabilità di sopravvivenza:

$$\hat{S}_{KM} = \prod_{t_i<t} \frac{Y(t_i) - d(t_i)}{Y(t_i)} \tag{7.2}$$

Graficamente la funzione di Kaplan-Meier è una curva a gradini che ha una caduta in ogni istante t_i in cui si verificano dei decessi. Solitamente, su questa curva vengono riportati con un simbolo grafico (in R un "+") i tempi a cui un paziente esce dallo studio a causa del censoring.

Il modello di KM è valido a patto che siano rispettate alcune assunzioni:

- Controllo delle variabili di confondimento
- Indipendenza del censoring (troncamento)
- Numero limitato di dati censored
- Campione di dimensione sufficientemente grande

7.2.1 Controllo delle variabili di confondimento

Come in ogni esperimento, nel raccogliere i dati bisogna assicurarsi di misurare sui pazienti tutte le variabili che si sospetta possano influenzarne la sopravvivenza. Nel caso in cui ci si attende che la stima della sopravvivenza possa essere diversa all'interno del campione a causa dell'effetto di una di tali variabili è necessario apportare una correzione all'analisi, ricorrendo alla stratificazione del modello, una tecnica analoga all'introduzione di variabili di blocco nell'analisi di una modello ANOVA. Si cercherà quindi di introdurre degli strati all'interno dei quali i pazienti siano il più omogenei possibile fra loro.

7.2.2 Indipendenza del censoring

Perché il modello sia valido è necessario che il censoring sia indipendente dal gruppo di classificazione. Se pazienti troppo malati sono eliminati dallo studio, o viceversa pazienti che sopravvivono molto a lungo vanno persi al follow-up si introducono distorsioni sulle stime di sopravvivenza. In secondo luogo bisogna essere certi che la perdita al follow-up di un paziente non influenzi la perdita di altri pazienti (indipendenza fra censoring e soggetti).

7.2.3 Numero limitato di dati censored

Uno studio può terminare con molti dati censored, o perché si hanno perdite al follow-up o perché molti soggetti sono ancora vivi al termine dello studio. In questo secondo caso vi è il sospetto di una cattiva pianificazione della durata dell'esperimento, che avrebbe dovuto essere maggiore. La presenza di molti dati censored influisce sulla stima della curva di KM dato che diminuisce il numero di pazienti a rischio, rendendo la stima di sopravvivenza meno precisa di quanto non si avrebbe in presenza di un numero ridotto di dati censored.

stato	mesi	età	gruppo	stato	mesi	età	gruppo	stato	mesi	età	gruppo
1	23	50	0	0	104	56	1	1	22	56	0
0	134	57	1	0	125	57	1	0	69	54	1
1	20	49	0	0	121	57	1	0	54	66	0
0	116	67	1	1	24	60	1	1	14	66	1
1	7	63	0	0	137	67	1	0	65	64	1
1	84	72	1	0	109	72	1	1	18	64	1
0	80	70	1	0	107	68	1	0	39	66	1
0	41	69	1	0	121	71	1	0	123	72	1
0	20	64	0	1	10	46	0	0	137	71	1
0	137	63	1	0	16	69	0	0	104	66	1
1	38	66	1	0	6	72	1	1	29	67	1
0	117	50	1	1	31	58	0	0	123	71	0
1	23	57	0	1	35	66	1	0	67	72	1
1	18	69	1	0	18	70	0	1	12	69	0
1	9	62	0	0	104	66	1	1	12	63	0
0	76	58	0	0	108	68	1	1	16	71	0
1	104	55	1	0	9	53	0	1	13	69	1
0	20	68	1	1	50	55	0	1	49	64	0
1	16	65	1	0	50	70	1	0	110	73	1
1	54	71	0	0	128	68	1	1	5	63	0

Tabella 7.1: Valori registrati su un campione di 60 pazienti con carcinoma polmonare nel caso dell'Esempio 7.2.4.

7.2.4 Campione di dimensione sufficientemente grande

Questo punto è strettamente legato al precedente. Le curve di KM hanno una precisione accettabile solo in presenza di un numero sufficientemente grande di osservazioni.

Esempio

In uno studio longitudinale un gruppo di 60 pazienti che hanno sviluppato un carcinoma polmonare di tipo NSCLC (Non Small Cell Lung Cancer) viene seguito dopo un intervento chirurgico volto a rimuovere la massa tumorale. I pazienti sono classificati per età al momento dell'intervento e a seconda della valutazione del tumore eseguita sulla scala T (dimensione tumorale, $T = 1, 2, 3$), N (infiltrazione linfonodale, $N = 0, 1, 2$. $N = 0$ indica assenza di infiltrazione). La variabile *gruppo* viene costruita in modo che valga 1 per pazienti di classe $N0$ e $T1$ o $T2$, mentre valga 0 altrimenti. Si vuole valutare la sopravvivenza generale e in relazione alle due variabili in studio. I dati sono presentati in Tab. 7.1. La variabile *stato* vale 1 se il paziente è deceduto e 0 se è ancora vivo al termine del periodo di osservazione, mentre *mesi* è il periodo di tempo per cui il paziente è stato seguito dopo l'intervento chirurgico.

La stima di KM per la sopravvivenza si ottiene facilmente facendo uso delle funzioni $Surv$ e $survfit$ della libreria standard *survival*. Se i dati sono stati inseriti nel data frame *car* si può avere la stima della funzione di sopravvivenza generale con le chiamate:

```
> library(survival)
> mod <- survfit( Surv(mesi, stato), data=car)
> mod
Call: survfit(formula = Surv(mesi, stato), data = car)

        n    events     rmean se(rmean)    median   0.95LCL   0.95UCL
     60.0      26.0      85.1       7.6       Inf      49.0       Inf
```

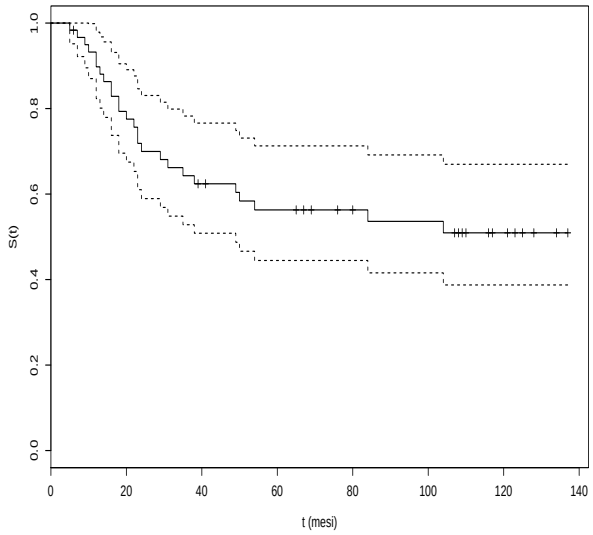

Figura 7.1: Stimatore di KM della curva di sopravvivenza generale.

La funzione *Surv* accetta in questo caso due argomenti: il primo è il tempo per cui il paziente è stato seguito dal momento in cui entra nello studio, il secondo lo stato alla fine del periodo di osservazione. Tale variabile deve essere codificata 1/0 (o TRUE/FALSE) con il valore 1 (TRUE) a indicare il decesso. L'output della funzione *Surv* viene usato dalla funzione *survfit* come variabile dipendente nella costruzione dei modelli di sopravvivenza di KM. Nel caso di sopravvivenza generale il modello non prevede nessun predittore. In output la funzione riporta il numero di pazienti in studio, il numero di decessi (events), la stima del tempo medio di sopravvivenza con il suo errore standard (vedi [54] per i dettagli di calcolo) e la mediana di sopravvivenza con il suo intervallo di confidenza. In questo caso si vede che la mediana non è calcolabile (Inf); ciò dipende dal fatto che tale valore è definito come l'istante di tempo in cui la curva stimata passa per il valore $S(t) = 0.5$, cioè l'istante in cui il 50% dei pazienti è deceduto. Dato che nell'esempio in questione ciò non avviene mai tale valore risulta non calcolabile. L'intervallo di confidenza della mediana è valutato in modo analogo considerando i tempi in cui le curve che delimitano l'intervallo di confidenza di $S(t)$ passano per il valore 0.5. Ovviamente se la mediana risulta non calcolabile altrettanto avviene per l'estremo superiore del suo intervallo di confidenza.

Il grafico della funzione di sopravvivenza di KM, dato in Fig. 7.1, si ottiene con la chiamata:

```
> plot(mod, xlab="t (mesi)", ylab="S(t)")
```

Le curve tratteggiate rappresentano l'intervallo di confidenza di $S(t)$.

La dipendenza della sopravvivenza dalle variabili *gruppo* ed *eta* può essere studiata aggiungendo i predittori al modello da fittare. Per valutare l'effetto della classificazione tumorale sulla sopravvivenza si usa la chiamata:

```
> mod2 <- survfit( Surv(mesi, stato) ~ gruppo, data=car)
> mod2
Call: survfit(formula = Surv(mesi, stato) ~ gruppo, data = car)

              n events rmean se(rmean) median 0.95LCL 0.95UCL
gruppo=FALSE 22     15  42.7      9.72     23      16     Inf
gruppo=TRUE  38     11 105.6      8.18    Inf     Inf     Inf
```

Si vede che la media dei tempi di sopravvivenza nel gruppo 0 (FALSE) è notevolmente inferiore a quelle del gruppo 1 (43 mesi contro 106). Il plot delle due curve (Fig. 7.2) evidenzia come la variabile

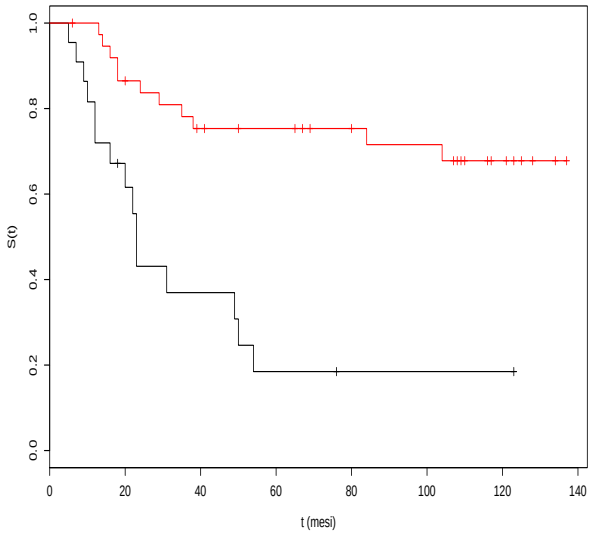

Figura 7.2: Stimatori di KM delle curve di sopravvivenza in relazione alla classificazione tumorale. La curva rossa si riferisce a pazienti di gruppo 1 ed evidenzia sopravvivenza più lunga rispetto ai pazienti di gruppo 0.

in esame sia importante per prevedere la sopravvivenza di un soggetto. Il comando per ottenere il grafico è:

```
> plot(mod2, xlab="t (mesi)", ylab="S(t)", col=c("black","red"))
```

in cui si specificano i colori con cui disegnare le due curve. Si noti che, nel caso le curve di sopravvivenza siano più di una, i loro intervalli di confidenza non vengono rappresentati. Tale scelta consente una più facile lettura del grafico evitando di appesantirlo con la visualizzazione contemporanea di troppe curve. Eventualmente si può far ricorso all'opzione $conf.int = TRUE$ che cambia questo comportamento:

```
> plot(mod2, xlab="t (mesi)", ylab="S(t)", col=c("black","red"), conf.int=TRUE)
```

Per valutare precisamente la differenza fra le due curve si può ricorrere al comando $survdiff$, che esegue un log-rank test per confrontare le curve di sopravvivenza (si veda [2] pag 576 per la matematica alla base del test):

```
> survdiff(Surv(mesi, stato) ~ gruppo, data=car)
Call:
survdiff(formula = Surv(mesi, stato) ~ gruppo, data = car)

              N Observed Expected (O-E)^2/E (O-E)^2/V
gruppo=FALSE 22       15     6.26     12.22      16.9
gruppo=TRUE  38       11    19.74      3.87      16.9

 Chisq= 16.9  on 1 degrees of freedom, p= 3.99e-05
```

Dal valore P del test si conclude che la differenza tra le due curve è altamente significativa. La funzione $survdiff$ accetta anche l'opzione rho che può assumere valori fra 0 (valore di default) e 1. Tale opzione controlla il tipo di test da eseguire secondo la famiglia G^ρ di Harrington e Fleming [32]. Il valore $rho = 0$ corrisponde al log-rank test, mentre $rho = 1$ esegue il test di Peto-Wilcoxon. I test differiscono nel modo in cui pesano i decessi: dato che essi vengono pesati con la funzione $\hat{S}(t)^\rho$, il log-rank test equivale al caso di decessi non pesati.

Per quanto riguarda l'analisi rispetto alla variabile continua *eta* si procede introducendo al suo posto una variabile categoriale *lev.eta* che separa i pazienti in due gruppi: quelli che hanno età inferiore alla mediana delle età (a cui si assegna il livello 1) e quelli che invece sono più anziani (livello 0):

```
> car$lev.eta <- car$eta < median(car$eta)
> mod3 <- survfit( Surv(mesi, stato) ~ lev.eta, data=car)
> mod3
Call: survfit(formula = Surv(mesi, stato) ~ lev.eta, data = car)

               n events rmean se(rmean) median 0.95LCL 0.95UCL
lev.eta=FALSE 33     10 100.9      9.45    Inf      84     Inf
lev.eta=TRUE  27     16  66.9     11.17     31      22     Inf
```

In questo caso si vede che la sopravvivenza media dei pazienti più anziani risulta maggiore. Le due curve differiscono in maniera significativa come risulta dal log-rank test:

```
> survdiff( Surv(mesi, stato) ~ lev.eta, data=car)
Call:
survdiff(formula = Surv(mesi, stato) ~ lev.eta, data = car)

               N Observed Expected (O-E)^2/E (O-E)^2/V
lev.eta=FALSE 33       10     15.7      2.07       5.3
lev.eta=TRUE  27       16     10.3      3.16       5.3

 Chisq= 5.3  on 1 degrees of freedom, p= 0.0214
```

Per valutare la sopravvivenza in relazione alla variabile *gruppo* tenendo conto dell'effetto della variabile di confondimento *lev.eta*, si può utilizzare un modello stratificato. In R la funzione per introdurre la stratificazione è *strata*:

```
> survdiff( Surv(mesi, stato) ~ gruppo + strata(lev.eta), data=car)
Call:
survdiff(formula = Surv(mesi, stato) ~ gruppo + strata(lev.eta),
    data = car)

              N Observed Expected (O-E)^2/E (O-E)^2/V
gruppo=FALSE 22       15     7.68      6.97      11.3
gruppo=TRUE  38       11    18.32      2.92      11.3

 Chisq= 11.3  on 1 degrees of freedom, p= 0.000778
```

Come si vede il log-rank test paragona solamente le sopravvivenze classicando i pazienti per *gruppo*, ma tenendo conto della contemporanea classificazione per *lev.eta* (si veda ad esempio [2] pag 581 per i dettagli del test in presenza di stratificazione). Anche in questo caso si conclude che vi è differenza altamente significativa fra le sopravvivenze dei pazienti nelle due classi.

7.3 Modello di Cox di rischio proporzionale

Il test log-rank sulle stime di KM è utile se si vuole paragonare l'effetto che un fattore di rischio ha sulla sopravvivenza. Quando i fattori in studio sono più di uno e si vogliono valutare contemporaneamente, come si fa nel caso di regressione multipla, si può fare uso della regressione di Cox di rischio proporzionale.

Si supponga che vi siano p predittori legati a diversi fattori di rischio. La tecnica di Cox assume che la funzione di rischio sia modellizzabile nel modo seguente:

$$h(t) = h_0(t) \exp(b_1 x_1 + b_2 x_2 + ... + b_p x_p) = h_0(t) \exp(bX) \tag{7.3}$$

dove nell'ultimo passaggio si è fatto uso della notazione matriciale introducendo il vettore b dei coefficienti e la matrice X dei predittori.

Il valore $h_0(t)$ è assunto come livello di rischio di riferimento al tempo t, e rappresenta il valore del rischio per un paziente che abbia tutti i valori dei predittori pari a 0. Con un semplice passaggio algebrico si ha:

$$\log\left(\frac{h(t)}{h_0(t)}\right) = bX \tag{7.4}$$

dove il rapporto $\frac{h(t)}{h_0(t)}$ è detto rapporto di rischio. I coefficienti b sono quindi interpretabili in modo analogo a quanto si fa in ambito di regressione multipla. Ad esempio si consideri il predittore x_i dicotomo che assume valore 1 se il fattore di rischio associato è presente e 0 se è assente. Allora la quantità e^{b_i} può essere interpretata come il rischio relativo istantaneo del "decesso" per un individuo in cui il fattore di rischio è presente rispetto ad uno in cui esso è assente, a parità di valori di tutti gli altri predittori. Analogamente, se il predittore in questione è continuo, il valore e^{b_i} è il rischio relativo istantaneo del "decesso" per un incremento di una unità nel valore di x_i, a parità di valori di tutti gli altri predittori. Dal fatto che questi valori non dipendano dall'istante t in cui viene fatta la valutazione deriva il nome di modello a rischio proporzionale (il rapporto di rischio è lo stesso indipendentemente dal tempo).

Il modello di Cox è detto *semiparametrico* dato che la dipendenza dal tempo del livello di rischio di riferimento $h_0(t)$ non viene specificata in modo parametrico, ma può essere di qualsiasi tipo.

Il fit del modello, che si basa su tecniche di massima verosimiglianza parziale (si veda [54] per i dettagli matematici), si esegue con la chiamata:

```
> mod.cox <- coxph( Surv(mesi, stato) ~ lev.eta + gruppo, data=car)
> mod.cox
Call:
coxph(formula = Surv(mesi, stato) ~ lev.eta + gruppo, data = car)


              coef exp(coef) se(coef)     z       p
lev.etaTRUE  0.598     1.818    0.418  1.43 0.15000
gruppoTRUE  -1.424     0.241    0.428 -3.33 0.00087

Likelihood ratio test=16.3  on 2 df, p=0.000284  n= 60
```

In output si ha la stima dei coefficienti b_i e e^{b_i}, l'errore standard su b_i, il test di Wald per la significatività dei singoli coefficienti. Infine è presentato un test globale sul modello che testa l'ipotesi che almeno uno dei predittore sia necessario. L'interpretazione dei coefficienti è più agevole se si considerano i loro esponenziali. Per quanto riguarda la variabile discreta *lev.eta* si ha che un paziente che abbia valore di tale variabile pari a 1 ha un rischio più elevato di un fattore 1.818 di uno che, a parità di classificazione per la variabile *gruppo* (l'unico altro predittore), abbia *lev.eta* = 0. Al contrario appartenere al gruppo con tumori classificati come *gruppo* = 1 ha l'effetto di ridurre il rischio di "decesso" di un fattore 0.241, cioè del 100 $(1 - 0.241) = 75.9\%$, rispetto a un paziente della stessa classe di età ma con un tumore di gruppo 0.

Nel calcolo del modello si vede che la dipendenza dal predittore *eta* non risulta significativa. Per vedere se è possibile eliminarlo si prova a fittare il modello che non lo contiene:

```
> mod.cox2 <- coxph( Surv(mesi, stato) ~ gruppo, data=car)
> mod.cox2
Call:
coxph(formula = Surv(mesi, stato) ~ gruppo, data = car)


            coef exp(coef) se(coef)     z       p
gruppoTRUE -1.56     0.211    0.413 -3.77 0.00016
```

```
Likelihood ratio test=14.2  on 1 df, p=0.000162  n= 60
```

Si comparano quindi i due modelli:

```
> anova(mod.cox2, mod.cox, test="Chisq")
Analysis of Deviance Table

Model 1: Surv(mesi, stato) ~ gruppo
Model 2: Surv(mesi, stato) ~ lev.eta + gruppo
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        59    178.773
2        58    176.672  1    2.101     0.147
```

Dal test risulta che non si può rifiutare l'ipotesi che i due modelli siano equivalenti. Il modello ristretto che contiene il solo predittore *gruppo* può essere usato per descrivere le sopravvivenze.

7.3.1 Calcolo dei residui

Nel caso di modello di Cox sono stati proposti diversi approcci per la definizione e il calcolo dei residui. Il residuo di più semplice introduzione è quello a martingala, che per il soggetto i-esimo al tempo t è la differenza tra il valore $Y_i(t)$ e il valore $\hat{H}(t)$:

$$r_i = Y_i(t) - \hat{H}(t)$$

Usualmente il suo valore viene valutato al tempo in cui il soggetto esce dall'insieme di rischio. L'impiego dei residui a martingala è suggerito da Fleming e Harrington (1991) per valutare la forma funzionale del modello a rischi proporzionali [59]. Viene suggerito di visualizzarli in vari scatterplot aventi in ordinata il loro valore e in ascissa i valori delle covariate del modello.

I residui a martingala hanno distribuzione fortemente asimmetrica, dato che il loro valore massimo è pari a 1, mentre il valore minimo può essere arbitrariamente negativo. I residui di devianza si ottengono dai residui a martingala con una trasformazione che intende ridurre l'asimmetria:

$$r_i^d = sign(r_i)\ \sqrt{-r_i - Y_i \log(Y_i - r_i)}$$

dove Y_i è il valore della variabile Y valutata al tempo in cui il soggetto esce dall'insieme di rischio. Dal grafico dei residui di devianza si evidenziano i casi che vengono fittati male dal modello.

Un'altra possibilità è quella di calcolare i residui di Schoenfeld. Essi sono definiti per ognuna delle covariate x e per ogni istante che si verifica un decesso come $x_i - \bar{x}(t_i)$, dove $\bar{x}(t_i)$ è la media pesata dei valori x con pesi $\exp b^T x$, calcolata solo sui soggetti ancora a rischio al tempo t_i. Il risultato è quindi una matrice che ha tante righe quanti sono i soggetti che decedono e tante colonne quante sono le covariate. Il calcolo di questi residui è alla base del test di rischio proporzionale di cui si parla in Sec. 7.3.2.

Infine è possibile introdurre i residui di score, che costituiscono una matrice di n righe e colonne pari al numero di predittori. Vengono impiegati per valutare se una osservazione è o meno influente nel calcolo dei valori stimati dei coefficienti di regressione. Per la loro definizione si rimanda a [59, 54].

Tutti questi tipi di residui si possono valutare con la chiamata alla funzione *residuals*.

Esempio

Nel caso del modello di Cox a una covariata fittato in precedenza è possibile ottenere i vari tipi di residui con le chiamate:

```
> r.m <- residuals(mod.cox2, type="martingale")   # residui a martingala
> r.d <- residuals(mod.cox2, type="deviance")     # residui di devianza
> r.s <- residuals(mod.cox2, type="schoen")       # residui di Schoenfeld
> r.score <- residuals(mod.cox2, type="score")    # residui di score
```

Si può verificare che il valore dei residui a martingala è identico al risultato della chiamata diretta:

```
> car$stato - predict(object, type="expected")
```

7.3.2 Test di rischio proporzionale

Il modello di rischio proporzionale ipotizza che i coefficienti del modello non dipendano dal tempo t. Tale ipotesi deve essere verificata prima di dichiarare valido un modello. In R è disponibile la funzione *cox.zph* che testa questa ipotesi per ognuna delle variabili del modello e per il modello nel suo complesso. Per i dettagli sulla matematica alla base del test si rimanda a [54, 29]. Per i modelli fittati in precedenza i test si eseguono con le chiamate:

```
> cox.zph(mod.cox)
                rho   chisq     p
lev.etaTRUE -0.0131 0.00387 0.950
gruppoTRUE  -0.0160 0.00591 0.939
GLOBAL           NA 0.00813 0.996
> cox.zph(mod.cox2)
              rho  chisq     p
gruppoTRUE 0.0212 0.0111 0.916
```

In entrambi i casi l'ipotesi di rischio proporzionale su cui si basano i modelli non può essere respinta.

Capitolo 8

Analisi multivariata: tecniche esplorative

Con il termine analisi multivariata ci si riferisce a un insieme di tecniche con le quali è possibile studiare un insieme di più variabili misurate sullo stesso soggetto. Il tentativo è quello di descrivere in modo conciso la natura e la struttura dei dati. In questo contesto i test d'ipotesi hanno scarsa importanza, a vantaggio di metodologie di statistica descrittiva, spesso di natura grafica.

8.1 Analisi in componenti principali (PCA)

Si supponga che in uno studio su un gruppo di n soggetti vengano misurate p variabili $x_1, \dots x_p$ su ognuno di essi. Ci si chiede se la differenza fra i soggetti può essere equivalentemente descritta da un sottoinsieme di esse, combinate in modo opportuno. In altre parole si tenta di costruire per ogni soggetto delle nuove variabili $y_1, \dots, y_p$ definite dalla relazione matriciale:

$$y = Ax \tag{8.1}$$

dove $y = (y_1, \dots, y_p)$, $x = (x_1, \dots, x_p)^T$ e A una matrice scelta in modo tale che i suoi elementi soddisfino la relazione di somma $\sum_{j=1}^{p} a_{ij}^2 = 1$ (ossia $|a| = 1$). Gli elementi di A, che determina la trasformazione lineare, andranno stimati secondo la seguente logica: dapprima si scelgono i valori di $a_{11}, \dots, a_{1p}$ che massimizzano la varianza di y_1 (quindi y_1 rappresenta meglio di qualunque altra combinazione lineare delle variabili x la differenza generale fra individui), poi si scelgono i valori $a_{21}, \dots, a_{2p}$ in modo tale che y_2 sia scorrelata da y_1 e abbia la massima varianza possibile. Si itera il procedimento fino ad arrivare all'ultima variabile y_p. Il vettore y_1 viene detto prima componente principale, y_2 seconda componente principale, e il piano da loro generato è detto *piano principale*.

La procedura per la determinazione dei valori di A passa attraverso il calcolo degli autovalori λ_i (con $i = 1, \dots, p$) della matrice di correlazione delle variabili x e dei relativi autovettori, che risultano essere le p componenti principali cercate. Il fatto di operare sulla matrice di correlazione significa riscalare implicitamente le variabili x_i in modo tale che abbiano tutte varianza unitaria. Questo procedimento è essenziale quando si hanno variabili misurate su scale molto diverse, dato che in questi casi l'analisi PCA assegnerebbe peso eccessivo alle variabili misurate su scale più grandi a discapito delle altre.

La somma dei p autovalori è esattamente uguale alla variabilità totale delle x:

$$\sum Var(x_i) = \sum Var(y_i) = \sum \lambda_i = p,$$

quindi, considerata la generica componente principale i-esima, si ha che la proporzione di variabilità totale delle x che essa è in grado di spiegare è:

$$\lambda_i / p$$

$x1$	$x2$	$x3$	$x4$
24.4	21.3	30.4	10.7
23.6	20.3	32.2	8.8
20.9	17.5	29.6	8.9
22.7	19.2	28.4	11.0
20.7	17.7	27.7	10.1
25.6	21.8	33.8	9.4
21.4	17.7	30.9	11.6
26.9	23.5	33.9	12.6
24.5	21.2	32.8	10.1
24.1	21.1	33.2	10.0
26.6	22.8	33.8	10.4
20.9	17.6	27.9	10.4
24.6	21.1	34.1	10.4
25.0	22.1	35.6	11.6
24.5	21.1	34.0	10.5

Tabella 8.1: Misurazioni di 15 anfore risalenti alla civiltà cretese.

L'interpretazione delle componenti in termini delle variabili iniziali viene fatta valutando i p elementi di ogni autovettore (in questo contesto le componenti degli autovettori sono comunemente detti loadings). Un alto valore dell'elemento i-esimo di un vettore di loadings (> 0.5) indica che la variabile x_i è associata alla componente in esame. Per meglio chiarire la struttura delle associazioni fra variabili iniziali e componenti principali si ricorre spesso a delle rotazioni della matrice degli autovettori, la più comune è quella denominata *varimax*. Con questa tecnica si cerca una rotazione tale per cui si abbiano, per ogni componente principale, pochi valori di loadings elevati e gli altri prossimi a zero.

Se è possibile spiegare buona parte della variabilità dell'insieme di dati usando solo poche delle componenti principali la tecnica raggiunge lo scopo prefisso. In particolare se le prime due componenti risultano sufficienti sarà possibile proiettare i dati sul piano principale, in modo tale da poter studiare graficamente la presenza di eventuali raggruppamenti o strutture nei dati. Dato che non esiste un metodo universalmente accettato per decidere quante componenti debbano essere incluse, lo sperimentatore ha varie alternative, tra cui: graficare in istogramma le varie componenti e scegliere quelle prima delle quali il grafico (*scree plot*) cambia bruscamente pendenza; scegliere tutte le componenti con $\lambda_i > 1$; scegliere un numero tale di componenti per spiegare una quantità fissata di variabilità, ad esempio l'80%.

Esempio

Quindici anfore risalenti alla civiltà cretese vengono studiate per vedere se è possibile catalogarle secondo quattro diversi parametri di dimensione. Tali parametri sono l'altezza dell'anfora fino alla sommità dell'impugnatura $x1$, l'altezza fino alla base dell'impugnatura $x2$, l'altezza totale dell'anfora $x3$ e la larghezza dell'imboccatura $x4$. Tutte le misure, riportate in Tab. 8.1, sono espresse in cm. Ci si chiede se è possibile ridurre il numero di variabili in studio.

Si supponga di avere inserito le variabili nel data frame X. L'analisi in componenti principali si conduce usando la funzione *princomp*:

```
> prc <-  princomp(X, cor=TRUE)
```

l'opzione $cor = TRUE$ specifica che l'analisi va condotta sulla matrice di correlazione delle variabili in X. L'importanza delle componenti si esamina con la chiamata:

```
> summary(prc)
Importance of components:
                          Comp.1    Comp.2     Comp.3      Comp.4
Standard deviation     1.6977838 0.9467987 0.46008591 0.097074866
```

```
Proportion of Variance 0.7206174 0.2241069 0.05291976 0.002355882
Cumulative Proportion  0.7206174 0.9447244 0.99764412 1.000000000
```

Dall'ultima riga della tabella in output si conclude che le prime due componenti interpretano ben il 94% della variabilità totale.

Una particolare rappresentazione grafica delle misurazioni proiettate sul piano principale (riscalate in modo che ogni componente abbia uguale varianza), sovraimposte alle proiezioni degli assi lungo cui crescono le variabili originarie è il cosidetto *biplot* che si ottiene con la chiamata (Fig. 8.1):

```
> biplot(prc)
```

Le variabili sui due assi sono definite come:

$$y_i^* = \frac{y_i}{\sqrt{n\,\lambda_i}}.$$

Esistono anche definizioni alternative di biplot, per le quali si rimanda a [59]. Dal grafico si osserva che i vettori $x1$, $x2$ sono quasi coincidenti e entrambi simili al vettore $x3$, suggerendo correlazione fra queste variabili. In effetti l'analisi dei valori di loadings:

```
> prc$loadings

Loadings:
   Comp.1 Comp.2 Comp.3 Comp.4
x1 -0.574 -0.103  0.404  0.704
x2 -0.576         0.396 -0.710
x3 -0.531 -0.223 -0.817
x4 -0.237  0.965 -0.109
```

e della rotazione *varimax* delle prime due componenti principali:

```
> varimax(prc$loadings[,1:2])
$loadings

Loadings:
   Comp.1 Comp.2
x1 -0.583
x2 -0.581
x3 -0.569
x4         0.994
```

conferma quanto intuibile graficamente. La prima componente principale può essere vista come una combinazione di $x1$, $x2$ e $x3$ con pesi di egual segno e quasi uguali fra loro, mentre la seconda componente dipende solo dal valore di $x4$. In questo caso l'analisi porta a un risultato che è facilmente interpretabile dato che le prime tre variabili sono legate all'altezza dei manufatti, mentre la quarta alla loro larghezza.

Il problema iniziale di ridurre il numero di variabili con cui rappresentare i dati ha quindi la sua soluzione. La combinazione lineare di $x1$, $x2$ e $x3$ con i pesi ottenuti in fase di analisi è la migliore possibile, tuttavia questa conclusione risente del fatto che tali pesi sono ottimali solo per il set di dati considerato. Una scelta più conservativa e generalizzabile è assumere, per ogni componente, pesi uguali (ovviamente a meno del segno) per ogni variabile che risulti ad essa associata. Così nell'esempio in questione e relativamente alla prima componente principale, si può assumere il vettore di pesi (1,1,1,0).

8.2 Cluster Analysis

Talvolta l'obiettivo di uno studio è vedere se gli n soggetti in esame possano essere suddivisi in un numero (non noto a priori) di gruppi, a seconda dei valori di p variabili $x_1, \ldots, x_p$ che vengono su di

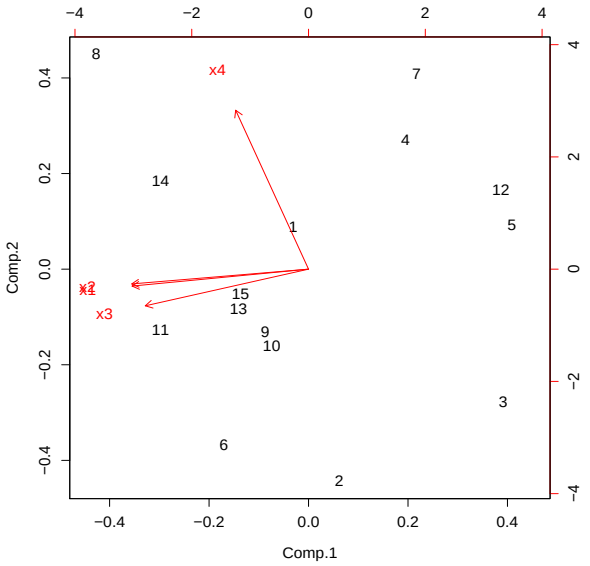

Figura 8.1: Biplot relativo alla misurazione di 15 anfore cretesi. Sono mostrate le proiezioni dei dati sperimentali sul piano principale (dopo oppurtuna riscalatura) e degli assi originari (in rosso) lungo cui crescono i valori delle variabili rilevate sul campione.

essi misurate. La tecnica di raggruppamento dovrà essere tale per cui individui nello stesso gruppo si assomiglino maggiormente (rispetto ai valori di x) di quanto non facciano con individui appartenenti a gruppi diversi.

Le tecniche di raggruppamento cadono in due categorie distinte:

1. *Algoritmi di partizionamento*: l'algoritmo suddivide i dati in k gruppi, con k specificato dall'utente. Generalemente la tecnica viene impiegata per diversi valori del numero di gruppi e viene scelto il raggruppamento che risulta migliore secondo un dato indice di qualità.

2. *Algoritmi gerarchici*: il problema della costruzione dei gruppi viene affrontato in due modi complementari. Uno di questo consiste nel partire con gli n individui separati in n classi. I due individui più "vicini" fra loro vengono quindi combinati in una classe mentre le altre $n-2$ classi constano ancora di un solo individuo. Si continua con il procedimento aggregando le classi più vicine fino a rimanere con una unica classe che contiene tutti i soggetti. Tale algoritmo viene detto di *agglomerative hierarchical clustering*. In alternativa è possibile partire da una unica classe contenente tutti gli individui e procedere all'indietro facendo uscire ad ogni passo la classe più "distante" dal gruppo principale (algoritmo di *divisive hierarchical clustering*). Per i dettagli matematici sulle due tecniche si rimanda a [40]. Lo svantaggio principale di queste tecniche è che, come per qualsiasi algoritmo gerarchico, le scelte fatte a un certo passo non possono essere riconsiderate a un passo successivo; si può quindi ottenere un partizionamento finale che non sia il migliore possibile.

8.2.1 Algoritmi gerarchici: distanze e dissimilarità

Sia X la matrice delle osservazioni di p variabili su n soggetti:

$$X = \begin{pmatrix} x_{11} & \dots & x_{1p} \\ \dots & \dots & \dots \\ x_{n1} & \dots & x_{np} \end{pmatrix}$$

i/j	1	0
1	a	b
0	c	d

Tabella 8.2: Tabella di contingenza per variabili binarie.

Si definisce matrice di dissimilarità D la matrice:

$$D = \begin{pmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \dots & \dots & \dots & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{pmatrix}$$

dove $d(i,j) = d(j,i)$ misura la distanza o dissimilarità fra i soggetto i e j. La costruzione di tale matrice è di fondamentale importanza, sia perché alcune funzioni di raggruppamento lavorano su di essa per generare il loro risultato, sia perché il suo calcolo è indispensabile se non tutti i p predittori sono quantitativi. Il calcolo della dissimilarità fra due soggetti dipende da quali siano i tipi di variabili in gioco (quantitative, ordinali, nominali).

Per variabili quantitative è possibile introdurre una vera e propria metrica; ad esempio si può far uso della distanza euclidea:

$$d(i,j) = \sqrt{\sum_{l=1}^{p} (x_{il} - x_{jl})^2}$$

o della distanza Mahattan:

$$d(i,j) = \sum_{l=1}^{p} |(x_{il} - x_{jl})| .$$

Ovviamente le scelte fatte si riflettono sul risultato finale, con esiti spesso molto diversi l'uno dall'altro. Si noti anche che le variabili con maggiore dispersione sono quelle che maggiormente influenzano il raggruppamento: per ovviare a questo problema, se si ritiene che le variabili abbiono tutte uguale rilevanza, è possibile lavorare sui dati standardizzati.

Per variabili ordinali si procede solitamente sostituendo alla variabile x_{il} il suo rango $r_{il} \in 1, \dots, R_l$. Si effettua quindi un cambiamento di scala definendo la variabile:

$$z_{il} = \frac{r_{il} - 1}{R_l - 1}$$

che ha range $[0, 1]$. Si utilizza quindi la procedura descritta precedentemente per variabili quantitative.

Per variabili nominali che possono assumere M valori distinti, sia N_d il numero di variabili che assumono valori diversi fra i soggetti i e j e N_{tot} il numero totale di variabili. Si definisce la dissimilarità fra i soggetti i e j (*simple matching coefficient*):

$$d(i,j) = \frac{N_d}{N_{tot}}.$$

Per variabili binarie simmetriche (variabili codificabili come 0 e 1 in cui gli stati hanno la medesima importanza, come maschio/femmina, animale/vegetale, ecc.) si considera la tabella di contingenza 8.2.

Si ha:

$$d(i,j) = \frac{b + c}{a + b + c + d}.$$

Infine, per variabili binarie asimmetriche (in cui uno dei due stati è più importante dell'altro; casi tipici sono variabili in cui il valore 1 indica la presenza di una proprietà - tipicamente rara - e il valore 0 la sua assenza, per cui due soggetti che presentano entrambi valore 1 per tale variabile sono ritenuti

"più simili" di due che presentano entrambi valore 0) si definisce la dissimilarità tra i soggetti i e j come (vedi Tab. 8.2):

$$d(i,j) = \frac{b+c}{a+b+c}.$$

La costruzione di una matrice di dissimilarità in R è possibile mediante l'uso di due differenti funzioni della libreria *cluster*: *dist*, che opera solo su variabili quantitative e *daisy*, più versatile, che tratta anche set di dati contenenti variabili categoriali. In particolare, quando fra i dati compaiono variabili di differenti tipi, la funzione *daisy* calcola la dissimilarità tra i soggetti i e j secondo la formula seguente:

$$d(i,j) = \frac{\sum_{l=1}^{p} \delta_{ij}^{(l)}\, d_{ij}^{(l)}}{\sum_{l=1}^{p} \delta_{ij}^{(l)}}$$

dove $\delta_{ij}^{(l)} = 0$ se x_{il} o x_{jl} sono mancanti, $\delta_{ij}^{(l)} = 0$ se $x_{il} = x_{jl}$ e la variabile l è binaria simmetrica, $\delta_{ij}^{(l)} = 1$ altrimenti. $d_{ij}^{(l)}$ è il contributo della variabile l e dipende dal suo tipo:

1. Se l è binaria o nominale $d_{ij}^{(l)} = 0$ se $x_{il} = x_{jl}$ e $d_{ij}^{(l)} = 1$ altrimenti.
2. Se l è quantitativa:
$$d_{ij}^{(l)} = \frac{|x_{il} - x_{jl}|}{\max_h x_{hl} - \min_h x_{hl}}$$
3. Se l è ordinale si calcolano i ranghi r_{il} e z_{il} come detto sopra e si trattano le variabili z_{il} come quantitative.

Esempio

Per verificare il funzionamento della funzione *daisy* si può lavorare su un dataset standard della libreria *cluster*, di nome *flower*. I dati si riferiscono a 18 specie di fiori, su cui vengono rilevati i valori di 8 variabili di tipi diversi:

- V1: variabile binaria, che indica se la pianta può essere lasciata all'aperto durante le gelate.
- V2: variabile binaria, che indica se la pianta necessita di stare all'ombra.
- V3: variabile binaria (asimmetrica), che distingue fra piante con tuberi e senza. Si considera la variabile come asimmetrica dato che due piante che non hanno tuberi (V3 = 0) possono essere in realtà molto diverse fra loro.
- V4: variabile nominale, che indica il colore dei fiori.
- V5: variabile ordinale, che indica le caratteristiche del suolo dove cresce la pianta (1 = secco, 2 = normale, 3 = umido).
- V6: variabile ordinale, che riporta la valutazione di preferenza che la pianta ha ricevuto dal compilatore del dataset.
- V7: variabile quantitativa, che indica l'altezza in cm della pianta.
- V8: variabile quantitativa, che indica la distanza a cui vanno disposte le piante fra loro.

La funzione *daisy* è in grado di leggere i tipi delle variabili in un dataset e di utilizzare le corrette definizioni di distanze. L'unica necessità è quella di specificare quali variabili binarie trattare come asimmetriche. Per comodità di lettura degli output nel corso dell'esempio si utilizzano solo le prime 6 righe del dataset:

```
> library(cluster)
> data(flower)
> flow <- flower[1:6, ]

> flow
  V1 V2 V3 V4 V5 V6  V7 V8
1  0  1  1  4  3 15  25 15
2  1  0  0  2  1  3 150 50
3  0  1  0  3  3  1 150 50
4  0  0  1  4  2 16 125 50
5  0  1  0  5  2  2  20 15
6  0  1  0  4  3 12  50 40
```

La matrice di dissimilarità si ottiene con la chiamata:

```
> daisy(flow, type=list(asymm=c(3)))
Dissimilarities :
          1         2         3         4         5
2 0.9701923
3 0.6118590 0.5904762
4 0.4169872 0.5698718 0.5865385
5 0.4256410 0.7952381 0.5095238 0.7176282
6 0.2633242 0.8078493 0.3983255 0.4536630 0.4445317

Metric :  mixed ;  Types = N, N, A, N, O, O, I, I
Number of objects : 6
```

La funzione accetta come primo argomento il dataframe su cui operare. Altri argomenti sono opzionali; in questo caso si usa l'argomento *type* che deve essere una lista contenente la specificazione di come trattare le variabili. Nell'esempio in questione si forza l'algoritmo a trattare la variabile 3 come variabile binaria asimmetricha (mediante la parola chiave *asymm*).

Se le variabili sono tutte quantitative è possibile utilizzare l'argomento *metric* (che può assumere i valori "euclidean" e "manhattan") per scegliere la metrica con cui misurare le distanze.

8.2.2 Algoritmi gerarchici: dendrogrammi

Una volta definita la distnza tra due soggetti, resta ancora da specificare come si misuri la "vicinanza" fra classi di soggetti. Anche in questo caso esistono svariate possibilità, le tre più comuni sono:

- Complete link: la distanza fra due classi viene definita come il massimo delle distanze fra coppie di individui nelle due classi.
- Single link: la distanza fra due classi viene definita come il minimo delle distanze fra coppie di individui nelle due classi.
- Average link: la distanza fra due classi viene definita come la media delle distanze fra coppie di individui nelle due classi.

Una tecnica più recente fa uso della distanza di Ward, basata sul concetto di inerzia (si veda [31] per i dettagli matematici).

Solitamente il risultato di una cluster analysis è un grafico ad albero rovesciato, detto dendrogramma. Sull'asse verticale di tale grafico sono rappresentate le altezze a cui si congiungono i vari rami. Più un nodo che unisce due gruppi è basso, più tali gruppi sono simili fra loro. La validità di un cluster viene talvolta valutata facendo uso del cosidetto coefficiente di agglomerazione AC (agglomerative coefficient) che si calcola nel modo seguente:

- si considera l'altezza h_i a cui ciascuno degli n soggetti entra nel dendrogramma;

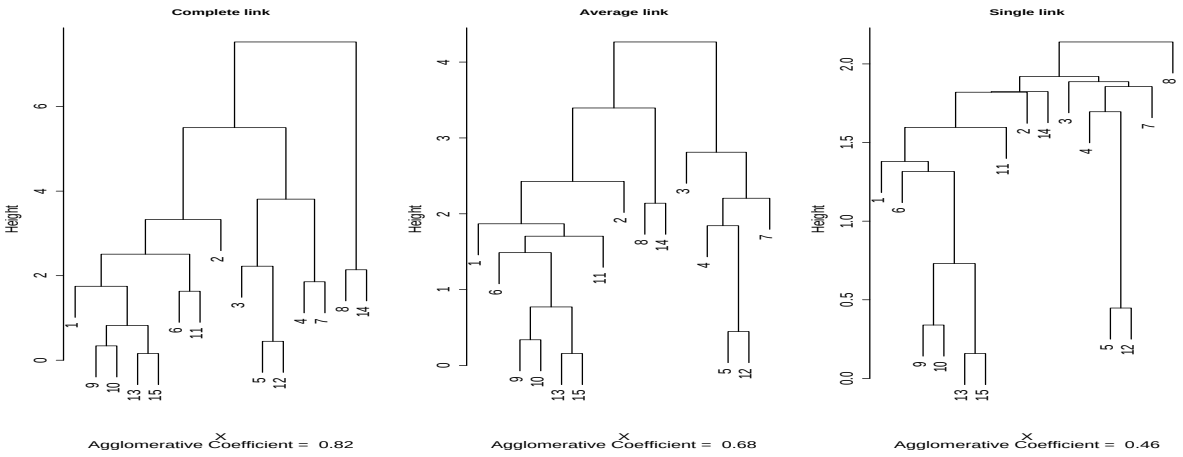

Figura 8.2: Dendrogrammi ottenuti con un algoritmo *agglomerative hierarchical clustering* per i dati dell'Esempio 8.1.

- si considera l'altezza massima del dendrogramma, cioè la quota h in cui tutti i punti convergono in un solo gruppo;
- si calcola infine il coefficiente di agglomerazione:

$$AC = \frac{1}{n}\sum_{i=1}^{n}(1 - \frac{h_i}{h}).$$

La quantità AC è compresa tra 0 e 1 e assume valori elevati per dendrogrammi in cui si ha una forte clusterizzazione, cioè in cui i soggetti tendono a entrare a basse quote nel grafico.

In R le funzioni per eseguire una analisi di questo tipo sono raccolte nella libreria standard *cluster*.

Esempio

Si vuole vedere se i dati dell'Esempio 8.1 (tutti quantitativi) suggeriscono un qualche raggruppamento delle anfore a seconda dei valori delle 4 variabili in esame.

La funzione con cui R esegue una cluster analysis mediante algoritmo *agglomerative hierarchical clustering* è *agnes* (AGglomerative NESting):

```
> library(cluster)
> agn <- agnes(X, method="complete", stand=TRUE)
> plot(agn, main="Complete link", which=2)
```

oltre al data frame su cui eseguire l'analisi si specifica il tipo di funzione di link con l'opzione *method* (che può assumere i valori "complete", "average", "single", "ward" e "weighted") e il fatto che l'analisi deve essere condotta sulle variabili standardizzate ($stand = TRUE$).

I tre dendrogrammi che si ottengono usando le funzioni di link descritte in Sec. 8.2 sono in Fig. 8.2. Mentre alcune caratteristiche risultano indipendenti dal tipo di funzione di link scelta (ad esempio le coppie 9-10, 13-15, 5-12 sono molto simili in tutti e tre i grafici), altre cambiano in modo anche drastico (ad esempio si veda la classificazione della coppia 8-14).

Come detto in precedenza l'analisi può anche essere condotta utilizzando un algoritmo *divisive hierarchical clustering*. Il grande vantaggio è che le prime scelte di suddivisione operate dall'algoritmo (ovvero quelle che si ripercuotono su tutta la costruzione del dendrogramma) sono in questo caso a livello più alto (si cercano per primi i gruppi fondamentali, poi si passa a suddividerli in sotto gruppi). Si ha quindi minor possibilità che una cattiva scelta agglomerativa iniziale impedisca la rilevazione

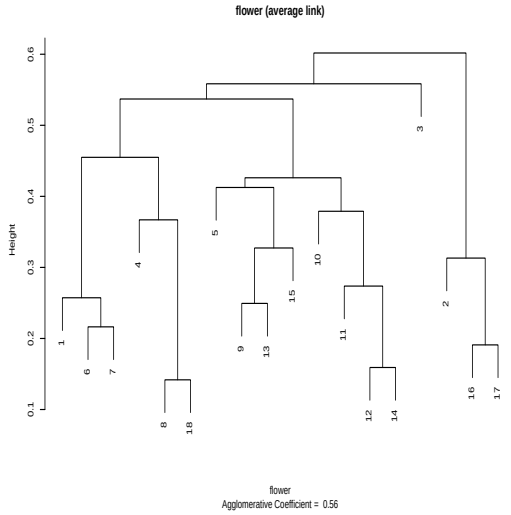

Figura 8.3: Dendrogramma ottenuto con un algoritmo *agglomerative hierarchical clustering* per i dati della classificazione di 18 fiori con 8 diverse variabili.

delle "vere" strutture macroscopiche insite nei dati. Il rovescio della medaglia è che questo approccio richiede di trovare la migliore partizione di un gruppo in due sottogruppi. Quando si ha a che fare con un numero elevato di soggetti e variabili il problema diventa computazionalmente troppo oneroso per essere affrontato nella sua completezza e si deve ricorrere ad algoritmi che ottengano un buon compromesso tra efficienza e prestazioni. In R è possibile farlo tramite la funzione *diana* (DIvisive ANAlysis clustering):

```
> dia <- diana(X, stand=TRUE)
> plot(dia, main="DIvisive ANAlysis clustering", which=2)
```

In questo caso, il dendrogramma che si ottiene è molto simile a quello ottenuto usando la funzione average link con algoritmo di tipo *agglomerative hierarchical clustering*.

Come visto in Sec. 8.1, l'uso delle prime due componenti principali (legate rispettivamente ad altezza e larghezza delle anfore), spiegava circa il 94% della variabilità dei dati. In questo caso lo studio di eventuali raggruppamenti fra i soggetti può essere condotto più opportunamente sulla proiezione dei dati sul piano principale, in Fig. 8.1. Confrontando quanto si vede in questa figura con il primo dendrogramma di Fig. 8.2, da cui si evince la presenza di alcuni gruppi (ad esempio 5-12, 4-7, 8-14), si osserva che la stessa struttura appare all'analisi in componenti principali. Le anfore 8-14 sono caratterizzate da elevati valori di entrambe le componenti principali, quindi sono manufatti particolarmente grandi, le anfore 4-7 hanno tutte altezza medio-bassa e larghezza elevata, mentre le anfore 5-12 sono basse.

Esempio

Nel caso del dataset *flower*, contenente variabili di tipi diversi, si può far uso della funzione *agnes*, chiamata non sul dataset originario, ma sulla corrispondente matrice di dissimilarità:

```
> ds.fl <- daisy(flower, type=list(asymm=c(3)))   # matrice di dissimilarita'
> ag <- agnes(ds.fl)     # con average link
> plot(ag, main="flower (average link)", xlab="flower", which=2)
```

Il risultato dell'analisi è in Fig. 8.3.

Silhouette media	interpretazione
0.71-1.0	Il raggruppamento evidenziato è molto valido
0.51-0.70	Il raggruppamento evidenziato è ragionevole
0.26-0.50	Il raggruppamento evidenziato è debole e può essere fasullo
≤ 0.25	Non è stata evidenziata nessuna struttura

Tabella 8.3: Interpretazione schematica dei valori di silhouette media, ottenuti da un silhouette plot.

8.2.3 Silhouette plot

Una volta ottenuta un raggruppamento dei dati si presenta il problema di decidere quale sia il numero "appropriato" di cluster che si possono inferire dall'analisi e quanto valido sia tale raggruppamento. Una delle tecniche proposte per rispondere a queste domande è la tecnica delle silhouette.

Si supponga che lo sperimentatore, analizzato un dendrogramma, ritenga di poter evidenziare un certo numero k di sotto-raggruppamenti, ottenuti tagliando il dendrogramma a una certa altezza con una linea orizzontale. La tecnica diagnostica in esame produce quindi una silhouette per ogni cluster evidenziato e le rappresenta tutte insieme in un grafico (silhouette plot) da cui è possibile stabilire la bontà dei singoli raggruppamenti e della struttura globale.

Più in dettaglio, per ogni osservazione i si definisce $a(i)$ come la media delle dissimilarità tra l'oggetto i e tutti gli altri rappresentati del cluster. Per ogni altro cluster C sia $d(i, C)$ la media delle dissimilarità tra l'oggetto i e gli oggetti del cluster C. Sia $b(i)$ il minore fra i valori $d(i, C)$. Si definisce quindi la larghezza di silhouette $s(i)$:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \in [-1, 1].$$

Se un cluster contiene un solo elemento allora si pone per definizione $s(i) = 0$.

Osservazioni con un alto valore di $s(i)$ sono ben raggruppate. Un valore di $s(i)$ prossimo a 0 indica che l'osservazione giace a metà strada fra il cluster di apparteneza e quello più vicino. Un valore negativo di $s(i)$ implica che l'osservazione è probabilmente raggruppata nel cluster sbagliato. Il silhouette plot riporta i valori di silhouette media di ognuno dei gruppi identificati e anche la silhouette media globale. Questo indice può essere utilizzato a fini diagnostici, come mostrato schematicamente in Tab. 8.3.

I silhouette plot sono molto usati per decidere in quanti gruppi sia appropriato dividere gli oggetti in studio. Per far ciò si ricorre alla costruzione di svariati silhouette plot, cambiando il numero di gruppi in cui suddividere gli oggetti. Il grafico con il miglior indice di silhouette media sarà quello corrispondente al raggruppamento migliore.

Esempio

Nal caso del dataset *flower* si vuole ottenere la divisione in gruppi più appropriata risultante dall'analisi del dendrogramma in Fig. 8.3. La funzione da impiegare è *silhouette*, definita nella libreria *cluster*. Lavorando su un dendrogramma essa accetta due argomenti: il primo è la classificazione dei soggetti nei diversi gruppi ottenuti mediante taglio dell'albero a una certa quota, il secondo la matrice di dissimilarità da utilizzare. La classificazione dei soggetti mediante taglio si ottiene dalla funzione *cutree* che accetta a sua volta due argomenti: l'albero su cui lavorare e il numero di gruppi da ottenere.

Per trovare il numero ottimale di gruppi è opportuno chiamare ciclicamente la funzione *silhouette*, con il comando:

```
> sil <- NULL
> for (i in 2:9) sil <- c(sil, list(silhouette(cutree(ag, i), ds.fl)))
```

In questo caso si cerca il numero di gruppi ottimale, partendo da un minimo di 2 fino a un massimo di 9. La funzione *list* serve a preservare l'integrità dell'informazione dei vari studi di silhouette. Gli indici di silhouette media negli 8 casi esaminati si ottengono con le chiamate:

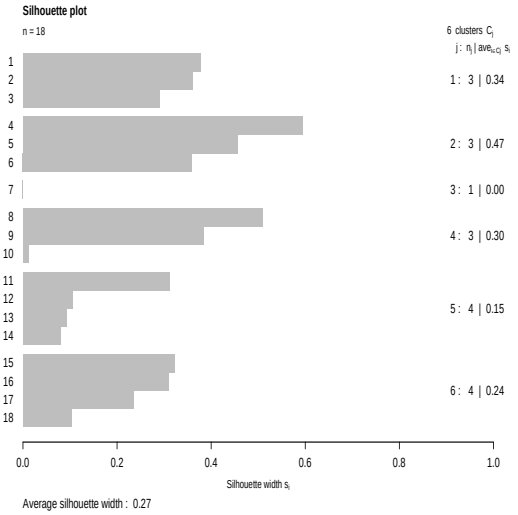

Figura 8.4: Silhouette plot per la classificazione di 18 fiori in 6 distinti gruppi.

```
> SM <- NULL; for(i in 1:8) SM <- c(SM, summary(sil[[i]])$avg.width)
> SM
[1] 0.2351306 0.1773975 0.1868441 0.2685201 0.2721285 0.2536208 0.2486248
[8] 0.2496006
```

Il miglior raggruppamento ha silhouette media pari a 0.272 in corrispondenza del quinto valore (6 gruppi). Il valore dell'indice di silhouette è piuttosto basso e porta a conclude che il raggruppamento trovato è probabilmente artificiale. Il silhouette plot corrispondente alla classificazione in 6 gruppi (Fig. 8.4) si ottiene con la chiamata:

```
> plot(sil[[5]], main="Silhouette plot")
```

8.2.4 Confronto di matrici di dissimilarità

Si supponga di rilevare su un campione di soggetti sperimentali i valori di due classi distinte di varibili. Si pensi ad esempio al caso di pazienti di cui vengano registrati sia dei dati clinici sia dei dati di espressioni geniche, o a siti che vengano classificati sia sulla base della loro distanza geografica sia sulla base di differenze ecologiche. In questi casi è spesso di interesse verificare se i soggetti che tendono a essere distanti relativamente alla prima misurazione tendano ad esserlo altrettanto relativamente alla seconda.

Anziché confrontare tra loro i cluster, si opera sulle matrici di dissimilarità mediante un test dovuto a Mantel. Tale test verifica l'ipotesi nulla di assenza di correlazione tra le matrici. La statistica di Mantel è definita sul campione come:

$$r = \frac{1}{N-1} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{x_{ij} - \bar{x}}{s_x} \frac{y_{ij} - \bar{y}}{s_y}$$

dove x_{ij} e y_{ij} sono gli elementi delle due matrici di dissimilarità, $\bar{x}$ e $\bar{y}$ i valori medi degli elementi delle due matrici, s_x e s_y del deviazioni standard degli elementi delle due matrici, N il numero di elementi non nulli delle matrici pari a $n(n-1)/2$ con n il numero di soggetti sperimentali considerati. In maniera più compatta il valore di r si scrive in termini del prodotto scalare tra i due vettori z_x e z_y ottenuti concatenando gli elementi delle due matrici e standardizzando i loro valori:

$$r = \frac{1}{N-1} < z_x, z_y > .$$

Dato che gli elementi delle matrici di dissimilarità non sono fra loro indipendenti, per valutare la significatività del test si ricorre a una procedura Monte Carlo: vengono permutate simultaneamente in maniera casuale le righe e le colonne della prima delle due matrici e viene ricalcolato il valore della correlazione. La procedura viene ripetuta un consistente numero di volte (da 1000 a 10000) per costruire la distribuzione empirica del coefficiente di correlazione. Disponendo di questa distribuzione è quindi possibile valutare la significatività della correlazione campionaria confrontando tale valore con i percentili simulati.

In R tale test è disponibile in diverse librerie. Si farà uso della funzione *mantel.randtest* della libreria *ade*4 e della funzione *mantel* della libreria *vegan*.

Esempio

Il dataset *yanomama* della libreria *ade*4 riporta le matrici di dissimilarità fra 19 villaggi di indiani Yanomama in base a misure geografiche, genetiche e antropometriche (Spielman, *Differences among Yanomama Indian villages: do the patterns of allele frequencies, anthropometrics and map locations correspond?*, American Journal of Physical Anthropology 39:461-480, 1973). Si vuole verificare se sono tra loro correlate le dissimilarità tra i villaggi ottenute dalle prime due matrici.

L'analisi inizia caricando il dataset e trasformando le matrici numeriche disponibili in oggetti che il programma riconosca come matrici di dissimilarità:

```
> library(ade4)
> data(yanomama)
> gen <- as.dist(yanomama$gen)
> geo <- as.dist(yanomama$geo)
```

La funzione *as.dist* ha il compito di convertire le matrici in input in oggetti di classe *dist* (la classe appropriata per una matrice di dissimilarità).

Il test di Mantel si esegue con la chiamata:

```
> set.seed(100)  # per una analisi ripetibile
> mantel.randtest(geo, gen, 10000)
Monte-Carlo test
Observation: 0.5098684
Call: mantel.randtest(m1 = geo, m2 = gen, nrepet = 10000)
Based on 10000 replicates
Simulated p-value: 0.00089991
```

dove i primi due argomenti sono le matrici di dissimilarità da confrontare e il terzo il numero di ripetizioni Monte Carlo da effettuare per il calcolo del valore P. In output si osserva il valore di correlazione campionaria (*Observation*) e il valore di significatività. Si conclude che la correlazione tra le due matrici è altamente significativa e che le due classificazioni, in base alle distanze geografiche e genetiche, sono fra loro in accordo.

La stessa analisi condotta con la funzione *mantel* della libreria *vegan* porta alle medesime conclusioni:

```
> mantel(geo, gen, permutations=1000)

Mantel statistic based on Pearson's product-moment correlation

Call:
mantel(xdis = geo, ydis = gen, permutations = 1000)

Mantel statistic r: 0.5099
      Significance: 0.001

Empirical upper confidence limits of r:
```

```
  90%   95% 97.5%   99%
0.218 0.279 0.342 0.387

Based on 1000 permutations
```

Si osservi che la funzione *mantel.randtest* è notevolmente più veloce della funzione *mantel* dato che si appoggia su un codice compilato scritto in linguaggio C. A suo vantaggio la funzione *mantel* ha il fatto di poter lavorare anche con correlazioni non parametriche (correlazione di Spearman e di Kendall), il che risolve eventuali casi di correlazione non lineare fra le matrici. Nell'esempio in questione per operare con una correlazione dei ranghi di Spearman la sintassi è:

```
> mantel(geo, gen, method="spearman", permutations=1000)

Mantel statistic based on Spearman's rank correlation rho

Call:
mantel(xdis = geo, ydis = gen, method = "spearman", permutations = 1000)

Mantel statistic r: 0.5361
      Significance: 0.001

Empirical upper confidence limits of r:
  90%   95% 97.5%   99%
0.190 0.242 0.287 0.361

Based on 1000 permutations
```

Il valore della statistica r in questo caso non è altro che il valore della statistica di Mantel che si ottiene sostituendo ai dati il loro rango.

8.2.5 Algoritmi di partizionamento

L'approccio alternativo a quello gerarchico è quello del partizionamento. In questo caso l'utente specifica il numero k di gruppi che devono essere individuati; l'algoritmo individua quindi k oggetti che siano "rappresentativi" dei vari gruppi e forma i gruppi assegnando ogni oggetto al gruppo dell'oggetto rappresentattivo più vicino. La chiave dell'algoritmo è una valida individuazione degli oggetti rappresentativi che dovranno essere posizionati verso il centro del gruppo che definiranno. L'algorimo implementato in R utilizza come oggetti rappresentativi quelli che minimizzano la dissimilarità media dagli altri componenti del gruppo stesso. Tali oggetti sono detti *medoidi* da cui il nome dell'algoritmo: Partitioning Around Medoids o PAM.

Gli algorimi di partizionamento vengono usualmente lanciati per diversi valori di k e viene alla fine scelto il partizionameto che ottiene la miglior silhouette media.

Esempio

Facendo sempre uso del dataset $flower$ si vuole ottenere la divisione in gruppi più appropriata e individuare i vari oggetti rappresentativi. L'analisi si conduce sulla matrice di dissimilarità calcolata come in precedenza e chiamando la funzione *pam* della libreria *cluster*:

```
> ds.fl <- daisy(flower, type=list(asymm=c(3)))   # matrice di dissimilarita'
> pam.4 <- pam(ds.fl, 4)
```

La funzione *pam* accetta un minimo di due argomenti: un datatset (o la corrispondente matrice di dissimilarità) e il numero di gruppi da costruire. Nell'esempio in questione si chiede di partizionare il gruppo di dati in quattro sottogruppi. Per valutare la bontà della suddvisione si costruisce il silhouette plot (Fig. 8.5 a sinistra):

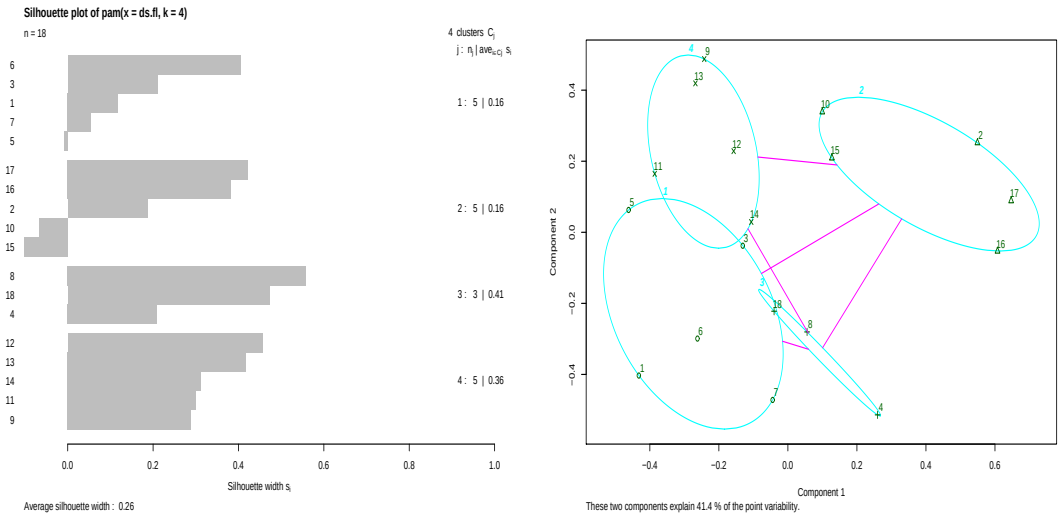

Figura 8.5: Silhouette plot per la classificazione di 18 fiori in 4 gruppi mediante algoritmo di partizionamento *pam*. A destra: *clusplot* relativo alla divisione dei dati in 4 gruppi.

```
> plot(pam.4, which.plots=2)
```

Il confronto del grafico con quelli ottenibili con un numero differente di gruppi porta a concludere che questa è la divisione ottimale.

Le informazioni calcolate dalla funzione *pam* riguardano sia l'individuazione degli oggetti rappresentativi, sia l'assegnazione dei vari oggetti nei gruppi. L'output della funzione può essere esaminato nel modo seguente:

```
>  pam.4
Medoids:
     ID
[1,] "6"  "6"
[2,] "17" "17"
[3,] "8"  "8"
[4,] "12" "12"
Clustering vector:
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
 1  2  1  3  1  1  1  3  4  2  4  4  4  4  2  2  2  3
Objective function:
    build      swap
0.2245250 0.2142037
```

Vengono fornite: la lista dei medoidi (oggetti 6, 17, 8, 12 rispettivamente); il vettore delle appartenenze dei 18 oggetti ai quattro gruppi; alcune informazioni riguardanti i passi dell'algoritmo nel trovare la soluzione ottimale (zona "Objective function"). Ulteriori informazioni sui cluster individuati si ottengono con la chiamata:

```
>  pam.4$clusinfo
     size  max_diss   av_diss  diameter separation
[1,]    5 0.3935574 0.2360096 0.5595705  0.2926062
[2,]    5 0.4032680 0.2324930 0.5178338  0.2908964
[3,]    3 0.2960376 0.1459423 0.4378268  0.3417892
[4,]    5 0.4194036 0.2150654 0.4396650  0.2908964
```

per colonna si hanno: la cardinalità dei gruppi individuati, la dissimilarità massima e media tra il medoide e gli altri componenti del gruppo, il diametro del cluster (dissimilarità massima tra due

oggetti del gruppo) e la separazione dal gruppo più vicino (minimo delle dissimilarità tra un oggetto del cluster e un oggetto al di fuori di esso).

Una rappresentazione grafica dei gruppi e delle loro distanze si può ottenere mediante la chiamata:

```
> plot(pam.4, which.plots=1, label=2, main="")
```

dove l'opzione *label* serve per richiedere l'etichettatura dei punti e dei cluster. I dati vengono proiettati su un piano mediante l'uso delle funzioni *princomp* e *cmdscale*, i gruppi sono rappresentati dalle ellissi e le linee che le congiungono danno un'idea delle distanze tra i cluster. L'output della chiamata è in Fig. 8.5 a destra.

8.3 Scaling multidimensionale

Si supponga di avere un set di n punti e di conoscere solo le distanze (o le dissimilarità) tra di essi. Il problema di interesse è ricostruire le coordinate dei punti, una volta specificato il numero p di dimensioni dello spazio in cui mapparli.

La tecnica tradizionale per affrontare il problema va sotto il nome di scaling multidimensionale (MDS), nota anche come analisi in coordinate principali (PCoA); i due metodi giungono agli stessi risultati facendo uso due approcci algoritmici differenti.

Le applicazioni della tecnica sono molteplici; innanzitutto lo studio di problemi in cui non esiste un sistema naturale di coordinate, come lo studio di similarità fra oggetti, come proteine o volti. In questi casi è possibile quantificare (oggettivamente o soggettivamente) la dissimilarità fra due soggetti e rappresentarla graficamente utilizzando la tecnica MDS. In seconda battuta la tecnica può essere impiegata per ricostruire una struttura spaziale, una volta note solo le distanze tra le varie parti (ad esempio si può ricostruire una configurazione molecolare dalle distanze degli atomi che la compongono).

Come nota finale, nell'ambito delle scienze naturali, ci si riferisce spesso a queste metodologie con il nome di tecniche di ordinamento.

8.3.1 Analisi in coordinate principali

L'analisi in coordinate principali (PCoA) propriamente detta è simile all'analisi in componenti principali, con alcune differenze. Anziché operare sulla matrice di correlazione delle variabili, mediante PCoA si ricercano le somiglianze tra i vari soggetti sperimentali partendo dall'analisi della matrice delle distanze (o di dissimilarità). In altri termini, l'analisi in PCoA è l'analogo dell'analisi PCA in cui si cambia solo la matrice di partenza su cui operare. Senza scendere nel dettaglio, le funzioni per eseguire questo tipo di analisi in R sono implementate in diverse librerie; ad esempio la funzione *pco* della libreria *ecodist* e la funzione *dudi.pco* della libreria *ade*4.

8.3.2 Scaling multidimensionale

L'approccio algoritmico più comune nel campo è quella della scaling multidimensionale. Questo metodo tenta di assegnare ai punti una configurazione p-dimensionale (con p parametro di input) in modo tale che le distanze tra i punti rispecchino nel modo migliore quelle definite dalla matrice di distanza. L'algoritmo MDS minimizza una funzione di stress, che può essere ad esempio la somma dei quadrati delle differenze tra le distanze in input e quelle ottenute dalle coordinate dei punti in output. La configurazione finale proposta dall'algoritmo è definita sempre a meno di rotazioni o riflessioni.

Altre forme di scaling multidimensionale non metriche (NMDS) sono state proposte negli anni. In R sono disponibili le funzioni *cmdscale* (scaling multidimensionale classico), *sammon* e $isoMDS$ (entrambe definite nella libreria $MASS$) che implementano differenti definizioni delle funzioni di stress da minimizzare (si veda [59] per le definizioni matematiche di tali funzioni).

Esempio

Il dataset standard *eurodist* contiene le distanze kilometriche fra 21 città europee. Mediante la tecnica MDS è possibile mappare queste città in uno spazio bidimensionale (sullo stile di una carta

geografica). La mappatura risentirà ovviamente del fatto che in realtà le città sono disposte in uno spazio tridimensionale quale la superficie terrestre.

Si inizia l'analisi usando la funzione *cmdscale*:

```
> library(MASS)
> data(eurodist)                # carica il dataset eurodist
> loc <- cmdscale(eurodist, 2)
```

La funzione *cmdscale* accetta un minimo di due argomenti: la matrice di distanze su cui operare e il numero di dimensioni in cui mappare i punti. In output si ottengono le coordinate dei punti generati:

```
> loc
                      [,1]        [,2]
Athens           2290.274680  1798.80293
Barcelona        -825.382790   546.81148
[...]
Vienna            911.230500   205.93020
```

Si noti che le coordinate vengono centrate, in modo tale che la media di ogni colonna sia nulla. I punti così ottenuti possono essere rappresentati in un piano per evidenziare eventuali configurazioni o raggruppamenti.

La stessa analisi può essere condotta mediante scaling non metrico, facendo uso della funzione $isoMDS$:

```
> loc.iso <- isoMDS(eurodist, k=2)
initial  value 7.505733
final  value 7.505688
converged
```

La funzione si appoggia su un processo iterativo per trovare la configurazione ottimale, la quale dipende dal punto di partenza scelto. Le informazioni presentate sulle linee "inital" e "final" sono il valore (in percentuale) della funzione di stress sulla configurazione di punti iniziale (ottenuta mediante una chiamata a *cmdscale*) e su quella finale come risultante dal processo di ottimizzazione. Si nota che i valori iniziale e finale sono molto simili; in questo caso le due funzioni di scaling ottengono una configurazione di punti quasi identica. L'output della funzione $isoMDS$ è:

```
> loc.iso
$points
                      [,1]        [,2]
Athens           2290.272645  1798.80178
Barcelona        -825.382640   546.81213
[...]
Vienna            911.232682   205.93089

$stress
[1] 7.505688
```

oltre alle coordinate delle città viene visualizzato anche il valore della funzione di stress.

Un grafico standard per valutare la bontà della mappatura è il grafico di Shepard in cui le distanze tra i punti calcolati sono graficate contro le distanze osservate. Una versione particolarmente gradevole di tale grafico è realizzabile dopo l'installazione della libreria *vegan*, mediante la funzione *stressplot*:

```
> library(vegan)
> stressplot(loc.iso, eurodist)
```

Il risultato di questa chiamata è in Fig. 8.6. La linea a scala rossa è il fit dei punti mediante funzione a scala. Le due statistiche di correlazione mostrate sono basate sul valore della funzione di stress S

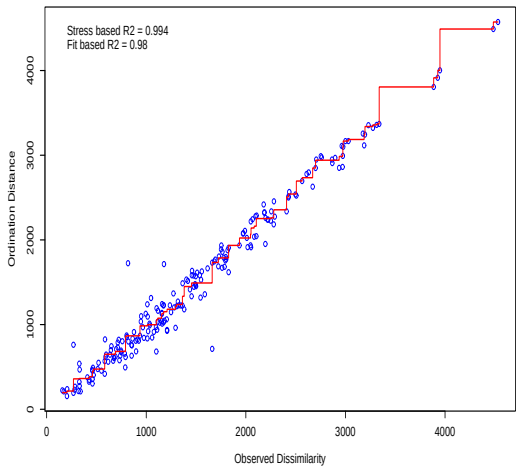

Figura 8.6: Shepard plot per la mappatura bidimensionale di 21 città europee.

(stress based, definito come $R^2 = 1 - S^2$) e sulla correlazione tra la funzione a scala (valori fittati, calcolati internamente mediante chiamata alla funzione *Shepard*) e i punti osservati (fit based).

Come ultima nota, si deve prestare attenzione a non utilizzare come statistica di correlazione quella tra le distanze osservate e quelle calcolate dall'algoritmo. Dato che la tecnica MDS non è lineare questa statistica potrebbe portare a preferire una cattiva mappatura lineare a una migliore mappatura non lineare. □

In alcuni casi, come nell'esempio precedente, la scelta del numero di dimensioni in cui mappare i punti è abbastanza semplice. In altri casi invece non esiste una dimensione "naturale" per lo spazio di mappatura. Per affrontare questo problema e scegliere il numero di dimensioni ottimali taluni autori suggersicono di ricorrere a uno scree plot in cui viene graficato il valore della funzione di stress al variare del numero di dimensioni dello spazio di mappatura. Si sceglie quindi la dimensione spaziale per cui il grafico presenta un cambio di pendenza, in modo tale che l'aggiunta di nuove dimensioni non porti a un vantaggio apprezzabile.

8.4 Analisi della corrispondenza (CA)

L'analisi della corrispondenza è una tecnica esplorativa che consente l'analisi di tabelle di contingenza a due vie. Lo scopo dell'analisi è quello di mappare in uno spazio di bassa dimensione (tipicamente bidimensionale) le righe e le colonne della tabella in modo tale da rispettare le distanze originarie ed evidenziare quali siano le righe (o colonne) simili tra loro e se sia possibile inferire un qualche andamento. Il metodo è particolarmente utile quando si analizza una tabella di grandi dimensioni e si vuole cercare di riassumere i risultati in un grafico di immediata interpretabilità.

Questa tecnica, sviluppata inizialmente in Francia da Jean-Paul Benzérci negli anni '60-'70, è nota in letteratura anche sotto il nome di optimal scaling o reciprocal averaging. In campo ecologico – insieme alle tecniche di scaling multidimensionale – questi metodi vanno sotto il nome di tecniche di ordinamento. Tale nome, la cui introduzione è attribuita a Goodall (1954), si riferisce al fatto che esse vennero sviluppate per l'analisi di tabelle di contingenza in cui differenti siti di campionamento (fattore di riga) erano confrontati fra loro per l'abbondanza di varie specie (fattore di colonna). Questi metodi permettono di "ordinare" i siti di campionamento lungo degli assi, basandosi sulla composizione delle specie che li caratterizzano, e di evidenziare similarità e differenze tra di loro. Per una dettagliata trattazione dell'argomento e dei vari problemi statistici in campo ecologico si rimanda a [41].

Il punto di partenza della tecnica è la definizione della distanza tra righe (o colonne). Si consideri una tabella di contingenza di r righe e c colonne e sia x_{ij} la generica frequenza di posto (i, j); sia n la somma totale delle frequenza in tabella. Nel seguito si supporrà che sia $r \geq c$; nel caso che questa condizione non sia verificata è sempre possibile trasporre la matrice, dato che righe e colonne hanno un ruolo interscambiabile. Si definisce la distanza χ^2 tra la riga i-esima e quella k-esima come:

$$d(i,k) = \sqrt{n}\sqrt{\sum_{j=1}^{c} \frac{1}{x_{+j}} \left(\frac{x_{ij}}{x_{i+}} - \frac{x_{kj}}{x_{k+}} \right)^2} \tag{8.2}$$

dove x_{i+} e x_{k+} sono i totali marginali delle due righe considerate e x_{+j} i totali marginali delle colonne.

Il procedimento passa per il calcolo della matrice Q dei contributi di ogni cella al valore della variabile χ^2 per il test di indipendenza. Si definisce inerzia la somma dei quadrati degli elementi di Q.

A partire dalla matrice Q si conduce una decomposizione ai valori singolari:

$$Q = \hat{U}\ W\ U^T$$

con W matrice diagonale di dimensioni $c \times c$, $\hat{U}$ e U matrici colonna ortonormali di dimensioni $r \times c$ e $c \times c$ rispettivamente. I $c - 1$ valori di W non negativi sono i valori singolari della matrice Q. Si considera la matrice:

$$Q^T Q = UW\hat{U}^T\ \hat{U}WU^T = UWIWU^T = UWWU^T$$

dove si è tenuto conto che, essendo $\hat{U}$ ortonormale, vale la relazione $\hat{U}^T = \hat{U}^{-1}$. Posto $\Lambda = WW$ e tenuto conto della ortonormalità di U si può scrivere:

$$Q^T Q = U\Lambda U^{-1}$$

da cui si evince che la matrice diagonale Λ contiene gli autovalori di $Q^T Q$ e che la matrice U contiene gli autovettori. I valori contenuti nei vettori colonna di U si dicono loadings delle colonne della tabella di contingenza in studio. Per analogia, lavorando sulla matrice QQ^T, si ha che la matrice $\hat{U}$ contiene i valori di loadings delle righe della matrice di partenza.

Le matrici U e $\hat{U}$ possono essere usate per plottare le posizioni di righe e colonne. Se si vuole ottenere un grafico congiunto è possibile operare diverse operazioni di scalatura. Per prima cosa si calcolano le matrici diagonali D_r e D_c che contengono sulla diagonale principale i profili di colonna e di riga rispettivamente:

$$D_r(j,j) = \frac{\sum_{i=1}^{r} x_{ij}}{n} \qquad , \qquad D_c(i,i) = \frac{\sum_{j=1}^{c} x_{ij}}{n}$$

Si calcolano poi le matrici:

$$V = D_r^{-1/2}\ U \qquad , \qquad \hat{V} = D_c^{-1/2}\ \hat{U}$$

di dimensioni $c \times c$ e $r \times c$ rispettivamente. La parte rilevante di tali matrici, una volta scartato l'autovalore nullo è di dimensioni $c \times (c-1)$ per V e $r \times (c-1)$ per $\hat{V}$.

La matrice F che da la posizione delle righe nello spazio dell'analisi della corrispondenza si ottiene a partire dalla matrice V che da le posizioni delle colonne in detto spazio:

$$F = D_c^{-1}\ Q\ V = \hat{V}\ \Lambda^{1/2}$$

Si ha che la distanza euclidea fra le righe della matrice F corrisponde alla distanza χ^2 tra le righe della tabella di contingenza originaria. Analogamente la matrice $\hat{F}$ che da la posizione delle colonne nello spazio dell'analisi della corrispondenza si ottiene dalla matrice $\hat{V}$ che da la posizione delle righe in tale spazio:

$$\hat{F} = D_r^{-1}\ Q^T\ \hat{V} = V\ \Lambda^{1/2}$$

In questo caso la distanza euclidea fra le righe di $\hat{F}$ corrisponde alla distanza χ^2 tra le colonne della tabella di partenza. Con le procedure di scalatura sopra descritte le matrici V e F formano una coppia tale per cui le righe (date da F) sono poste ai centroidi (o baricentri) delle colonne, date dalla matrice V. Analogamente le matrici $\hat{F}$ e $\hat{V}$ sono tali per cui le colonne (date da $\hat{F}$) sono poste ai baricentri delle righe in $\hat{V}$.

Le matrici V e $\hat{V}$ sono legate tra loro dalla relazione:

$$\hat{V}\Lambda^{1/2} = D_c^{-1/2}\, Q\, D_r^{1/2} V$$

ossia l'ordinamento delle righe è legato – lungo l'i-esimo asse principale – dal valore $\sqrt{\lambda_i}$ a quello delle colonne. Tale valore è assunto come una misura di correlazione tra i due ordinamenti, cioè di quanto facilmente è possibile risalire all'ordinamento delle righe dato quello delle colonne (o viceversa).

Disponendo delle matrici V, $\hat{V}$, F e $\hat{F}$ è possibile realizzare una serie di grafici differenti, come nell'esempio seguente.

Esempio

Si valuta la presenza di una particolare specie in tre habitat, differenti per condizioni climatiche. Ogni habitat viene diviso in zone di campionamento e si conta il numero di zone in cui la specie è assente (livello 0), moderatamente presente (livello +) o abbondante (livello ++). Si ottiene la tabella seguente:

```
> M <- matrix( c(10,10,20, 10,15,10, 15,5,5), nrow=3, byrow=TRUE)
> rownames(M) <- c("Cold", "Medium", "Warm")
> colnames(M) <- c("0", "+", "++")
> M
        0  + ++
Cold   10 10 20
Medium 10 15 10
Warm   15  5  5
```

In R è possibile condurre l'analisi mediante la funzione *corresp* della libreria $MASS$:

```
> library(MASS)
> cp <- corresp(M, nf=2)
> cp
First canonical correlation(s): 0.3100532 0.2023408

 Row scores:
             [,1]       [,2]
Cold   -0.8489561  0.8827648
Medium -0.2204587 -1.3448200
Warm    1.6669718  0.4703244

 Column scores:
         [,1]       [,2]
0   1.3187071  0.3437359
+  -0.3721491 -1.4814987
++ -0.9997222  0.9261201
```

La funzione accetta in input la tabella da analizzare e il numero di componenti da calcolare (il valore di default è 1, ma per poter produrre un grafico bidimensionale è necessario richiedere 2 componenti). In output si hanno le radici dei 2 autovalori calcolati e le due matrici $\hat{V}$ e V.

Le matrici F e $\hat{F}$ si ottengono, a partire dalle loro definizioni, con le chiamate:

```
> F <- cp$rscore %*% diag(cp$cor)
```

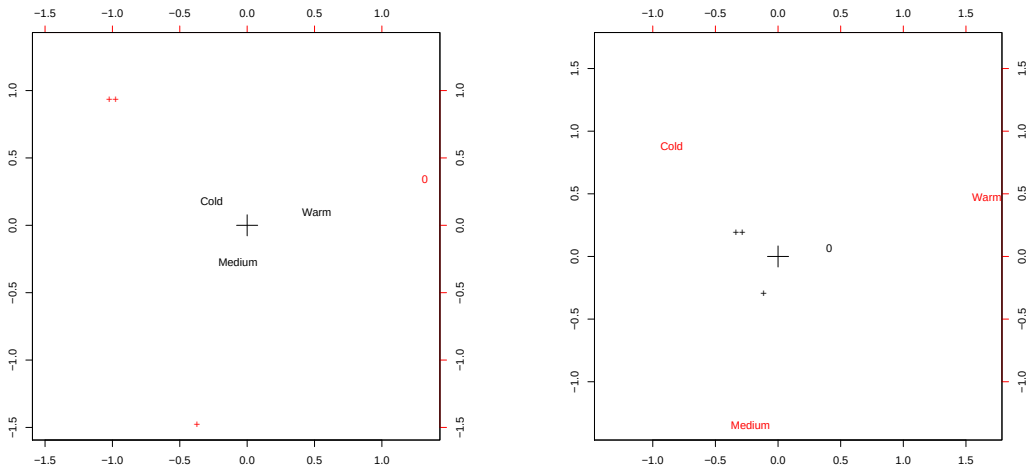

Figura 8.7: Analisi della corrispondenza per dati relativi all'abbondanza di una specie in tre siti di indagine (distinti per condizioni climatiche). A sinistra il grafico con scalatura di tipo 1 (*type="row"*) a destra con scalatura di tipo 2 (*type="column"*). Le coordinate dei punti vanno lette sulla coppia di assi contraddistinti dal loro stesso colore.

```
> F
              [,1]        [,2]
Cold   -0.26322158  0.17861936
Medium -0.06835392 -0.27211200
Warm    0.51685002  0.09516583

> F.hat <- cp$cscore %*% diag(cp$cor)
> F.hat
         [,1]        [,2]
0   0.4088694  0.06955181
+  -0.1153860 -0.29976768
++ -0.3099671  0.18739192
```

dove la matrice diagonale diag(cp$cor) corrisponde alla matrice $\Lambda^{1/2}$.

I grafici che si usano per rappresentare il risultato dell'analisi sono di tre classi. La prima (scalatura di tipo 1) consiste nel graficare congiuntamente i punti individuati dalle matrici F e V; questo ordinamento è da preferirsi quando si vogliano confrontare tra loro la disposizione delle righe (in questo caso i siti di campionamento) della tabella di contingenza, visto che le distanze date da F coincidono con le distanze fra le righe di detta tabella. Nel caso di scalatura di tipo 2 il ruolo tra righe e colonne si inverte, dato che si graficano i punti ottenuti dalle matrici $\hat{F}$ e $\hat{V}$ (la matrice $\hat{F}$ preserva le distanze χ^2 tra le colonne della tabella di contingenza). Una terza scalatura fa uso delle matrici F e $\hat{F}$ congiuntamente.

Analizzando i diversi grafici che si possono realizzare occorre prestare attenzione al fatto che è appropriato considerare la distanza tra punti "riga" fra loro e punti "colonna" tra loro, mentre non ha nessun senso interpretare le distanze tra punti "riga" e punti "colonna". Quello che è possibile fare è interpretare la posizione di un punto "riga" rispetto alla struttura complessiva dei punti "colonna" e viceversa.

In R si possono ottenere i tre tipi di grafici descritti con le chiamate seguenti:

```
> plot(cp, type="row")         # scalatura di tipo 1
> plot(cp, type="column")      # scalatura di tipo 2
> plot(cp, type="symmetric")   # scalatura di tipo 3
```

	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9	SP10
P1	96	85	12	2	0	0	0	0	0	0
P2	35	85	63	15	0	0	0	0	0	0
P3	5	25	98	67	5	0	0	0	0	0
P4	0	5	34	90	55	15	6	0	0	0
P5	0	0	6	3	99	80	41	0	0	0
P6	0	0	0	0	57	89	99	10	0	0
P7	0	0	0	0	10	31	64	60	12	3
P8	0	0	0	0	0	4	13	98	70	28
P9	0	0	0	0	0	0	0	43	94	93
P10	0	0	0	0	0	0	0	5	39	75
P11	0	0	0	0	0	0	0	0	5	16

Tabella 8.4: Presenza di 10 specie (per colonna) in 11 differenti habitat (per riga). I numeri in tabella sono il numero di soggetti di ciascuna specie rilevati all'interno dei diversi siti di campionamento. L'altitudine dei siti di campionamento aumenta da P1 a P11.

I risultati delle prime due chiamate sono mostrati in Fig. 8.7. Nel grafico a sinistra (scalatura di tipo 1) la distanza tra i siti di campionamento rispecchia la distanza nella tabella di contingenza; in questo grafico ogni sito che si trovi vicino ad un punto colonna porta a un rilevante contributo al valore di detta colonna. Nello specifico ad esempio, al sito a maggior temperatura (warm) è associata assenza della specie in esame (0). Nel grafico di destra di Fig. 8.7 sono invece le distanze tra le colonne a essere conservate. Ogni punto colonna (specie) che si trova vicino a un punto riga è particolarmente diffuso in quel particolare sito. In entrambi i grafici si può notare che le abbondanze della specie in studio sono ordinate in ordine decrescente al crescere della temperatura.

8.4.1 Detrending

In studi di tipo ecologico si ha spesso il caso in cui uno o più gradienti sottostanti (temperatura, umidità, abbondanza di fonti di nutrimento e altre) influenzi la diffusione delle specie in studio. Dato che la curva di densità di presenza di una specie in risposta a un gradiente ecologico è tipicamente unimodale – piccata intorno alle condizioni ottimali – se lo studio viene pianificato in modo da esplorare per intero il gradiente, campionando una quantità appropriata di siti differenti, la risposta (diffusione delle specie nei siti) sarà fortemente non lineare rispetto al gradiente. Questa semplice considerazione fa capire perché la tecnica di analisi in componenti principali, che presuppone dipendenze lineari, sia in questo ambito soppiantata dalla tecnica di analisi delle corrispondenze che ben lavora con dipendenze unimodali. Il modo con cui la presenza di risposte non lineari viene messa in luce nei diagrammi bidimensionali di ordinamento è sotto forma di particolari andamenti, a ferro di cavallo (per PCA) o ad arco (per CA).

Il tentativo di ricostruire i gradienti sottostanti alle osservazioni è alla base della tecnica di analisi della corrispondenza detrended (DCA), tecnica che lavora sugli assi individuati da una analisi della corrispondenza piegando gli archi in modo da far riapparire un andamento lineare tra i siti. Esistono due tipi di tecniche DCA: a segmenti o polinomiale; nel seguito viene descritta brevemente la prima, mentre per la seconda si rimanda a testi come [41].

Nella tecnica DCA a segmenti si divide l'asse principale dell'analisi CA in un numero variabile di segmenti. All'interno di ciascun segmento si spostano i punti in modo da rendere la media dei valori di detti punti sul secondo asse CA pari a 0. Ovviamente al termine della procedura la distanza tra i punti non ha più un immediato significato, anche perché la suddivisione iniziale in segmenti può portare a risultati finali piuttosto diversi. In aggiunta, per correggere l'effetto di accumulazione dei punti verso gli estremi dell'arco, viene eseguita una riscalatura non lineare all'interno dei segmenti in modo da rendere uguale la varianza interna a ogni segmento. In R la procedura DCA è disponibile mediante la routine DECORANA (sviluppata da Hill nel 1980 e successivamente emendata da alcuni errori), implementata nella funzione *decorana* della libreria *vegan*.

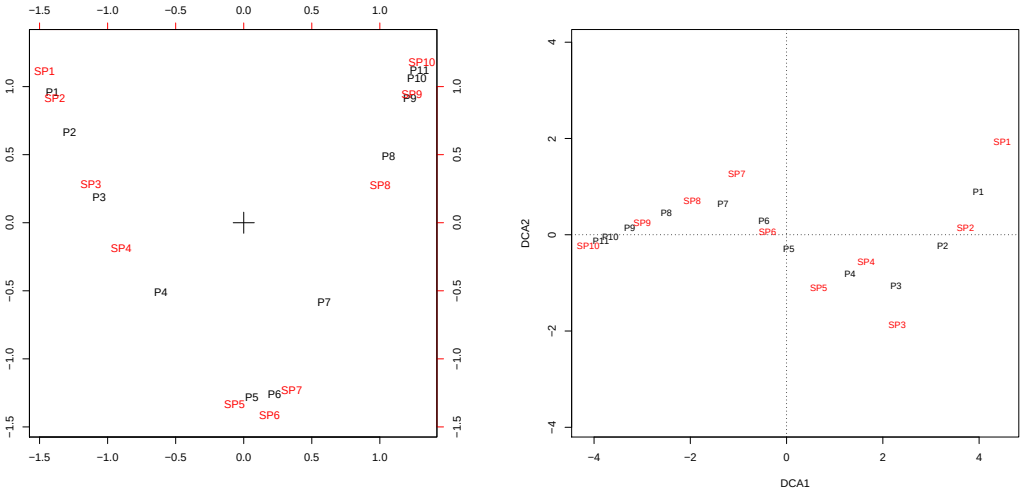

Figura 8.8: Analisi della corrispondenza per dati relativi all'abbondanza di 10 specie in 11 siti di indagine (distinti per altitudine). A sinistra il grafico dell'analisi non detrended mostra chiaramente la presenza di un arco. A destra il risultato della analisi detrended in cui l'arco è scomparso.

Attualmente in letteratura non vi sono opinioni omogenee sulla validità della procedura DCA. Molte critiche sono state mosse alla solidità e alla arbitrarietà delle sue basi teoriche. Inoltre in indagini su dataset simulati in cui si tentava di ricostruire gradienti ecologici complessi, la tecnica non ha ben figurato, rimuovendo, oltre ad archi fittizi, anche andamenti reali (si veda [41] per una discussione più approfondita su tutti questi argomenti). In particolare sembra che in presenza di più gradienti che determinano la diffusione delle specie in studio, la tecnica di scaling multidimensionale non metrico sia da preferirsi in quanto più precisa.

Esempio

In 11 differenti siti di campionamento, di altitudine crescente da P1 a P11, viene rilevata la presenza di 10 differenti specie. I risultati dell'indagine sono riepilogati in Tab. 8.4. Supponendo di aver caricato i valori nel dataset *ecologia* si può dapprima analizzare la situazione con una analisi della corrispondenza:

```
> ca <- corresp(ecologia, nf=2)
> plot(ca, type="row")
```

Il risultato, a sinistra in Fig. 8.8, mostra la presenza di un arco. Un semplice controllo della tabella dei dati rivela che questo è dovuto principalmente al fatto che il turnover delle specie segue il gradiente di altitudine in modo eccellente. In questi casi il primo asse di analisi è sufficiente a ordinare i siti in maniera appropriata e gli assi successivi presentano forme quadratiche dei valori del primo asse.

Per rimuovere questo andamento si può usare la funzione *decorana*:

```
> library(vegan)
> dca <- decorana(ecologia)
> plot(dca)
```

Il grafico, a destra in Fig. 8.8, evidenzia che la funzione ha rimosso l'artefatto quadratico, inserendo un andamento che non ha particolare significato ecologico. Si ricordi che in questo caso la distanza tra i punti è priva di significato.

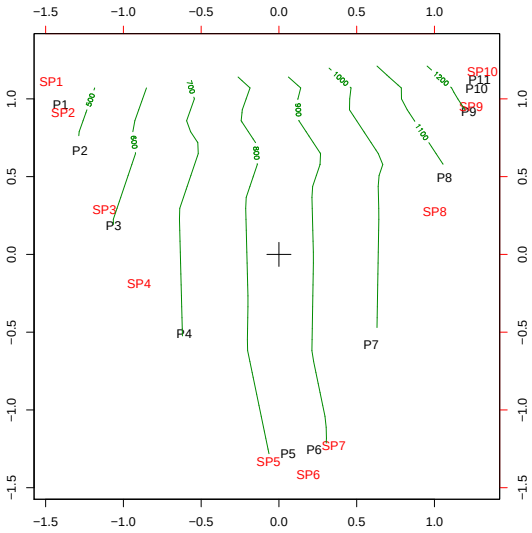

Figura 8.9: Analisi della corrispondenza per dati relativi all'abbondanza di 10 specie in 11 siti di indagine (distinti per altitudine). Le curve in verde evidenziano che la variabile altitudine cresce linearmente lungo il primo asse individuato dall'analisi di corrispondenza.

8.4.2 Interpretazione degli ordinamenti

Usualmente in campo ecologico è possibile disporre di misure di grandezze quali temperatura, elevazione, umidità e altro che caratterizzano i siti di campionamento. È allora interessante cercare di interpretare gli ordinamenti evidenziati in base a tali misurazioni, in modo da ricercare una possibile "spiegazione" di quanto evidenziato dallo studio. Per far ciò è possibile sovrapporre ai grafici di ordinamento ottenuti con tecnica CA o PCA, curve di livello che rappresentano l'andamento di ulteriori variabili in studio. Se le curve ottenute sono tra loro parallele e equispaziate l'ordinamento evidenziato sarà lineare nel fattore (la direzione ortogonale alle curve è quindi la cosiddetta direzione del gradiente). In R è possibile ricorrere alla funzione *ordisurf* della libreria *vegan*, come nell'esempio seguente.

Esempio

Si supponga che le altitudini dei siti campionati nell'esempio precedente vadano da 400 a 1400 metri sul livello del mare:

```
> altitudine <- seq(400, 1400, by=100)
```

La sovrapposizione delle curve di livello per la variabile *altitudine* si possono sovrapporre al grafico risultante dall'analisi CA nel modo seguente:

```
> plot(ca, type="row")
> ordisurf(ca$rscore, altitudine, add=TRUE, col="green4")
```

La funzione accetta come primo argomento gli score dei siti, individuati dall'analisi di corrispondenza, e come secondo la variabile i cui livelli vanno sovraimposti al grafico di ordinamento. L'opzione *add* serve a aggiungere il grafico sulla finestra esistente, senza creare un nuovo oggetto vuoto. In Fig. 8.9 si vede che la variabile *altitudine* cresce in maniera lineare lungo il primo asse individuato dalla tecnica CA.

Capitolo 9

Analisi multivariata: metodi di classificazione

9.1 Analisi discriminante lineare (LDA)

Se l'obiettivo dello studio è la divisione dei soggetti in r gruppi (noti a priori) a seconda dei valori delle p variabili $x_1, \ldots, x_p$ che vengono su di essi misurate, si può ricorrere alla tecnica dell'analisi discriminante. Questa metodica porta a costruire delle funzioni lineari dei predittori che permettono di classificare al meglio i soggetti all'interno dei gruppi. L'idea originaria risale e Fisher che pensò di ricercare la funzione lineare dei predittori che massimizza il test F per l'ANOVA a una via che usi i valori di tale funzione come variabile dipendente. A seconda del valore assunto dalla funzione si disporrà di un metodo automatico per allocare gli oggetti nei vari gruppi e per classificarne di nuovi.

Il maggiore impiego di questa tecnica si ha quando si vuole sostituire a una procedura di classificazione molto onerosa, o dal punto di vista economico o procedurale, una nuova metodologia basata su rilevazione di quantità che siano siano meno dispendiose da misurare o più rapidamente disponibili.

Il punto di partenza è la divisione della matrice della somma dei quadrati e dei prodotti crociati T (o analogamente della matrice di covarianza S) dei dati in due parti, in modo simile a quanto si fa nel caso di ANOVA. Una parte E è dovuta al contributo entro gruppi e una parte F è dovuta al contributo fra gruppi. Le tre matrici sono definite nel modo seguente. Sia X la matrice dei predittori con i soggetti inseriti per riga, M la matrice in cui a ogni soggetto è sostituito il vettore del baricentro (o centroide) del gruppo a cui esso appartiene e G una matrice $n \times r$ le cui colonne segnano l'appartenenza dei dati ai gruppi: $g_{ij} = 1$ se il soggetto i-esimo è nel gruppo j-esimo, 0 altrimenti. Si ha:

$$T = (n-1)S = (n-1)\ \mathrm{Cov}(X) \tag{9.1}$$

$$E = \frac{(X - GM)^T (X - GM)}{n - r} \tag{9.2}$$

$$F = \frac{(GM - \mathbf{1}\bar{\boldsymbol{x}})^T (GM - \mathbf{1}\bar{\boldsymbol{x}})}{r - 1} \tag{9.3}$$

dove $\bar{\boldsymbol{x}}$ è il baricentro generale dei dati. Si ha la relazione di somma:

$$T = (n-r)E + (r-1)F.$$

La miglior funzione discriminante Xa sarà tale da massimizzare il rapporto fra la varianza fra gruppi rispetto a quella entro gruppi, ossia massimizzare il rapporto:

$$\frac{a^T F a}{a^T E a}.$$

Per risolvere il problema è consuetudine effettuare uno *sphering* delle variabili $X' = XS$, in modo tale che sulle variabili trasformate la matrice E' sia l'identità. Si deve quindi risolvere il problema di

massimizzare $a^T F' a$ con il vincolo $|a| = 1$; questo equivale a dire che a deve essere l'autovettore di F' corrispondente al suo massimo autovalore. Sia U la matrice che ha per colonna gli autovettori ordinati in modo tale che sulla prima colonna si abbia l'autovettore corrispondente all'autovalore massimo, e così via[1]. Le funzioni discriminanti sono quindi:

$$X'U = XSU.$$

In R questo tipo di analisi si può effettuare facendo uso della funzione *lda* disponibile all'interno della libreria $MASS$, la quale in realtà opera con una logica leggermente diversa preferendo alla ricerca degli autovettori la decomposizione ai valori singolari delle matrici in questione.

Esempio

Riprendendo l'esempio delle anfore cretesi, si supponga che sia nota la loro datazione. Le prime 5 anfore risalgono ad un periodo più antico, le ultime 6 sono le più recenti e le rimanenti 4 hanno un'età intermedia. Si cerca di ricavare una classificazione analoga utilizzando solo i parametri di dimensione $x1, \ldots, x4$. Per prima cosa si crea il fattore *grp* che tiene traccia del gruppo di appartenenza reale delle anfore:

```
> grp <- factor( c(1,1,1,1,1,2,2,2,2,3,3,3,3,3,3) )
```

A questo punto l'analisi discriminante si esegue con la semplice chiamata:

```
> library(MASS)
> discr <- lda(grp ~ ., data=X)
```

Dove X è, come in precendenza, il data frame che contiene i valori delle dimensioni, rilevate sulle 15 anfore. L'output della funzione è abbastanza ricco:

```
> discr
Call:
lda(grp ~ ., data = X)

Prior probabilities of groups:
        1         2         3
0.3333333 0.2666667 0.4000000

Group means:
        x1       x2    x3     x4
1 22.46000 19.20000 29.66  9.900
2 24.60000 21.05000 32.85 10.925
3 24.28333 20.96667 33.10 10.550

Coefficients of linear discriminants:
          LD1        LD2
x1 -1.8266939 -3.0112168
x2  2.2243913  2.7660701
x3 -0.6399994  0.3771298
x4 -0.7181312 -0.2750134

Proportion of trace:
   LD1    LD2
0.8942 0.1058
```

[1] In generale è possibile costruire al massimo $\min(p, r-1)$ funzioni discriminanti dato che questo è il numero massimo di autovalori non nulli che ammette la matrice F'.

Dapprima vengono valutate le probabilità a priori di far parte di un dato gruppo. In mancanza di informazioni specificate dallo sperimentatore (tramite l'opzione *prior* della funzione *lda*) esse sono esattamente le proporzioni di soggetti nei vari gruppi di *grp*. A questa informazione segue una tabella in cui vengono riepilogate le medie dei predittori all'interno dei tre gruppi definiti dal fattore *grp*. Questi vettori di coordinate definiscono i *centroidi* dei gruppi. Infine vi è la tabella dei coefficienti delle funzioni lineari che meglio separano i soggetti rispetto alle classi specificate dal fattore cronologico in esame. Dato che i gruppi sono tre è possibile costruire due funzioni di questo genere (indicate da R con le sigle LD1 e LD2). Secondo i dati dell'analisi, la funzione lineare dei predittori che meglio separa i dati, identificata da LD1, è:

$$LD1 = -1.83 * x1 + 2.22 * x2 - 0.64 * x3 - 0.72 * x4.$$

Tale funzione separa le tre popolazioni di anfore molto meglio di quanto non farebbe la funzione LD2, come si evince dall'ultima riga dell'output. Per ogni funzione discriminante il valore riportato è infatti il rapporto fra l'autovalore corrispondente della matrice F' e la somma degli autovalori stessi. Questa quantità rappresenta proprio la proporzione di varianza fra gruppi interpretata dalle funzioni lineari trovate.

Per verificare i calcoli dell'algoritmo è possibile effettuare manualmente la procedura:

```
> S <- cov(X)          # matrice di covarianza totale
> M <-  discr$means    # coordinate dei centroidi
> G <- NULL; for(i in 1:3) G <- cbind(G, as.numeric(grp == levels(grp)[i]))
> E <- t(as.matrix(anfore) - G %*% M) %*% (as.matrix(anfore) - G %*% M) / 12
> F <- (14 * S - 12 * E)/2

> eigen( solve(E) %*% F )
$values
[1]  8.892942e+00  1.052540e+00 -5.553715e-14 -1.207411e-14

$vectors
           [,1]       [,2]       [,3]        [,4]
[1,] -0.6019148 -0.7316967 -0.5933035 -0.68597423
[2,]  0.7329603  0.6721284  0.7546568  0.06526138
[3,] -0.2108865  0.0916389 -0.1208490  0.19837373
[4,] -0.2366318 -0.0668256  0.2527443  0.69701375
```

Si hanno le componenti degli autovettori (non scalate) e i corrispondenti autovalori. Per questo problema si possono avere un massimo di due funzioni discriminanti. In effetti si nota che, entro la precisione algoritmica, gli ultimi due autovalori sono nulli; nei calcoli seguenti si possono quindi trascurare i relativi autovettori. La proporzione di varianza fra gruppi spiegata dalla prima funzione discriminante è:

```
> 8.892942e+00/(8.892942e+00 + 1.052540e+00)
[1] 0.894169
```

risultato coincidente con quanto visto in precedenza. Per quanto riguarda la matrice degli autovettori si procede alla scalatura in modo da rendere approssimativamente circolare la dispersione dei punti nei gruppi. Detta U la matrice degli autovettori questo risultato si ottiene dividendo ogni autovettore u_k per la radice quadrata della quantità $u_k^T E u_k$ (la varianza entro gruppi dell'autovettore u_k). In notazione matriciale la matrice normalizzata C è:

$$C = U(U^T EU)^{-1/2}$$

dove la matrice di scalatura $U^T EU$ è diagonale. Nel caso dell'esempio in questione si ha:

```
> U <- eigen( solve(E) %*% F )$vectors[,1:2]
```

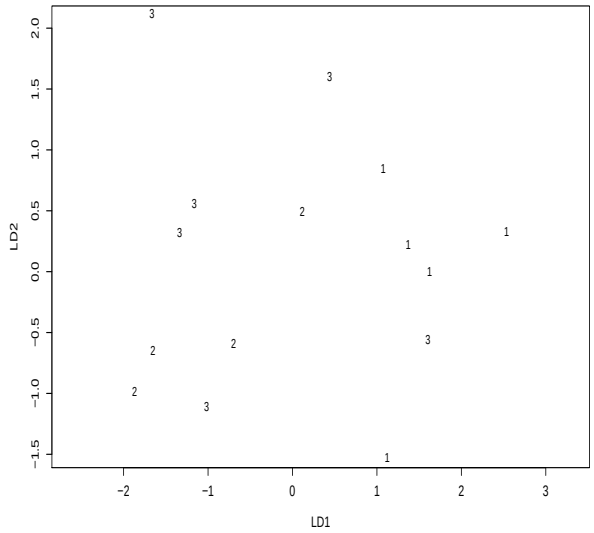

Figura 9.1: Grafico dei gruppi, corrispondenti alle diverse datazioni delle 15 anfore (dalle più antiche individuate dal numero 1, alle più recenti individuate dal numero 3), sul piano individuato dalle prime due funzioni lineari discriminanti. Si nota una buona separazione fra le tre popolazioni.

```
> scale <- sqrt(diag(diag((t(U) %*% E %*% U))))    # matrice di scalatura
> C <- U %*% solve(scale)

> C
           [,1]       [,2]
[1,] -1.8266939 -3.0112168
[2,]  2.2243913  2.7660701
[3,] -0.6399994  0.3771298
[4,] -0.7181312 -0.2750134
```

Si verifica che questo è quanto riportato dall'algoritmo *lda* (eventualmente a meno del segno).

Per avere un'idea della bontà delle funzioni discriminanti individuate è possibile graficare i valori che assumono LD1 e LD2 sugli individui e segnare ogni punto con il corrispondente valore del fattore *grp*. Se, in questa particolare visualizzazione, le popolazioni risultano ben separate si può concludere che la tecnica ha raggiunto il suo scopo (si noti per inciso che questo avviene quando la varianza all'interno dei gruppi è piccola rispetto a quella fra gruppi). In R è possibile realizzare questo grafico con la chiamata:

```
> plot(discr)
```

che produce l'output di Fig. 9.1, da cui si può concludere che i tre gruppi di anfore sono sufficientemente ben separate dalle due funzioni LD1 e LD2. Nel calcolare i valori che le funzioni assumono sui dati campionari, al posto dei vettori $x1, \ldots, x4$ vengono usati i vettori traslati $x1 - \bar{x1}, \ldots, x4 - \bar{x4}$.

9.1.1 Allocazione dei soggetti nelle classi

Per assegnare un soggetto sperimentale a una classe è necessario introdurre un algoritmo di allocazione. Il più semplice di essi è quello di calcolare il valore delle funzioni discriminanti sul soggetto in questione, fare lo stesso sui centroidi delle classi e assegnare il dato alla classe il cui centroide è più vicino. Nel far questo si trascurano le probabilità (a priori) di far parte delle diverse classi.

Un approccio differente fa uso della tecnica di massima verosimiglianza, introducendo nel contempo alcune ipotesi sulla distribuzione dei dati. Sia $p(c)$ la probabilità a priori di appartenere alla c-esima classe. Sia $p(x|c)$ la densità delle osservazioni x nella classe c e $p(c|x)$ la probabilità a posteriori di appartenere alla c-esima classe. Dalla regola di Bayes segue:

$$p(c|x) = \frac{p(c)\ p(x|c)}{p(x)} \propto p(c)\ p(x|c)$$

La regola di classificazione è quella di assegnare il soggetto alla classe per cui è massima l'espressione data. Supponendo che $p(x|c)$ sia normale multivariata con media μ_c e matrice di covarianza Σ_c, la regola di Bayes equivale a minimizzare:

$$Q_c = -2 \ln p(x|c) - 2 \ln p(c) = (x - \mu_c)\Sigma_c^{-1}(x - \mu_c)^T + \ln |\Sigma_c| - 2 \ln p(c) \tag{9.4}$$

Il primo termine è il quadrato della distanza di Mahalanobis dell'osservazione dal centroide del gruppo. Dato che la funzione Q_c dipende quadraticamente da x questa tecnica è nota come analisi discriminante quadratica (implementata nella funzione *qda*). Supponendo ulteriormente che le classi abbiano comune matrice di covarianza Σ il problema si semplifica ed è possibile massimizzare la funzione $-Q_c/2$ o:

$$L_c = x\Sigma^{-1}\mu_c^T - \mu\Sigma^{-1}\mu_c^T + \ln p(c) \tag{9.5}$$

si nota che la funzione è lineare in x. Secondo la tecnica plug-in nel calcolo di L_c (o Q_c) si sostituiscono a Σ, Σ_c e μ le loro stime campionarie. Si noti anche che l'algoritmo citato in inizio di sezione differisce da questa ultima tecnica di allocazione solo per il fatto di trascurare i valori delle probabilità a priori.

Per concludere, è bene essere consci del fatto che le ipotesi che stanno alla base di questa teoria classificatoria possono facilmente cadere, dato che non è infrequente che le osservazioni abbiano distribuzioni non normali multivariate. Le tecniche LDA/QDA sono ben lontane dall'essere robuste quando queste ipotesi vengono a cadere. Altre più moderne tecniche, illustrate nel seguito, vengono in aiuto per affrontare problematiche di classificazione.

Tornando all'esempio discusso in precedenza, per studiare se le funzioni lineari individuate dall'analisi classificano in maniera accettabile le anfore rispetto alla realtà si può far uso della funzione *predict*:

```
> pred <- predict(discr)$class
```

che estrae la classificazione delle anfore ottenuta dai valori simultanei di LD1 e LD2. In alternativa è possibile utilizzare la classificazione basandosi sul solo valore della migliore funzione discriminante, cioè LD1, con la chiamata:

```
> pred.ld1 <- predict(discr, dimen=1)$class
```

Per confrontarle con la classificazione reale si possono utilizzare le semplici chiamate:

```
> table(grp, pred)
   pred
grp 1 2 3
  1 5 0 0
  2 0 3 1
  3 1 1 4

> table(grp, pred.ld1)
   pred.ld1
grp 1 2 3
  1 5 0 0
  2 0 2 2
  3 1 1 4
```

Per riga si legge la classificazione reale e per colonna quella inferita dalle funzioni lineari utilizzate. Sulla diagonale principale si hanno i casi in cui le classificazioni coincidono. Ad esempio, nel primo caso, si nota che 12 anfore su 15 sono classificate correttamente dall'analisi e che i maggiori problemi si hanno per le 6 anfore di classe $grp = 3$, dato che 2 di esse vengono classificate non correttamente dalla funzione lineare individuata. Solitamente si riporta l'*error rate* di classificazione, definito come il rapporto fra i soggetti classificati in modo errato e il totale dei soggetti. In questo caso si ha un error rate di 3/12 ossia del 25%.

Questa stima è comunque ottimistica dato che vi è sicuramente il problema della effetiva generabilizzabilità della funzione lineare al di fuori del campione con cui essa viene costruita. Una possibile soluzione, nel caso di un grande numero di dati, è usare parte del campione per valutare le funzioni discriminanti (training sample) e parte per validarne i risultati (test sample). Dato che solitamente le dimensioni del campione non sono tali da permettere un approccio di questo genere senza perdere troppo in potenza, è possibile ricorrere a tecniche bootstrap o jackknife con cui testare l'efficacia effettiva delle funzioni lineari individuate della procedura di analisi discriminante (si veda ad esempio Sec. 9.1.2).

Un ulteriore valutazione della bontà delle funzioni discriminanti individuate è eseguire effettivamente il test ANOVA per vedere se i gruppi considerati differiscono per i valori medi di LD1 e LD2. Per l'esempio in questione si ha:

```
> ld1 <- predict(discr)$x[,1]     # valori di LD1 sul campione di 15 anfore
> ld2 <- predict(discr)$x[,2]     # valori di LD2 sul campione di 15 anfore
> anova(lm(ld1 ~ grp))
Analysis of Variance Table

Response: ld1
          Df  Sum Sq Mean Sq F value   Pr(>F)
grp        2 17.7859  8.8929  8.8929 0.004276 **
Residuals 12 12.0000  1.0000

> anova(lm(ld2 ~ grp))
Analysis of Variance Table

Response: ld2
          Df  Sum Sq Mean Sq F value Pr(>F)
grp        2  2.1051  1.0525  1.0525 0.3792
Residuals 12 12.0000  1.0000
```

Dai due risultati si conclude che la prima funzione discriminante raggiunge il suo scopo mentre la seconda non discrimina al livello di significatività statistica. In questo caso l'uso della sola LD1 è quindi perfettamente giustificato.

9.1.2 Leave-one-out cross-validation

Una metodologia largamente utilizzata per valutare la bontà delle funzioni discriminanti individuate in fase di analisi fa uso della tecnica di validazione *leave-one-out*. Essa consiste nel calcolare le funzioni discriminanti su tutti i possibili sottocampioni di $n - 1$ individui e classificare, in ognuna di queste occasioni, solamente l'individuo escluso. In ognuno di questi casi si classifica quindi un soggetto che non rientra nella costruzione delle funzioni discriminati. Si hanno così n classificazioni da confrontare con la situazione reale.

Nel caso dell'esempio delle anfore si procede nel modo seguente. La valutazione della rivalidazione si ottiene sempre con la funzione *lda*, utilizzando l'opzione $CV = TRUE$:

```
> discr.val <- lda(grp ~ ., data=X, CV=TRUE)
```

Il confronto tra la classificazione delle funzioni discriminanti e la realtà si ottiene facilmente con la chiamata:

```
> table(grp, discr.val$class)

grp 1 2 3
  1 4 0 1
  2 0 0 4
  3 1 1 4
```

Si nota che solamente 8 anfore sono classificate in maniera esatta alla rivalidazione, ossia 4 di meno di quanto ottimisticamente ottenuto in precedenza. Si può quindi concludere che le funzioni discriminati ottenute hanno un error rate stimato di $7/15 = 46.7\%$, con intervallo di confidenza al 95% valutabile con la chiamata:

```
> binom.test(7, 15)
[...]
95 percent confidence interval:
 0.2126667 0.7341387
```

9.2 Alberi di classificazione

Come la tecnica dell'analisi discriminante, anche la costruzione di un albero di classificazione (classification tree) è utile per effettuare una classificazione automatica di soggetti in base ai valori misurati di un set di variabili. La flessibilità di questo metodo lo fa talvolta preferire alle analisi più tradizionali, in particolar modo quando le assunzioni su cui queste ultime si poggiano vengono meno o non possono essere facilmente verificate.

La caratteristica che contraddistingue gli alberi di classificazione è la loro natura gerarchica. Per chiarire il concetto con un esempio si supponga di avere una pila di monete composta da pezzi di tre valori diversi che differiscono per il loro diametro. Per suddividerle velocemente si può pensare di setacciare il mucchio con un setaccio che abbia fori abastanza piccoli da lasciar pasare solo le monete di diametro inferiore, che vengono quindi rapidamente isolate. Il mucchio restante viene passato in un nuovo setaccio, dalle maglie di dimensioni tali che solo le monete più piccole fra quelle rimaste possano cadere. Questo è un esempio di albero di classificazione che, con una procedura gerarchica in due passi, arriva alla classificazione del mucchio di monete.

Un albero di classificazione si presenta come un diagramma ad albero rovesciato (il nodo radice, o root node, è in alto) che si biforca ogni volta che viene presentata una scelta, basata sul valore di una delle variabili in gioco (come nel caso dell'algoritmo implementato in R) o sul valore di una combinazione di più variabili (questo caso non è coperto dalle librerie disponibili). Si distinguono due tipologie di nodo: i nodi non terminali - i quali hanno due discendenti diretti - e i nodi terminali (o foglie) che non subiscono ulteriori bipartizioni. In questo tipo di albero ogni split, basato sul valore di una singola variabile, divide sempre un nodo genitore in due nodi figli. In questo tipo di analisi è possibile far rientrare sia variabili qualitative che quantitative.

L'algoritmo di costruzione degli alberi implementato in R, è del tipo CART (Classification And Regression Trees). Questo algoritmo parte dai dati raggruppati in un unico nodo (root node) ed esegue ad ogni passo una ricrca esaustiva fra tutte le possibili suddivisioni. A ogni passo viene scelta la suddivisione migliore, cioè quella che produce rami il più possibile omogenei fra loro. Per valutare la bontà di uno split si usa comunemente l'indice di impurità di *Gini*, che per il nodo i-esimo è definito come:

$$1 - \sum_k \hat{p}_{ik}^2$$

dove l'indice k corre sulle classi del fattore di classificazione e

$$\hat{p}_{ik} = \frac{n_{ik}}{n_i}$$

è la frazione degli n_i soggetti nel nodo i-esimo assegnato alla k-esima classe. Si sceglie come split quello che riduce l'impurità media delle due foglie.

Dopo che si è individuato lo split migliore per il nodo radice, l'algoritmo ripete il processo di ricerca per ogni nodo figlio continuando a bipartire finché ciò non è più possibile (se un nodo è costituito da un solo caso oppure quando tutti i casi che compongono un nodo afferiscono alla stessa classe) o finché il processo non è arrestato per qualche ragione (tipicamente perché un nodo è costituito da un numero troppo esiguo di casi; per la libreria *rpart* questa soglia è di 20 casi). Quando un nodo viene riconosciuto come terminale bisogna stabilire come classificare i casi che in esso sono contenuti. Un semplice criterio al riguardo consiste nell'assegnare al nodo l'etichetta del gruppo maggiormente rappresentato.

Originariamente la costruzione degli alberi di classificazione avveniva splittando ogni nodo finché non smetteva di essere soddisfatto un qualche criterio di bontà di bipartizione. Quando tutti i rami uscenti dal nodo radice raggiungevano dei nodi terminali, il processo di costruzione dell'albero veniva considerato ultimato. La filosofia dell'algoritmo CART è completamente diversa. Invece di cercare di stabilire se un nodo è terminale o meno, il CART procede diramando l'albero finché è possibile ed esaminando sotto-alberi ottenuti dalla potatura dei rami (*tree pruning*) dell'albero di partenza. Tra tutti i sotto-alberi possibili si sceglierà il migliore in base a un criterio del minimo costo-complessità. Sia R_i una misura di adeguatezza, valutata su ognuna delle foglie dell'albero (tale misura può ad esempio essere il numero di casi mal classificati all'interno della foglia i-esima). Sia R la somma di tali valori su tutte le foglie dell'albero. Si definisce l'indice R_α:

$$R_\alpha = R + \alpha\, nT$$

con nT pari alla dimensione dell'albero (ossia al numero delle sue foglie). All'aumentare di α si penalizzano alberi con molte foglie. Per valutare quale sia il valore di α ottimale gli algoritmi ricorrono a una procedura di rivalidazione interna, dividendo il set di dati in 10 sottoparti, usandone quindi 9 per sviluppare l'albero e usando la decima per una procedura di validazione. Questa procedura è ripetuta tenendo fuori a turno uno dei sottocampioni e mediando poi i risultati (questo processo è noto come 10-fold cross-validation). Per ulteriori informazioni si rimanda ad esempio a [45].

Il problema principale degli algoritmi stile CART è il fatto che essi non tengono assolutamente conto dell'influenza che la scelta di una particolare divisione ha sui futuri divisori. In altre parole, la decisione della divisione avviene ad ogni nodo dell'albero, in un preciso momento durante l'esecuzione dell'algoritmo, e non è mai più riconsiderata in seguito. Dato che tutte le suddivisioni vengono scelte sequenzialmente e ognuna di esse di fatto dipende dalle predecedenti, si ha che tutte le divisioni sono dipendenti dal nodo radice dell'albero; una modifica del nodo radice potrebbe portare alla costruzione di un albero completamente differente.

I dettagli matematici e algoritmici che sono alla base della costruzione e della selezione di un albero di classificazione sono numerosi e non possono essere trattati qui nel dettaglio. Per ulteriori particolari si rimanda a testi come [8, 45].

Esempio

Le funzioni con cui R tratta gli alberi di classificazione sono disponibili nella librerie aggiuntive *rpart* (che sarà qui utilizzata) e *tree*. Per un esempio del loro impiego si usa un datatset standard della distribuzione.

I dati impiegati si riferiscono a misure (in cm) di quattro caratteristiche (lungezza e larghezza dei sepali e lunghezza e larghezza dei petali) rispettivamente di 50 fiori di iris di tre specie differenti: *Iris setosa*, *Iris versicolor* e *Iris virginica*. Si inizia l'analisi caricando la libreria *rpart* e il dataset:

```
> library(rpart)
> data(iris)
> iris
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
```

```
[...]
150          5.9         3.0          5.1         1.8  virginica
```

Il fit dell'albero di classificazione è molto semplice:

```
> ir.rp <- rpart(Species ~ .,  data=iris)
```

Il risultato del fit si ottiene con la chiamata:

```
> ir.tr
n= 150

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 150 100 setosa (0.33333333 0.33333333 0.33333333)
  2) Petal.Length< 2.45 50   0 setosa (1.00000000 0.00000000 0.00000000) *
  3) Petal.Length>=2.45 100  50 versicolor (0.00000000 0.50000000 0.50000000)
    6) Petal.Width< 1.75 54   5 versicolor (0.00000000 0.90740741 0.09259259) *
    7) Petal.Width>=1.75 46   1 virginica (0.00000000 0.02173913 0.97826087) *
```

da cui si vede che l'albero ha 3 nodi terminali. La classificazione dei soggetti nelle tre specie come risulta dalle regole dell'albero si ottiene con la chiamata:

```
> pred <-  predict(ir.rp, type="class")
```

e la sua accuratezza si può valutare con la funzione *table*, confrontando la classificazione reale con quella ottenuta dall'algoritmo:

```
> table(iris$Species, pred)

            setosa versicolor virginica
  setosa        50          0         0
  versicolor     0         49         1
  virginica      0          5        45
```

Si nota che solo 6 fiori su 150 sono classificati in modo errato(error rate $ER = 4\%$). Ovviamente anche qui vale il discorso fatto per le funzioni discriminanti: la performance dell'albero è ottimale per il set di dati su cui è costruito, ma è necessario convalidarla su dati indipendenti in modo da avere una stima realistica della sua efficienza.

Per avere la rappresentazione grafica dell'albero, che di solito rappresenta l'obiettivo dell'analisi, si eseguono le chiamate:

```
> plot(ir.rp)
> text(ir.rp, use.n=TRUE)
```

che producono l'output di Fig. 9.2. La funzione *plot* disegna lo scheletro dell'albero, mentre la funzione *text* etichetta i nodi con le condizioni di divisione o, per i nodi terminali, con l'etichetta di gruppo. L'opzione $use.n = TRUE$ serve ad aggiungere ai nodi terminali il numero di soggetti delle varie classi che finiscono in quel nodo. Si vede ad esempio che il nodo relativo al gruppo "setosa" è un nodo puro, cioè contiene solo soggetti della specie *Iris setosa*, mentre entrambi gli altri non lo sono e contengono sia soggetti della classe dominante che da il nome al nodo, sia soggetti di un'altra classe. Come si vede solo due delle variabili (quelle riguardanti le dimensioni dei petali) entrano nella costruzione dell'albero.

Per verificare che l'abero completo sia il migliore o se sia invece opportuno effettuare una potatura (pruning) si può far uso della chiamata seguente:

```
> plotcp(ir.rp)
```

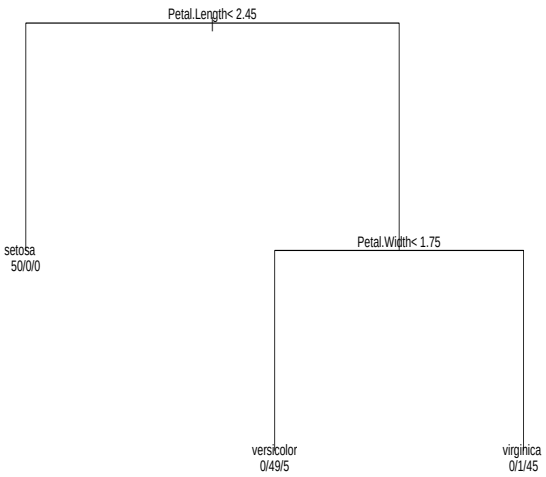

Figura 9.2: Albero di classificazione per i i dati relativi a tre specie di iris.

che produce il grafico di Fig. 9.3. In tale grafico si hanno sulle ascisse i valori del parametro α (che in R è chiamato *cp*) che corrispondono ad alberi con diversi numeri di foglie (valori riportati sull'asse delle ascisse addizionale, posto in alto). In ordinata si hanno i valori degli error rate in rivalidazione per ognuno degli alberi considerati, ottenuti con il processo 10-fold cross-validation. Le barre d'errore sono a loro volta ottenute dal processo bootstrap, semplicemente calcolando la deviazione standard delle 10 stime di error rate per ciscun albero. Gli error rate vengono espressi relativamente all'error rate dell'albero a un unico nodo terminale costruito con i dati campionari, al quale spetterebbe in ordinata il valore 1 (si noti che è possibile ottenere nel processo bootstrap error rate relativi superiori a 1). Solitamente si sceglie l'albero più piccolo, quindi più a sinistra possibile, il cui error rate sia al di sotto della linea orizzontale tratteggiata che coincide con il valore del minimo assoluto più il suo errore. Questo metodo di scelta, largamente usato in ambito multivariato, va sotto il nome di tecnica 1SE. Nel caso in questione l'albero ottimale è proprio quello con tre foglie.

9.3 Random Forests

La tecnica di Random Forests, estensione dell'approccio relativo alla costruzione degli alberi di classificazione, è stata recentemente proposta da Breiman [7].

L'algoritmo Random Forests si basa sulla costruzione di molti alberi di classificazione. Ogni singolo caso viene fatto passare in tutti gli alberi della foresta; ognuno di essi fornisce una classificazione. La catalogazione finale viene quindi fatta secondo il criterio di maggioranza: la classe con più voti è quella a cui viene assegnato il caso in esame.

Si abbia un campione di n soggetti di cui è disponibile la classificazione in k classi secondo un dato fattore di interesse; su ognuno dei soggetti si misurano p variabili (categoriali o continue). Per la crescita degli alberi della foresta si usano le seguenti regole:

- Si seleziona un set di n soggetti (campionamento con reinserimento) dal campione in esame. Questo set di dati verà usato nella costruzione dell'albero come training set. I casi non selezionati (*oob*, out-of-bag) vengono usati per valutare l'error rate dell'albero e per stabilire l'importanza delle p variabili in sede di classificazione.

- Si sceglie un numero $k \leq p$. Ad ogni split k predittori sono selezionati casualmente e solo essi vengono usati per valutare la bipartizione ottimale.

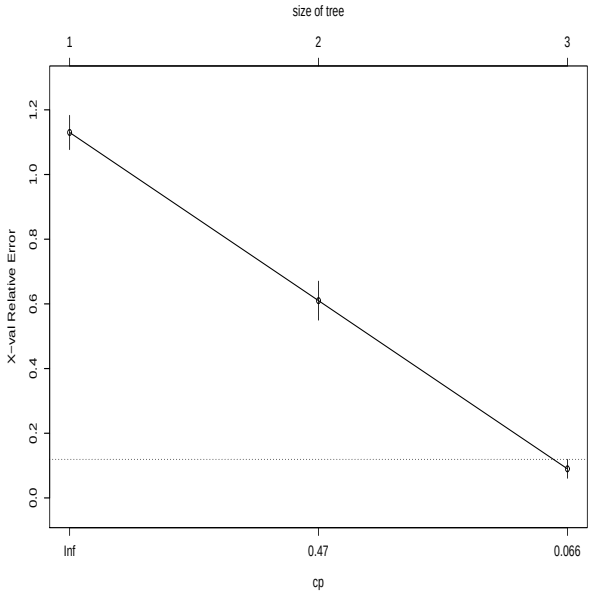

Figura 9.3: Scelta del miglior albero di classificazione per i dati relativi a tre specie di iris. La linea tratteggiata corrisponde all'estremo superiore dell'intervallo di confidenza per l'albero a error rate minore. L'albero migliore è quello con tre nodi terminali, l'unico con error rate inferiore al livello identificato dalla linea tratteggiata.

- Ogni albero viene fatto crescere fino alla sua massima estensione, senza utilizzare tecniche di pruning.

L'error rate globale della foresta dipende da due fattori: la correlazione esistente fra gli alberi e la bontà dei singoli alberi. Aumentando la correlazione aumenta l'error rate, mentre aumentando la bontà dei singoli alberi l'erorr rate diminuisce [7]. Diminuendo il valore di k si riducono correlazione fra gli alberi e bontà dei singoli alberi, mentre aumentandolo si aumentano entrambi. Un valore di k intermedio risulterà quindi ottimale.

Data la metodologia bootstrap con cui vengono costruite, le foreste offrono una stima non distorta dell'error rate, generalizzabile al di fuori del campione utilizzato per stimarlo. Non vi è quindi necessità di rivalidazione su un campione indipendente.

9.3.1 Importanza delle variabili

Un caratteristica particolarmente attraente delle Random Forests è che esse generano una stima di quali variabili sono importanti per la classificazione, offrendo la possibilità di selezionare solo un sottoinsieme che risulti ottimale dal punto di vista statistico.

La procedura che permette di fare questa valutazione è la seguente. Per ogni albero si classificano i casi *oob* per quel particolare albero. Per il j-esimo albero si abbiano C_j casi classificati correttamente. Si permutano quindi casualmente i valori della i esima variabile per i casi *oob* e si riclassificano i soggetti. Sia CP_{ij} i casi classificati correttamente dopo la permutazione. Si calcola il valore di importanza della i-esima variabile I_{ij}:

$$I_{ij} = C_j - CP_{ij}$$

Si ripete il procedimento su tutti gli alberi della foresta e si fa la media dei valori ottenuti:

$$I_i = mean(I_{ij})$$

l'indice I_i stima l'importanza della i-esima variabile. L'errore standard di tale stima viene valutato come se tali valori fossero indipendenti (si veda [7] per una giustificazione di questa procedura).

Dividendo la stima I_i per il suo errore standard si ha la stima normalizzata dell'importanza della i-esima variabile.

Un altro indice che permette di valuare l'importanza di una variabile è sommare su tutti gli alberi della foresta la decrescita dell'indice di impurità di Gini dovuto all'uso di ciascuna variabile. Questa stima di importanza è solitamente consistente con la precedente.

In R è possibile avvaleri di questa tecnica mediante le funzioni implementate nella libreria aggiuntiva *randomForest*.

Esempio

Per illustrare la costruzione di una foresta si fa uso del dataset *iris* già analizzato in precedenza. La funzione per costruire una Random Forest è *randomForest*, il cui uso è il seguente:

```
> library(randomForest)
> set.seed(100)     # scelta del seme random
                    # per un'analisi riproducibile
> iris.rf <- randomForest(Species ~ ., data=iris, importance=TRUE)
```

In questo caso si usa l'opzione $importance = TRUE$ per richiedere che, durante la costruzione della foresta, venga valutata l'importanza delle variabili. Alte possibili opzioni di uso comune sono *ntree* (per specificare il numero di alberi che fanno parte della foresta, di default 500) e *mtry* (il numero di variabili da considerare a ogni split, di default pari alla radice del numero totale delle variabili). Il risultato della chiamata è il seguente:

```
> iris.rf

Call:
 randomForest(formula = Species ~ ., data = iris, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 4%
Confusion matrix:
           setosa versicolor virginica class.error
setosa         50          0         0        0.00
versicolor      0         47         3        0.06
virginica       0          3        47        0.06
```

Si ha una stima dell'error rate - pari al 4% - e la matrice di classificazione dei vari casi.

Per visualizare la stima dell'importanza delle variabili si usa la chiamata seguente:

```
> importance(iris.rf)
               setosa versicolor virginica MeanDecreaseAccuracy
Sepal.Length 1.371545  1.6961553  1.895610           1.3133882
Sepal.Width  0.973110  0.3076256  1.165498           0.6912411
Petal.Length 3.731889  4.5337883  4.100242           2.5275034
Petal.Width  3.833735  4.6327332  4.365181           2.5581178

             MeanDecreaseGini
Sepal.Length         9.572410
Sepal.Width          2.221248
Petal.Length        42.076043
Petal.Width         45.369419
```

Per riga sono presentate le variabili in studio. Nelle prime tre colonne si ha una stima dell'importanza delle variabili per le tre classi del fattore *Species*, nella quarta una stima della importanza globale delle variabili (la colonna di maggior interesse), infine una stima dell'importanza delle variabili nella decrescita dell'indice di Gini. Si nota che le due variabili più importanti sono $Petal.Length$ e $Petal.Width$.

9.4 Reti neurali

Le reti neurali hanno conosciuto negli ultimi anni una diffusione impetuosa e sono state impiegate con successo in molti campi della scienza in cui vi siano problemi di classificazione e predizione. Il loro vasto impiego è dovuto principalmente alla loro potenza e versatilità nell'affrontare la modellizzazione di fenomeni fortemente non lineari in cui sono presenti un gran numero di variabli. Esse possono essere impiegate praticamente in ogni occasione in cui esista una relazione, quanto si voglia complessa, fra predittori (input) e variabile dipendente (output).

L'unità di base di una rete neurale, in analogia con un sistema neurale biologico, si chiama *neurone*. Un neurone artificiale si comporta nel modo seguente:

- Riceve una serie di input, o dai dati originali o da altri neuroni che costituiscono la rete.
- Ogni input viene pesato in modo diverso (questi pesi corrispondono alle diverse efficienze sinaptiche di un neurone biologico).
- Tutti i diversi input pesati sono sommati insieme. A tale somma si sottrae un valore, caratteristico di ogni neurone, che simula la presenza di una soglia di attivazione. Il valore così ottenuto rappresenta il segnale di attivazione del neurone.
- Il segnale di attivazione viene processato da una funzione di attivazione (o funzione di trasferimento) che produce l'output del neurone.

La rete neurale è l'insieme dei vari neuroni di output e di input, spesso collegati tramite un passaggio intermedio di neuroni nascosti (che prende il nome di hidden layer). Una rete semplice ha una struttura *feedforward*: il segnale fluisce dagli input attraverso le unità nascoste fino a raggiungere gli output. Questa struttura gode della proprietà di essere stabile. Ovviamente si può pensare di costruire reti neurali più complesse con proprietà di feedback, ma al momento le semplici e stabili strutture feedforward si sono rivelate più utili nella pratica.

Una rappresentazione diagrammatica di una rete neurale feedforward (FFNN) in cui si ritrovano tutte le strutture sin qui descritte si ha in Fig. 9.4.

Per quanto riguarda la forma matematica della funzione di attivazione, che permette di calcolare l'output di ogni neurone, essa è usualmente modellata da funzioni con dominio in $\mathbf{R}$ e codominio in suo sottoinsieme ristretto. Un tipico esempio è la funzione logistica:

$$f(x) = \frac{e^x}{1 + e^x},$$

che ha codominio in [0, 1], ed è usualmente impiegata come funzione di trasferimento.

A questo punto si dispone di tutti gli elementi per dare una descrizione matematica di una rete neurale. Si supponga di avere p variabili di input $x_1, \dots x_p$ misurate su n soggetti, classificati in q classi differenti. La variabile dipendente y_k, nel caso di classificazione, è un vettore di q componenti $y_1, \dots, y_q$ ognuno dei cui elementi rappresenta la probabilità che il soggetto appartenga alla classe k-esima. Il legame fra i vettori x e y è modellizzabile come:

$$y_k = f_0\left(b_k + \sum_h W'_{hk} f_h(b_h + \sum_i W_{ih} x_i)\right) \tag{9.6}$$

dove W' rappresentano i pesi che congiungono l'hidden layer agli output, W i pesi che congiungono input e hidden layer, b (bias) i valori che permettono di tener conto delle soglie di attivazione, f

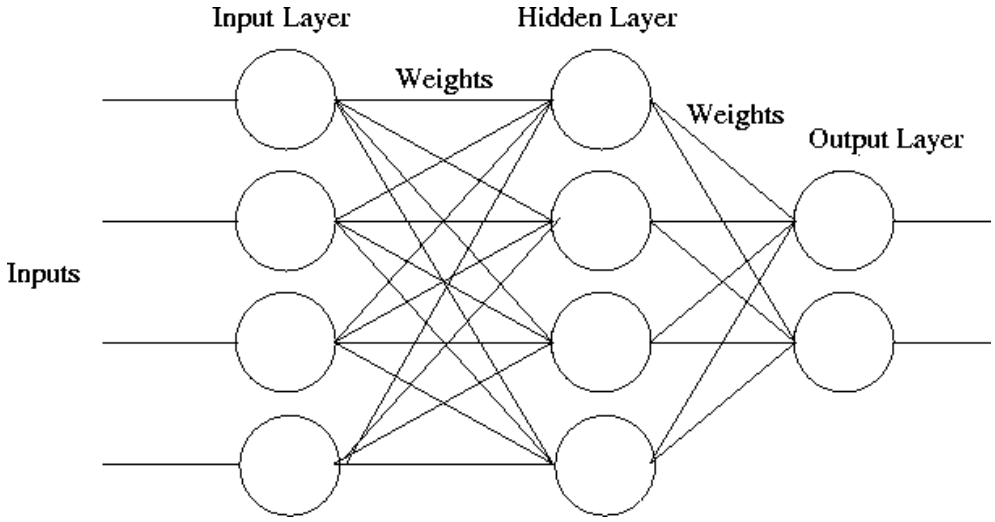

Figura 9.4: Schema generale di una rete neurale feedforward (FFNN). Ogni cerchio rappresenta un neurone e i tre layer sono connessi da un sistema di pesi.

le funzioni di attivazione. Ripetendo il processo su tutte gli n soggetti si ha una matrice y_{kj} con $k = 1, \ldots, q$ e $j = 1, \ldots, n$ delle probabilità di classificazione previste dalla rete neurale. La stima dei parametri W, W' e b si effettua con la cosidetta procedura di apprendimento (training o learning) in cui si minimizza, su un campione rappresentativo di soggetti, il quadrato degli scarti fra la classificazione reale e quella prevista. Algoritmicamente si tratta di trovare il minimo di una funzione non lineare dei parametri, problema che può essere affrontato con una varietà di tecniche.

Per il momento è stata lasciata in sospeso una questione: come decidere il numero di neuroni che compongono l'hidden layer. Questa decisione è lasciata allo sviluppatore della rete che deve cercare di bilanciare due effetti contrastanti. Un numero maggiore di neuroni permette di approssimare meglio sistemi di grande complessità; il problema è che più neuroni si introducono più sono i parametri da stimare e maggiore il rischio di "overfitting" (ossia i dati del trining set vengono fittati molto bene, ma la rete è difficilmente generalizzabile al di fuori di esso). D'altra parte un numero insufficiente di neuroni nell'hidden layer produce reti dalle scarse prestazioni. Per una trattazione dettagliata delle reti neurali si rimanda a [45, 5].

In R la funzione per fittare una rete neurale è *nnet*, inserita nell'omonima libreria. Negli esempi che seguono sono illustrate alcune applicazioni e alcuni problemi che si possono manifestare nel suo utilizzo.

Esempio

Si riprendano i dati relativi alle tre specie di iris su cui si rilevano quattro misurazioni di dimensione. L'adattamento di una rete neurale di tipo FFNN si può realizzare molto semplicemente con le chiamate:

```
> library(nnet)
> nn1 <- nnet(Species ~ ., data=iris, size=2, maxit=600)
# weights:  19
initial  value 190.465301
iter  10 value 69.655319
iter  20 value 69.318395
iter  30 value 69.314815
final  value 69.314798
converged
```

l'opzione $size = 2$ specifica che nell'hidden layer devono essere presenti 2 neuroni, mentre $maxit = 600$ specifica il numero massimo di iterazioni prima dell'arresto del ciclo di ottimizzazione. L'output della

funzione mostra il valore della funzione da minimizzare e, nell'ultima riga, segnala che il processo raggiunge la convergenza prima del numero massimo di ripetizioni specificate. Bisogna a questo punto accertarsi che il minimo trovato sia un "buon" minimo, ossia che l'algoritmo non abbia individuato un minimo locale e sia rimasta qui intrappolata. Per far ciò è possibile ripetere il fit più volte, dato che ogni volta viene generato un set random di pesi con cui iniziare la ricerca del minimo. In effetti in questo caso la ripetizione del processo porta a trovare un risultato assai migliore:

```
> nn1 <- nnet(Species ~ ., data=iris, size=2, maxit=600)
# weights:  19
initial  value 185.037723
iter  10 value 64.724960
iter  20 value 9.869698
iter  30 value 5.995920
iter  40 value 5.979222
iter  50 value 5.977537
[...]
iter 370 value 5.709308
final  value 5.709290
converged
```

Per esaminare i parametri di questa rete si usa la chiamata:

```
> summary(nn1)
a 4-2-3 network with 19 weights
options were - softmax modelling
  b->h1  i1->h1  i2->h1  i3->h1  i4->h1
   0.79  -77.36    7.42   37.57  133.61
  b->h2  i1->h2  i2->h2  i3->h2  i4->h2
  -1.94   -0.05   -0.23    0.35    0.67
  b->o1  h1->o1  h2->o1
  33.91    0.62 -117.70
  b->o2  h1->o2  h2->o2
   9.38  -16.85    4.17
  b->o3  h1->o3  h2->o3
 -43.17   15.87  112.35
```

I valori ottenuti si interpretano nel modo seguente: sulla prima riga si ha il bias del primo neurone dell'hidden layer e tutti i pesi che lo connettono con i neuroni di input; sulla seconda riga gli analoghi parametri per il secondo neurone dell'hidden layer; sulle ultime tre righe i parametri relativi alle connessioni dei tre neuroni di output con l'hidden layer.

Per verificare la bontà della rete neurale è possibile esaminare i valori fittati relativi alla classificazione di ciascuna unità:

```
> nn1$fitted
          setosa   versicolor    virginica
1   9.999993e-01 7.279985e-07 1.123193e-25
2   9.999974e-01 2.570835e-06 1.215706e-24
3   9.999988e-01 1.226702e-06 3.007681e-25
4   9.999966e-01 3.374830e-06 2.031968e-24
5   9.999994e-01 6.103881e-07 8.053718e-26
[...]
148 2.141261e-22 7.731672e-04 9.992268e-01
149 1.122273e-33 3.072375e-20 1.000000e+00
150 2.501119e-20 2.954116e-05 9.999705e-01
```

Ogni soggetto viene quindi assegnato alla classe il cui valore di output è maggiore; quindi i primi 5 soggetti sono classificati come *Iris setosa*, gli ultimi tre come *Iris virginica*. Questo processo può essere automatizzato usando la funzione *predict* nel modo seguente:

```
> predict(nn1, type="class")
  [1] "setosa"     "setosa"     "setosa"     "setosa"     "setosa"
  [6] "setosa"     "setosa"     "setosa"     "setosa"     "setosa"
[...]
[146] "virginica"  "virginica"  "virginica"  "virginica"  "virginica"
```

La classificazione comparata fra la situazione reale e quella ottenuta dalla rete neurale si ottiene, al solito, usando la funzione *table*:

```
> table(predict(nn1, type="class"), iris$Species)

            setosa versicolor virginica
  setosa     50      0          0
  versicolor  0     49          1
  virginica   0      1         49
```

da cui si conclude che sul training set, con i parametri ottenuti dall'algoritmo di minimizzazione, solo 2 fiori su 150 sono classificati in modo errato, con un error rate del 1.3%.

Ovviamente, come già detto in precedenza, anche per le reti neurali è opportuno testare la validità del modello ottenuto su un campione indipendente di dati (test sample). In questo caso, data la dimensione del campione in esame, è possibile utilizzare metà dati per adattare la rete neurale e riservare l'altra parte per la procedura di validazione. Come primo passo è necessario creare un vettore che contenga gli indici dei soggetti da usare nella procedura di fit. Se si sceglie di estrarre la metà dei soggetti per la procedura di apprendimento della rete, lo si può fare in modo bilanciato estraendo 25 fiori per ogni specie. Questo risultato si ottiene velocemente con il comando:

```
> training <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))
```

A questo punto si adatta la rete neurale solo sul training set:

```
> nn1 <- nnet(Species ~ ., data=iris, sample=training, size=2, maxit=600)
# weights:  19
initial  value 199.021850
iter  10 value 73.680598
iter  20 value 19.624933
iter  30 value 6.362878
iter  40 value 5.982060
iter  50 value 5.979595
iter  60 value 5.975253
iter  70 value 5.967165
iter  80 value 5.965253
iter  90 value 5.962355
iter 100 value 5.961253
iter 110 value 5.959566
iter 120 value 5.958422
iter 130 value 5.957413
final  value 5.957380
converged
```

Come si può vedere è sufficiente specificare l'opzione *sample* per ottenere il risultato voluto. Le prestazioni di questa rete possono quindi essere valutate sul test sample nel modo seguente:

```
> table(predict(nn1, iris[-training,], type="class"), iris$Species[-training])

            setosa versicolor virginica
  setosa     25      0          0
  versicolor  0     25          0
  virginica   0      0         25
```

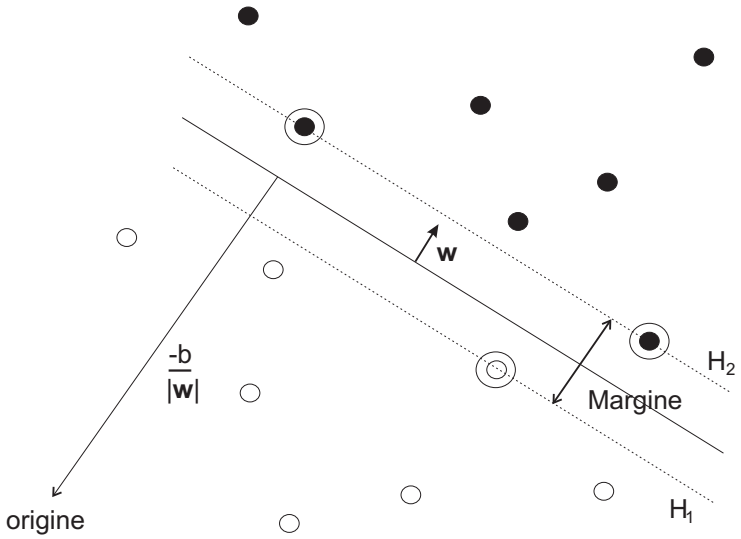

Figura 9.5: Iperpiano di separazione nel caso di due classi completamente separabili. I support vector sono indicati con un cerchio.

da cui si conclude che la rete neurale ottenuta si comporta egregiamente anche nella fase di rivalidazione classificando correttamente tutti e 75 i soggetti.

9.5 Support vector machines

Una delle tecniche più recenti nel campo della classificazione è quella delle Support Vector Machines (SVM). L'idea originale è in [6], lavoro ampliato in [58]. Un ottimo tutorial, in cui viene presentata la teoria alla base delle SVM, è [9]. Sulla falsariga di questo lavoro, sono riepilogati qui brevemente alcuni dei punti fondamentali della tecnica SVM.

Si consideri il caso in cui si abbiano n soggetti suddivisi in due uniche classi (il caso di un numero maggiore di classi è più complesso e verrà presentato brevemente in seguito). Su ognuno dei soggetti si definisce la variabile Y che assume i valori $y = \pm 1$ a seconda della classe di appartenenza del soggetto. Per ognuno dei soggetti si misurano anche i valori di p variabili $\mathbf{x} = (x_1, \ldots, x_p)$. Si supponga che le due classi siano separabili, nel senso che è possibile trovare un iperpiano tale da dividere lo spazio in due parti, una che contiene solamente soggetti con $y = 1$ e l'altra solo con soggetti $y = -1$. I punti $\mathbf{x} \in \mathbf{R}^p$ che giacciono sull'iperpiano soddisferanno l'equazione:

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{9.7}$$

con $\mathbf{w}$ vettore normale all'iperpiano, $|b|/||\mathbf{w}||$ distanza perpendicolare fra piano e origine e $||\mathbf{w}||$ norma euclidea di $\mathbf{w}$. Siano d_+ e d_- le distanze minime dell'iperpiano dai punti di classe $y = 1$ e $y = -1$ rispettivamente. La tecnica SVM tenta di individuare un tale iperpiano che abbia la caratteristica di avere massimo margine $d_+ + d_-$. I punti che giacciono esattamente a distanza minima dall'iperpiano sono detti *support vector*, da cui il nome della tecnica (si veda Fig. 9.5).

Si supponga che tutti i dati soddisfino alle seguenti condizioni:

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq +1 \qquad \text{per } y_i = +1 \tag{9.8}$$

$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -1 \qquad \text{per } y_i = -1 \tag{9.9}$$

che possono essere combinate in

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0 \qquad i = 1, \ldots, n \tag{9.10}$$

Si considerino i punti che soddisfano strettamente l'eguaglianza di Eq. 9.8 (imporre l'esistenza di questi punti equivale a scegliere una scala per $\mathbf{w}$ e b). Questi punti giacciano su un iperpiano H_1, di distanza dall'origine pari a $|1-b|/||\mathbf{w}||$. Equivalentemente, i punti che soddisfano l'eguaglianza di Eq. 9.9 giacciano su un iperpiano H_2, di distanza dall'origine pari a $|-1-b|/||\mathbf{w}||$. Si ha quindi $d_+ = d_- = 1/||\mathbf{w}||$, e il margine vale $2/||\mathbf{w}||$. Il problema di determinare il massimo margine si riduce dunque a trovare il minimo di $||\mathbf{w}||$, (o equivalentemente, ma più convenientemente, di $\frac{1}{2}||\mathbf{w}||^2$) con il vincolo di Eq. 9.10.

Si noti che la soluzione è determinata solamente dai support vector, e che le posizioni degli altri punti non ha nessuna influenza. Per risolvere questo problema è opportuno introdurre dei moltiplicatori di Lagrange $\alpha_i \geq 0$. Si costruisce quindi il lagrangiano da minimizzare L_p sottraendo i vincoli di Eq. 9.10, moltiplicati per il rispettivo moltiplicatore di Lagrange, dalla funzione obiettivo da minimizzare:

$$L_p \equiv \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i y_i (\mathbf{w} \cdot \mathbf{x}_i + b) + \sum_{i=1}^{n} \alpha_i \tag{9.11}$$

Si deve quindi minimizzare L_p rispetto a $\mathbf{w}$ e b, richiedendo che le derivate di L_p rispetto a tutti gli α_i siano nulle, con i vincoli $\alpha_i \geq 0\ \forall i$. Data la natura di questo problema la soluzione equivale a quella del problema duale: massimizzare L_p con i vincoli che il suo gradiente rispetto a $\mathbf{w}$ e b sia nullo, e con gli ulteriori vincoli $\alpha_i \geq 0\ \forall i$ (problema duale di Wolfe) [9]. Le condizioni sul gradiente rispetto a $\mathbf{w}$ e b portano rispettivamente alle due condizioni:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \tag{9.12}$$

$$\sum_{i=1}^{n} \alpha_i \mathbf{x}_i = 0 \tag{9.13}$$

Risostituendo queste condizioni nell'Eq. 9.11 si ottiene il lagrangiano duale, che viene indicato con L_D:

$$L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{9.14}$$

che deve essere massimizzato rispettando i vincoli:

$$\alpha_i \geq 0\ \forall i \quad , \quad \sum_{i=1}^{n} \alpha_i \mathbf{x}_i = 0.$$

I punti per cui risulterà $\alpha_i > 0$ sono i support vector, mentre tutti gli altri avranno $\alpha_i = 0$.

9.5.1 Caso di classi non separabili

Nel caso generale le due classi non saranno perfettamente separabili con un iperpiano in $\mathbf{R}^p$. Per risolvere questo problema si consente ad alcuni punti di giacere dal lato sbagliato dell'iperpiano, rilassando le condizioni di Eq. 9.8 e Eq. 9.9, introducendo della variabili ξ_i:

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq +1 - \xi_i \qquad \text{per } y_i = +1 \tag{9.15}$$

$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -1 - \xi_i \qquad \text{per } y_i = -1 \tag{9.16}$$

$$\xi_i \geq 0\ \forall i$$

Un modo naturale di tener conto delle variabili ξ_i è quello di definire la funzione obiettivo da minimizzare da $||\mathbf{w}||^2/2$ a $||\mathbf{w}||^2/2 + C\sum_i \xi_i$, con C parametro scelto dall'utente. Un alto valore di C corrisponde dunque a una grande penalità per un errore. In questo caso il lagrangiano duale L_D da massimizzare coincide con quello precedente in Eq. 9.14, con le condizioni:

$$0 \leq \alpha_i \leq C\ \forall i \quad , \quad \sum_{i=1}^{n} \alpha_i \mathbf{x}_i = 0.$$

Si verifica facilmente che la soluzione per $\mathbf{w}$ coincide con quella di Eq. 9.12.

In R le funzioni per utilizzare la tecnica di classificazione SVM sono implementate nella libreria aggiuntiva *e*1071, scaricabile dal sito della distribuzione.

Esempio

Si consideri nuovamente il dataset *iris*, oggetto di analisi nei paragrafi precedenti. Per illustrare anche graficamente il funzionamento della tecnica SVM si fa uso soltanto di una sua parte, relativa alle specie *Iris setosa* e *Iris versicolor* (le prime 100 linee del dataset) e alle variabili $Sepal.Length$ e $Sepal.Width$ (le prime due colonne):

```
> iris3 <- iris[1:100,c(1,2,5)]
> iris3$Species <- factor(iris3$Species)   # ricodifico i 2 livelli residui
```

La seconda linea serve a eliminare dalla codifica il livello *virginica* del fattore *Species*, che non fa più parte del dataset. Le due classi originali sarebbero separabili in fase di analisi, quindi per complicare un po' le cose si perturba uno dei dati in modo da eliminare la separabilità perfetta:

```
> iris3[1,1] <- 6.5
```

La classificazione SVM si effettua mediante la chiamata della funzione *svm*:

```
> model <- svm(Species ~ . , data=iris3, kernel="linear", cost=1)
```

Oltre alla formula relativa al modello da fittare, la funzione accetta altri parametri che ne determinano il comportamento. L'opzione *cost* permette di specificare il valore della costante C di cui sopra, mentre l'opzione *kernel* consente di trattare modelli SVM non lineari (si veda [9] per una loro trattazione). In questo caso si sceglie un modello lineare con $C = 1$. Il risultato del fit è:

```
> model

Call:
 svm(formula = Species ~ ., data = iris3, kernel = "linear", cost = 1)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1
      gamma:  0.5

Number of Support Vectors:  12
```

da cui si vede solamente che sono necessari 12 support vector per trattare il problema. Per visualizzare i valori di α_i (in realtà di $\alpha_i y_i$) si usa la chiamata:

```
> model$coefs
            [,1]
 [1,]  1.0000000
 [2,]  0.8499092
 [3,]  0.7543446
 [4,]  1.0000000
 [5,]  1.0000000
 [6,]  1.0000000
 [7,] -0.6042538
 [8,] -1.0000000
 [9,] -1.0000000
[10,] -1.0000000
[11,] -1.0000000
[12,] -1.0000000
```

da cui si conclude che sono stati usati 6 punti per il gruppo $y = +1$ (i primi 6) e 6 per il gruppo $y = -1$. Per sapere quali punti sono stati utilizzati si usa la chiamata:

```
> model$index
 [1]  1  2 21 26 32 42 58 60 67 85 86 89
```

che stampa in output il numero di riga del dataset di ognuno dei soggetti utilizzati come support vector.

Il valore di b si ottiene con la chiamata:

```
> b <- -model$rho
> b
[1] -0.2302362
```

mentre quello di **w** va costruito a partire dalle coordinate dei vettori di supporto, ottenibili con la chiamata:

```
> model$SV
   Sepal.Length Sepal.Width
1     1.5645337   0.8376174
2    -0.9017263  -0.2067933
21   -0.1310201   0.6287352
26   -0.7475851  -0.2067933
32   -0.1310201   0.6287352
42   -1.5182913  -1.6689683
58   -0.9017263  -1.4600862
60   -0.4393026  -0.8334397
67    0.1772624  -0.2067933
85   -0.1310201  -0.2067933
86    0.7938274   0.6287352
89    0.1772624  -0.2067933
```

Si osservi che tali valori non coincidono con i valori delle variabili nel datatset originario, dato che per motivi di stabilità numerica i dati vengono automaticamente standardizzati dalla funzione *svm* prima dell'analisi. Le coordinate del vettore **w** si ottengono quindi con la chiamata:

```
> w <- t(model$SV) %*% model$coefs
> w
                  [,1]
Sepal.Length -1.730741
Sepal.Width   1.596466
```

La classificazione del modello, confrontata con quella reale può essere ottenuta nel modo seguente:

```
> table(iris3$Species, predict(model))

            setosa versicolor
  setosa         48          2
  versicolor      0         50
```

Come di consueto la valutazione dell'error rate di classificazione sul dataset originario può essere eccessivamente ottimistica. Si può ottenere una stima più realistica ricorrendo a un procedimento di auto rivalidazione. Ad esempio una stima dell'error rate tramite processo 10-fold cross-validation si ottiene mediante l'opzione $cross = 10$ della funzione *svm*:

```
> set.seed(100)
> model.cv <- svm(Species ~ . , data=iris3, kernel="linear", cost=1, cross=10)
> summary(model.cv)
```

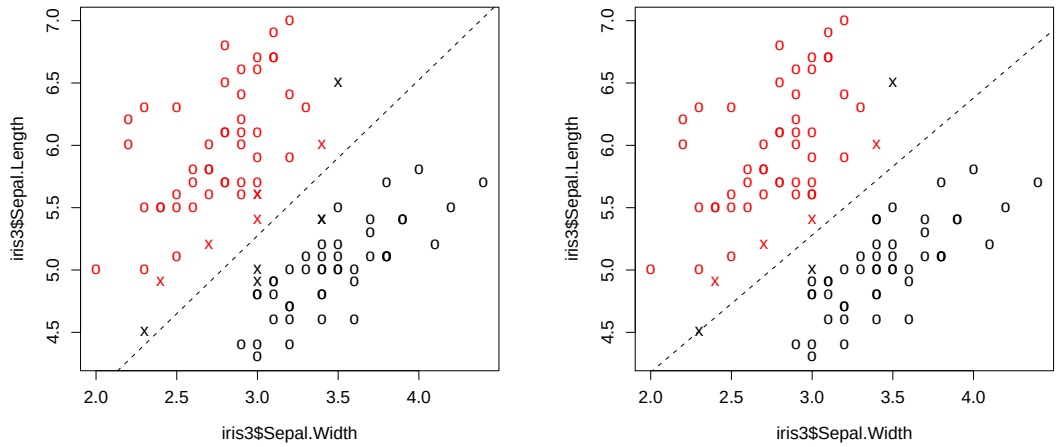

Figura 9.6: Retta di separazione nel caso di due classi (di colori differenti) e due variabili (*Sepal.Length* e *Sepal.Width*). I support vector sono indicati con una crocetta. A sinistra si ha $C = 1$, a destra il costo dell'errore è molto più penalizzato dal valore $C = 100$. Nel grafico di sinistra due punti sono mal classificati. Nel grafico di destra la retta tratteggiata che segna il confine tra le classi tende a spostarsi in modo da classificare correttamente uno dei due punti (quello in basso a sinistra).

```
[...]
10-fold cross-validation on training data:

Total Accuracy: 98 Single Accuracies:
 100 90 100 90 100 100 100 100 100 100
```

Si conclude che, per quanto riguarda il processo di rivalidazione, l'accuratezza del classificatore SVM è del 98%.

Dato che si hanno solo due gruppi e due variabili è interessante rappresentare graficamente la situazione. Nel grafico di Fig. 9.6 vi sono i dati originari, distinti per colore nelle due classi. Le crocette identificano i support vector mentre la retta tratteggiata è il confine di separazione, identificato dalla tecnica SVM, di equazione:

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

con $\mathbf{x}$ vettore di due componenti: *Sepal.Length* e *Sepal.Width*. Per rappresentare graficamente tale retta è necessario tener conto del fatto che, come detto sopra, le variabili vengono internamente standardizzate. I parametri della standardizzazione si ricavano con la chiamata:

```
> model$x.scale
$"scaled:center"
Sepal.Length  Sepal.Width
       5.485        3.099

$"scaled:scale"
Sepal.Length  Sepal.Width
   0.6487556    0.4787389
```

Si definiscono due vettori in cui memorizzare queste informazioni:

```
> media <- model$x.scale[[1]]
> varianza <-  model$x.scale[[2]]
```

Scelto arbitrariamente $x_1 = Sepal.Length$ e $x_2 = Sepal.Width$ si procede ricercando i valori di x_1 in funzione di x_2 per cui sia soddisfatta l'equazione della retta di separazione:

$$x_1 = \frac{-b - w_2\, x_2}{w_1}$$

Si definisce quindi un range appropriato per x_2:

```
> x2 <- seq(2, 5, by=0.1)
```

e si calcolano i corrispondenti valori di x_1, tenendo conto della standardizzazione:

```
> x1 <- ((-b - w[2]*(x2-media[2])/varianza[2])/w[1])*varianza[1] + media[1]
```

A questo punto il grafico di Fig. 9.6 si realizza con le chiamate seguenti:

```
> plot(iris3$Sepal.Width, iris3$Sepal.Length, col=as.numeric(iris3$Species),
+  pch=c("o","x")[1:100 %in% model$index +1])
> lines(x2, x1, lty=2)
```

Le opzioni passate alla funzione *plot* servono ad attribuire un colore diverso ai punti delle due classi e a rappresentare con una crocetta solo i punti corrispondenti ai support vector.

Per confronto, in Fig. 9.6 a destra viene presentato anche il modello con penalizzazione degli errori molto più alta ($C = 100$). Questo modello classifica in maniera scorretta solamente un caso.

9.5.2 Estensioni della tecnica SVM

Il metodo descritto finora è applicabile qualora vi siano due sole classi fra cui discriminare. Quando ve ne sono un numero $k > 2$ la funzione *svm* costruisce tutti i $k(k-1)/2$ classificatori binari possibili e assegna il caso alla classe maggiormente rappresentata (si veda [11] per una descrizione approfondita di questo e degli altri algoritmi implementati nella libreria $e1071$).

Una seconda estensione, che porta a una tecnica estremamente versatile, è la possibilità di implementare meccanismi di classificazione complessi, ossia classificatori che non siano funzioni lineari dei dati. Per una trattazione completa di questo tipo di classificatori si rimanda a [9]. L'idea di base è quella di mappare i dati in uno spazio differente e di costruire un classificatore lineare in questo spazio. Il punto di partenza è il fatto che i vettori dei dati $\mathbf{x}$ appaiono nell'Eq. 9.14 che definisce la funzione obiettivo solo tramite il loro prodotto scalare, così come appaiono nella funzione che definisce la classificazione in Eq. 9.7 solo mediante il prodotto scalare con il vettore $\mathbf{w}$.

Considerato che qualsiasi funzione $K(\mathbf{x}, \mathbf{z})$ simmetrica e semidefinita positiva è un prodotto scalare in un qualche spazio si ha che tale funzione definisce intrinsecamente una mappatura $\phi : \mathbf{x} \to \phi(\mathbf{x})$ tale che:

$$K(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{z})$$

La funzione K si dice kernel. Si noti che non è necessario conoscere esplicitamente la funzione di mappatura ϕ dato che essa non rientra mai nei calcoli; quello che occorre è solamente la forma di K. Ad esempio il funzionale L_D diventa, con l'introduzione della funzione K:

$$L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{9.17}$$

Nella implementazione di R è possibile scegliere fra quattro diversi kernel, usando l'opzione *kernel* della funzione *svm*. Oltre al caso *kernel="linear"* considerato fino a questo momento sono disponibili un kernel polinomiale (*kernel="polynomial"*), uno gaussiano (*kernel="radial"*, opzione impostata come default della funzione) e uno sigmoide (*kernel="sigmoid"*). Per ulteriori dettagli sui parametri di cui essi necessitano e sulla loro forma analitica si rimanda alla pagina di manuale della funzione *svm*.

9.6 Shrunken centroid

Un metodo recente, che consente contemporaneamente di ottenere la classificazione di soggetti in diverse classi e di valutare l'importanza delle singole variabili per la classificazione è quello degli shrunken centroids, descritto in [57]. Questa tecnica è stata sviluppata nell'ambito dell'analisi di DNA microarrays per far fronte a situazioni in cui si hanno moltissime variabili, solitamente fortemente correlate tra loro, e pochi soggetti sperimentali.

La tecnica si basa sull'estensione dalla tecnica di nearest centroid. Si supponga di avere n soggetti, classificati in K classi, su cui si misurano p variabili. Per ognuna delle classi si calcola il centroide, che per la i-esima variabile nella k-esima classe C_k è definito come:

$$\bar{x}_{ik} = \sum_{j \in C_k} \frac{x_{ij}}{n_k}$$

dove n_k è la numerosità del gruppo C_k. Per ogni soggetto la classificazione avviene misurando la sua distanza euclidea p-dimensionale dai K centroidi e assegnando il soggetto al gruppo rispetto a cui tale distanza è minore.

Nella tecnica di shrunken centroid la procedura subisce una importante modifica. Si calcola innanzitutto il centroide generale dei dati, la cui i-esima componente è:

$$\bar{x}_i = \sum_{j=1}^{n} \frac{x_{ij}}{n}.$$

L'idea della tecnica è quella di contrarre i centroidi delle classi verso il centroide generale dopo aver standardizzato l'espressione di ciascuna variabile mediante la loro deviazione standard entro classi. Il motivo di questa standardizzazione è quello di dare maggior peso a quelle variabili la cui espressione è stabile all'interno delle singole classi.

Matematicamente si valuta la quantità:

$$d_{ik} = \frac{\bar{x}_{ik} - \bar{x}_i}{m_k(s_i + s_0)} \tag{9.18}$$

con:

$$m_k = \sqrt{1/n_k - 1/n}$$

e s_i la deviazione standard entro gruppi per la i-esima variabile:

$$s_i^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{j \in C_k} (x_{ij} - \bar{x}_{ik})^2$$

dove s_0 è una costante positiva il cui compito è di "filtrare" le variabili con un basso livello di espressione. Solitamente si pone s_0 pari alla mediana dei valori di s_i (si veda [57]). Con un semplice passaggio algebrico si può riscrivere l'Eq. 9.18 nel modo seguente:

$$\bar{x}_{ik} = \bar{x}_i + m_k(s_i + s_0)d_{ik}$$

L'idea della tecnica è quindi di contrarre i valori di d_{ik} verso zero, ottenendo dei valori d'_{ik} tramite i quali si calcolano i centroidi contratti:

$$\bar{x}'_{ik} = \bar{x}_i + m_k(s_i + s_0)d'_{ik}$$

La tecnica di contazione utilizzata è detta di soft thresholding: si riduce ogni d_{ik} in modulo di una quantità Δ e si pone d'_{ik} pari a zero se si ottiene una quantità negativa:

$$d'_{ik} = \text{sign}(d_{ik})(|d_{ik}| - \Delta)_+$$

dove il pedice "+" indica di prendere solo la parte positiva della parentesi e zero altrimenti. Dopo la procedura di contrazione i soggetti sono classificati come nel metodo nearest centroid, facendo però uso dei nuovi centroidi.

Al variare del parametro di soglia Δ si avrà che per alcune variabili il valore di d'_{ik} sarà nullo per tutti i K gruppi. In tal caso esse non contribuiscono alla classificazione finale dei soggetti e possono essere eliminate. Si ha quindi un modo per estrarre dall'insieme di p variabili un sottoinsieme ottimale per discriminare fra i gruppi: si ricerca il valore di Δ che minimizza l'errore di classificazione degli n soggetti (usualmente valutato mediante 10-fold cross validation) e si utilizzano per la classificazione solo le variabili i cui valori d'_{ik} non siano nulli per tutte le classi.

In R è possibile utilizzare questa tecnica dopo aver installato la libreria aggiuntiva *pamr*, scaricabile dal sito della distribuzione.

Esempio

Si consideri nuovamente il dataset *iris*. In questo caso la situazione non è particolarmente appropriata per la tecnica in esame, pensata per lavorare al meglio quando il numero delle variabili è molto maggiore del numero delle osservazioni. In ogni caso la funzione per classificare i 150 fiori in base alle 4 variabili su di essi misurate è disponibile con la chiamata alla funzione *pamr.train*, che necessita di un poco di lavoro per formattare i dati in input:

```
> x <- t(iris[,1:4])
> y <- iris[,5]
> label <- dimnames(iris)[[2]][1:4]
> mydata <- list(x=x, y=y, geneid=1:4, genenames=label)
```

Si deve cioè preparare una lista di quattro elementi: x è la matrice che contiene per colonna i soggetti e per riga le variabili (chiamate geni, dato l'ambito in cui la funzione è stata sviluppata), y è il vettore con le classificazioni dei vari soggetti, *geneid* e *genenames* sono gli identificatori e le etichette con cui identificare le variabili (in questo caso i numeri progressivi da 1 a 4 e i nomi delle variabili, estratti dal dataframe originario). Si passa quindi a classificare i 150 fiori:

```
> sc.train <- pamr.train(mydata)
> sc.train
Call:
pamr.train(data = mydata)
   threshold nonzero errors
1   0.000    4       6
2   0.841    4       7
3   1.682    4       10
[...]
29 23.546    1       57
30 24.387    0       100
```

Il risultato è una tabella di tre colonne. Sulla prima vi è il valore progressivo del parametro di soglia, sulla seconda il numero di variabili che sopravvivono a quella soglia, sulla terza il numero di soggetti mal classificati. Il risultato migliore si ha per parametro di soglia pari a 0 (prima riga della tabella di output).

Per convalidare il risultato si utilizza comunemente il metodo di rivalidazione 10-fold, che può essere applicato nel modo seguente:

```
> set.seed(120)   # per una analisi riproducibile
> sc.cv <- pamr.cv(sc.train, mydata)
>  sc.cv
Call:
pamr.cv(fit = sc.train, data = mydata)
   threshold nonzero errors
```

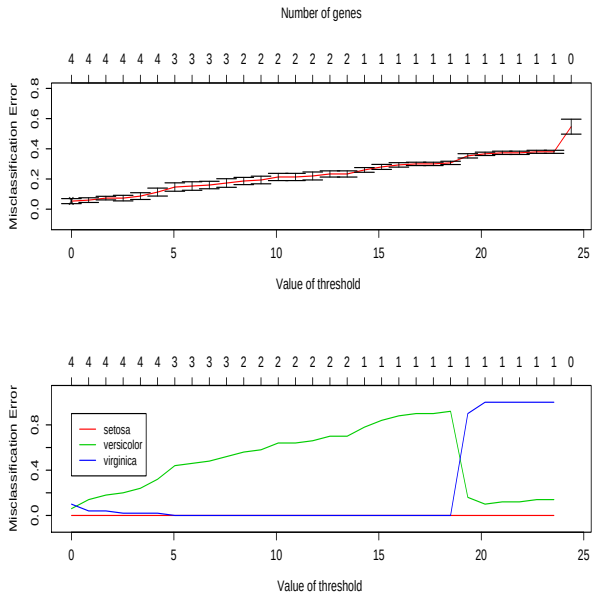

Figura 9.7: In alto: andamento dell'error-rate in rivalidazione al variare del parametro di soglia (e del corrispondente numero di geni residui). Si vede che la miglior classificazione si ottiene per soglia pari a 0. In basso: error-rate nelle singole classi.

```
1   0.000    4       8
2   0.841    4       10
3   1.682    4       10
[...]
29 23.546    1       57
30 24.387    0       84
```

La funzione *pamr.cv* accetta in input il risultato della funzione *pamr.train* e il set di dati utilizzato. La tabella in output è analoga a quella precedente. Anche in questo caso si vede che il minor numero di errori di classificazione si ottiene per soglia pari a zero, facendo uso di tutte e 4 le variabili.

Per esaminare la classificazione ottenuta in rivalidazione si può usare la seguente chiamata:

```
> pamr.confusion(sc.cv, 0)
           setosa versicolor virginica Class Error rate
setosa         50          0         0             0.00
versicolor      0         47         3             0.06
virginica       0          5        45             0.10
Overall error rate= 0.053
```

La funzione *pamr.confusion* accetta due argomenti: la classificazione da valutare (che può essere indistintamente il risultato della funzione *pamr.train* o di *pamr.cv*) e il valore di soglia da utilizzare. Oltre all'error-rate globale, dato sull'ultima riga, si hanno anche le stime degli error-rate nelle singole classi. Una visualizzazione grafica dell'andamendo dell'error-rate in rivalidazione al variare del parametro di soglia si può ottenere con la chiamata:

```
> pamr.plotcv(sc.cv)
```

che produce il grafico di Fig. 9.7.

In questo particolare esempio tutte le variabili sono selezionate come rileventi. Più comunemente si avrà il caso in cui la miglior classificazione si ottiene per parametro di soglia diverso da zero, con solo alcune delle variabili importanti per la classificazione. Supponiamo che si sia trovata la miglior

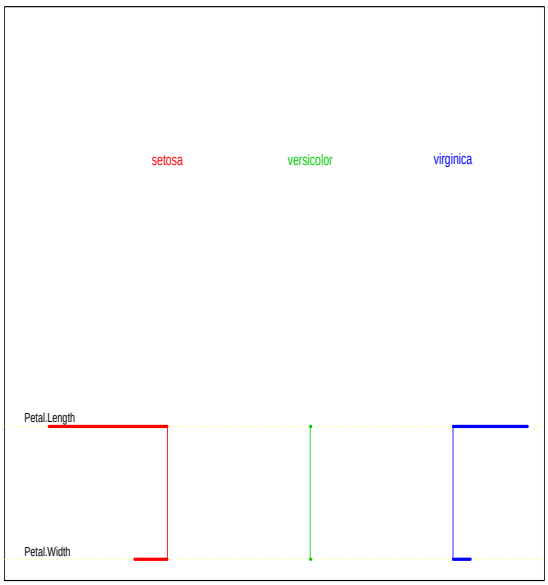

Figura 9.8: Posizione dei centroidi dei tre gruppi per parametro di soglia uguale a 10. Si nota che solo due variabili contribuiscono alla classificazione.

classificazione in corripondenza di un parametro di soglia pari a 10. Ci si chiede quali siano le variabili (i geni nella notazione della tecnica) che contribuiscono a questo livello. Per rispondere a tale domanda si usa la chiamata:

```
> pamr.listgenes(sc.train, mydata, 10, sc.cv)
     id setosa-score versicolor-score virginica-score av-rank-in-CV prop-selected-in-CV
[1,] 3  -1.6613      0                1.0456          1             1
[2,] 4  -0.462       0                0.2472          2             1
```

La funzione *pamr.listgenes* accetta un minimo di tre argomenti: la classificazione ottenuta dalla funzione *pamr.train*, i dati su cui lavorare e il valore di soglia, in questo caso posto uguale a 10. È possibile passare, come in questo caso, un quarto argomento ossia il risultato della funzione *pamr.cv*. Il risultato della chiamata è una tabella che ha per riga solo i geni che sopravvivono nella classificazione alla soglia impostata. Nella prima colonna della tabella si ha l'identificatore del gene; si vede che le sole variabili rilevanti sono la terza e la quarta, ossia lunghezza e larghezza del petalo. Nelle successive tre colonne si hanno le coordinate dei centroidi dei tre gruppi di iris. Nelle ultime due colonne – che appaiono solo se si passa alla funzione anche il quarto argomento opzionale – si ha lo studio in rivalidazione dei geni selezionati: la loro importanza (av-rank-in-CV) e la proporzione di volte in cui tali geni sono stati selezionati nel processo bootstrap (prop-selected-in-CV). Si vede che, a questo livello di soglia, entrambe le variabili sono state selezionate nel 100% dei campioni di rivalidazione e che la variabile più importante risulta essere la terza.

È anche possibile visualizzare i centroidi dei tre gruppi, come in Fig. 9.8, mediante la chiamata:

```
> pamr.plotcen(sc.train, mydata, 10)
```

Il grafico ottenuto mette in evidenza quali siano le variabili rilevanti per la classificazione.

9.7 Metodi di selezione di variabili

Nel campo della indagine genetica è oggi abbastanza facile essere alle prese con campioni multivariati contenenti moltissime variabili. Lo sviluppo della tecnologia di DNA micro-array ha infatti reso possibile agli sperimentatori di disporre simultaneamente dell'esperessione di un numero consistente di

geni in un singolo esperimento. Questo spiega come mai nel corso degli ultimi anni si siano sviluppate una schiera di tecniche tese a ricercare in un campione multivariato le caratteristiche rilevanti per spiegare la classificazione clinica sotto indagine.

9.7.1 Selezione di variabili: tecnica RFE

Una delle metodologie più recenti – che coinvolge la tecnica SVM – è l'algoritmo Recursive Feature Elimination (RFE), proposto da Guyon et al. nel 2002 [30]. L'algoritmo originale, sviluppato nel caso di classificazione dicotomica, usa una tecnica SVM con kernel lineare per identificare, con un processo ricorsivo, le caratteristiche più importanti del campione multivariato. La tecnica parte considerando tutte le variabili. A ogni passo vengono eseguite le seguenti operazioni:

1. si fitta il classificatore SVM, ottenendo i valori w_i;
2. si classificano le variabili in gioco a seconda del valore di w_i^2;
3. si rimuove la variabile con il valore di w_i^2 minore e si ritorna al punto 1.

Per ragioni computazionali il passo 3 può essere modificato rimuovendo più di una variabile per volta, escludendo ad esempio il 10% delle variabili ancora in esame con pesi più bassi. In questo caso l'ordinamento delle variabili è chiaramente più impreciso.

La metodologia proposta è implementata nella libreria rfe (a cura di C. Ambroise), attualmente non disponibile on-line, e che diverrà presumibilmente accessibile all'URL:

http://www.hds.utc.fr/∼ambroise/doku.php?id=softwares:rfe

La libreria rfe può essere usata sia in caso di classificazione dicotomica che policotomica (per i dettagli dell'implementazione si rimanda alla pagina web segnalata in precedenza).

Le funzioni della libreria rfe hanno la limitazione di non accettare in input il valore della costante C[2]. Per risolvere il problema è possibile scaricare una versione della libreria, modificata dall'autore di queste note, dall'indirizzo web:

http://mail.df.unipi.it/∼valle/statistics.html

In questa versione è implementata la possibilità di passare parametri aggiuntivi, tra cui *cost*, alle funzioni della libreria. Sono anche implementate due funzioni aggiuntive, $rfe.cv2$ e $plot.rfe.cv2$, che verranno usate nel seguente esempio.

Esempio

Si può utilizzare la tecnica RFE facendo uso del dataset *iris*. Il primo passo è caricare la libreria:

```
> library(rfe)
```

La procedura di classificazione delle variabili è disponibile tramite la funzione $rfe.fit$:

```
> fit <- rfe.fit(iris[,1:4], iris$Species, speed="low", cost=10)
```

Il primo argomento della funzione è il dataset contenente le variabili multivariate da esaminate, il secondo la variabile che specifica la classificazione dei soggetti. Il terzo argomento serve per scegliere la tecnica di eliminazione: *speed="low"* richiede che a ogni passo l'algoritmo elimini solo una variabile, mentre *speed="high"* (valore di default) rimuove la metà delle variabili a ogni passo. Infine vi è il valore del parametro *cost*. Il vettore contenente le variabili in ordine di importanza si può ottenere con la chiamata:

```
> fit$Flist
[1] 3 4 1 2
```

da cui si deduce che la variabile più importante sembra essere *Petal.Length*. Occorre comunque prestare attenzione al fatto che valori diversi del parametro *cost* producono ordinamenti differenti:

[2]La versione attualmente disponibile, a cui si riferiscono queste note, è la 0.2.

```
> fit <- rfe.fit(iris[,1:4], iris$Species, speed="low", cost=1)
> fit$Flist
[1] 4 3 2 1
```

Per scegliere in modo appropriato un sottoinsieme di variabili che descivono bene la classificazione in esame la procedura migliore è sottoporre a rivalidazione bootstrap la classificazione ottenuta considerando tutti i sottoinsiemi di variabili identificati dall'algoritmo ricorsivo. Si sceglierà quindi il modello di complessità minore (con meno variabili) che soddisfi alla condizione 1SE (ossia, come visto per gli alberi di classificazione, l'error rate del modello selezionato deve essere minore o uguale all'error rate del modello migliore più il suo errore standard).

La procedura che implementa questo schema è definita nella funzione $rfe.cv2$:

```
> fit <- rfe.cv2(iris[,1:4], iris$Species, speed="low", cost=10, nfold=150)
> fit
$Flist
[1] 3 4 1 2

$error.rate
[1] 0.04666667 0.04666667 0.03333333 0.04000000

$conf.int.low
[1] 0.02938711 0.02938711 0.01862767 0.02394640

$conf.int.up
[1] 0.06394623 0.06394623 0.04803900 0.05605360
```

La funzione accetta, oltre agli argomenti discussi in precedenza, l'argomento $nfold$. Il valore di tale parametro serve per la procedura k-fold cross-validation, che divide il campione in $nfold$ sottocampioni. Se si desidera utilizzare la procedura leave-one-out cross-validation è necessario impostare il valore di $nfold$ alla dimensione campionaria (in questo caso 150 soggetti). In output si ha la lista della variabili in ordine di importanza, la valutazione dell'error rate dei modelli annidati (il primo valore di error-rate si riferisce al modello con la sola variabile migliore, il secondo al modello con le due variabili migliori e così via) e gli estremi dell'intervallo (*error.rate*-1SE, *error.rate*+1SE). Una visualizzazione grafica del risultato, che semplifica la sua interpretazione, si ottiene con la chiamata:

```
> plot.rfe.cv2(fit)
```

Il risultato, in Fig. 9.9, mostra che il modello migliore in assoluto comprende tre variabili, e ha un error rate del 3.3%, ma sulla base della maggior semplicità si preferirà il modello a una sola variabile (error-rate del 4.7%), che non differisce significativamente dal modello ottimale.

Come nota finale, si ricordi sempre che il risultato del processo di selezione dipende dal valore del parametro *cost*: scelte diverse possono portare a modelli con un numero differente di variabili.

9.7.2 Selezione di variabili: Random Forests

Fra le ultime tecniche proposte per realizzare una selezione automatica di variabili vi è quella di Diaz-Uriarte e Alvarez de Andrez nel 2005 [21]. Questa metodologia sfrutta la costruzione di una Random Forest e la sua capacità di classificare in ordine di importanza le variabili che rientrano nel processo decisionale da essa implementato.

La tecnica proposta è di tipo ricorsivo. Il punto di partenza è il fit di una Random Forest sul campione di partenza. A ogni ciclo vengono quindi compiuti i seguenti due passi:

1. Si elimina dal set di variabili quelle che hanno minor indice di importanza. Una buona scelta è quella di eliminare il 20% di variabili a ogni passo [21]. Per evitare problemi di overfitting è buona norma non ricalcolare gli indici di importanza delle variabili, ma utilizzare l'ordinamento ottenuto sul campione di partenza [53].

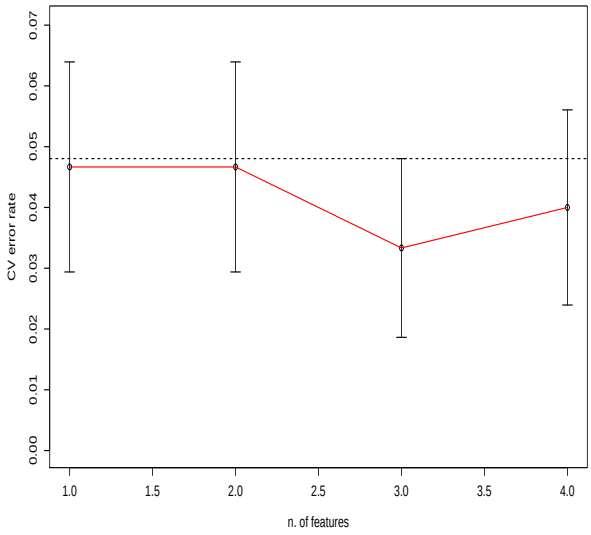

Figura 9.9: Error-rate in rivalidazione leave-one-out. Risultati ottenuti con la tecnica RFE. Il modello da adottare sulla base del criterio di massima semplicità contiene solo una variabile, mentre il modello migliore in assoluto ne contiene tre.

2. Si fitta nuovamente una Random Forest, utilizzando le variabili sopravvissute al passo precedente, e si calcola l'OOB error.

I valori degli errori OOB sono quindi usati per scegliere il modello migliore, secondo la regola 1SE.

In R è possibile avvalersi di questa tecnica installando la libreria aggiuntiva *varSelRF*, scaricabile presso il sito standard di distribuzione.

Esempio

Usando il dataset standard *iris* si può illustrare un'applicazione della tecnica in esame. Il primo passo consiste nel caricare la libreria aggiuntiva:

```
> library(varSelRF)
```

La funzione per ricercare il miglior sottoinsieme di variabili è *varSelRF*, che può essere usata nel modo seguente:

```
> fit.rf <- varSelRF(iris[,1:4], iris$Species)
```

È possibile passare alla funzione alcune opzioni per controllarne il comportamento. Fra le principali si ricordano: *ntree* (default 5000), il numero di alberi che fanno parte della foresta iniziale (quella usata per ordinare le variabili per indice di importanza); *ntreeIterat* (default 2000), il numero di alberi delle foreste successive alla prima; *mtryFactor* (default 1), un fattore moltiplicativo per il valore del parametro *mtry* della funzione *randomforest* (si veda Sec. 9.3 per i dettagli sull'uso del parametro *mtry*), utile per selezionare il numero di variabili da considerare a ogni split; *vars.drop.frac* (default 0.2), frazione di variabili residue da eliminare a ogni passo.

Il risultato della chiamata precedente è:

```
> fit.rf

Backwards elimination on random forest; ntree =  5000 ;  mtryFactor =  1
```

```
 Selected variables:
[1] "Petal.Length" "Petal.Width"

 Number of selected variables: 2
```

da cui si conclude che il metodo seleziona un modello con le variabili $Petal.Length$ e $Petal.Width$.

9.7.3 Stabilità del processo di selezione delle variabili

Il problema di estrarre un set di variabili che riassumano al meglio le capacità classificatorie dell'intero insieme di partenza può spesso avere diverse soluzioni, tutte ugualmente valide dal punto di vista della discriminazione, ma che condividono solo poche variabili fra loro. Questo inconveniente, detto mancanza di stabilità della soluzione deve essere sempre ben presente in fase di analisi (si veda [51] per una discussione accurata delle problematiche connesse). Sottostimarne l'importanza porta a un falso senso di sicurezza nei confronti del modello individuato dagli algoritmi di selezione. Ovviamente il problema principale che la mancanza di stabilità pone al ricercatore è che causa seri dubbi sulla interpretazione biologica dei risultati.

Per stimare quanto incida la mancanza di affidabilità del modello individuato in fase di analisi è possibile affidarsi a tecniche bootstrap. Si ripete cioè il processo di selezione su sottocampioni bootstrap ricavati dal set di dati originario. Si controlla poi quante volte i geni selezionati sul campione di partenza sono selezionati anche nei campioni bootstrap. È chiaro che questa procedura non costituisce una cura al problema, che deriva dal fatto che le procedure di selezione sono troppo strettamente legate al campione in esame, ma almeno costituisce una spia di quanto possa essere ritenuto valido e generalizzabile un risultato. L'unica vera cura consite nell'allargare il campione in esame, nella speranza di riuscire a isolare un comportamente generale da quello dovuta a semplice casualità.

9.8 Significance Analysis of Microarrays (SAM)

Nel campo dell'analisi dei dati ottenuti da DNA microarray è stata recentemente proposta una tecnica mirata a selezionare, tra un vastissimo insieme di geni, quelli che sono espressi in modo differente tra diversi stati biologici. Tale metodica, i cui dettagli matematici sono presentati in [55], ha avuto un'immediata diffusione tanto da essere tra i metodi più popolari per l'analisi di microarray.

Nel seguito si considera il caso di due soli gruppi, ma la tecnica è generalizzabile anche per tre o più classi. Siano A e B sono due stati biologici tra cui si vuole valutare se p geni sono espressi in maniera differente. La tecnica SAM valuta la differenza di espressione dei geni fra i due stadi in modo simile al test t. Per l'i-esimo gene si definisce la statistica:

$$d_i = \frac{\bar{x}_{iA} - \bar{x}_{iB}}{s_i + s_0}$$

dove $\bar{x}_{iA}$ e $\bar{x}_{iB}$ sono le medie dell'espressione del gene nei due diversi stadi, s_i è la deviazione standard pool del gene in questione (valutata in modo identico al caso di test t). Dato che la valutazione di s_i nel caso di piccoli campioni, situazione tipica nel caso di indagine su DNA microarray, è imprecisa e soggetta a forte variabilità si introduce a denominatore il parametro s_0 (da calcolare a partire dai dati, si veda [55] per ulteriori dettagli) il cui compito è quello di fungere da stabilizzatore, riducendo il valore di d_i per geni che assumono valore pressoché costante sui due stati biologici.

Per valutare la significatività della statistica d_i si procede permutando casualmente i valori del gene tra i due gruppi e ricalcolando il valore di d_i. Si ripete questa procedura un certo numero di volte e si assume la media d_{Ei} dei valori calcolati come valore atteso per la statistica in esame. Un output standard della tecnica SAM è il grafico dei valori campionari d_i contro i valori attesi d_{Ei}; saranno espressi in modo significativamente differente nei due stati biologici quei geni il cui valore campionario di d_i si discosta (in eccesso o in difetto) dal valore atteso più di una soglia Δ.

9.8.1 Il problema della molteplicità

Come in tutti gli ambiti in cui si ha a che fare con test multipli si pone il problema di come correggere per l'effetto di molteplicità. Cercare di controllare il family-wise error rate (FWER), ad esempio mediante correzione di Bonferroni o simili, porta a test troppo conservativi soprattutto in questo ambito in cui si hanno spesso migliaia di test simultanei su variabili fortemente correlate tra loro. Dato che in molti casi il controllo completo del FWER è un criterio eccessivamente restrittivo è possibile rilassare le richieste accettando un certo numero di falsi positivi a patto che il loro rate sia basso. Questo approccio conduce all'impiego del false discovery rate (FDR) come criterio di valutazione (si veda [52] per una dettagliata trattazione).

Per stimare il FDR (ossia la proporzione di geni identificati come significativi per pure fluttuazioni), fissato un valore di Δ, si definiscono due valori di soglia, pari al minimo d_{Mi} tra i geni significativamente espressi e il massimo d_{mi} tra quelli significativamente repressi. Tali valori dipendono chiaramente dalla scelta di Δ. Si conta quindi, per ogni permutazione casuale, il numero di geni per cui si ha $d_i > d_{Mi}$ o $d_i < d_{mi}$. Mediando su tutte le permutazioni si ha il numero stimato di geni falsamente significativi. Dalla descrizione della procedura risulta chiaro che il parametro di soglia permette di selezionare diversi insiemi di geni, con differente FDR.

In R la tecnica SAM è implementata nella libreria *samr*, scaricabile dal sito della distribuzione.

Esempio

Si supponga di avere un set di dati relativi a 20 pazienti, divisi in due gruppi, entrambi di 10 soggetti. Su ogni soggetto si valuta l'espressione di 1000 geni e si vuol verificare quali geni caratterizzino con la loro espressione i due gruppi. Per comodità si lavora su dati simulati:

```
> set.seed(123)   # per una analisi riproducibile
> x <- matrix(rnorm(1000*20), ncol=20)  # matrice di espressione genica
> y <- gl(2, 10)   # fattore di gruppo
> x[1:5, 1:10] <- x[1:5, 1:10] + 2
> x[6:10, 1:10] <- x[6:10, 1:10] - 2
```

Le ultime due istruzioni modificano l'espressione dei primi 5 geni nel primo gruppo (aumentandola di 2 punti) e riducono quella dei successivi 5 di una analoga quantità. Si vuole verificare se la tecnica SAM riesce a selezionare questi geni modificati, separandoli dagli altri. Come visto nel caso di shrunken centroid è necessario inserire i dati in una lista, contenente la matrice delle espressioni geniche su cui operare (con in soggetti per colonna), il vettore della classificazione in gruppi, gli identificatori e i nomi dei geni (in questo caso semplicemente il numero progressivo di riga):

```
> data <- list(x=x, y=y, geneid=1:nrow(x), genenames=1:nrow(x), logged2=TRUE)
```

l'opzione *logged*2 serve per specificare che le espressioni geniche sono state normalizzate (mediante trasformazione logaritmica). Per condurre l'analisi si usa la funzione *samr*:

```
> samr.obj <- samr(data, resp.type="Two class unpaired", nperms=100)
perm= 1
[...]
perm= 100
```

che accetta vari argomenti. Nell'esempio in questione essi sono: i dati su cui lavorare, il tipo di problema in esame *resp.type* (si veda la pagina di manuale della funzione per la lista completa dei problemi trattabili) e il numero di permutazioni random da eseguire per valutare il FDR.

La valutazione del FDR al variare del parametro di soglia Δ si ottiene con la seguente chiamata:

```
> delta.table <- samr.compute.delta.table(samr.obj)
```

L'output della funzione è una tabella di 8 colonne:

```
> delta.table
             delta # med false pos 90th perc false pos # called median FDR
 [1,] 0.0000000000        1021.952           1022.9760     1000  1.0219520
 [2,] 0.0006055627        1014.784           1018.8800      995  1.0198834
 [3,] 0.0024222509         989.184            996.3520      973  1.0166331
 [4,] 0.0054500646         948.224            960.5120      941  1.0076769
 [5,] 0.0096890038         542.208            563.2000      544  0.9967059
[...]
[17,] 0.1550240604           5.120              9.3184       19  0.2694737
[18,] 0.1750076307           4.096              8.1920       17  0.2409412
[19,] 0.1962023264           2.048              5.1200       13  0.1575385
[20,] 0.2186081477           0.000              2.0480       11  0.0000000
[21,] 0.2422250944           0.000              2.0480       11  0.0000000
[22,] 0.2670531665           0.000              2.0480       11  0.0000000
[23,] 0.2930923642           0.000              2.0480       11  0.0000000
[24,] 0.3203426873           0.000              1.0240       10  0.0000000
[25,] 0.3488041359           0.000              1.0240       10  0.0000000
[26,] 0.3784767099           0.000              1.0240        9  0.0000000
      90th perc FDR         cutlo        cuthi
 [1,]     1.0229760 -0.0007324644 1.284597e-03
 [2,]     1.0240000 -0.0057877900 3.325467e-03
 [3,]     1.0240000 -0.0057877900 2.846975e-02
 [4,]     1.0207354 -0.0328393668 4.194779e-02
 [5,]     1.0352941 -0.0423624139 5.911797e-01
[...]
[17,]     0.4904421 -1.1440994822 1.109009e+00
[18,]     0.4818824 -1.1440994822 1.178326e+00
[19,]     0.3938462 -1.2213644144 1.238785e+00
[20,]     0.1861818 -1.3484447126 1.559516e+00
[21,]     0.1861818 -1.3484447126 1.559516e+00
[22,]     0.1861818 -1.3484447126 1.559516e+00
[23,]     0.1861818 -1.3484447126 1.559516e+00
[24,]     0.1024000 -1.4097076850 1.559516e+00
[25,]     0.1024000 -1.4097076850 1.559516e+00
[26,]     0.1137778 -1.8073971630 1.559516e+00
```

I valori interessanti sono sulla prima colonna, dove si hanno valori progressivi del parametro di soglia Δ, nella quarta dove vi è il numero di geni dichiarati significativi a quella soglia, nella seconda con il numero stimato di geni dichiarato significativo in maniera errata e nella quinta con il valore stimato di FDR (il rapporto tra seconda e quarta colonna). Si vede che nella ventesima riga della tabella, per $\Delta = 0.2186$, si ha un FDR stimato pari a zero e 11 geni identificati come significativi.

Per avere l'elenco dei geni dichiarati significativi ad una certa soglia si usa la chiamata:

```
> genes <- samr.compute.siggenes.table(samr.obj, 0.2186, data, delta.table)
```

dove il primo argomento è il risultato della chiamata a *samr*, il secondo il valore di Δ desiderato, il terzo la lista dei dati e il quarto il risultato della precedente chiamata a *samr.compute.delta.table*. La lista dei geni significativamente più espressi nel secondo gruppo si può visualizzare con la sintassi:

```
> genes$genes.up
     Row Gene ID Gene Name Score(d) Numerator(r) Denominator(s+s0) Fold Change
[1,]  11      10        10 2.049343     2.183624          1.065524    4.964468
[2,]   7       6         6 1.754805     2.025525          1.154274    3.873756
[3,]  10       9         9 1.632573     1.898432          1.162846    4.053598
[4,]   9       8         8 1.583674     1.784795          1.126996    2.823322
```

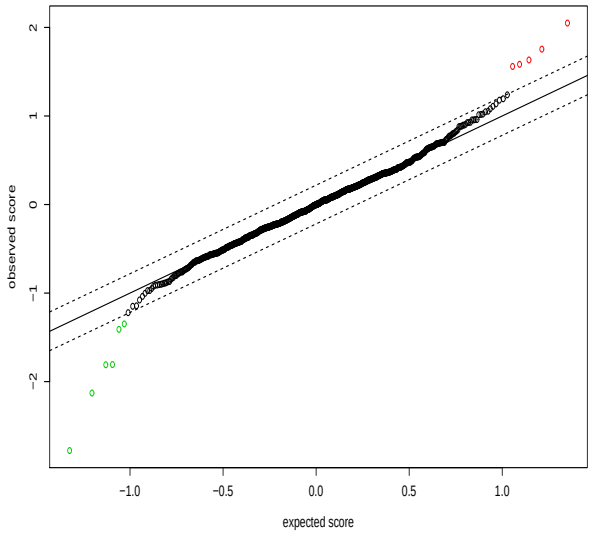

Figura 9.10: Analisi di significatività dei valori campionari della statistica d_i per parametro di soglia $\Delta = 0.2186$. 5 geni (in rosso) risultano over-espressi nel secondo gruppo e 6 (in verde) soppressi.

```
[5,]   8       7         7 1.559516     1.974864          1.266331    5.763571
     q-value(%)
[1,]          0
[2,]          0
[3,]          0
[4,]          0
[5,]          0
```

Dalla seconda colonna si vede che sono esattamente i geni da 6 a 10, che in fase di input erano stati repressi nel primo campione. La colonna etichettata come "q-value" riporta le percentuale di volte in cui ciascun gene risulta come falsamente espresso nei campioni permutati (in questo caso tutti i geni hanno valore q pari a 0). La tabella analoga per i geni significativamente soppressi nel secondo gruppo si ottiene con la chiamata:

```
> siggenes.table$genes.lo
     Row Gene ID Gene Name  Score(d) Numerator(r) Denominator(s+s0) Fold Change
[1,]   3       2         2 -2.778476    -2.948745          1.061281   0.1288060
[2,]   5       4         4 -2.128768    -2.235772          1.050266   0.2035256
[3,]   6       5         5 -1.809455    -2.308352          1.275716   0.1317912
[4,]   4       3         3 -1.807397    -2.115356          1.170388   0.2005731
[5,]   2       1         1 -1.409708    -1.686497          1.196345   0.3692400
[6,] 750     749       749 -1.348445    -1.606363          1.191271   0.3620386
     q-value(%)
[1,]          0
[2,]          0
[3,]          0
[4,]          0
[5,]          0
[6,]          0
```

In questo caso, oltre ai 5 geni alterati in fase di input, viene selezionato anche il gene 749.

Il classico plot dei valori di d_i contro d_{Ei}, presentato in Fig. 9.10, si realizza con la chiamata:

```
> samr.plot(samr.obj, 0.2186)
```

Esempio

Si supponga ora che i dati dell'esempio precedente si riferiscano a una situazione in cui i campioni sono appaiati. Ad esempio questa situazione si presenta quando, su uno stesso paziente, si campiona del tessuto sano e del tessuto tuomorale. Su ogni campione di tessuto si valuta l'espressione di 1000 geni e si vuol verificare quali geni caratterizzino con la loro espressione i due tessuti. Il problema si tratta in modo simile al precedente, con la variante che il vettore y che classifica i pazienti deve tenere conto dell'appaiamento individuale. Per convenzione i campioni prelevati dallo stesso paziente vengono identificati con coppie di numeri del tipo (-k, k). Se i primi 10 campioni si riferiscono a tessuti sani e i seguenti 10 a tessuti tumorali, con i pazienti disposti nello stesso ordine, il vettore di classificazione si definisce nel modo seguente:

```
> y.p <- c(-(1:10), 1:10)
> y.p
 [1]  -1 -2 -3 -4 -5 -6 -7 -8 -9 -10  1  2  3  4  5  6  7  8  9  10
```

L'analisi procede quindi come visto in precedenza:

```
> data.p <- list(x=x, y=y.p, geneid=1:nrow(x), genenames=1:nrow(x), logged2=TRUE)
> samr.obj.pair <- samr(data.p, resp.type="Two class paired", nperms=100)
```

Si noti che l'opzione *resp.type* riflette il fatto che l'analisi viene fatta su dati appaiati.

9.9 Selezione delle variabili per modelli lineari e GLM

Spesso, data una variabile dipendente y e un insieme di p predittori, si vuole di stabilire quale sia il "migliore" sottoinsieme di predittori da utilizzare in una modellizzazione del fenomeno (sia esso un modello lineare, un GLM o una regressione di Cox). A disposizione dello sperimentatore vi sono numerose tecniche che permettono di eseguire tale operazione in maniera automatizzata. Questa procedure sono al tempo stesso vantaggiose e pericolose. La loro utilità consiste nel fatto che è spesso desiderabile selezionare fra molti modelli che riescono a spiegare un fenomeno il più semplice (quindi quello con meno predittori), dato che l'aggiunta di un predittore inessenziale ad un modello comporta una inutile perdita di gradi di libertà e può riflettersi in un peggioramento delle stime dei coefficienti. Ciò è particolarmente vero se il predittore che si aggiunge è quasi collineare agli altri. D'altra parte l'uso di tecniche automatiche può escludere dal modello finale variabili la cui rilevanza nel contesto è manifestamente accertata da studi esistenti in letteratura. In questi casi è preferibile far rientrare tali variabili nel modello, anche se non risultano statisticamente significative, dato che la loro presenza può alterare le stime degli altri coefficienti. In secondo luogo è fondamentale comprendere che il fatto che una variabile non compaia nel modello finale non implica affatto che essa sia scorrelata dalla variabile dipendente. La conclusione corretta è che essa non aggiunge nessuna spiegazione della variabilità fra soggetti oltre a quella fornita dal modello che non la contiene. Se l'obiettivo dell'indagine è quello di stabilire se i valori della variabile dipendente siano correlati con quelli di tale variabile, concludere solo sulla base del modello finale multivariato può condurre a fraintendere completamente le indicazioni dei dati. Per una discussione più dettagliata dei problemi che stanno alla base dell'uso di algoritmi di selezione automatica si veda Sec. 9.9.3.

9.9.1 Procedure di selezione

Le tecniche di selezione automatica delle variabili si dividono in tre categorie generali. Nel seguito esse sono brevemente esaminate una per una.

Backward elimination

La tecnica più semplice si basa sull'eliminazione gerarchica di variabili. Il punto di partenza è il modello completo che contiene tutti i predittori. Ad ogni passo si elimina il predittore meno significativo a patto che la sua significatività sia superiore a un livello α_c scelto a priori. La procedura si arresta quando nessun predittore può essere eliminato. Il valore α_c (valore P per rimozione) viene comunemente scelto tra 0.15-0.20.

Forward selection

È la tecnica inversa della precedente. Si parte con il modello nullo che non contiene nessun predittore e ad ogni passo si verifica la significatività di ognuno di essi aggiungendolo al modello. Si sceglie il predittore con valore P più basso, a patto che sia inferiore a α_c. La procedura si arresta quando nessun predittore può essere aggiunto.

In genere le due tecniche portano a modelli finali diversi. Il modello ottenuto con tecniche di eliminazione tende ad avere più predittori rispetto a quello ottenuto con tecniche di inserimento.

Stepwise regression

Questa tecnica mista combina le due precedenti. Ad ogni passo si tenta sia una eliminazione che un inserimento di predittore, dato che variabili inserite o eliminate in stadi precedenti possono cambiare la loro influenza a seconda di quali altre variabili siano state in seguito aggiunte o tolte dal modello.

Il notevole vantaggio portato da tecniche gerarchiche come quelle appena presentate è quello di richiedere la valutazione di un numero non troppo elevato di modelli. Se ci sono p predittori una tecnica di eliminazione o inserimento ha una complessità computazionale dell'ordine di p^2, mentre tutti i possibili modelli sono ben 2^p. Visto che non vengono valutati tutti i modelli, ma solo una porzione, vi è però la possibilità di lasciarsi sfuggire il modello "migliore".

9.9.2 Procedure basate su criteri

Il problema principale degli algoritmi appena presentati è che risiedono sulla valutazione di un valore P per i coefficienti del modello e sul suo confronto con un valore α_c deciso a priori. Dato che vengono eseguiti una serie di confronti simultanei la validità di questi valori P è certamente dubbia. Tecniche più moderne si basano sulla valutazione dei modelli fittati in accordo con un criterio dato. La valutazione di tutti i modelli viene evitata fittando solo i modelli più probabili, in modo simile alle tecniche stepwise. I criteri più comunemente usati sono AIC (Akaike Information Criterion) e BIC (Bayes Information Criterion) che contengono informazioni sia sulla bontà del modello sia sul numero di predittori che esso contiene. Essi sono definiti come:

$$\begin{aligned} AIC &= -2\ \log L + 2p \\ BIC &= -2\ \log L + p\ \log n \end{aligned}$$

dove n è il numero di osservazioni, p il numero di predittori e L la likelihood del modello fittato. Per modelli lineari $-2\ \log L$ è un valore legato alla devianza d'errore d_E^2 dalla relazione:

$$-2\ \log L = n\ \log \frac{d_E^2}{n}$$

Si ha quindi che i due criteri assumo valore elevato se il modello fitta male i dati (alto valore di $-2\ \log L$) oppure se ci sono molti predittori (alto valore di p). L'obiettivo è trovare un modello che minimizzi il criterio scelto.

Nelle definizioni di AIC e BIC si vede che l'unica differenza è nel peso che i due criteri danno alla presenza dei predittori. Dato che già per campioni di taglia modesta ($n \geq 8$) si ha $\log n > 2$, il criterio BIC porta a prediligere modelli con meno predittori rispetto a AIC.

In R è possibile utilizzare una tecnica di questo genere tramite la funzione *step* che permette di usare questi e altri criteri per valutare la bontà di un modello. La funzione *step* accetta, oltre al nome del modello su cui lavorare, vari argomenti. L'opzione k permette di specificare il peso da attribuire al parametro p (numero di predittori nel modello) durante la valutazione dell'indice del modello. Il suo valore di default è $k = 2$, che corrisponde all'uso del criterio AIC; modificando tale valore si può quindi richiedere facilmente l'uso del criterio BIC. Un'altra opzione è *direction*, che può assumere i valori "backward", "forward" o "both", la quale permette di scegliere il tipo di valutazione gerarchica dei modelli (quindi a eliminazione, a inserimento o mista). Nel caso si voglia procedere con una valutazione a inserimento è necessario utilizzare anche l'opzione *scope* che deve essere una formula corrispondente al modello contenente tutti i predittori.

Esempio

Per illustrare le procedure di selezione di variabili nell'ambito di una regressione lineare si fa uso di un dataset standard di R, utilizzabile con la chiamata:

```
> data(swiss)        # richiama il datatset standard swiss
```

Il dataset *swiss* contiene misure di alcuni indicatori socio economici delle 47 provincie francofone della Svizzera registrati durante una ricerca condotta nel 1888. Le 6 variabili riportate (misurate in percentuale) sono:

- *Fertility*: indice di fertilità standardizzata
- *Agriculture*: percentuale di maschi impegnati in attività agricole
- *Examination*: percentuale di coscritti che ricevono alto punteggio alla visita militare
- *Education*: percentuale di coscritti con livello educativo superiore alla scuola primaria
- *Catholic*: percentuale di persone di religione cattolica
- $Infant.Mortality$: percentuale di bambini nati vivi che sopravvivono meno di 1 anno.

Si vuole vedere se è possibile costruire un modello linerare per legare la variabile *Fertility* alle altre cinque o a un loro sottoinsieme opportunamente determinato con le tecniche presentate precedentemente.

Nel caso più semplice di backward elimination è sufficiente chiamare la funzione *step* passandole il nome del modello da ridurre:

```
> mod <- lm(Fertility ~ ., data = swiss)     # fit del modello completo
> step(mod)
```

L'output di questa funzione presenta la valutazione successiva dei modelli in cui, a ogni passo, viene eliminato un predittore. Nelle tabelle si legge nella prima colonna il nome del predittore eliminato (preceduto dal segno "-") e nell'ultima l'indice AIC del modello che non lo contiene. Il modello identificato dalla stringa "<none>" corrisponde a quello contenente tutti i predittori elencati. Le variabili che possono essere eliminate sono quelle che portano a un modello con indice AIC minore di quello del modello completo. Il primo passo di eliminazione è:

```
Start:  AIC= 190.69
 Fertility ~ Agriculture + Examination + Education + Catholic +
    Infant.Mortality

                  Df Sum of Sq    RSS    AIC
- Examination      1      53.0 2158.1  189.9
<none>                         2105.0  190.7
- Agriculture      1     307.7 2412.8  195.1
```

```
- Infant.Mortality  1     408.8 2513.8  197.0
- Catholic          1     447.7 2552.8  197.8
- Education         1    1162.6 3267.6  209.4
```

Si nota che l'eliminazione del predittore *Examination* porta a un indice AIC di 189.9, valore inferiore a quello del modello con tutti i predittori (190.7). L'algoritmo elimina quindi tale predittore e prosegue per vedere se è possibile eliminarne altri:

```
Step:  AIC= 189.86
 Fertility ~ Agriculture + Education + Catholic + Infant.Mortality

                  Df Sum of Sq    RSS    AIC
<none>                         2158.1  189.9
- Agriculture       1     264.2 2422.2  193.3
- Infant.Mortality  1     409.8 2567.9  196.0
- Catholic          1     956.6 3114.6  205.1
- Education         1    2250.0 4408.0  221.4

Call:
lm(formula = Fertility ~ Agriculture + Education + Catholic +
             Infant.Mortality, data = swiss)

Coefficients:
     (Intercept)     Agriculture      Education       Catholic
         62.1013         -0.1546         -0.9803          0.1247
Infant.Mortality
          1.0784
```

Al secondo passo, le esclusioni dei quattro predittori residui producono modelli peggiori (per quanto riguarda il criterio AIC) rispetto al modello che li contiene tutti. Nessuno di questi può essere quindi eliminato e l'algoritmo si arresta presentando i coefficienti del modello finale.

L'uso di una tecnica di forward selection richiede di partire dal fit del modello nullo e di usare l'opzione *scope* per dichiarare quali variabili debbano entrare nel modello completo:

```
> mod <- lm(Fertility ~ 1, data=swiss)     # modello nullo
> attach(swiss)
> step(mod, scope=Fertility ~ Agriculture + Examination +
+ Education + Catholic + Infant.Mortality, direction="forward")
```

si nota che nell'output che segue i nomi dei predittori sono preceduti da un segno "+" che indica il fatto che l'algoritmo cerca di inserire tali predittori nel modello. L'output della funzione è piuttosto lungo:

```
Start:  AIC= 238.35
 Fertility ~ 1

                  Df Sum of Sq    RSS    AIC
+ Education         1    3162.7 4015.2  213.0
+ Examination       1    2994.4 4183.6  215.0
+ Catholic          1    1543.3 5634.7  229.0
+ Infant.Mortality  1    1245.5 5932.4  231.4
+ Agriculture       1     894.8 6283.1  234.1
<none>                         7178.0  238.3
```

Il primo passo indica che tutte le variabili producono indice AIC migliore di quello del modello nullo. A ogni step viene inclusa solo quella che produce indice minore.

```
Step:  AIC= 213.04
 Fertility ~ Education

                  Df Sum of Sq    RSS    AIC
+ Catholic         1     961.1 3054.2  202.2
+ Infant.Mortality 1     891.2 3124.0  203.2
+ Examination      1     465.6 3549.6  209.2
<none>                         4015.2  213.0
+ Agriculture      1      62.0 3953.3  214.3

Step:  AIC= 202.18
 Fertility ~ Education + Catholic

                  Df Sum of Sq     RSS     AIC
+ Infant.Mortality 1    631.92 2422.25  193.29
+ Agriculture      1    486.28 2567.88  196.03
<none>                         3054.17  202.18
+ Examination      1      2.46 3051.71  204.15

Step:  AIC= 193.29
 Fertility ~ Education + Catholic + Infant.Mortality

              Df Sum of Sq     RSS     AIC
+ Agriculture  1    264.18 2158.07  189.86
<none>                     2422.25  193.29
+ Examination  1      9.49 2412.76  195.10

Step:  AIC= 189.86
 Fertility ~ Education + Catholic + Infant.Mortality + Agriculture

              Df Sum of Sq     RSS     AIC
<none>                     2158.07  189.86
+ Examination  1     53.03 2105.04  190.69

Call:
lm(formula = Fertility ~ Education + Catholic + Infant.Mortality +
             Agriculture, data = swiss)

Coefficients:
     (Intercept)         Education          Catholic  Infant.Mortality
         62.1013           -0.9803            0.1247            1.0784
     Agriculture
         -0.1546
```

L'ultimo passo di inserimento verifica che il predittore *Examination* produce un modello "peggiore" di quello che non lo contiene.

Al termine di questa procedura si trova che in questo particolare caso i modelli ottenuti con le tecniche di inserimento ed eliminazione coincidono. Come detto sopra, questa è tuttaltro che una regola generale dato che solitamente si giunge a modelli differenti.

Se nelle valutazioni precedenti si fosse voluto usare il criterio BIC sarebbe stato sufficiente usare l'opzione k della funzione *step*. Nel caso di tecnica di eliminzaione la chiamata sarebbe stata semplicemente:

```
> step(mod, k=log(47))
```

dato che il dataset è composto da 47 osservazioni. Si noti comunque che nell'output della funzione *step* l'ultima colonna viene sempre etichettata come AIC, indipendentemente dal criterio realmente usato.

9.9.3 Alcuni problemi degli algoritmi di selezione automatica

Con la diffusione di calcolatori sempre più potenti le tecniche di selezione automatica delle variabili sono divenute computazionalmente accessibili a qualunque ricercatore. L'uso indiscriminato di questi algoritmi ha però molti svantaggi, come segnalato in una serie di lavori [1, 38, 46, 20, 56] che mettono in luce le difficoltà teoriche e pratiche che sorgono utilizzando tecniche sia di tipo stepwise sia di ricerca esaustiva fra tutti i possibili modelli.

In particolare questi metodi tendono a fornire modelli con R^2 fortemente distorto verso valori elevati e intervalli di confidenza per i valori predetti ingannevolmente piccoli. Inoltre i valori P di significatività delle variabili sono ottenuti da distribuzioni che spesso si discostano fortemente da quelle teoriche alla base del loro calcolo. In aggiunta a tutto ciò, il grado di collinerità fra i predittori influisce pesantemente sulla selezione delle variabili che appaiono nel modello finale così come anche il numero stesso di predittori totali. Infine i modelli ottenuti con queste tecniche sono troppo legati al particolare set di dati utilizzato e non sono facilmente generalizzabili al di fuori di esso.

Per tutti questi motivi la costruzione di un modello di regressione dovrebbe sempre poggiare principalmente sulla competenza del ricercatore nell'individuare le variabili rilevanti nello specifico campo di indagine.

Capitolo 10

Geostatistica

La Geostatistica è la branca della Statistica che si occupa dell'analisi di dati spaziali. La prima sistematizzazione in questo campo risale al 1970, ad opera di George Matheron del Centre de Morophologie Mathematicque (Fontainebleau). Nata nell'ambito di analisi minerarie, le sue metodiche sono oggi largamente impiegate in numerose aree della Geologia, Ecologia, Agronomia.

Le tecniche di Geostatistica permettono di utilizzare la struttura spaziale inferita dai dati campionari, valutando la varianza spaziale, per fornire delle stime sui parametri che caratterizzano la variabilità spaziale o sul valore assunto da una variabile in una posizione in cui la misurazione non è stata effettuata. Secondo l'approccio classico, la struttura spaziale non viene formalmente modellizzata e la variabilità spaziale viene elaborata mediante lo strumento del semivariogramma, che permette di valutare il grado di variabilità di punti a distanze crescenti. Questo studio di variabilità spaziale può essere effettuato sia come indagine esplorativa, sia per effettuare una predizione spaziale mediante interpolazione. Il metodo di interpolazione più usato in questa fase è il kriging.

Nell'ambito della Geostatistica Classica si considera che il fenomeno di interesse sia descrivibile in termini di variabili regionalizzate, ossia variabili né completamente casuali, né completamente deterministiche. Si assume cioè che il valore $z(x)$ assunto dalla variabile di interesse Z nel punto x sia ascrivibile a tre componenti:

$$z(x) = m(x) + S(x) + \varepsilon$$

dove $m(x)$ è una funzione deterministica della posizione x, $S(x)$ è la parte di residuo dipendente alla posizione x, e ε è una perturbazione indipendente dalla posizione x. Le tecniche geostatistiche classiche permettono di caratterizzare l'andamento di $S(x)$, senza supporre che esso segua una qualche distribuzione data (approccio non parametrico).

Il campo della variabile Z si dice stazionario (in senso forte) se la sua legge di distribuzione è uguale per tutti i punti x nell'area di studio. Questo equivale a dire che il campo definito da Z è invariante per traslazione. Le stime di varianza spaziale si appoggiano sulle ipotesi di variabili regionalizzate stazionarie. Tuttavia, dato che la stazionarietà forte è difficilmente soddisfatta in pratica, per procedere alle stime di varianza spaziale si richiede almeno la condizione di quasi stazionarietà del secondo ordine (o stazionarietà debole del secondo ordine). Questa implica che, per ogni posizione x, la media della variabile sia costante, la sua varianza sia finita ed esista una funzione di covarianza dipendente solo dalla separazione tra i punti: $Cov(Z(x), Z(y)) = C(x - y)$.

Un approccio differente è quello della Geostatistica model-based in cui $S(x)$ è visto come un processo stocastico latente (e quindi non direttamente osservabile), dipendente unicamente dalla posizione, e z_i è la misura effettuata nella posizione x_i, intesa come il valore assunto da una variabile Z_i, dipendente da $S(x)$. La caratteristica di questo approccio è che per lo sperimentatore è necessario ipotizzare una certa distribuzione per $S(x)$ e per ε. Per stimare i parametri delle distribuzioni si utilizzeranno poi generalmente tecniche di maximum likelihood. Un caso particolare si ha se le misure effettuate non sono soggette a errore dovuto al campionamento e si può assumere $Z_i = S(x_i)$. In questo caso, ripetendo la misura z_i, si suppone di ottenere sempre lo stesso valore, entro la precisione strumentale. Per realizzare una mappa di $S(x)$ ha quindi senso adottare tecniche di interpolazione. Viceversa se si

suppone che Z_i assuma valori diversi ripetendo la misura, è opportuno adottare modelli statistici che separino la componente stocastica da quella legata al processo di misura.

Nella prima parte di questo Capitolo si tratterà delle tecniche classiche, rimandando alla seconda parte le metodiche model-based.

10.1 Semivariogramma

Il semivariogramma è una funzione che valuta il grado di dipendenza spaziale di dati osservati in punti georeferiti. Dato un campo spaziale $Z(x)$ si definisce variogramma la funzione:

$$2\,\gamma(x,y) = E[(Z(x) - Z(y))^2]$$

dove x e y sono due differenti località. Se il processo in studio è stazionario e isotropico si ha che la funzione dipende solo dalla distanza h tra le località: $\gamma(x,y) = \gamma(h)$. In questo caso la funzione variogramma si scrive come:

$$2\,\gamma(h) = E[(Z(x) - Z(x+h))^2]. \tag{10.1}$$

È possibile considerare anche casi in cui la correlazione spaziale dipende dalla direzione. Per la trattazione di questi problemi, che vengono affrontati con il calcolo di più variogrammi, si rimanda a testi come [27, 41] e alle referenze in essi citate. Un esempio di calcolo di variogrammi dipendenti dalla direzione è dato al termine dell'Esempio trattato in seguito.

Se il processo è stazionario, la media e la varianza della variabile Z non dipendono dalla posizione e si può scrivere $m = E[Z(x)]$, $\sigma^2 = Var(Z(x))$ indipendentemente dal valore di x. Si introduce anche una seconda misura statistica, la covarianza spaziale:

$$C(h) = E[Z(x)Z(x+h)] - m^2$$

con $C(0) = \sigma^2$. Espandendo i quadrati nell'Eq. 10.1, con un po' di algebra si arriva alla relazione tra covarianza spaziale e variogramma, valida nel caso di campo stazionario:

$$\gamma(h) = C(0) - C(h). \tag{10.2}$$

In caso di isotropia, la varianza spaziale di una variabile quantitativa viene stimata sul campione dalla funzione di semivarianza $\hat{\gamma}(h)$:

$$\hat{\gamma}(h) = \frac{1}{2\,n(h)} \sum_{i=1}^{n(h)} [z(x_i) - z(x_i + h)]^2$$

dove $n(h)$ è il numero di coppie di dati posti a distanza h uno dall'altro. Il fattore 2 a denominatore tiene conto del fatto che ogni coppia viene contata due volte e spiega anche perché la funzione viene detta semivarianza. Il grafico della funzione di semivarianza contro i valori della distanza h è il semivariogramma sperimentale, anche se per semplicità è spesso detto variogramma. In pratica, dato che le osservazioni non si presenteranno su un grigliato ordinato, occorrerà modificare leggermente la tecnica di calcolo del variogramma. Si introduce cioè un parametro di tolleranza sulla distanza Δh in modo che contribuiscano al calcolo al lag h tutti i punti situati a distanza $h \pm \Delta h$.

In presenza di correlazione spaziale ci si attende che la semivarianza aumenti fino a un certo valore di distanza, oltre la quale la dipendenza spaziale viene perduta. Corrispondentemente, la covarianza spaziale manifesta andamento opposto, diminuendo fino a raggiungere il valore $C(h) = 0$. Dopo tale punto il grafico della semivarianza presenterà una regione piatta detta sella (*sill*). Dall'Eq. 10.2 si nota che il valore di sill coincide con la varianza σ^2 del campo spaziale. Vale la pena di sottolineare che la varianza campione s^2 è una stima di σ^2 solo quando l'area campionata tende all'infinito; usualmente questa condizione non è soddisfatta [3].

La distanza per la quale si raggiunge il regime stazionario è detta *range* e rappresenta la regione entro cui le misure sono spazialmente correlate tra loro. Le località che sono situate a distanze minori del range sono considerate vicine e sono positivamente correlate. Il terzo parametro che caratterizza un

variogramma è il *nugget* ossia la discontinuità che il grafico presenta all'origine, dove teoricamente ci si aspetterebbe semivarianza nulla. In un variogramma sperimentale la sua presenza può essere dovuta a errori di misura o a strutture con correlazione che occorre a scale più piccole della risoluzione campionaria.

Ai fini delle tecniche di interpolazione, il semivariogramma sperimentale deve essere fittato con funzioni matematiche che rispettino determinate caratteristiche. Si ottiene in questo modo un semivariogramma teorico. Tale modello di semivariogramma si desume a partire dall'osservazione del semivariogramma sperimentale. Il tipo di funzione da scegliere dovrà essere la più semplice possibile in grado di interpretare l'andamento dei punti sperimentali.

Tra i tipi di modello utilizzati per l'approssimazione dei variogrammi sperimentali si possono ricordare:

- il modello esponenziale: $\gamma(h) = C_0 + C_1[1 - \exp(-3\frac{h}{a})]$;
- il modello sferico: $\gamma(h) = C_0 + C_1[1.5\frac{h}{a} - 0.5(\frac{h}{a})^3]$ per $h \leq a$ e $\gamma(h) = C$ per $h > a$;
- il modello gaussiano: $\gamma(h) = C_0 + C_1[1 - \exp(-3\frac{h^2}{a^2})]$;
- il modello lineare: $\gamma(h) = C_0 + bh$ con b il coefficiente angolare della retta.

In questi modelli a è il parametro di range, C_0 il parametro di nugget e $C = C_0 + C_1$ il parametro di sill.

Oltre a questi è possibile utilizzare altri modelli, per la cui definizione matematica si rimanda, ad esempio, a [27, 41].

Dato che la dipendenza spaziale si realizza spesso su distanze brevi sarà particolarmente importante modellizzare correttamente il semivariogramma per piccoli valori di h. Attualmente non esistono tecniche affidabili che guidino nella selezione di un semivariogramma e che permettano di stabilirne la significatività. La scelta si basa quindi o sull'esperienza dello sperimentatore o sulla implementazione di tecniche bootstrap di cui è dato un esempio nel seguito (si veda ad esempio [27] per ulteriori dettagli su questi argomenti).

In R è possibile costruire variogrammi sperimentali e teorici mediante le funzioni della libreria aggiuntiva *gstat*, la quale necessita a sua volta dell'installazione della libreria *sp*. In alternativa le funzioni per costruire variogrammi sono implementate nella libreria *geoR*, la quale mette a disposizione numerose funzioni per Geostatistica model-based.

Esempio

Il dataset *meuse*, fornito con la libreria *gstat*, registra il contenuto di metalli pesanti (in ppm) del suolo, campionati nella piana del fiume Meuse, vicino al villaggio di Stein (Olanda). I dati sono georeferenziati (in m) usando coordinate del sistema cartografico olandese. Si vuole studiare l'andamento spaziale della concentrazione di rame.

Come prima cosa si caricano la libreria e il dataset:

```
> library(gstat)
> data(meuse)
```

È quindi necessario specificare quali siano le variabili che vanno utilizzate per georeferenziare il campione. In questo caso si tratta delle variabili x e y, contenute nel dataset. Si procede quindi utilizzando la funzione *coordinates*:

```
> coordinates(meuse) <- ~ x + y
```

Tale funzione converte il data frame in input restituendo in output un oggetto della classe SpatialPointDataFrame.

Il variogramma sperimentale per la concentrazione al suolo di rame si valuta con la chiamata alla funzione *variogram*:

```
> vgm <- variogram(copper ~ 1, data=meuse)
```

La funzione accetta almeno due argomenti: il modello da costruire (in questo caso non si ipotizza la presenza di nessun regressore) e il dataset da utilizzare. Per una lista esaustiva delle numerose opzioni della funzione si rimanda alla sua pagina di manuale. Ad esempio, la funzione ritorna di default 15 bin spaziali di uguali dimensioni, con distanza massima pari a un terzo della massima separazione rilevabile dai dati campionari. Questi valori possono essere modificati con le opzioni *width* e $cutoff$ rispettivamente. Il risultato della chiamata può essere esaminato graficamente:

```
> plot(vgm)
```

Nella costruzione del variogramma teorico si deve prestare attenzione al fatto che in ogni bin ci sia un numero sufficiente di punti sperimentali di modo che le stime ottenute possano essere ritenute attendibili. Usualmente si assume che un centinaio di punti siano sufficienti e che trecento garantiscano un risultato pienamente affidabile. Per il variogramma in questione si può esaminare il numero di punti nei 15 bin nel modo seguente:

```
> vgm$np
 [1]  57 299 419 457 547 533 574 564 589 543 500 477 452 457 415
```

Si nota che, a parte il primo bin, gli altri sono ben popolati.

A partire dal variogramma sperimentale si procede quindi a fittare un variogramma teorico. In particolare si vogliono valutare un variogramma lineare e un variogramma sferico. In entrambi i casi si fa uso della funzione $fit.variogram$:

```
> fit.lin <- fit.variogram(vgm, model=vgm(psill=1, "Lin", nug=50))
> fit.sph <- fit.variogram(vgm, model=vgm(psill=1, "Sph",range=800, nug=50))
```

La funzione richiede due argomenti: il variogramma sperimentale e il modello di variogramma teorico, costruito mediante la chiamata alla funzione *vgm*. Questa funzione accetta vari argomenti, che dipendono dal modello da fittare. Oltre al nome del modello, nel caso di fit lineare si specificano i valori iniziali per il parametro di nugget e per il parametro *psill* (che è interpretato come il coefficiente angolare della retta), mentre nel caso di fit sferico occorre specificare anche il valore iniziale del parametro di range. Come in tutti i casi di fit non lineare la scelta dei parametri iniziali può influenzare il risultato finale ritornato dalla funzione. Il metodo di fit impiegato di default dalla funzione è quello dei minimi quadrati ordinari; è anche possibile richiedere un fit mediante tecnica dei minimi quadrati generalizzati (si veda la pagina di manuale della funzione).

I risultati dei due fit si possono ispezionare nel modo seguente:

```
> fit.lin
  model       psill range
1   Nug 270.0608103     0
2   Lin   0.3880909     0

> fit.sph
  model    psill    range
1   Nug 175.9665   0.0000
2   Sph 428.6551 707.2165
```

Nel caso di fit lineare si ottiene parametro di nugget pari a 270 e coefficiente di regressione 0.388; per il modello sferico si ha nugget 176, range 707 e sill 429.

I risultati grafici dei due fit (Fig. 10.1) si esaminano con le seguenti chiamate:

```
> plot(vgm, fit.lin)
> plot(vgm, fit.sph)
```

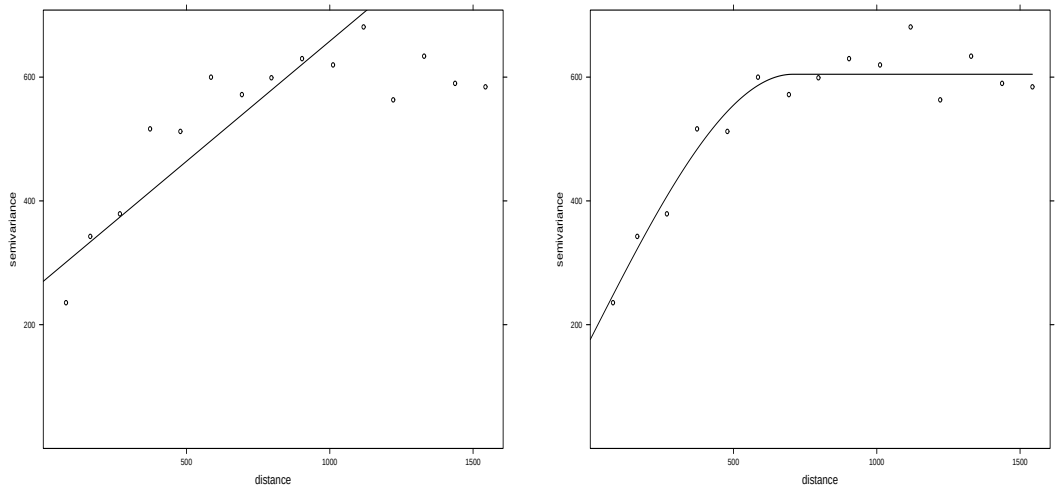

Figura 10.1: Variogramma sperimentale con fit di tipo lineare (a sinistra) e sferico (a destra). Il modello sferico sembra descrivere meglio la struttura dei dati.

Dai grafici sembra che il modello sferico descriva meglio la struttura spaziale dei dati. Da questo modello si evidenzia che la relazione di dipendenza spaziale viene persa per distanze dal luogo di campionamento superiori a circa 700 m.

Se si utilizza la libreria *geoR* è necessario per prima cosa convertire i dati nel formato accettato dalle funzioni di questa libreria. Questa operazione si realizza con la chiamata della funzione *as.geodata*:

```
> library(geoR)
> meuse.geo <- as.geodata(meuse, data.col=2)
```

dove l'opzione *data.col* serve a specificare in quali colonna si trovano i dati di misura. La funzione accetta numerose altre opzioni che verranno descritte parzialmente nel seguito del Capitolo. Gli oggetti di classe geodata possono essere esaminati con la funzione *summary*, che fornisce un riepilogo sia della parte relativa alle coordinate e alle distanze, sia della zona dati:

```
> summary(meuse.geo)
Number of data points: 155

Coordinates summary
         x      y
min 178605 329714
max 181390 333611

Distance summary
       min        max
  43.93177 4440.76435

Data summary
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  14.00   23.00   31.00   40.32   49.50  128.00
```

Il variogramma sperimentale è visualizzabile con la chiamata alla funzione *variog*:

```
> vg.geo <- variog(meuse.geo, max.dist=1600)
variog: computing omnidirectional variogram
```

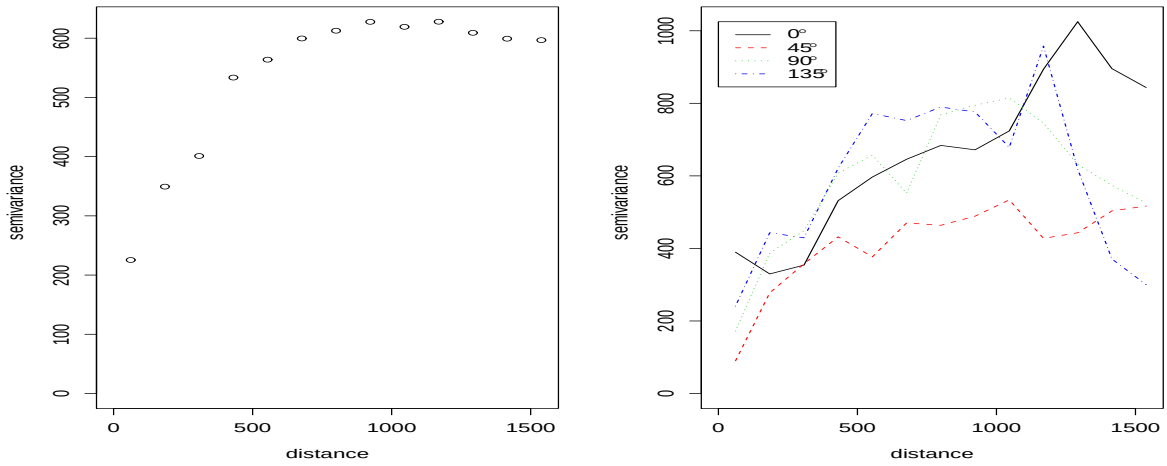

Figura 10.2: Variogramma sperimentale prodotto dalla funzione *variog* (a sinistra) e variogramma direzionale mediante funzione *variog4* (a destra). I variogrammi direzionali soffrono di mancanza di precisione, soprattutto a grandi distanze, dato lo scarso numero di punti che fanno da supporto al calcolo.

```
> plot(vg.geo)
```

L'opzione *max.dist* serve a specificare la distanza massima da includere nel variogramma ed è stata scelta per renderlo confrontabile con il grafico prodotto in precedenza. Come si può osservare in Fig. 10.2 a sinistra, il risultato è molto simile a quello di Fig. 10.1. Le piccole differenze sono dovute a un diverso schema di suddivisione in bin spaziali.

Per concludere questa sezione si noti che talvolta, quando si sospetta la presenza di anisotropia nei dati, è di interesse costruire più che un unico variogramma, un set di variogrammi differenti in dipendenza non solo dalla distanza delle stazioni di misura, ma anche della direzione. Questo approccio soffre particolarmente della mancanza di dati, soprattutto ad alte distanze, e si rivela quindi applicabile con successo solamente qualora si abbia a che fare con dataset di dimensione notevole. Sui dati in questione la tecnica si applica con la chiamata alla funzione *variog4* della libreria *geoR*, che consente di calcolare il variogramma empirico in 4 direzioni definibili dall'utente. L'opzione di default utilizza le direzioni 0^o, 45^o, 90^o e 135^o:

```
> plot(variog4(meuse.geo, max.dist=1600))
```

Il risultato della chiamata è a destra in Fig. 10.2.

10.1.1 Validazione Monte Carlo di un variogramma

Quando il variogramma empirico sembra suggerire una dipendenza spaziale debole o nulla può essere di interesse testare formalmente l'ipotesi di indipendenza spaziale. Dato che il modello che descrive le osservazioni è scrivibile in termini di un termine legato al trend spaziale $\mu(x_i)$ che descrive la media del fenomeno e di un residuo z_i come:

$$y_i = \mu(x_i) + z_i$$

è possibile eseguire un test formale del modello utilizzando un metodo Monte Carlo. Per eseguire il test si procede valutando il termine $\mu(x)$ che può essere semplicemente la media generale del fenomeno spaziale, se non si suppone la presenza di alcun trend, o il modello adeguato che incorpora le variabili legate al fenomeno di trend ipotizzato (si veda il seguito per il fit dei modelli); si calcolano i residui nei vari punti ; si permutano casualmente i residui e si costruisce un nuovo variogramma. I variogrammi

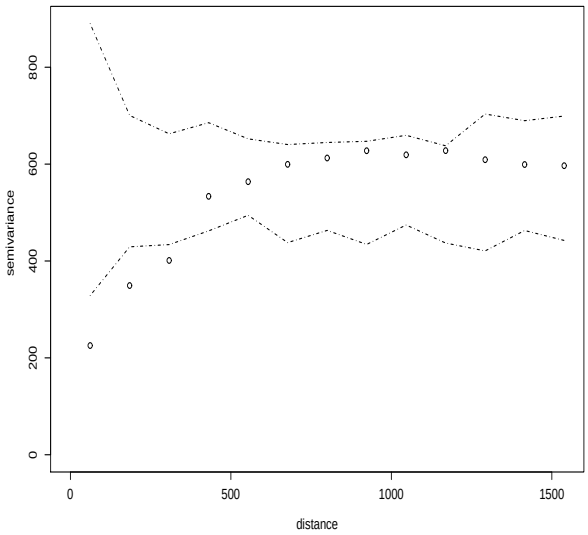

Figura 10.3: Envelop Monte Carlo (linee tratteggiate) del variogramma empirico (punti). Dato che non tutti i punti giacciono all'interno dell'envelop si conclude che la dipendenza spaziale influenza significativamente il processo.

ottenuti dalle simulazioni sono suddivisi con gli stessi bin spaziali di quello empirico. La procedura viene ripetuta un certo numero di volte e per ogni bin si prende il valore massimo e quello minimo risultanti dalle simulazioni. Al termine si ottiene una envelop Monte Carlo per il variogramma. Se tutti i punti del variogramma empirico giacciono all'interno dell'envelop, si ha indicazione che la variabile in studio non presenta dipendenza spaziale.

Con i dati dell'Esempio la procedura è attuabile sfruttando la funzione *variog.mc.env* della libreria *geoR*:

```
> vg.geo.mc <- variog.mc.env(meuse.geo, obj=vg.geo)
variog.env: generating 99 simulations by permutating data values
variog.env: computing the empirical variogram for the 99 simulations
variog.env: computing the envelops
```

La funzione accetta almeno due argomenti: il set di dati su cui lavorare e il variogramma sperimentale da validare. Di default vengono eseguite 99 permutazioni casuali, ma tale valore è facilmente modificabile mediante l'opzione *nsim*.

Il risultato può essere esaminato graficamente:

```
> plot(vg.geo, env=vg.geo.mc)
```

Come si vede in Fig. 10.3, i punti nei primi bin sono all'esterno della envelop, confermando che la dipendenza spaziale influisce significativamente sulle misure della concentrazione di rame nel suolo.

10.2 Kriging

Per poter ricostruire l'andamento spaziale della variabile $Z(x)$ di interesse è necessario interpolare i dati disponibili per stimarne i valori dove non si hanno misurazioni.

Nell'ambito della teoria delle variabili regionalizzate la tecnica di kriging fornisce un metodo di interpolazione ottimale, nel senso che la varianza della differenza tra il valore predetto e quello reale è minima.

Come in tutte le tecniche interpolatorie, la stima del valore da valutare $\hat{z}(x)$ viene effettuata calcolando una media pesata dei valori osservati $z(x)$. In particolare, nella tecnica kriging i pesi non vengono ottenuti utilizzando funzioni arbitrarie (ad esempio, inverso della distanza o del quadrato delle distanza), ma a partire dai valori di covarianza spaziale. Per ottenere questi valori si sfruttano le informazioni sul fenomeno spaziale costruendo un modello di variogramma teorico.

I risultati della tecnica kriging sono spesso simili a quelli di altre tecniche. In particolare, se le valutazioni della variabile $z(x)$ sono dense e ben distribuite si ottengono buoni risultati con qualsiasi tecnica; se al contrario i dati sono fortemente raggruppati in cluster le stime saranno cattive con qualsiasi tecnica; tutte le tecniche interpolatorie, per loro natura, tendono ad appiattire il campo dei valori della variabile sottostimando i picchi e sovrastimando le valli [39].

Tra i vantaggi della tecnica, oltre a quella di valutare la varianza del valore predetto, vi è quello di compensare gli effetti del clustering, pesando i punti clusterizzati meno di quelli isolati.

10.3 Tipi di interpolazione Kriging

Esistono diversi tipi di kriging, tra cui kriging semplice (simple kriging), kriging ordinario (ordinary kriging) e kriging universale (universal kriging). Ciò che li differenzia è il tipo di variabile usata: il kriging ordinario può lavorare solo con variabili stazionarie del secondo ordine (che presentano media costante e covarianza dipendente solo dal lag muovendosi da punto a punto); il kriging universale può invece lavorare anche con variabili non stazionarie (che presentano un trend). In questo caso la condizione di stazionarietà del dato può essere ristabilita attraverso l'introduzione di una funzione deterministica che descriva il trend, cioè l'andamento della media, in modo da poter isolare il residuo, che costituisce la parte aleatoria del dato. Il kriging universale procede modellando e sottraendo il trend presente nel dato tramite una funzione deterministica e analizza la sola componente aleatoria.

Tutte le varianti della tecnica si basano sul calcolo di uno stimatore lineare del tipo:

$$\hat{z}(x) - m(x) = \sum_{i=1}^{n(x)} w_i(x)[z(x_i) - m(x_i)] \tag{10.3}$$

dove x e x_i sono le localizzazioni del punto in cui si desidera la stima e dei punti prossimi al luogo della stima, utilizzati nel processo; $n(x)$ il numero di detti punti vicini; $m(x)$ e $m(x_i)$ i valori attesi delle variabili $Z(x)$ e $Z(x_i)$; w_i i pesi kriging da assegnare ai valori $z(x_i)$ nel calcolo della stima. La variabile $Z(x)$ è scomposta nella sua componente di trend $m(x)$ e nella componente residua $R(x) = Z(x) - m(x)$. La tecnica kriging stima i valori dei residui nel punto x come media pesata dei residui dei dati vicini. I pesi sono ricavati dalla funzione di covarianza spaziale o equivalentemente dal semivariogramma.

I pesi vengono determinati ricercando quei valori che minimizzino la varianza kriging:

$$\sigma_E^2(x) = Var(\hat{z}(x) - Z(x))$$

con il vincolo di non distorsione:

$$E[\hat{z}(x) - Z(x)] = 0.$$

Per questo motivo la stima kriging viene anche detta Best Linear Unbiased Predictor (BLUP). Si noti che la varianza kriging non coincide con la varianza dello stimatore $Var(\hat{z}(x))$.

Data la scomposizione della variabile $Z(x)$ nella componente di trend e nella parte residua si ipotizza che la parte residua abbia media nulla e covarianza dipendente solo dal lag spaziale ma non dalla posizione x:

$$E[R(x)] = 0 \quad , \quad Cov(R(x), R(x+h)) = C_R(h)$$

La funzione di covarianza spaziale viene usualmente ricavata dal semivariogramma teorico:

$$C_R(h) = C_R(0) - \hat{\gamma}(h) = sill - \hat{\gamma}(h).$$

10.3.1 Kriging semplice

Nel caso di kriging semplice si suppone che la variabile $Z(x)$ abbia media nota pari a m. In questo caso l'Eq. 10.3 si riduce a:

$$\hat{z}_{SK}(x) = m + \sum_{i=1}^{n(x)} w_i[z(x_i) - m]$$

che è automaticamente non distorta dato che: $E[z(x_i) - m] = 0$. Si ha quindi che l'errore della stima è:

$$\hat{z}_{SK}(x) - z(x) = R_{SK}(x) - R(x)$$

e la sua varianza:

$$\sigma^2_E(x) = Var(R_{SK}(x)) + Var(R(x)) - 2\, Cov(R_{SK}(x), R(x)) =$$

$$= \sum_{i=1}^{n(x)} \sum_{j=1}^{n(x)} w_i w_j C_R(x_i - x_j) + C_R(0) - 2 \sum_{i=1}^{n(x)} w_i C_R(x_i - x)$$

I valori dei pesi sono scelti in modo tale da minimizzare questo valore. Per far ciò si deriva l'espressione per $\sigma^2_E(x)$ rispetto ai pesi e si pongono a zero i risultati. Si arriva così al sistema di equazioni lineari:

$$\sum_{j=1}^{n(x)} w_j C_R(x_i - x_j) = C_R(x_i - x) \quad i = 1, \ldots, n(x).$$

Dato che la media è costante, la matrice di covarianza dei residui è identica alla matrice di covarianza della variabile $Z(x)$. Il sistema finale si può dunque scrivere, in forma matriciale:

$$\boldsymbol{C}\boldsymbol{w} = \boldsymbol{c}$$

dove $\boldsymbol{C}$ è la matrice di covarianza tra i punti del campione, con elementi $C_{ij} = C(x_i - x_j)$, $\boldsymbol{c}$ il vettore che contiene le covarianze tra i punti del campione e il nodo in cui effettuare la stima e $\boldsymbol{w}$ il vettore dei pesi kriging. Il vettore dei pesi sarà quindi:

$$\boldsymbol{w} = \boldsymbol{C}^{-1}\boldsymbol{c}$$

da cui si ricavano sia il valore stimato che la varianza kriging.

Si osservi che la tecnica è disegnata in modo tale da tenere in considerazione i raggruppamenti di punti. Infatti punti raggruppati avranno un elevato coefficiente C_{ij} che condurrà a un basso peso, dato che questi ultimi dipendono dall'inverso della matrice $\boldsymbol{C}$.

Esempio

Con il dataset dell'esempio precedente si vuole stimare mediante kriging semplice il valore di concentrazione di rame su un grigliato, definito nel dataset *meuse.grid*. Come primo passo si carica quindi il nuovo dataset:

```
> data(meuse.grid)
> coordinates(meuse.grid) <- ~ x + y
```

Per utilizzare una stima simple kriging è necessario disporre della media della variabile *copper*, che viene considerata stazionaria. Tale valore viene poi impiegato nella chiamata alla funzione *krige*:

```
> media <- mean(meuse$copper)
> media
[1] 40.31613

> kr.sk <- krige(copper ~ 1, meuse, newdata=meuse.grid, model=fit.sph, beta=media)
[using simple kriging]
```

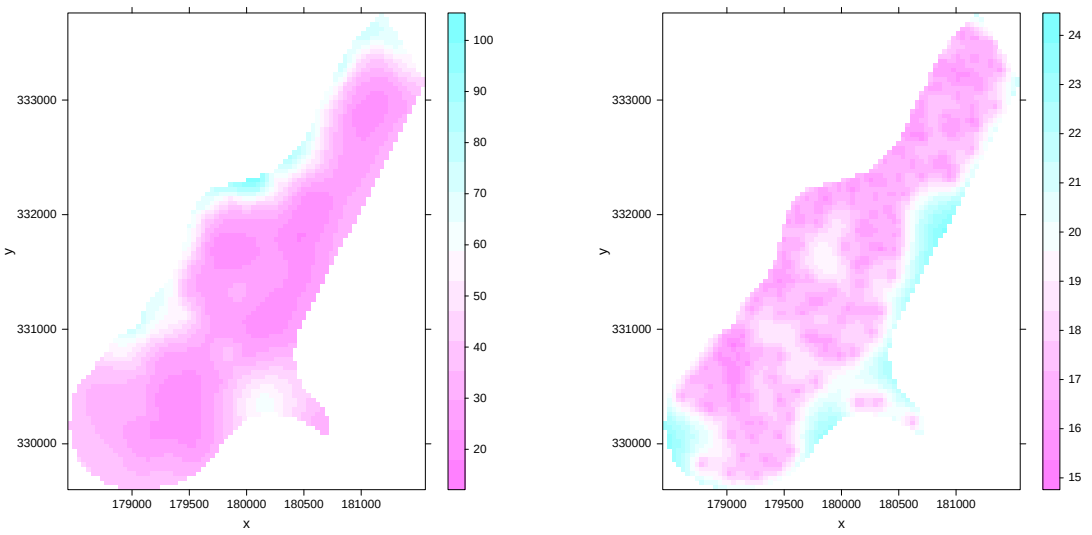

Figura 10.4: Mappa di interpolazione ottenuta mediante simple kriging (sinistra) e mappa delle radici quadrate degli errori kriging (destra).

La funzione *krige* accetta come argomenti: il modello da interpolare; il dataset contenente le misure effettuate; il dataset con i luoghi di interpolazione; il modello di variogramma teorico; il parametro *beta* che serve per specificare la media della variabile sull'area di studio.

Il risultato della chiamata può essere visualizzato creando una mappa di interpolazione sfruttando la funzione *levelplot* della libreria aggiuntiva *lattice*:

```
> library(lattice)
> levelplot(var1.pred ~ x + y, as.data.frame(kr.sk), aspect="iso")
```

Gli argomenti della funzione sono facilmente interpretabili: in primo luogo il modello da graficare che contiene i valori predetti e le variabili geografiche; in secondo luogo il dataframe che contiene le variabili del modello (la chiamata alla funzione $as.data.frame$ è necessaria per convertire in formato appropriato il risultato della chiamata alla funzione *krige*); in terzo luogo una stringa che specifica l'aspetto del grafico (in questo caso si richiede una scala isometrica sui due assi).

Analogamente si può realizzare una mappa degli errori kriging, o meglio delle radici quadrate di tali valori:

```
> levelplot(sqrt(var1.var) ~ x + y, as.data.frame(kr.sk), aspect="iso")
```

I risultati delle due chiamate sono in Fig. 10.4. Nel grafico di destra si possono evidenziare le aree in cui la predizione è meno accurata e che beneficerebbero di ulteriore campionamento.

Come nota operativa, si osservi che per ottenere una copia postscript a colori dei grafici è necessario modificare le impostazioni di default del device grafico. Prima di aprire il device postscript si salvano le impostazioni del device grafico standard:

```
> par <- trellis.par.get()
```

Le quali vengono usate per modificare le impostazioni del device postscript:

```
> postscript("output.ps", paper="special", width=8, height=8, horizontal=FALSE)
> trellis.par.set(par)
[...]
> dev.off()
```

Seguendo questo schema l'output manterrà i colori che si vedono durante le analisi a schermo.

10.3.2 Kriging ordinario

Nel caso di kriging ordinario si suppone che la media della variabile da interpolare sia costante non su tutta l'area, ma almeno nelle vicinanze di ogni punto, ossia $m(x_i) = m(x)$. In questo caso l'Eq. 10.3 si scrive come:

$$\hat{z}_{OK}(x) = m(x) + \sum_{i=1}^{n(x)} w_i[z(x_i) - m(x)] = \sum_{i=1}^{n(x)} w_i z(x_i) + m(x)[1 - \sum_{i=1}^{n(x)} w_i].$$

Imponendo il vincolo:

$$\sum_{i=1}^{n(x)} w_i = 1$$

ci si riconduce a una stima simple kriging con $m = 0$:

$$\hat{z}_{OK}(x) = \sum_{i=1}^{n(x)} w_i z(x_i)$$

La varianza di errore deve essere minimizzata rispettando il vincolo. Per questo si minimizza un funzionale in cui si introduce un moltiplicatore di Lagrange μ:

$$L = \sigma_E^2(x) + 2\mu(x)[1 - \sum_{i=1}^{n(x)} w_i]$$

In questo sistema infatti, la minimizzazione rispetto a μ impone il vincolo sulla somma dei pesi:

$$\frac{1}{2}\frac{\partial L}{\partial \mu} = 0 \quad \rightarrow \quad 1 - \sum_{i=1}^{n(x)} w_i = 0.$$

Le equazioni di kriging sono quindi:

$$\begin{cases} \sum_{j=1}^{n(x)} w_j C_R(x_i - x_j) + \mu(x) = C_R(x_i - x) & i = 1, \ldots, n(x) \\ \sum_{j=1}^{n(x)} w_j = 0 \end{cases}$$

In questo caso, dato che la media della variabile $Z(x)$ non è costante su tutta l'area, l'identificazione $C_R(h) = C(h)$ (valida nel caso di kriging semplice) non è corretta. Tuttavia essa viene ugualmente fatta supponendo che il semivariogramma teorico filtri appropriatamente il trend spaziale della media su larga scala.

Esempio

Riprendendo l'esempio precedente, la stima di kriging ordinario per la concentrazione di rame si ottiene utilizzando, anche in questo caso, la funzione *krige*:

```
> kr.ok <- krige(copper ~ 1, meuse, newdata=meuse.grid, model=fit.sph)
[using ordinary kriging]
```

L'unica differenza rispetto alla chiamata precedente è che alla funzione non si passa il parametro *beta*.

Le differenze tra le stime di interpolazione di kriging ordinario e kriging semplice sono solitamente contenute. Nel caso dell'esempio in studio, in Fig. 10.5 è riportata la mappa delle differenze tra le due stime, ottenuta con la chiamata:

```
> levelplot(var1.pred-kr.sk$var1.pred ~ x+y,data=as.data.frame(kr.ok),aspect="iso")
```

Come si può osservare le due stime differiscono ben poco su tutta l'area in studio.

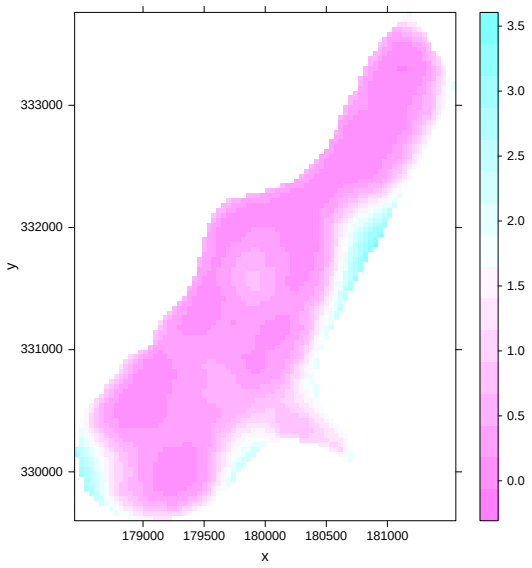

Figura 10.5: Mappa delle differenze tra le interpolazioni ottenuta mediante ordinary kriging e simple kriging.

10.3.3 Kriging universale

La tecnica di kriging universale viene impiegata quando si suppone che l'ipotesi di stazionarietà sia violata e che i dati presentino un andamento (drift), lineare o di ordine superiore, ascrivibile alla posizione o al valore di qualche altra variabile contestualmente misurata. Ad esempio, un tipico caso di campo non stazionario si ha in Climatologia, in cui l'altitudine del luogo viene spesso utilizzata nelle stime kriging di temperatura. Rimuovendo la componente di trend dai dati è possibile eseguire una stima kriging sui residui, che soddisfano l'ipotesi di stazionarietà. I valori interpolati vengono quindi aggiunti alla componente sistematica per avere le stime finali dal modello.

La presenza di un trend nei dati può essere identificata dall'esame del variogramma sperimentale, che in questi casi mostra un andamento sempre crescente. Questo è indice del fatto che all'aumentare della distanza tra località aumenta anche la differenza tra i valori misurati in tali località.

In questo tipo di analisi c'è bisogno di stimare il variogramma dei residui, ottenuti rimuovendo la componente sistematica di drift. L'operazione viene illustrata nel seguente esempio.

Esempio

Con i dati relativi agli esempi precedenti, si supponga di voler interpolare il valore di concentrazione di rame usando un modello di kriging universale con termine di drift lineare nelle coordinate x e y. Il primo passo è costruire il variogramma dei residui:

```
> vgm.uk <- variogram(copper ~ x + y, data=meuse)
```

Nella chiamata alla funzione *variogram* si specifica il tipo di modello da utilizzare nel calcolo del trend.

Il variogramma viene quindi fittato con le tecniche illustrate precedentemente:

```
> fit.sph.uk <- fit.variogram(vgm.uk, model=vgm(psill=1,"Sph",range=800,nug=50))

> fit.sph.uk
  model    psill    range
1   Nug 181.7899   0.0000
```

```
2   Sph 262.8878 624.4587
```

La stima di kriging universale si ottiene quindi con la chiamata:

```
> kr.uk <- krige(copper ~ x + y, meuse, newdata=meuse.grid, model=fit.sph.uk)
[using universal kriging]
```

Anche in questo caso viene specificato il modello utilizzato nel calcolo del trend.

10.3.4 Rivalidazione bootstrap

Nell'ambito delle stime kriging è possibile avvalersi di tecniche bootstrap per testare la validità di un modello di interpolazione. In particolare queste tecniche sono utili per verificare quanto la scelta di due variogrammi teorici differenti si ripercuote sulle stime al termine della procedura.

Di particolare impiego è la metodica leave-one-out cross-validation in cui ogni dato viene lasciato a turno fuori dalla costruzione del modello e il valore osservato viene quindi confrontato con la predizione del modello.

La funzione per eseguire questo tipo di analisi è *krige.cv*. Ad esempio, nel caso di kriging ordinario si può procedere nel modo seguente:

```
> ok.cv <- krige.cv(copper ~ 1, meuse, model=fit.sph)
[using ordinary kriging]
...
```

L'oggetto in output contiene varie informazioni, tra cui i valori dei residui (differenza tra punti osservati e predetti) e i rispettivi valori z (residui diviso la deviazione standard kriging). Nel caso dell'esempio in questione si ha:

```
> summary(ok.cv$residual)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-35.6900  -8.2690  -2.4240   0.1514   5.9910  50.1900

> summary(ok.cv$zscore)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-2.13600 -0.45290 -0.13710  0.00455  0.35350  2.83100
```

Da queste informazioni si vede che molti punti presentano scarti piuttosto marcati. Gli score z del modello hanno invece valori abbastanza piccoli in modulo, indice del fatto che la varianza kriging delle stime è elevata.

Come indicazione di bontà del modello si possono calcolare varie statistiche, come la media dei residui, la media dei loro quadrati o la somma dei quadrati degli score z:

```
> mean(ok.cv$residual)
[1] 0.1513638
> mean(ok.cv$residual^2)
[1] 247.4845
> sum(ok.cv$zscore^2)
[1] 118.3454
```

Queste statistiche permettono di confrontare modelli differenti, scegliendo quello che produce risultati migliori.

10.4 Geostatistica basata su modello

L'approccio classico, basato sul calcolo e il fit del variogramma empirico, è largamente diffuso e presenta il chiaro vantaggio di non necessitare di nessuna ipotesi sulla forma di $S(x)$. L'unico vincolo al suo utilizzo è dato dalla assunzione di stazionarietà del campo di $S(x)$. Come sempre avviene per le

tecniche non parametriche, questa semplicità porta con sé delle limitazioni. La più evidente dipende dal fatto che il calcolo del variogramma empirico dipende dalla scelta dei bin spaziali: una scelta diversa si riflette in un variogramma diverso, che può portare a un fit teorico anche sostanzialmente differente.

L'approccio basato su modello invece richiede che si specifichi la distribuzione congiunta del processo stocastico $S(x)$. Un caso frequentemente trattato è quello in cui tale distribuzione è gaussiana multivariata $S(x_1) \dots S(x_n)$ per ogni intero n e per ogni posizione in esame; si dice allora che il processo $S(x)$ ha modello gaussiano. Nel caso di processo stazionario del secondo ordine $S(x)$ ha lo stesso valore di aspettazione e la stessa varianza per ogni x, e la correlazione tra $S(x)$ e $S(x')$ è esprimibile come:

$$\rho(h) = Corr(S(x), S(x'))$$

dove h è la distanza tra x e x'.

Le misure z_i sono viste come realizzazioni tra loro indipendenti di variabili casuali Z_i, normalmente distribuite e con valore atteso condizionato $E[Z_i|S] = S(x_i)$ e varianza τ^2. Equivalentemente si può scrivere:

$$z_i = S(x_i) + \varepsilon$$

con $S(x)$ che soddisfa le assunzioni precedenti e $\varepsilon \sim N(0, \tau^2)$.

Il caso appena descritto può essere immediatamente esteso per campi $S(x)$ non stazionari introducendo esplicitamente la modellizzazione del trend atteso, che può dipendere sia dalle coordinate geografiche, sia da altre covariate. Un esempio del primo caso è un problema in cui si studia la concentrazione di una data sostanza nel suolo in una zona in cui è presente una fonte di inquinamento: è logico attendersi che la concentrazione diminuisca allontanandosi da tale fonte. Un esempio del secondo tipo di problema è lo studio della resa agricola di un terreno suddiviso in aree trattate in diversi modi: il tipo di trattamento sarà una variabile da tenere in considerazione nella modellizzazione. In tutti questi casi invece di ipotizzare che il valore di aspettazione delle variabili casuali Z_i sia $E[Z_i|S] = S(x_i)$ si pone:

$$E[Z_i|S] = S(x_i) + \sum_{k=1}^{p} \beta_k d_k(x_i)$$

dove d_k sono le variabili esplicative inserite nel modello e β_k i coefficienti di regressione da stimare.

Si osservi che il processo descritto si presta al trattamento di campi $s(x)$ non gaussiani. In questo caso è possibile trasformare le variabili in studio, ad esempio mediante trasformazione di Box-Cox, in modo da rendere accettabile l'ipotesi di gaussianità.

Nel seguito viene presentata una panoramica per forza di cose limitata dell'utilizzo delle tecniche basate su modello. Per una trattazione dettagliata di questo tipo di approccio si rimanda a [22].

10.4.1 Funzioni di correlazione

Un modello appropriato è tale solo se la funzione di correlazione $\rho(h)$ è positiva definita. Questo implica che per ogni intero m e per ogni set di posizioni x_i e per ogni set di costanti reali a_i la combinazione lineare $\sum_{i=1}^{m} a_i S(x_i)$ abbia varianza non negativa. Per soddisfare il vincolo si è soliti scegliere la funzione $\rho(h)$ all'interno di una classe standard di modelli parametrici. Oltre alle funzioni di correlazione sferiche e gaussiane, trattate in precedenza, una funzione molto frequentemente adottata è dovuta a Matérn e ha la forma seguente:

$$\rho(h; \phi, \kappa) = \left(2^{\kappa-1}\Gamma(\kappa)\right)^{-1} (h/\phi)^{\kappa} K_{\kappa}(u/\phi)$$

dove K_κ indica le funzioni di Bessel modificate di secondo tipo di ordine κ. Il parametro $\phi > 0$ determina la velocità con cui la funzione tende a 0 al crescere di h. Il parametro $\kappa > 0$ è detto ordine del modello di Matérn ed è legato alla differenziabilità del processo stocastico $S(x)$; al crescere del suo valore la mappa del processo stocastico diventa sempre più omogenea.

10.4.2 Stima dei parametri del modello

Nel caso di modello gaussiano stazionario i parametri che entrano nel modello e che devono essere stimati sono la media μ, la varianza σ^2, la varianza legata al processo di misura τ^2 e i parametri della funzione di correlazione assunta. Se invece si suppone che il modello non sia stazionario allora sarà necessario specificare come la media dipenda dalle covariate adottate. Ad esempio se si ipotizza un modello lineare dalla covariata d_1 del tipo:

$$\mu(x) = \beta_0 + \beta_1 d_1$$

occorrerà stimare anche i valori di β_0 e β_1. Un modo per stimare efficientemente i parametri del modello, sia nel caso stazionario che in quello non stazionario, è far uso di metodi di maximum likelihood (ML). La stima nel caso di modello gaussiano è molto semplice. Tale semplicità viene persa, almeno dal punto di vista computazionale, per modelli diversi; tuttavia grazie a tecniche di trasformazione è frequentemente possibile ricondursi al modello gaussiano.

Si consideri il caso più generale di modello gaussiano in cui si abbiano n determinazioni del valore della variabile in studio in località georiferite. La media del processo si suppone modellizzabile come:

$$\mu(x) = D\beta$$

dove D è la matrice del modello, che contiene tutti 1 sulla prima colonna e i valori delle covariate sulle colonne seguenti. Le variabili di risposta Z seguono quindi la distribuzione normale multivariata:

$$Z \sim N(D\beta, \sigma^2 R(\phi) + \tau^2 I) \tag{10.4}$$

con R che determina la correlazione tra le misure e dipende dal parametro (o parametri) ϕ. In questo caso la log-likelihood è:

$$\begin{aligned} \ln L(\beta, \tau^2, \sigma^2, \phi) &= -0.5[n \ln(2\pi) + \ln|\sigma^2 R(\phi) + \tau^2 I| \\ &+ (z - D\beta)^T (\sigma^2 R(\phi) + \tau^2 I)^{-1} (z - D\beta)], \end{aligned} \tag{10.5}$$

come si può verificare facilmente calcolando il logaritmo naturale dell'espressione data nella Eq. 1.1. Per massimizzare la log-likelihood si può procedere in diversi modi. Ad esempio, posto $\nu^2 = \tau^2/\sigma^2$ e $V = R(\phi) + \tau^2 I$, derivando rispetto a β e a σ^2 e ponendo a zero i risultati si trovano i valori dei due parametri che massimizzano la likelihood, in funzione degli altri:

$$\hat{\beta}(V) = (D^T V^{-1} D)^{-1} D^T V^{-1} z$$

e

$$\hat{\sigma}^2(V) = n^{-1} \left(z - D\hat{\beta}(V)\right)^T V^{-1} \left(z - D\hat{\beta}(V)\right)$$

Sostituendo questa ultima uguaglianza nella Eq. 10.5 si ha, con pochi semplici passaggi:

$$\ln L_0(\nu^2, \phi) = -0.5[n \ln(2\pi) + n \ln \sigma^2(V) + \ln|V| + n]$$

Questa espressione può essere massimizzata numericamente rispetto ai parametri ν e ϕ; con una sostituzione all'indietro nelle espressioni date precedentemente si ottengono quindi i valori di $\hat{\sigma}$ e $\hat{\beta}$. Per le interessanti questioni algoritmiche concernenti la massimizzazione della likelihood e sulle possibili parametrizzazioni alternative si rimanda a [22].

Un'alternativa alla stima di maximum likelihood è quella dell'approccio basato su maximum likelihood ristretta (REML). In questo caso si trasformano i dati originali mediante una trasformazione lineare:

$$Z^* = AZ$$

scegliendo la matrice A in modo tale che la distribuzione di Z^* non dipenda da β. Si massimizza quindi la likelihood relativa ai dati trasformati rispetto ai parametri (ν^2, σ^2, ϕ). Una scelta possibile per la matrice A è la matrice di proiezione sui residui:

$$A = I - D(D^T D)^{-1} D^T$$

Dato che la trasformazione con cui si ottengono i valori Z^* è lineare, la distribuzione di Z^* è ancora multivariata normale. Gli stimatori di REML si ottengono quindi massimizzando la log likelihood ristretta, che si può scrivere in termini dei dati originali Z nel modo seguente:

$$\begin{aligned} \ln L^*(\tau^2, \sigma^2, \phi) &= -0.5[n\ln(2\pi) + n\ln|\sigma V| + \ln|D^T(\sigma^2 V)^{-1}D| \\ &+ \left(z - D\hat{\beta}(V)\right)^T (\sigma^2 V)^{-1} \left(z - D\hat{\beta}(V)\right)] \end{aligned} \tag{10.6}$$

Si noti che la matrice A non appare esplicitamente in questa espressione e che quindi le stime di REML non dipendono dalla scelta della trasformazione lineare utilizzata.

In generale le stime di REML sono meno distorte nel caso di piccoli campioni, tuttavia si può notare che nella valutazione di L^* appare la matrice del modello D e che quindi differenti scelte di covariate possono influenzare anche pesantemente le stime dei parametri. In definitiva, nessuna delle due tecniche sovrasta chiaramente l'altra.

In R le tecniche di stima di ML e REML sono disponibili tramite chiamata alla funzione *likfit* della libreria *geoR*.

Esempio

Riprendendo i dati già utilizzati più volte negli Esempi precedenti, si vuole calcolare il modello gaussiano che meglio si adatta a descriverli, assumendo che non vi siano covariate che entrano nel calcolo della media regionalizzata e che la matrice di covarianza sia adeguatamente descritta dal modello sferico.

```
> ml <- likfit(meuse.geo, ini=c(400,800), cov.model="spherical", lik.method="ML")
kappa not used for the spherical correlation function
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

La funzione accetta svariati argomenti: il dataset su cui operare, i valori di partenza dei parametri σ^2 e ϕ (opzione *ini*), il modello della matrice di covarianza e la tecnica di fit. Di default la funzione assume di dover stimare anche il valore di τ^2, con valore di partenza pari a 0. Questo comportamento può essere modificato agendo sulle opzioni *fix.nugget* (con valore di default FALSE, che consente di specificare se il valore di τ^2 deve essere assunto noto) e *nugget* (che permette di impostare il valore iniziale di τ^2, se è richiesta la sua stima, o il valore fisso da assumere nel modello in caso contrario). Come da avvertenza del messaggio di output, come per tutte le tecniche di ottimizzazione numerica, è consigliabile lanciare più volte la funzione con differenti valori iniziali dei parametri; questa tecnica permette di esplorare al meglio il profilo della funzione ed evitare di finire "intrappolati" in massimi locali.

L'output della funzione è il seguente:

```
> ml
likfit: estimated model parameters:
     beta     tausq   sigmasq       phi
"  54.48" "  52.66" "1008.96" "1204.86"
Practical Range with cor=0.05 for asymptotic range: 1204.865

likfit: maximised log-likelihood = -663.5
```

Vengono presentati nell'ordine la media stimata della concentrazione al suolo di rame (*beta*), i valori della varianza legata al procedimento di misura (*tausq*), quello legato alla variabilità intrinseca (*sigmasq*), il valore del range entro cui i dati sono spazialmente correlati (*phi*). Come si può notare i valori stimati mediante maximum likelihood sono sostanzialmente differenti con quanto ottenuto in Sec. 10.1 adattando al variogramma sperimentale un modello di variogramma teorico con funzione di covarianza sferica. In quel caso infatti il valori di nugget, equivalente di τ^2, risultava essere circa 176, quello di σ^2 di circa 429, mentre il valore del range era di circa 707. Il fatto non deve sorprendere, date le profonde differenze teoriche alla base dei due approcci.

È anche possibile esaminare i risultati del fit mediante l'uso della funzione *summary*, che ritorna numerose informazioni sul metodo di fit impiegato e sulle trasformazioni applicate sui dati:

```
> summary(ml)
Summary of the parameter estimation
-----------------------------------
Estimation method: maximum likelihood

Parameters of the mean component (trend):
   beta
54.4822

Parameters of the spatial component:
   correlation function: spherical
      (estimated) variance parameter sigmasq (partial sill) =  1009
      (estimated) cor. fct. parameter phi (range parameter)  =  1205
   anisotropy parameters:
      (fixed) anisotropy angle = 0  ( 0 degrees )
      (fixed) anisotropy ratio = 1

Parameter of the error component:
      (estimated) nugget =  52.66

Transformation parameter:
      (fixed) Box-Cox parameter = 1 (no transformation)

Practical Range with cor=0.05 for asymptotic range: 1204.865

Maximised Likelihood:
   log.L n.params      AIC      BIC
"-663.5"      "4"   "1335"   "1347"

non spatial model:
   log.L n.params      AIC      BIC
  "-710"      "2"   "1424"   "1430"
```

Per paragone si riporta di seguito il risultato del fit con tecnica di REML, ottenuto con l'opzione *lik.method="REML"*.

```
> ml.reml <- likfit(meuse.geo, ini=c(400,800), cov.model="spherical",
+  lik.method="REML")
[...]

> ml.reml
likfit: estimated model parameters:
     beta     tausq   sigmasq       phi
"  54.69" "  49.79" "1039.23" "1208.46"
```

```
Practical Range with cor=0.05 for asymptotic range: 1208.457

likfit: maximised log-likelihood = -657.7
```

Come si nota, le stime dei parametri non differiscono di molto.

Un'altra possibilità è quella di includere la dipendenza della media da altre covariate. Nella chiamata alla funzione *likfit* è possibile inserire tale informazione mediante l'opzione *trend*. Nell'esempio seguente si assume che la media della concentrazione di rame nel suolo dipenda esplicitamente dalle coordinate geografiche, che all'interno del dataset vengono chiamate x[1] e y:

$$\mu = \beta_0 + \beta_1 x + \beta_2 y$$

Dato che le coordinate sono disponibili nell'oggetto *coords* del dataset *meuse.geo* la chiamata per il fit del modello è:

```
> ml.coord <- likfit(meuse.geo, trend=~coords, ini=c(400,800),
+  cov.model="spherical", lik.method="ML")
[...]

> ml.coord
likfit: estimated model parameters:
       beta0        beta1        beta2        tausq      sigmasq          phi
"-4970.8001" "   -0.0349" "    0.0341" "   59.6850" "  858.2583" " 1190.7852"
Practical Range with cor=0.05 for asymptotic range: 1190.785

likfit: maximised log-likelihood = -657.7
```

Si nota che l'introduzione degli ulteriori parametri nel fit altera la stima degli altri in modo sensibile.

10.4.3 Mappe predittive basate su modello

Spesso in ambito geostatistico, più che una stima dei parametri del modello che si suppone sottendere i dati, è di interesse andare a utilizzare tale modello per realizzare una mappa predittiva da cui si possa desumere il valore di una variabile casuale $T = T(S)$, anche nei punti dove non si disponga di misura diretta. In ambito classico il problema viene trattato mediante interpolazione kriging.

Si supponga che sia di interesse la stima di $T = S(x)$, ossia il caso più semplice in cui si vuole stimare il valore del segnale anche dove non è stato direttamente misurato. Nel caso di modello gaussiano, come si osserva dall'Eq. 10.4 la variabile Z ha distribuzione multivariata gaussiana con matrice di covarianza:

$$\sigma^2 R(\phi) + \tau^2 I = \sigma^2 V$$

siano $r_{ij} = \rho(|x_i - x_j|)$ gli elementi di R, r il vettore di elementi $r_i = \rho(|x - x_i|)$ e $\mu\mathbf{1}$ il vettore delle medie della distribuzione di Z. Allora il predittore dei minimi quadrati per T è (si veda [22] per i dettagli):

$$\hat{T} = \mu + r^T V^{-1}(Z - \mu\mathbf{1}) \tag{10.7}$$

con varianza:

$$Var(T|Z) = \sigma^2(1 - r^T V^{-1} r) \tag{10.8}$$

In ambito geostatistico classico l'Eq. 10.7, utilizzata con la stima plugin dei parametri del modello (compreso il valore di μ), coincide con la stima di simple kriging. Se invece si assume che tutti i parametri siano fissati ai loro valori plugin, eccetto la media, stimata mediante stimatore di minimi quadrati generalizzati come:

$$\hat{\mu} = (\mathbf{1}^T V^{-1} \mathbf{1})^{-1} \mathbf{1}^T V^{-1} Z$$

allora si ottiene la stima di kriging ordinario.

Lavorando in ambito model-based, in R è possibile ottenere una mappa predittiva utilizzando la funzione *krige.conv* della libreria *geoR*.

[1] Il simbolo x qui usato si riferisce alla coordinata E-W e non va confuso con l'eguale simbolo utilizzato nel resto del Capitolo

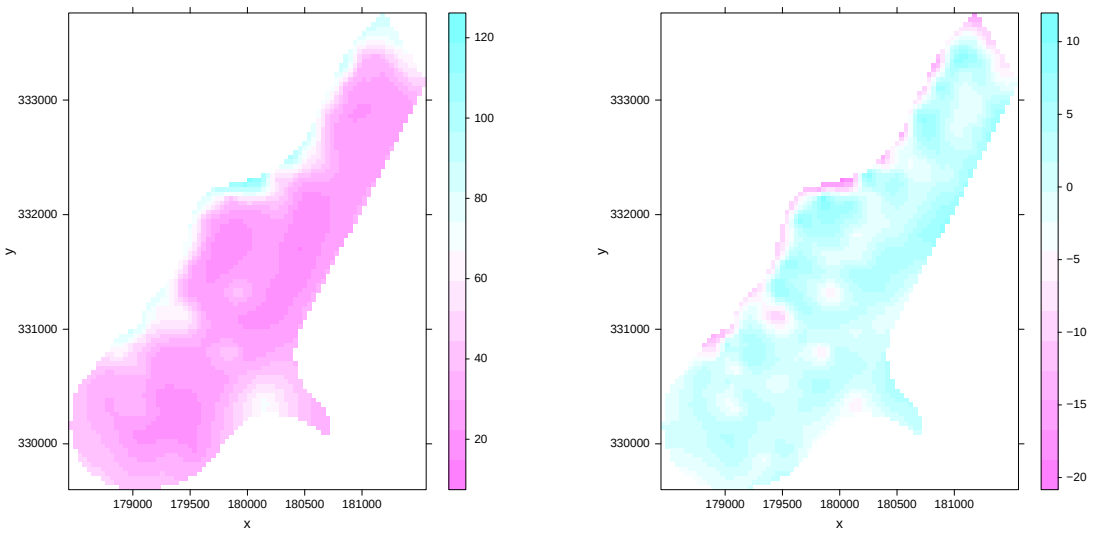

Figura 10.6: A sinistra: mappa previsionale ottenuta mediante kriging a partire da un modello gaussiano, fittato con tecnica di maximum likelihood. A destra: differenze tra le stime di kriging ordinario e quella equivalente ottenuta da modello.

Esempio

Nel caso dell'Esempio già ampiamente discusso nel Capitolo, occorre per prima cosa estrarre le coordinate delle località in cui effettuare la stima:

```
> coord <- coordinates(meuse.grid)
```

La predizione kriging si effettua tramite la funzione *krige.conv*, che sfrutta delle funzioni aggiuntive per inizializzare in modo appropriato i parametri del modello. In particolare la chiamata a *krige.control* serve per specificare i parametri del modello:

```
> KC <- krige.control(obj.model=ml)
```

In questo caso si assumono i parametri ottenuti in precedenza dal fit di maximum likelihood. Per la predizione spaziale la sintassi di base è:

```
> pred <- krige.conv(meuse.geo, loc=coord, krige=KC)
```

La funzione accetta numerosi argomenti; in questo caso sono stati utilizzati il dataset su cui operare, le località in cui effettuare le previsioni, i parametri del modello da utilizzare.

Il risultato della chiamata è un oggetto che contiene, come minimo, le stime del valore di interesse nelle località del grigliato

```
> pred$predict
   [1]  84.45799  84.32614  83.57463  83.13417  84.29153  82.90553  82.02963
[...]
```

sia le loro varianze:

```
> pred$krige.var
   [1] 403.85204 310.46068 338.41685 368.84586 211.26537 243.16560 276.83018
[...]
```

Tra gli argomenti opzionali della funzione *krige.conv*, molto interessante è l'opzione *borders*, utile quando si ha un poligono che definisce i confini della zona su cui operare. In questo caso si può definire un grigliato di interpolazione rettangolare, che comprenda completamente la regione di interesse, e utilizzare l'opzione suddetta per restringere l'interpolazione ai soli punti della griglia contenuti all'interno del poligono di confine. Nei punti esterni verrà assegnato il valore di interpolazione NA. Questa metodica consente di utilizzare la funzione *image* per realizzare la mappa previsionale. Dato che nell'esempio in questione il grigliato di interpolazione non è rettangolare, questa funzione non è in grado di operare e si può procedere come visto in precedenza con la chiamata a *levelplot*.

```
> levelplot(pred$predict ~ coord[,1] + coord[,2], xlab="x", ylab="y", aspect="iso")
```

In Fig. 10.6 a sinistra si ha il risultato della chiamata. Nella medesima figura a destra si ha la mappa delle differenze tra le stime derivanti dal modello e quelle ottenute mediante kriging ordinario, date in Sec. 10.3.2. Il comando per realizzare tale figura è:

```
> levelplot(var1.pred-pred$predict~x+y, data=as.data.frame(kr.ok), aspect="iso")
```

Come si nota le differenze sono modeste, con la discrepanza maggiore lungo il confine nord dell'area in esame.

10.5 Modelli geostatistici lineari generalizzati

Quando la variabile di risposta è di tipo particolare come un conteggio o un indicatore dicotomico, le tecniche finora elencate non possono affrontare il problema. In ambito classico, per variabili di risposta non dipendenti, la questione viene affrontata nell'ambito dei modelli lineari generalizzati. Quando le variabili sono dipendenti, come in ambito geostatistico, esistono soluzioni che permettono di trattare il problema, ad esempio nell'ambito dei modelli lineari generalizzati misti.

Il modello in questo caso si compone di un processo gaussiano stazionario $S(x)$, con le stesse caratteristiche dette in precedenza, e della descrizione del meccanismo che genera i dati, condizionati al segnale ignoto $S(x)$. Come nel caso di modello generalizzato classico, si suppone che, condizionate a S le risposte Z_i con $i = 1, \ldots, n$ nelle posizioni x_i siano variabili casuali indipendenti il cui valore di aspettazione $\mu_i = E[Z_i|S]$ è dato da:

$$h(\mu_i) = S(x_i) + \sum_{k=1}^{p} \beta_k d_k(x_i) \tag{10.9}$$

dove $h()$ è una appropriata funzione di link, d_k sono i predittori inclusi nel modello e β_k i coefficienti ignoti di regressione. La distribuzione di Z_i dato S, è detta distribuzione degli errori.

Solitamente la struttura del problema suggerisce il tipo di modello da utilizzare; ad esempio se si ha a che fare con conteggi, la distribuzione di Poisson è solitamente un buon modello per la distribuzione degli errori così come il modello log-lineare per descrivere i dati. Se invece si ha a che fare con una variabile di risposta dicotomica, la distribuzione degli errori è binomiale, e il modello logit-lineare è frequentemente adottato per descrivere i dati.

10.5.1 Stime dei parametri per modelli geostatistici lineari generalizzati

L'applicazione di metodologie di maximum likelihood nel caso di modelli geostatistici lineari generalizzati è resa problematica da difficoltà computazionali, essenzialmente dovute alla alta dimensionalità del vettore $S = S(x_1), \ldots, S(x_n)$. Dato che le osservazioni $Z = (Z_1, \ldots, Z_n)$ sono condizionalmente indipendenti dato S, la funzione di likelihood assume una forma semplice. Sia θ i parametri che determinano la distribuzione di Z dato S, e sia $f_i(z_i|S, \theta)$ la distribuzione condizionata di Z_i dati S e θ. Si ha allora, per la funzione di likelihood per θ condizionata a S:

$$L(\theta|S) = \prod_{i=1}^{n} f_i(z_i|S, \theta)$$

Sia $g(S;\phi)$ la distribuzione congiunta di S con parametro ϕ. La funzione di likelihood basata su Z si ottiene marginalizzando rispetto alle variabili stocastiche S:

$$L(\theta,\phi) = \int_S \prod_{i=1}^{n} f_i(z_i|S,\theta)\; g(s|\phi)\; ds$$

Se le S_i sono mutualmente indipendenti l'integrale si riduce al prodotto di integrali unidimensionali e la soluzione numerica è banale. Dato che invece in ambito geostatistico le S_i sono dipendenti, l'integrale ha la stessa dimensione di Z e le tecniche numeriche di integrazione non riescono a trattare il problema.

Oltre a possibili approcci di tipo Monte Carlo o di quasi likelihood, per i quali si rimanda a [22], è possibile affrontare efficacemente la questione in un ambito di analisi Bayesiana. Nella sezione seguente viene presentato un approccio possibile per modelli gaussiani e non gaussiani in ambito geostatistico.

10.6 Tecniche Bayesiane applicate a problemi di Geostatistica

In ambito Bayesiano cade la distinzione formale tra il processo latente S e il vettore dei parametri θ; entrambe vengono considerate variabili casuali non direttamente osservabili. Il punto di partenza è la specificazione della distribuzione congiunta dei dati Z, del segnale S e dei parametri θ. Il modello è scrivibile in maniera gerarchica nel modo seguente:

$$f(Z,S,\theta) = f(\theta)f(S|\theta)f(Z|S,\theta) \tag{10.10}$$

dove $f(\theta)$ indica la distribuzione a priori per θ, che riassume la conoscenza dello sperimentatore prima di condurre l'esperimento. In pratica una corretta specificazione di $f(\theta)$ è proprio l'ostacolo e la difficoltà maggiore che si incontra nell'adozione di un approccio Bayesiano. In particolare in ambito geostatistico, come riportato in [22], le difficoltà sono accresciute dalla dipendenza esistente tra i dati. Infatti questo effetto riduce l'influenza della likelihood dei dati sulla distribuzione a priori dei parametri nell'ottenerne la stima a posteriori. Quindi, anche per campioni apparentemente grandi, la scelta di diversi prior porta a stime anche sostanzialmente differenti della distribuzione a posteriori dei parametri.

La distribuzione predittiva di S, definita come $f(S|Z)$, si può ottenere a partire dall'Eq. 10.10 applicando il teorema di Bayes:

$$f(S|Z) = \int f(S|Z,\theta)f(\theta|Z)d\theta$$

ossia una media pesata della distribuzione predittiva $f(S|Z,\theta)$ plugin, pesata dalla distribuzione a posteriori dei parametri θ. Inoltre, posto $T = T(S)$ il target di predizione, la distribuzione predittiva per T segue immediatamente da quella di S, dato che la trasformazione da S a T è deterministica. In pratica basterà campionare dalla distribuzione predittiva di S e e per ogni campione calcolare il corrispondente valore di T.

Nel caso di modello gaussiano, se $T = S(x)$ e con una appropriata scelta delle distribuzioni a priori dei parametri, i risultati Bayesiani per la distribuzione di T possono essere ottenuti esplicitamente.

10.6.1 Stima Bayesiana per i parametri di un modello gaussiano

La stima Bayesiana dei parametri parte dalla funzione di likelihood $L(\theta;z)$ e la combina con la stima a priori dei parametri $f(\theta)$ per ottenere la distribuzione a posteriori di θ:

$$f(\theta|z) = \frac{L(\theta;z)\; f(\theta)}{\int L(\theta;z)\; f(\theta)\; d\theta}$$

La distribuzione a posteriori può essere quindi utilizzata per ottenere un inferenza sui parametri, esprimibile in termini di intervallo di credibilità.

All'interno del modello gaussiano, con apposite specificazioni della forma della distribuzione a priori, è possibile ottenere in maniera esplicita la distribuzione a posteriori di θ. In particolare, posto per semplicità $\tau^2 = 0$ e considerando noti tutti i parametri della funzione di correlazione (ipotesi semplicistica che verrà rilassata in seguito), la famiglia di prior coniugati per (β, σ^2) è la Gaussian-Scaled-Inverseχ^2:

$$f(\beta|\sigma^2, \phi) \sim N(m_b, \sigma^2 V_b) \quad , \quad f(\sigma^2|\phi) = \chi^2_{ScI}(n_\sigma, S^2_\sigma)$$

dove la distribuzione $\chi^2_{ScI}(n_\sigma, S^2_\sigma)$ ha densità:

$$f(x) \propto x^{-(n_\sigma/2+1)} \exp(-n_\sigma S^2_\sigma/(2x)) \quad x < 0$$

Per brevità spesso queste espressioni si riassumono in:

$$f(\beta, \sigma^2|\phi) \sim N\chi^2_{ScI}(m_b, V_b, n_\sigma, S^2_\sigma)$$

Da queste espressioni, usando il teorema di Bayes e l'espressione di likelihood data in Eq. 10.5, si ottengono le distribuzioni a posteriori dei parametri:

$$f(\beta, \sigma^2|z, \phi) \sim N\chi^2_{ScI}(\tilde{\beta}, V_{\tilde{\beta}}, n_\sigma + n, S^2) \tag{10.11}$$

con:

$$\tilde{\beta} = V_{\tilde{\beta}}(V_b^{-1} m_b + D^T R^{-1} z) \quad , \quad V_{\tilde{\beta}} = (V_b^{-1} + D^T R^{-1} D)^{-1}$$

e:

$$S^2 = \frac{n_\sigma S^2_\sigma + m_b^T V_b^{-1} m_b + z^T R^{-1} z - \tilde{\beta}^T V_{\tilde{\beta}}^{-1} \tilde{\beta}}{n_\sigma + n}$$

Si osservi che il grado di incertezza nei prior è controllato dai parametri delle distribuzioni a priori; in particolare S^2_σ e m_b stimano i valori di σ e β, mentre n_σ e V_b sono legate all'incertezza di tali stime.

Per fare un passo in una direzione più realistica, si può abbandonare l'ipotesi che i parametri della funzione di covarianza siano noti, pur supponendo che vi sia un unico parametro ϕ da stimare. In questo caso è possibile adottare un prior del tipo:

$$f(\beta, \sigma^2, \phi) = f(\beta, \sigma^2|\phi) f(\phi)$$

utilizzando i prior visti sopra per $f(\beta, \sigma^2|\phi)$ e una distribuzione appropriata per ϕ. Per motivi computazionali spesso il supporto per $f(\phi)$ viene discretizzato (si veda [22]). La distribuzione a posteriori dei parametri sarà quindi:

$$f(\beta, \sigma^2, \phi|z) = f(\beta, \sigma^2|z, \phi) f(\phi|z)$$

con $f(\beta, \sigma^2|z, \phi)$ dato in Eq. 10.11 e:

$$f(\phi|z) \propto f(\phi)|V_{\tilde{\beta}}|^{1/2}|R|^{-1/2}(S^2)^{-(n+n_\sigma)/2} \tag{10.12}$$

Per simulare valori dalle distribuzioni a posteriori si inizia con il calcolare la distribuzione $f(\phi|z)$; quindi si campiona un valore di ϕ da tale distribuzione, si inserisce nelle espressioni per le distribuzioni a posteriori per β e σ^2 e si campionano valori da queste. Il processo viene ripetuto il numero desiderato di volte. Si è quindi in presenza di un processo di simulazione diretta e non di un processo MCMC. In particolare quindi la varianza legata al processo simulativo scala con la radice quadrata del numero di repliche, il che offre un metodo per controllarne la grandezza.

L'introduzione della stima di τ^2 non presenta ulteriori complicazioni teoriche. Basterà utilizzare un supporto discreto sia per ϕ che per $\nu^2 = \tau^2/\sigma^2$ e sostituire a R nelle equazioni precedenti il valore $V = R + \nu^2 I$.

parametro	stima	95% CI
β	69.1	[27.3-128.9]
σ^2	2396.7	[633.4-5857.3]
ϕ	455.8	[230-815]
ν^2	0.094	[0.031-0.281]

Tabella 10.1: Medie e intervallo di credibilità (CI) dei parametri del modello Bayesiano, ottenuti dalle loro distribuzioni a posteriori.

Esempio

In R è possibile utilizzare la funzione *krige.bayes* della libreria *geoR* per un approccio Bayesiano nel caso di modello gaussiano. Per definire le informazioni relative al modello, ai prior e all'output della funzione sono disponibili tre funzioni di servizio: *model.control*, *prior.control* e *output.control*. Nell'esempio seguente si imposta un modello con funzione di correlazione di Matérn (scelta di default), con $k = 1.5$, si specificano i prior per ϕ e ν^2 (chiamato *tausq.rel*) su un supporto discreto con 21 bin per ϕ e 17 per ν^2, e si richiedono in output 1000 simulazioni nella valutazione delle distribuzioni a posteriori dei parametri del modello.

```
> MC <- model.control(kappa=1.5)
> PC <- prior.control(phi.discrete=seq(100,1400,l=21), phi.prior="reciprocal",
+  tausq.rel.prior="unif", tausq.rel.discrete=seq(0,0.5,l=17))
> OC <- output.control(n.post=1000)
```

Per le altre numerose opzioni di queste funzioni di appoggio si rimanda alle rispettive pagine di manuale. La scelta dell'intervallo discreto per i prior deve essere fatto in maniera da coprire il range di valori che si suppongono ragionevoli. Un buon metodo per capire se la scelta è stata fatta in modo non troppo restrittivo è quello di controllare se le distribuzioni a posteriori assegnano valori di probabilità trascurabile agli estremi dell'intervallo; in caso contrario è meglio ripetere l'analisi selezionando un range più grande.

Con gli input così preparati è possibile effettuare in un passo solo l'inferenza e la predizione Bayesiana:

```
> skb <- krige.bayes(meuse.geo, loc=coord, model=MC, prior=PC, output=OC)
```

Gli input della funzione sono il dataset su cui operare, un argomento opzionale che contiene le posizioni dei punti in cui stimare il segnale, e gli input preparati in precedenza. In assenza del secondo elemento la funzione si limita alla sola stima a posteriori dei parametri. Nell'utilizzare questa chiamata è indispensabile essere consci del fatto che il tempo di calcolo si misura in ore[2], determinate in larga misura dalla presenza del grigliato di stima. Quindi prima di inserire anche la parte relativa alla stima del segnale è opportuno valutare esclusivamente la parte inferenziale, controllando le distribuzioni a posteriori dei parametri.

Al termine del calcolo, l'oggetto *skb* contiene numerose informazioni. In particolare a partire da *skb*$*posterior* si trovano le informazioni relative alle distribuzioni a posteriori dei parametri, mentre in *skb*$*predictive* si hanno le stime predittive del modello. Nel dettaglio, le informazioni relative alle distribuzioni a posteriori dei parametri si possono trovare nell'oggetto *skb*$*posterior*$*sample*, che ospita il dataframe con i valori generati nel processo di simulazione:

```
> skb$posterior$sample
         beta  sigmasq phi tausq.rel
1    46.01254 1261.637 360   0.12500
2    93.61865 3944.963 490   0.03125
[...]
1000 58.77660 1792.513 360   0.09375
```

[2]Circa 4-5 ore di calcolo con una CPU Intel Core 2 Duo a 1.83 GHz.

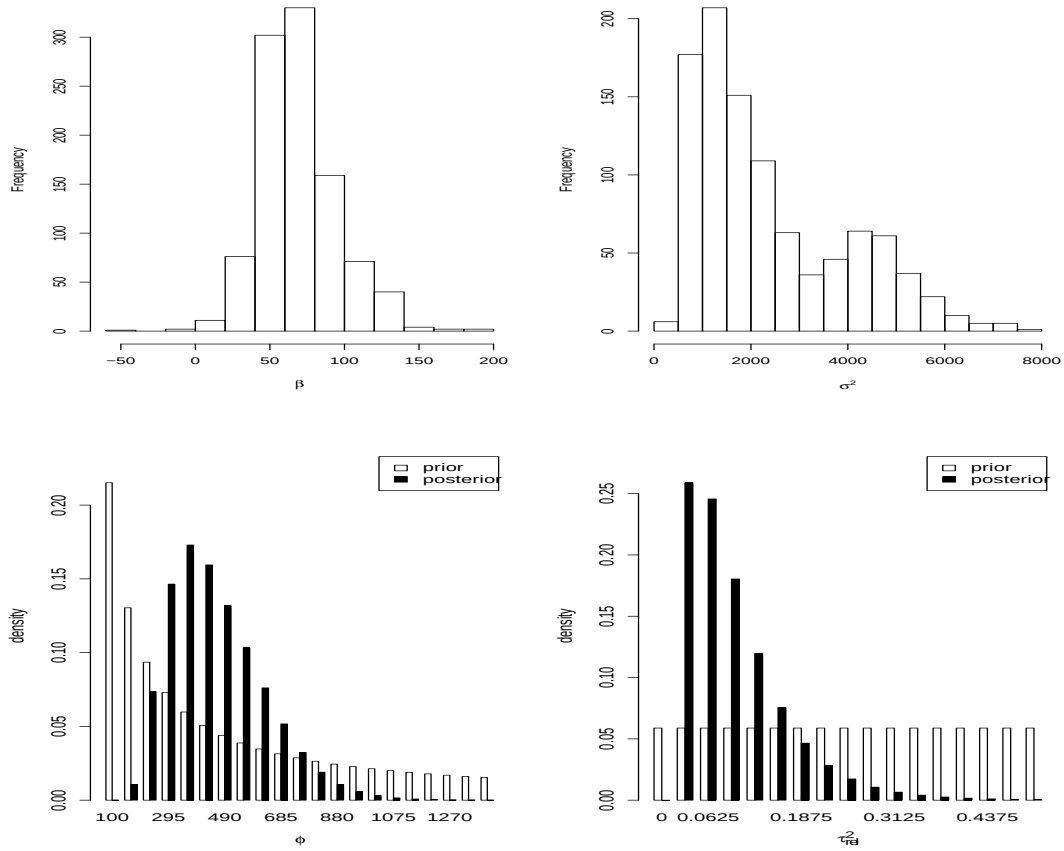

Figura 10.7: Istogrammi delle distribuzioni a posteriori dei parametri del modello Bayesiano.

Un riepilogo rapido sui valori dei quattro parametri può essere ottenuto nel modo seguente:

```
> Post <- skb$posterior$sample
> summary(Post)
      beta             sigmasq             phi             tausq.rel
 Min.   :-44.39   Min.   : 398.4   Min.   : 165.0   Min.   :0.03125
 1st Qu.: 52.07   1st Qu.:1196.5   1st Qu.: 360.0   1st Qu.:0.03125
 Median : 64.85   Median :1861.0   Median : 425.0   Median :0.06250
 Mean   : 69.11   Mean   :2396.7   Mean   : 455.8   Mean   :0.09450
 3rd Qu.: 82.72   3rd Qu.:3508.7   3rd Qu.: 555.0   3rd Qu.:0.12500
 Max.   :190.87   Max.   :7804.6   Max.   :1205.0   Max.   :0.46875
```

mentre per valutare l'intervallo di credibilità al 95% dei parametri si può ricorrere alla funzione *quantile*. Ad esempio, per il parametro β, la chiamata è la seguente:

```
> quantile(Post[,1], c(0.025, 0.975))
    2.5%    97.5%
 27.2955 128.9435
```

Gli argomenti della funzione sono il set di dati su cui operare – in questo caso la prima colonna del dataset *Post* – e il vettore contenente i quantili di interesse (il 2.5% e il 97.5%). I risultati relativi a tutti e quattro i parametri sono riepilogati in Tab. 10.1.

È anche possibile visualizzare gli istogrammi delle distribuzioni a posteriori dei parametri del modello:

```
> hist(Post$beta, xlab=bquote(beta), main="")
> hist(Post$sigmasq, xlab=bquote(sigma^2), main="")

> plot(skb)
```

Si osservi che la funzione *plot*, chiamata sull'oggetto *skb* ritorna automaticamente i grafici per i parametri ϕ e ν^2. I grafici così ottenuti sono riportati in Fig. 10.7.

Per quanto riguarda la parte predittiva, a partire da *skb*$*predictive* si trovano i vettori con i valori delle medie e varianze stimate:

```
> skb$predictive$mean[1:5]                    # primi 5 valori
[1] 90.22357 88.20401 87.70465 87.55790 85.84244

> skb$predictive$variance[1:5]                # primi 5 valori
[1] 274.9304 178.1189 198.5023 227.3427 109.3192
```

Questi valori sono calcolati a partire dalle distribuzioni congiunte dei parametri e non coinvolgono il processo di simulazione Monte Carlo eseguito per ogni località; i valori che risultano da tale processo sono:

```
> skb$predictive$mean.simulations[1:5]       # primi 5 valori
[1] 90.67137 88.44848 88.13973 88.10859 85.91451

> skb$predictive$variance.simulations[1:5]   # primi 5 valori
[1] 280.4722 184.8383 206.0743 233.8012 115.8742
```

che, come si osserva, sono del tutto confrontabili con i precedenti. Il dataset delle simulazioni è a sua volta disponibile in *skb*$*predictive*$*simulations* ed è una matrici di dimensione pari al numero di punti del grigliato di simulazione per il numero di simulazioni Monte Carlo eseguite (in questo caso 3103×1000).

10.6.2 Stima Bayesiana per modelli lineari generalizzati

Le tecniche basate sulla massimizzazione della funzione di likelihood non sono computazionalmente adeguate ad affrontare modelli geostatistici lineari generalizzati. La soluzione è quindi spesso quella di ricorrere a tecniche Bayesiane e in particolare a tecniche Monte Carlo basate su catene di Markov (MCMC). In questo ambito, la stima dei parametri del modello si basa sulla generazione di un campione Monte Carlo estratto dalla distribuzione a posteriori dei parametri $f(\theta, \beta|Z)$, mentre per avere una mappa predittiva sarà necessario campionare la distribuzione $f(S^*|Z)$ dove S^* è grave il vettore $S(x)$ nei luoghi in cui si vuole avere la previsione. Per i numerosi dettagli matematici relativi alle problematiche sia teoriche che computazionali si rimanda il lettore interessato a [22].

In R è possibile utilizzare la libreria *geoRglm* per analisi di questo tipo.

Esempio

Il dataset *gambia* distribuito con la libreria *geoRglm* riporta dati relativi a una indagine sulla prevalenza infantile della malaria realizzata in villaggi del Gambia (Thomson et al., *Predicting malaria infection in Gambian children from satellite data and bednet use surveys: the importance of spatial correlation in the interpretation of results*, American Journal of Tropical Medicine and Hygiene 61:2-8, 1999). L'indagine originaria era stratificata a due livelli (livello di villaggio e livello di individuo). Qui interessa solamente un disegno semplificato in cui i dati vengono analizzati a livello di villaggio. La prima cosa da farsi è quindi predisporre i dati da analizzare estraendo una riga per ogni villaggio, contenente le variabili di interesse a tale livello, ossia le coordinate geografiche, un indice vegetativo relativo alle vicinanze del villaggio (*green*), l'indicatore di presenza di un centro medico nel villaggio (*phc* codificato come assenza (0)/presenza (1)).

```
> data(gambia)
> ind <- paste("x",gambia[,1], "y", gambia[,2], sep="")
> village <- gambia[!duplicated(ind),c(1:2,7:8)]
```

A queste variabili si aggiungono il conteggio delle positività per malaria e il numero di individui testati:

```
> village$pos <- as.vector(tapply(gambia$pos, ind, sum))
> village$n <- as.vector(tapply(gambia$pos, ind, length))

> village
            x       y green phc pos  n
1850 349631.3 1458055 40.85   1  17 33
1696 358543.1 1460112 40.85   1  19 63
744  360308.1 1460026 40.10   0   7 17
[...]
```

Si deve prestare particolare attenzione alla dimensione campionaria in ogni località; infatti per esperimenti di tipo binomiale è necessario avere un buon numero di individui testati in ogni luogo in modo che le stime locali della prevalenza siano affidabili. Una buona scelta è quella di testare almeno 30 individui per ogni località inclusa nello studio. Lo studio in esame include un numero soddisfacente di individui in quasi tutti i punti, come si può verificare visualizzando i quantili del numero di individui testati:

```
> summary(village$n)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   8.00   27.00   30.00   31.31   33.00   63.00
```

Il dataset *village* deve essere trasformato in un oggetto della classe geodata per procedere con le analisi. In questo caso, oltre alla colonna contenente i dati (ossi il numero di individui positivi), è indispensabile disporre dell'informazione di quanti individui sono stati testati in ogni villaggio e di quali sono le covariate da includere nel modello geostatistico. La prima operazione si effettua utilizzando l'argomento *units.m.col*, mentre la seconda richiede l'argomento *covar.col*:

```
> vil <- as.geodata(village, data.col=5, units.m.col=6, covar.col=c(3,4))
```

La parte relativa all'inclusione del trend dovuto agli effetti delle covariate nel modello si specifica con la funzione *trend.spatial* della libreria *geoR*:

```
> TS <- trend.spatial(trend= ~ green + factor(phc), vil)
```

Il modello lineare generalizzato può quindi essere specificato a partire dal trend e dalla funzione di correlazione desiderata. Come è uso comune nel caso di modelli logistici con componente spaziale, si fa uso di una funzione di covarianza esponenziale:

```
> MOD <- model.glm.control(trend.d=TS, cov.model ="exponential")
```

L'analisi Bayesiana necessita della specificazione dei prior. In questo caso si richiede per il parametro ϕ una distribuzione a priori proporzionale a $1/\phi$, con supporto discreto da 10 a 100000 m, scandito ogni 200 m, e per σ^2 un prior improprio proporzionale a $1/\sigma^2$:

```
> PGC <- prior.glm.control(phi.prior="reciprocal",
+   phi.discrete=seq(10,100000,by=200), sigmasq.prior="reciprocal")
Warning message:
In prior.glm.control(phi.prior = "reciprocal", phi.discrete = seq(10,  :
  This choice of sigmasq.prior gives an improper posterior !!!!!!!
```

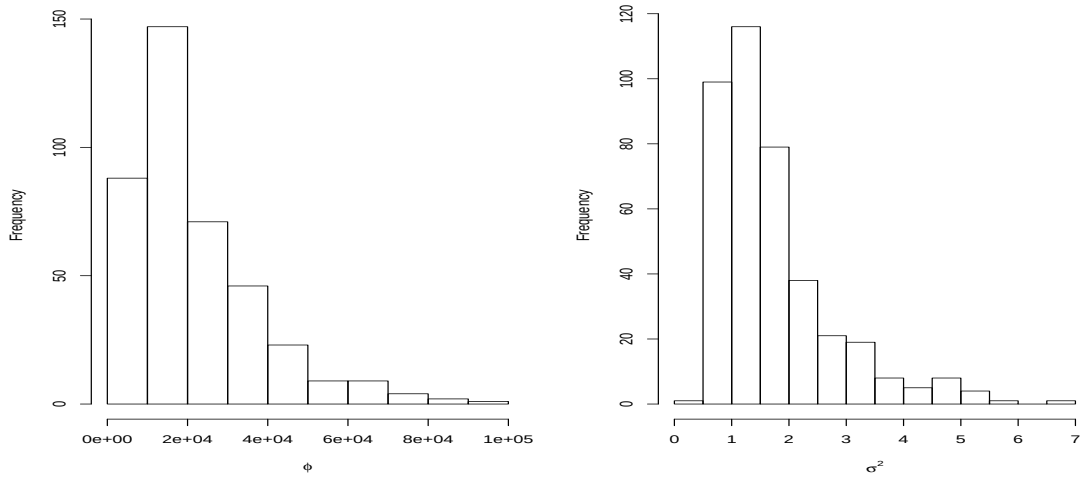

Figura 10.8: Istogrammi delle distribuzioni a posteriori dei parametri stimati con tecnica Bayesiana del modello generale linearizzato relativo alla prevalenza infantile di malaria nel Gambia.

La scelta di distribuzioni a priori diffuse riflette il grado di ignoranza dello sperimentatore e consente ai dati di avere una maggiore influenza nella determinazione delle distribuzioni a posteriori.

L'ultimo parametro da impostare riguarda le scelte relative al processo di simulazione Monte Carlo. La funzione *mcmc.control* consente di specificare le iterazioni richieste (*n.iter*), la lunghezza della coda iniziale di iterazioni scartate (fase di burn-in) necessaria per portare la catena in convergenza (*burn.in*), il thinning ossia ogni quante iterazioni campionare il valore (*thin*). Questa ultima fase è utile per migliorare le caratteristiche di mixing della catena, evitando di avere dati eccessivamente correlati tra loro. Le ultime due opzioni riguardano il fattore di scala della varianza attesa per le distribuzioni di S e di ϕ. Il suggerimento dato dagli sviluppatori del software è quello di agire su questi due parametri fino ad ottenere, nell'algoritmo di campionamento Metropolis-Hastings, un acceptance rate di circa il 60% per S e un acceptance rate di circa il 20-30% per ϕ. I parametri adottati in questo caso permettono di avere valori di acceptance rate simili a quelli consigliati.

```
> MCc <- mcmc.control(S.scale=0.025, n.iter=40000, thin=100,
+   phi.sc=2e+8, burn.in=50000)
```

Si dispone ora di tutti gli elementi per eseguire la simulazione MCMC. Per eseguirla si ricorre alla funzione *binom.krige.bayes*, la quale accetta il dataset su cui operare, e le opzioni impostate nei passi precedenti:

```
> bkb <- binom.krige.bayes(vil, model=MOD, prior=PGC, mcmc=MCc)
binom.krige.bayes: model with mean defined by covariates provided by the user
burn-in =  50000 is finished; Acc.-rate = 0.46 ; Acc-rate-phi = 0.38
iter. numb. 51000 ; Acc.-rate = 0.84 ; Acc-rate-phi = 0.07
iter. numb. 52000 ; Acc.-rate = 0.77 ; Acc-rate-phi = 0.12
iter. numb. 53000 ; Acc.-rate = 0.87 ; Acc-rate-phi = 0.06
iter. numb. 54000 ; Acc.-rate = 0.64 ; Acc-rate-phi = 0.20
[...]
iter. numb. 90000 ; Acc.-rate = 0.73 ; Acc-rate-phi = 0.14
MCMC performed: n.iter. =  40000 ; thinning =  100 ; burn.in =  50000
Only Bayesian estimation of model parameters
```

I risultati relativi alle distribuzioni a posteriori dei parametri del modello si trovano nell'oggetto *bkb*$*posterior*. In particolare la mediana e l'intervallo Bayesiano di credibilità (CI) al 95% per ϕ e σ^2 si ottengono con le chiamate:

parametro	mediana	95% CI
Intercetta	-0.17	(-3.10, 3.51)
green	-0.001	(-0.062, 0.056)
phc=1	-0.40	(-0.78, -0.03)
ϕ	16810	(4410, 67020)
σ^2	1.44	(0.66, 4.60)

Tabella 10.2: Risultati del fit del modello Bayesiano. Si nota in particolare l'ampiezza degli intervalli di credibilità.

```
> quantile(bkb$posterior$phi$sample, c(0.025,0.5,0.975))
 2.5%   50% 97.5%
 4410 16810 67020

> quantile(bkb$posterior$sigmasq$sample, c(0.025,0.5,0.975))
     2.5%       50%     97.5%
0.6553979 1.4384906 4.6003716
```

Mentre quelle relative ai tre coefficienti di regressione possono essere valutate con la chiamata seguente:

```
> for(i in 1:3) print(quantile(bkb$posterior$beta$sample[i,], c(0.025,0.5,0.975)))
      2.5%        50%      97.5%
-3.1032740 -0.1687295  3.5099021
        2.5%          50%        97.5%
-0.061656696 -0.001209571  0.056158573
       2.5%         50%       97.5%
-0.77643510 -0.40079602 -0.02743690
```

È anche possibile visualizzare gli istogrammi delle distribuzioni a posteriori dei vari parametri. Ad esempio quelli per ϕ e σ^2, mostrati Fig. 10.8, possono essere realizzati con le seguenti chiamate:

```
> hist(bkb$posterior$phi$sample, xlab=bquote(phi), main="")
> hist(bkb$posterior$sigma$sample, xlab=bquote(sigma^2), main="")
```

Le informazioni relative la modello sono riepilogate in Tab. 10.2, dove si osserva che l'effetto relativo all'indice vegetativo non risulta significativo, al contrario di quello dovuto alla presenza di un centro medico nel villaggio.

È particolarmente interessante esaminare cosa succederebbe esaminando i dati di prevalenza con un modello di regressione logistica semplice, trascurando la componente spaziale. Il modello assume che vi sia indipendenza tra le varie osservazioni.

```
> mod <- glm( cbind(pos,n) ~ green + factor(phc), binomial(logit),
+   data=village)

> summary(mod)
[...]
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.202685   0.300240  -7.336 2.19e-13 ***
green         0.027218   0.006008   4.531 5.88e-06 ***
factor(phc)1 -0.186699   0.091129  -2.049   0.0405 *

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 195.07  on 64  degrees of freedom
```

```
Residual deviance: 169.08  on 62  degrees of freedom
AIC: 415.43

Number of Fisher Scoring iterations: 4
```

Confrontando questi risultati con quelli riportati in Tab. 10.2 si osserva che il modello logistico sottostima fortemente la dimensione degli errori sui parametri, portando a risultati significativi su entrambi. Questo effetto è legato a due cause: la prima è di ordine puramente computazionale, dato che la stima dei parametri spaziali aggiuntivi introduce una ulteriore fonte di variabilità nel modello Bayesiano; la seconda è di carattere teorico, dato che l'effetto della componente spaziale è quella di ridurre la dimensione "effettiva" del campione, data la dipendenza che viene a essere introdotta tra le varie osservazioni. Dall'esame dei risultati si nota anche che l'effetto legato all'indice vegetativo risulta altamente significativo nel modello logistico non spaziale a differenza di quanto trovato con l'analisi Bayesiana. Questo semplice confronto deve mettere sull'avviso lo sperimentatore che analizzi dati in cui si può supporre una dipendenza spaziale con tecniche che invece assumono indipendenza tra le misurazioni: nel migliore dei casi le conclusioni sulle significatività degli effetti saranno enfatizzate.

Ovviemente una scelta diversa dei prior in fase di analisi Bayesiana porterebbe a risultati differenti. Questo è particolarmente vero tutte le volte in cui il campione ha taglia modesta e la likelihood dei dati non può determinare le conclusioni dello studio. In generale, la scelta di un prior piuttosto diffuso tende a rendere meno potente lo studio, ma protegge lo sperimentatore da conclusioni affrettate su significatività che non sono tali.

Per vedere questo effetto in azione si consideri la seguente scelta dei prior:

```
> beta.in <- coefficients(mod)
> beta.in
 (Intercept)        green factor(phc)1
 -2.20268542   0.02721765  -0.18669867

> beta.var <- vcov(mod)
> beta.var
              (Intercept)         green  factor(phc)1
(Intercept)   0.090144322 -1.749002e-03 -6.666948e-03
green        -0.001749002  3.609092e-05  2.643413e-05
factor(phc)1 -0.006666948  2.643413e-05  8.304475e-03

> PGC <- prior.glm.control(phi.prior="uniform", phi.discrete=seq(10,100000,by=200),
+  sigmasq.prior="sc.inv.chisq", sigmasq=1, df.sigmasq=50,
+  beta.prior="normal", beta=beta.in, beta.var.std=2*beta.var )

> MCc <- mcmc.control(S.scale=0.03, n.iter=40000, thin=100, phi.sc=0.5e+8,
+  burn.in=50000)
```

dove si fa uso delle informazioni ricavate dal modello di regressione logistica per inizializzare i prior per i coefficienti β e per la loro varianza. In particolare si richiede per i coefficienti un prior gaussiano multivariato, con vettore delle medie ottenuto dal modello non spaziale e matrice di covarianza pari al doppio di quanto ottenuto nel modello logistico. Le informazioni relative alla catena MCMC sono state adattate per ottenere dei valori di acceptance rate in linea con quanto detto in precedenza. La simulazione può essere lanciata nel modo consueto:

```
> bkb <- binom.krige.bayes(vil, model=MOD, prior=PGC, mcmc=MCc)
binom.krige.bayes: model with mean defined by covariates provided by the user
burn-in =  50000 is finished; Acc.-rate = 0.68 ; Acc-rate-phi = 0.28
iter. numb. 51000 ; Acc.-rate = 0.66 ; Acc-rate-phi = 0.27
iter. numb. 52000 ; Acc.-rate = 0.73 ; Acc-rate-phi = 0.18
iter. numb. 53000 ; Acc.-rate = 0.73 ; Acc-rate-phi = 0.23
```

```
[...]
```

I risultati delle simulazioni suggeriscono una conclusione alquanto differente da quanto visto sopra. Si osserva che i valori dei parametri sono molto simili a quelli del modello logistico e che i loro intervalli di credibilità al 95% sono assai più stretti di quanto ottenuto con prior diffusi. Anche per i parametri spaziali si ottengono dei range di variabilità di dimensione notevolmente inferiore.

```
> for(i in 1:3) print(quantile(bkb$posterior$beta$sample[i,], c(0.025,0.5,0.975)))
     2.5%       50%     97.5%
-2.828610 -2.019844 -1.258032
      2.5%        50%      97.5%
0.01054562 0.02436659 0.04229308
       2.5%         50%       97.5%
-0.49409150 -0.26610613 -0.03137294

> quantile(bkb$posterior$phi$sample, c(0.025, 0.5, 0.975))
 2.5%   50% 97.5%
 6005 13310 27615

> quantile(bkb$posterior$sigmasq$sample, c(0.025, 0.5, 0.975))
     2.5%       50%     97.5%
0.7930178 1.0756042 1.5125268
```

Capitolo 11

Analisi genomica

In questo capitolo vengono descritte alcune tecniche statistiche e computazionali impiegate nel campo dell'analisi genomica. La panoramica offerta è per il momento limitata a tecniche piuttosto semplici. Per una descrizione più approfondita di queste e altre metodiche si rimanda ad altri testi come [19] e [28].

Le metodiche affrontate nel seguito sono comprese nell'ambito della Biologia Computazionale, quella branca della Scienza che si occupa di studiare gli organismi a differenti livelli di complessità (dal livello molecolare a quello cellulare), con particolare riguardo alle questioni evolutive. I problemi che si affrontano sono quindi piuttosto differenti tra loro (studi anatomici, biologici, biomolecolari); ognuno di essi necessita di un insieme di tecniche specifiche e di competenze più generali.

Con il termine genoma si intende il repertorio di DNA che determina l'identità di un organismo. Con l'avvento dell'era genomica è divenuto possibile disporre di tecniche sperimentali atte a sequenziare il DNA di un grandissimo numero di organismi e in maniera sempre più completa. Queste sequenze sono in massima parte pubblicamente accessibili via web in database internazionali. L'accesso a questa mole sempre crescente di informazioni, se da un lato mette lo sperimentatore in grado di disporre di una conoscenza impensabile solo una decina di anni or sono, dall'altro pone il problema di dover disporre di tecniche e software adeguati per le elaborazioni.

11.1 Bioconductor

Il progetto Bioconductor (http://www.bioconductor.org) mette a disposizione un insieme di librerie e pacchetti software che permettono allo sperimentatore di interagire con le banche dati internazionali e analizzare dati biomedici e genomici. Le librerie del progetto sono basate su R e distribuite in pacchetti. Data la loro utilità è opportuno installare i pacchetti base del progetto. La procedura è piuttosto semplice. Una volta avviato R e disponendo di una connessione Internet si possono usare i due comandi:

```
> source("http://bioconductor.org/biocLite.R")

> biocLite()
Using R version 2.10.1, biocinstall version 2.5.10.
Installing Bioconductor version 2.5 packages:
 [1] "affy"         "affydata"     "affyPLM"      "annaffy"      "annotate"
 [6] "Biobase"      "biomaRt"      "Biostrings"   "DynDoc"       "gcrma"
[11] "genefilter"   "geneplotter"  "hgu95av2.db"  "limma"        "marray"
[16] "multtest"     "vsn"          "xtable"       "affyQCReport"
Please wait...
[...]
```

Il primo comando preleva dal server del progetto la versione aggiornata della routine di installazione, che viene lanciata dal secondo comando. A seconda della versione di R utilizzata verrà scaricata e installata la versione appropriata di Bioconductor. Il comando *biocLite* installa le librerie che fanno parte delle versione base del progetto. Se necessario è possibile installarne altre, passando il nome della libreria alla funzione di installazione. Ad esempio per la libreria *EBImage* la sintassi è:

```
> biocLite("EBImage")
```

11.2 Frequenze dei nucleotidi

Talvolta, data una sequenza di DNA o di RNA, è di interesse valutare le frequenze dei nucleotidi (A, C, G, T in DNA; A, C, G, U in RNA) che la compongono. Lo scopo di tale valutazione può essere o meramente conoscitivo o volto al confronto delle frequenze osservate in diversi tratti dello stesso genoma o tra genomi di diversi organismi. Oltre alla valutazione delle frequenze dei singoli nucleotidi possono essere di interesse la valutazione delle frequenze di dinucleotidi, trinucleotidi o oligonucleotidi di lunghezza definita dallo sperimentatore.

I trinucleotidi (o codoni, se trascritti in mRNA) costituiscono l'unità di lettura del codice genetico. Dato che l'alfabeto degli acidi nucleici è composto da 4 nucleotidi, esistono $4^3 = 64$ codoni differenti. A essi, al termine della fase di traduzione, corrisponde un amminoacido, con tre sole eccezioni; esistono infatti tre codoni (UAG, UGA, UAA) che non codificano per nessun amminoacido e sono perciò detti codoni di stop. Dato che in natura gli amminoacidi che costituiscono le proteine sono solo 20, alcuni codoni diversi codificano per lo stesso amminoacido.

Nel valutare le frequenze di oligonucleotidi in una sequenza di DNA o RNA bisogna prestare attenzione a considerare tutte le possibili finestre di lettura. Ad esempio, data la sequenza 5'-TAAGATC-3', e detto n_i $(i = 1, \ldots, 7)$ il nucleotide di posto i-esimo, i dinucleotidi che la compongono sono tutte le coppie $n_i n_{i+1}$ con $i = 1, \ldots, 6$. Quindi il primo dinucleotide è TA, il secondo AA, il terzo AG e così via. Una ovvia estensione vale per sequenze di oligonucleotidi più lunghe.

Nel seguito vengono presentati alcuni esempi in cui si illustra la procedura di recupero delle informazioni genomiche da banche dati disponibili via web (principalmente GenBank) e si testa un modello probabilistico per la composizione delle triplette di nucleotidi.

Esempio

Si vuole verificare se le frequenze osservate delle triplette di nucleotidi del genoma di *Mycoplasma genitalium* (GenBank ID L43967) sono descritte correttamente da un modello probabilistico che, stimate le frequenze dei singoli nucleotidi sul genoma, ipotizzi che le disposizioni dei nucleotidi siano indipendenti tra loro.

Il genoma può essere importato e analizzato in R mediante le funzioni della libreria *annotate* o *Biostrings*, a seconda che i dati debbano essere scaricati dal database GenBank o siano invece disponibili in un file locale. Nel primo caso si può utilizzare la funzione *getSEQ* della libreria *annotate*:

```
> library(annotate)
> mg.seq <- getSEQ("L43967")
```

L'oggetto *mg.seq* contiene la sequenza desiderata. Per potere utilizzare le funzioni descritte nel seguito è necessario trattare la sequenza con la funzione *DNAString* della libreria *Biostrings*; questo passaggio crea un nuovo oggetto – di classe DNAString – che consente di operare efficientemente su lunghe sequenze di nucleotidi.

```
> library(Biostrings)
> mg.seq <- DNAString(mg.seq)
```

Per la lettura dei dati da file locale si possono usare diverse alternative. Prima di procedere è necessario accertarsi che la sequenza sia stata scaricata in un formato leggibile dal software; solitamente per evitare problemi si usa un formato standard detto FASTA. Se non ci sono problemi di

standardizzazione si può usare la funzione *read.XStringViews*. Se il file contenente i dati si chiama *m_genitalium.txt* si procede nel modo seguente:

```
> library(Biostrings)
> mg.seq2 <- read.XStringViews("m_genitalium.txt", "fasta", "DNAString")
```

La funzione accetta tre argomenti: il nome del file la leggere, il formato dei dati (attualmente è supportato solo il formato "fasta") e la classe di dati ("DNAString" per DNA, "RNAString" per RNA, "AAString" per amminoacidi). Il risultato della chiamata è un oggetto piuttosto complesso che contiene numerose informazioni. Un semplice output si ottiene nel modo seguente:

```
> mg.seq2
  Views on a 580076-letter DNAString subject
subject: TAAGTTATTATTTAGTTAATACTTTTAACAATAT...TGATTATAATATTTCTTTAATACTAAAAAAATAC
views:
    start    end  width
[1]     1 580076 580076 [TAAGTTATTATTTAGTTAATACTTTT...TATTTCTTTAATACTAAAAAAATAC]
```

Per il conteggio delle frequenze dei singoli nucleotidi, così come di sequenze più lunghe, è possibile usare la funzione *oligonucleotideFrequency* che accetta in input la sequenza genomica da elaborare e la dimensione del gruppo di nucleotidi da considerare (1 per le frequenze dei singoli nucleotidi, 2 per quelle dei dinucleotidi e così via).

```
> o1 <- unlist(oligonucleotideFrequency(mg.seq2, 1))  # conteggio singoli nucleotidi
> o2 <- unlist(oligonucleotideFrequency(mg.seq2, 2))  # conteggio coppie nucleotidi
> o3 <- unlist(oligonucleotideFrequency(mg.seq2, 3))  # conteggio triplette
```

La chiamata alla funzione *unlist* è necessaria per trasformare l'output in un vettore, formato più comodo per le successive operazioni di elaborazione. Sulle sequenze create da *DNAString* tale funzione non risulta necessaria. Il risultato delle chiamate sono dei vettori contenenti le frequenze assolute richieste. Ad esempio per le triplette si ha:

```
> o3
  AAA   AAC   AAG   AAT   ACA   ACC   ACG   ACT   AGA   AGC   AGG   AGT   ATA
34109 16098 14040 20289  9988  6998  1962 11456  9388  7415  5166 11735 11813
  ATC   ATG   ATT   CAA   CAC   CAG   CAT   CCA   CCC   CCG   CCT   CGA   CGC
10701  9020 20365 16552  5123  6282  8565  7426  3150   911  5095  1447  1414
  CGG   CGT   CTA   CTC   CTG   CTT   GAA   GAC   GAG   GAT   GCA   GCC   GCG
  910  1874  9556  3533  6113 13563 11317  2505  3956 11030  6876  1719  1448
  GCT   GGA   GGC   GGG   GGT   GTA   GTC   GTG   GTT   TAA   TAC   TAG   TAT
 7334  4823  1858  3311  6463  6410  2512  5587 15157 22558  6679  9426 12015
  TCA   TCC   TCG   TCT   TGA   TGC   TGG   TGT   TTA   TTC   TTG   TTT
12232  4715  1324  8880 13150  6690  7068  9594 22898 10405 15782 32295
```

Il modello teorico ipotizzato permette il calcolo delle frequenze dei codoni in modo diretto. Sia N_i $(i = 1, \ldots, 3)$ la i-esima posizione nella tripletta e sia n_i $(i = 1, \ldots, 3)$ il nucleotide in posizione i-esima. La probabilità del trinucleotide considerato è semplicemente:

$$P(N_1 = n_1, N_2 = n_2, N_3 = n_3) = P(N_1 = n_1)P(N_2 = n_2)P(N_3 = n_3) = P(n_1)P(n_2)P(n_3)$$

Il codice per effettuare il calcolo è elementare. Si calcolano dapprima le frequenze relative dei singoli nucleotidi:

```
> P1 <- o1/sum(o1)        # vettore delle probabilta' dei nucleotidi
> P1
        A         C         G         T
0.3457202 0.1577638 0.1591274 0.3373885
```

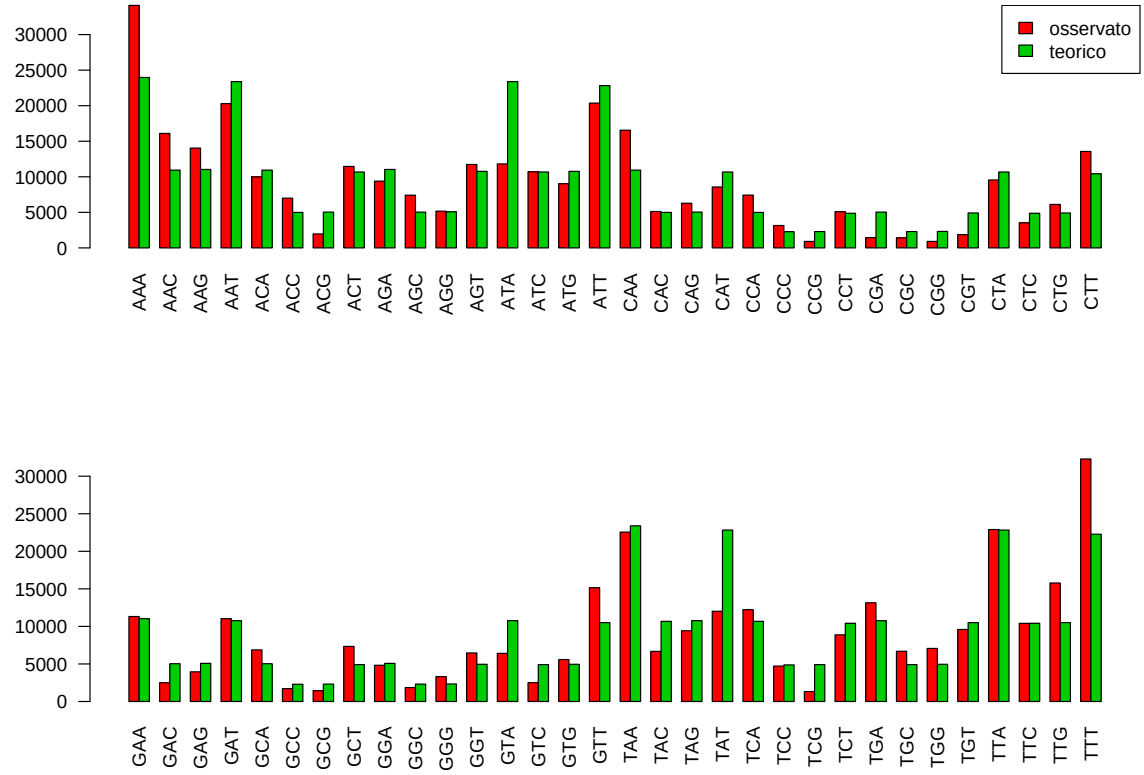

Figura 11.1: Confronto tra la distribuzione osservata di trinucleotidi e quella stimata mediante modello probabilistico che suppone l'indipendenza nella disposizione dei nucleotidi.

Quindi si costruisce il vettore delle frequenze (relative) teoriche $res1$ sfruttando tre cicli annidati (si osservi che la chiamata ordina le frequenze teoriche nello stesso modo in cui sono ordinate quelle osservate):

```
> res1 <- vector()
> for(i in 1:4) for(j in 1:4) for(k in 1:4)
+ res1[16*(i-1)+4*(j-1)+k] <- P1[i]*P1[j]*P1[k]
```

Le frequenze assolute si ottengono normalizzando al totale delle triplette:

```
> e3 <- res1*sum(o3)
```

L'ipotesi nulla che le disposizioni dei nucleotidi all'interno dei trinucleotidi siano indipendenti può essere esaminata mediante test χ^2.

```
> x <- sum((o3 - e3)^2/e3)
> x
[1] 60964.41

> pchisq(x, 63, lower.tail=FALSE)
[1] 0
```

Il valore p del test è talmente basso che il software non lo distingue da 0. Si respinge quindi l'ipotesi di indipendenza nella disposizione dei nucleotidi.

Le serie di frequenze possono essere confrontate graficamente. La Fig. 11.1, ottenuta con il codice sottostante, mostra chiaramente che il modello teorico non descrive correttamente la realtà.

```
> TAB <- rbind(o3, e3)
> par(mfrow=c(2,1))
> barplot(TAB[,1:32], beside=TRUE, las=2, col=c(2,3))
> legend("topright", c("osservato", "teorico"), fill=c(2,3))
> barplot(TAB[,32+(1:32)], beside=TRUE, las=2, col=c(2,3))
```

11.3 Trascrizione e traduzione di sequenze di DNA

In R sono disponibili alcune funzioni per riprodurre in silico i processi cellulari di trascrizione di DNA in RNA, traduzione di RNA in proteine, sintesi di DNA complementare (cDNA) a partire da RNA maturo. Le funzioni sono raccolte nella libreria *Biostrings*.

Per simulare il processo di trascrizione si può utilizzare la funzione *transcribe*. Nel seguente esempio si effettua la trascrizione delle prime 60 basi della sequenza di *M. genitalium* introdotta precedentemente.

```
> mg <- DNAString(as.character(mg.seq2), nchar=60)  # solo 60 basi
> mg
  60-letter "DNAString" instance
seq: TAAGTTATTATTTAGTTAATACTTTTAACAATATTATTAAGGTATTTAAAAAATACTATT

> transcribe(mg)
  60-letter "RNAString" instance
seq: AUUCAAUAAUAAAUCAAUUAUGAAAAUUGUUAUAAUAAUUCCAUAAAUUUUUUAUGAUAA
```

Si noti l'uso della funzione *as.character*, necessaria nella conversione dell'oggetto *mg.seq*2. L'argomento *nchar* serve a selezionare 60 basi della sequenza (la funzione accetta anche un argomento opzionale *start*, con valore di default 1, che permetterebbe di selezionare un certo numero di basi a partire da una qualsiasi posizione di inizio). Nell'uso della funzione si deve tenere presente che essa assume che la sequenza in ingresso sia orientata 5'-3', e produce in output la sequenza di RNA orientata 3'-5'.

Il processo di traduzione può essere simulato mediante l'uso della funzione *translate*. Mediante l'uso del codice genetico standard, a partire da una stringa di DNA o RNA, viene prodotta la sequenza amminoacidica corrispondente. Ad esempio, le prime 60 basi del genoma di *M. genitalium* vengono così tradotte:

```
> translate(mg)
  20-letter "AAString" instance
seq: *VII*LILLTILLRYLKNTI
```

dove con "*" viene identificato un codone di stop. Il codice genetico utilizzato nel processo è definito all'interno dell'oggetto *GENETIC_CODE* e si può visualizzare con la seguente chiamata:

```
> GENETIC_CODE
TTT TTC TTA TTG TCT TCC TCA TCG TAT TAC TAA TAG TGT TGC TGA TGG CTT CTC CTA CTG
"F" "F" "L" "L" "S" "S" "S" "S" "Y" "Y" "*" "*" "C" "C" "*" "W" "L" "L" "L" "L"
CCT CCC CCA CCG CAT CAC CAA CAG CGT CGC CGA CGG ATT ATC ATA ATG ACT ACC ACA ACG
"P" "P" "P" "P" "H" "H" "Q" "Q" "R" "R" "R" "R" "I" "I" "I" "M" "T" "T" "T" "T"
AAT AAC AAA AAG AGT AGC AGA AGG GTT GTC GTA GTG GCT GCC GCA GCG GAT GAC GAA GAG
"N" "N" "K" "K" "S" "S" "R" "R" "V" "V" "V" "V" "A" "A" "A" "A" "D" "D" "E" "E"
GGT GGC GGA GGG
"G" "G" "G" "G"
```

Il codice utilizzato per la traduzione a partire da RNA è definito in *RNA_GENETIC_CODE*.

La sintesi di DNA complementare a partire da RNA maturo può essere effettuata usando la funzione . Nell'esempio seguente si trascrive la stringa di 60 basi in RNA e poi si produce il DNA complementare, con l'effetto di riprodurre la sequenza di DNA di partenza:

```
> cDNA(transcribe(mg))
  60-letter \index{matchPattern@{\sl matchPattern\/}}"DNAString" instance
seq: TAAGTTATTATTTAGTTAATACTTTTAACAATATTATTAAGGTATTTAAAAAATACTATT
```

Per concludere la sezione, meritano un cenno anche le funzioni *complement* e *reverseComplement* dedicate a produrre la sequenza complementare di un filamento di DNA, rispetto all'appaiamento di Watson e Crick. Le due funzioni ritornano la medesima sequenza, con diverso orientamento. La prima ritorna il filamento complementare orientato 3'-5', la seconda lo restituisce orientato 5'-3'. Il tutto è illustrato nelle chiamate seguenti.

```
> mg                    # sequenza di partenza
  60-letter "DNAString" instance
seq: TAAGTTATTATTTAGTTAATACTTTTAACAATATTATTAAGGTATTTAAAAAATACTATT

> complement(mg)
  60-letter "DNAString" instance
seq: ATTCAATAATAAATCAATTATGAAAATTGTTATAATAATTCCATAAATTTTTTATGATAA

> reverseComplement(mg)
  60-letter "DNAString" instance
seq: AATAGTATTTTTTAAATACCTTAATAATATTGTTAAAAGTATTAACTAAATAATAACTTA
```

11.4 Mappe di restrizione

Prima dello sviluppo di efficienti tecniche di PCR e sequenziamento, le mappe di restrizione erano uno degli strumenti più utili per verificare se un frammento di DNA contenesse o meno uno specifico gene o una sua sequenza. Ancora oggi questa metodica è impiegata per applicazioni in cui un sequenziamento completo eccede gli scopi sperimentali, come ad esempio verificare il verso di un inserto di DNA in un vettore.

La tecnica si basa sulla individuazione dei siti di restrizione in una sequenza di DNA, ossia delle posizioni in cui uno o più enzimi di restrizione tagliano la molecola. Questi ultimi sono enzimi in grado di riconoscere specifiche sequenze di DNA a doppia o singola elica, note come siti di restrizione, di legarsi e operare un taglio della catena in posizioni ben precise. Le sequenze riconosciute dai vari enzimi variano, con lunghezze tra 4 e 8 basi, ma molte di esse condividono la caratteristica di essere palindrome, ossia di leggersi con la stessa sequenza di basi azotate sul filamento senso e su quello antisenso. Ad esempio, *EcoR*I (isolato da *Escherichia coli* RY13) riconosce la sequenza GAATTC, mentre *BamH*I (isolato da *Bacillus amyloliquefaciens* H) la sequenza GGATCC.

Sperimentalmente, si digerisce la sequenza di DNA su cui si vuole indagare con uno o più enzimi di restrizione, quindi si controllano le dimensioni dei frammenti che si generano. Questa semplice conoscenza è spesso sufficiente per inferire le caratteristiche di interesse per lo sperimentatore.

Dal punto di vista computazionale può essere interessante valutare la mappa di restrizione in silico di una data sequenza o confrontare tra loro mappe prodotte dallo stesso enzima di restrizione in regioni di DNA codificanti e non codificanti.

Esempio

Usando la sequenza di DNA di *M. genitalium* data sopra, si vuole verificare se i siti di restrizione di *Alu*I sono distribuiti in modo casuale. In tale ipotesi ci si trova di fronte a un processo poissoniano

che può essere descritto dal rate di occorrenza λ di siti di restrizione per bp. La probabilità di k siti in un intervallo di lunghezza l bp sarà:

$$P(N = k) = \frac{e^{-\lambda l}(\lambda l)^k}{k!} \quad , \quad k = 0, 1, 2, \ldots$$

Si può quindi calcolare la probabilità che un frammento di restrizione abbia lunghezza X superiore a x, ossia che, se un frammento inizia in una posizione y, non vi siano siti di restrizione nell'intervallo $(y,\, y + x)$. Essa è semplicemente:

$$P(X > x) = e^{-\lambda x} \quad , \quad x > 0$$

da cui si ha la funzione di ripartizione della variabile casuale X:

$$P(X \leq x) = 1 - e^{-\lambda x}$$

e la sua densità:

$$f(x) = \lambda\, e^{-\lambda x} \quad , \quad x > 0$$

che altro non è che la funzione di densità esponenziale con parametro λ, legato alla lunghezza media dei frammenti pari a $1/\lambda$.

Per ricavare le dimensioni dei frammenti di digestione, noto che *Alu*I riconosce la sequenza AGCT, si può procedere utilizzando la funzione *matchPattern* della libreria *Biostrings*:

```
> library(Biostrings)
> rs <- matchPattern(DNAString("AGCT"), mg.seq2)
> rs
  Views on a 580076-letter DNAString subject
subject: TAAGTTATTATTTAGTTAATACTTTTAACAATAT...TGATTATAATATTTCTTTAATACTAAAAAAATAC
views:
        start    end width
   [1]    560    563     4 [AGCT]
   [2]   1219   1222     4 [AGCT]
   [3]   1376   1379     4 [AGCT]
   [4]   1606   1609     4 [AGCT]
   [5]   2208   2211     4 [AGCT]
   [6]   2219   2222     4 [AGCT]
   [7]   2289   2292     4 [AGCT]
   [8]   2358   2361     4 [AGCT]
   [9]   2393   2396     4 [AGCT]
   ...    ...    ...   ... ...
[2775] 577062 577065     4 [AGCT]
[2776] 577537 577540     4 [AGCT]
[2777] 578201 578204     4 [AGCT]
[2778] 578225 578228     4 [AGCT]
[2779] 578396 578399     4 [AGCT]
[2780] 579341 579344     4 [AGCT]
[2781] 579386 579389     4 [AGCT]
[2782] 579404 579407     4 [AGCT]
[2783] 579416 579419     4 [AGCT]
```

La funzione accetta due argomenti: la sequenza da individuare e quella in cui cercarla. In output si hanno i siti in cui la sequenza bersaglio è stata individuata. In dettaglio, vengono fornite sia la posizione di inizio sia quella di fine delle 2783 sequenze individuate. Ad esempio, la prima occorrenza della sequenza AGCT si trova in posizione 560. Trascurando i frammenti localizzati alle estremità della sequenza di DNA di *M. genitalium*, le lunghezze di frammenti di digestione si possono ottenere per differenza:

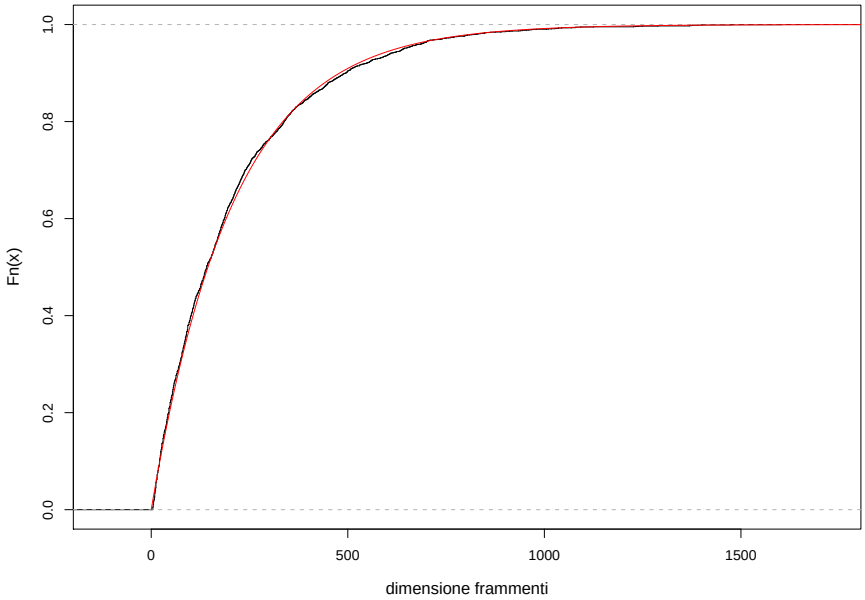

Figura 11.2: Distribuzione cumulativa sperimentale delle dimensioni dei frammenti di digestione del genoma di *M. genitalium* mediante *Alu*I (in nero) e la equivalente distribuzione teorica (in rosso) nell'ipotesi che i siti di restrizione si susseguano in maniera casuale.

```
> frag <- rs@start[-1] - rs@start[-2783]
```

La lunghezza media dei frammenti risulta:

```
> mean(frag)
[1] 208.0719
```

Il paragone tra la mappa sperimentale e quella teorica si può eseguire in vari modi, ad esempio con un test di Kolmogorov-Smirnov:

```
> ks.test(frag, "pexp", rate=1/mean(frag))

        One-sample Kolmogorov-Smirnov test

data:  frag
D = 0.0192, p-value = 0.2593
alternative hypothesis: two-sided
```

Dato che il rate della distribuzione esponenziale è stato stimato sul campione, la procedura per il calcolo del valore P non è corretta e il test risulta troppo conservativo. Comunque, alla luce del risultato, non si evidenza differenza significativa tra la distribuzione sperimentale e quella teorica. Non si respinge quindi l'ipotesi che i siti di restrizione di *Alu*I nel genoma di *M. genitalium* siano modellizzabili da un processo poissoniano.

Il paragone grafico tra le distribuzioni sperimentale e teorica è presentato in Fig. 11.2, ottenuta con le seguenti chiamate:

```
> x <- 0:2000
> y <- pexp(x, 1/mean(frag))
> plot(ecdf(frag), do.points=FALSE, verticals=TRUE, xlab="dimensione frammenti",
```

```
+   main="")
> lines(x, y, col="red")
```

Si nota l'eccellente accordo tra il modello probabilistico testato e la situazione reale.

11.5 Allineamento di sequenze

Dato il grande numero di progetti di sequenziamento sistematico che vengono attualmente condotti in numerosi istituti di ricerca, il problema dell'assegnazione delle funzioni ai geni che vengono individuati diventa sempre più rilevante. Infatti, quando si ottiene una sequenza di DNA o aminoacidi si è spesso interessati a capire se tale sequenza sia o meno già nota, e in caso di risposta negativa a scoprire la sua funzione. In alcuni casi infatti la sequenza stessa è già presente nei database e la sua funzione si trova descritta (si dice che la sequenza è annotata). Se invece la sequenza non compare nei database, si potrebbe tentare di ipotizzarne la funzione cercando sequenze simili già annotate. Questa procedura trova giustificazione nel fatto che, se le sequenze di due proteine sono molto simili, allora lo saranno anche le strutture e le funzioni, mentre non vale il viceversa. Per questo motivo la ricerca di similarità di sequenze è diventato un compito fondamentale nell'ambito della Biologia Computazionale. Infatti queste corrispondenze permettono di stabilire relazioni evolutive tra i diversi organismi e, in seconda battuta, il confronto puó consentire l'identificazione di porzioni funzionali conservate all'interno del genoma. Il grado di corrispondenza tra i nucleotidi di due sequenze viene comunemente espresso come similarità (spesso espressa in percentuale). Ovviamente la similarità tra due sequenze può avere radici profonde, indicando regioni omologhe (ossia con origine filogentica comune) nel genoma di due individui, oppure essere semplicemente frutto del caso.

Un compito importante sarà quindi quello di determinare quanto due sequenze, che siano di DNA, RNA o amminoacidi, siano simili tra loro, affiancando le due sequenze e valutando in quali posizioni esse corrispondano o meno. Questa operazione va sotto il nome di allineamento, e presenta numerosi punti critici sia procedurali che computazionali.

11.5.1 Evoluzione di sequenze genetiche e allineamento

Prima di procedere, è opportuno un breve excursus riguardante l'evoluzione delle sequenze genetiche. Tale fenomeno è dovuto principalmente a:

1. sostituzioni: CCA $\rightarrow$ CGA
2. cancellazioni: ATTG $\rightarrow$ ATG
3. inserzioni: AAG $\rightarrow$ AATG
4. inversioni: AACTG $\rightarrow$ AAATCG
5. ricombinazioni: dovuti a scambi di porzioni di DNA

Gli algoritmi di allineamento considerano i primi tre tipi di mutazioni. In questo ambito inserzioni e cancellazioni vanno cumulativamente sotto il nome di indels. Nell'allineare due sequenze si potranno presentare quindi i seguenti casi: una base della prima sequenza è appaiata con la stessa base nella seconda sequenza (match); una base della prima sequenza è appaiata con una diversa base nella seconda sequenza (mismatch); una base della prima o seconda sequenza è appaiata con una lacuna o gap (indel), rappresentata dal carattere "-".

Data la possibilità di aprire lacune in una sequenza, vi sono parecchi modi in cui allineare due sequenze. Ad esempio l'allineamento:

```
ATTCAGGACT
ATCAGACT--
```

presenta 2 match, 6 mismatch e 2 gap; in questo caso semplice si può fare ovviamente di meglio:

```
ATTCAGGACT
AT-CAG-ACT
```

Questo allineamento presenta 8 match, 0 mismatch e 2 gap. In casi estremi, quando una sequenza è sostanzialmente più corta dell'altra, aprendo un sufficiente numero di gap è quasi sempre possibile ottenere dei match per tutte le basi. È quindi chiaro che servirà un algoritmo di allineamento che sia efficiente nel trovare match tra basi e allo stesso tempo parsimonioso nell'aprire gap. Tale algoritmo dovrà consentire il confronto tra allineamenti alternativi, assegnando uno score a ogni soluzione proposta, e selezionando quella (o quelle) che presentano lo score più elevato.

Fino ad ora si è parlato indistintamente di allineamento tra sequenze, tuttavia è opportuno differenziare tra allineamento globale e allineamento locale. Nel caso di allineamento globale si tenta di allineare il massimo numero di caratteri delle due sequenze, incluse le parti finali. Candidate ideali sono le sequenze di lunghezza simile e quasi simili. Viceversa nel caso di allineamento locale si tenta di allineare solo pezzi di sequenze ad alta similarità, comprese all'interno delle due sequenze. L'allineamento termina quando termina l'isola di forte match. Candidate ideali sono sequenze con lunghezze diverse, che presentano regioni fortemente conservate.

Nel paragrafo successivo viene introdotto il concetto di matrice di score per valutare la bontà di un allineamento.

11.5.2 Score di un allineamento

Per confrontare due possibili allineamenti alternativi è indispensabile poter assegnare a ognuno di essi un punteggio che ne quantifichi numericamente la bontà. Un semplice metodo di score potrebbe essere quello di assegnare $+1$ a ogni posizione in cui si verifica un match tra basi, $-\mu$ ogni volta che si ha un mismatch, e $-\alpha$ per ogni posizione in cui si ha un gap. Dato che spesso, come risultato di processi biochimici, inserzioni e cancellazioni avvengono in blocchi, può essere interessante valutare anche un sistema di score che penalizzi in maniera diversa l'apertura di un gap e la sua estensione; detta k la dimensione (in basi) del gap, si assegna una penalità α per l'apertura del gap e una βk per la sua estensione.

Da quanto precede è chiaro che gli allineamenti "migliori" saranno funzioni del sistema di score scelto dallo sperimentatore. Lavorando con DNA (o RNA) il metodo di score sarà sostanzialmente più semplice che nel caso di allineamenti di proteine, di cui si tratterà nel seguito. Per l'allineamento di DNA si possono presentare infatti solo 16 casi di appaiamenti di basi (4 basi sulla prima sequenza, 4 basi sulla seconda sequenza) di cui valutare lo score. Tale situazione si può riepilogare in una matrice quadrata, detta matrice di score, che avrà sulla diagonale i casi di appaiamento e fuori diagonale tutti i possibili mismatch. Un semplice sistema di score è quello di assegnare +1 a tutti i match, -1 a tutti i mismatch. Questa situazione assume, ad esempio, che l'allineamento di A con G sia altrettanto cattivo di quello fra A e T, dato che le penalità sono le stesse.

In R è possibile costruire una matrice di score usando la funzione *nucleotideSubstitutionMatrix* della libreria Biostrings:

```
> mat <- nucleotideSubstitutionMatrix(match = 1, mismatch = -1, baseOnly = TRUE)
> mat
   A  C  G  T
A  1 -1 -1 -1
C -1  1 -1 -1
G -1 -1  1 -1
T -1 -1 -1  1
```

La funzione accetta come argomenti lo score da attribuire a ogni match e mismatch; l'argomento *baseOnly* specifica che si deve costruire una matrice per una sequenza di basi azotate e non di amminoacidi.

Esempio

Si vuole trovare il migliore allineamento globale delle due sequenze:

```
> s1 <- DNAString("ACTTCACCAGCTCAC")
> s2 <- DNAString("ACTTCAGCTCAC")
```

assumendo come matrice di score quella data nel paragrafo precedente. Per procedere bisogna definire le penalità da assegnare ai gap (sia per l'apertura che per l'estensione). Questa operazione si effettua passando due argomenti alla funzione *pairwiseAlignment*, che si occupa di ricercare l'allineamento migliore:

```
> gA <- pairwiseAlignment(s1, s2, substitutionMatrix = mat,
+  gapOpening = -1, gapExtension = -1)
```

In questo caso si richiede una penalità pari a 1 per ogni gap aperto (*gapOpening*) e una ulteriore penalità pari a 1 per ogni base di estensione del gap (*gapExtension*). Gli altri argomenti della funzione sono le sequenze da allineare e la matrice di score da utilizzare. Il risultato della chiamata è:

```
> gA
Global Pairwise Alignment (1 of 1)
pattern: [1] ACTTCACCAGCTCAC
subject: [1] ACTTCA---GCTCAC
score: 8
```

in output si ha l'allineamento migliore individuato dalla funzione e il suo score. In questo caso si hanno 12 match (score +12), e un gap (score -1) di dimensione 3 (score -3), per un score totale di $12 - 1 - 3 = 8$.

I gap possono essere aperti su entrambe le sequenze, come mostra l'allineamento seguente, in cui si modifica la sequenza $s1$:

```
> s1 <- DNAString("ACTTCACCAGCTAC")
> gA <- pairwiseAlignment(s1, s2, substitutionMatrix = mat,
+  gapOpening = -1, gapExtension = -1)
> gA
Global Pairwise Alignment (1 of 1)
pattern: [1] ACTTCACCAGCT-AC
subject: [1] ACTTCA---GCTCAC
score: 5
```

In questo caso lo score è determinato da 11 match e 2 lacune, la prima (nella sequenza $s1$) determina uno score di -2, la seconda (in $s2$) porta uno score di -4. In totale quindi lo score dell'allineamento è di 5.

Come accennato, i pesi scelti per penalizzare mismatch e gap influenzano la soluzione proposta. Si può infatti esaminare cosa succede attribuendo una maggiore penalità all'apertura di un gap:

```
> gA <- pairwiseAlignment(s1, s2, substitutionMatrix = mat,
+  gapOpening = -3, gapExtension = -1)
> gA
Global Pairwise Alignment (1 of 1)
pattern: [1] ACTTCACCAGCTAC
subject: [1] ACTTCAGCTC--AC
score: 1
```

come si vede, si ottiene una proposta di allineamento piuttosto differente dalla precedente.

Esempio

Si vuole trovare il migliore allineamento locale delle due sequenze:

```
> s1 <- DNAString("ACTTCACCAGCTAC")
> s2 <- DNAString("ACTTCAGCTCAC")
```

assumendo come matrice di score quella data in precedenza. Il problema si risolve usando la funzione *pairwiseAlignment*, specificando l'argomento *type*:

```
> gA <- pairwiseAlignment(s1, s2, substitutionMatrix = mat,
+  gapOpening = -1, gapExtension = -1,  type="local")
> gA
Local Pairwise Alignment (1 of 1)
pattern: [1] ACTTCA
subject: [1] ACTTCA
score: 6
```

Come si vede, l'allineamento isola una zona di perfetto match.

Capitolo 12

Tecniche bootstrap e metodi Monte Carlo

Uno degli scopi principali della statistica è quello di ricavare, tramite l'esame di un campione, alcune proprietà della popolazione da cui esso è stato estratto. In altri termini, si cerca di stimare un parametro di una popolazione, la cui distribuzione è ignota, tramite uno *stimatore*, cioè tramite una funzione dei dati campionari. Una volta scelto lo stimatore da usare non è sempre facile calcolare quanto esso sia accurato. In alcuni semplici contesti è facile ricavare una misura della sua variabilità (si pensi al parametro media aritmetica di una popolazione finita, il cui stimatore è la media campionaria, di cui si può calcolare l'errore standard). In altri casi, quando il parametro da stimare è più complesso, calcolare il suo errore standard può essere molto più complicato e richiedere assunzioni non verificabili o non giustificate. La conseguenza è che gli intervalli di confidenza per lo stimatore che si ottengono hanno copertura differente da quella nominale (molto spesso inferiore al valore nominale).

Se si potesse disporre di diversi campioni estratti dalla stessa popolazione, si potrebbe calcolare il valore dello stimatore su ogni campione e poi calcolarne la variabilità (ad esempio tramite la varianza o l'errore standard), ma questo caso si verifica raramente. Molto spesso si dispone di un solo campione e di conseguenza un solo valore dello stimatore, il che non permette di avere alcuna informazione sulla sua accuratezza.

L'idea alla base del bootstrap è quella di ricavare dalla distribuzione empirica del campione, l'unica informazione di cui si dispone sulla distribuzione della popolazione, numerosi campioni con una procedura di ricampionamento con reinserimento. In questo modo si possono calcolare diverse stime del parametro che interessa, con le quali si è in grado di ottenere misure di variabilità dello stimatore quali errore standard e intervalli di confidenza.

Dalla loro proposta originaria (introdotti in un lavoro di Efron del 1979 [24]) ad oggi, i metodi di ricampionamento hanno stimolano un enorme sviluppo, sia metodologico sia nelle applicazioni. Per una panoramica esaustiva sulla teoria alla base del loro funzionamento e su molte applicazioni si rimanda a [18, 25].

In generale la distinzione che occorre operare è fra tecniche bootstrap parametriche e non parametriche. La differenza è che, mentre nel primo caso si è in grado di ipotizzare la forma generale della distribuzione generatrice e stimarne i parametri sul campione, nel secondo non si ha nessuna informazione nemmeno sulla forma di tale distribuzione. In questa sezione vengono trattate alcune applicazioni della tecnica di bootstrap non parametrico facendo uso della libreria *boot*, parte dell'installazione standard di R.

Prima di iniziare una analisi di tipo bootstrap è necessario assicurarsi che i dati soddisfino due condizioni essenziali:

- Il campione rappresenta bene la popolazione, nel senso che la presenza di eventuali outliers deve essere valutata con cura. Eventuali dati sospetti vanno ricontrollati e nel caso corretti, dato che la loro presenza può alterare pesantemente le stime bootstrap.

- I dati del campione sono fra loro indipendenti. La presenza di correlazioni spazio-temporali rende assolutamente inaffidabile la tecnica bootstrap standard.

Per una discussione più approfondita delle condizioni di validità delle metodologie di ricampionamento si veda [18].

12.1 Applicazione: media campione

Il classico esempio introduttivo del funzionamento dei metodi bootstrap è il calcolo dell'errore standard della media campione. In questo caso dalla teoria asintotica (valida per campioni di taglia sufficientemente grande) si sa che detta s la stima campionaria della deviazione standard della popolazione, valutata su un campione di taglia n, l'errore da attribuire alla stima campionaria m della media della popolazione è $s/\sqrt{n}$. La validità di tale risultato si fonda sul teorema limite centrale. Nel caso in cui il campione sia di taglia troppo piccola perché tale teorema giustifichi le assunzioni, si pone il problema di come valutare l'errore da attribuire alla media campionaria m. La tecnica bootstrap permette di rispondere a questa domanda.

Si supponga di avere voler valutare l'efficienza di una particolare compagnia telefonica nell'intervenire a seguito di una segnalazione di un guasto. Si misurano in 25 casi i tempi trascorsi (in ore) dal momento della segnalazione del problema fino alla sua soluzione:

```
tempi <- c(9,10,16,17,19,21,21,24,31,32,32,32,34,35,39,41,41,42,44,
+  46,50,50,52,54,55,56,56,72,101,111)
```

La media campione e il suo errore standard si ottengono con le chiamate:

```
> mean(tempi)
[1] 41.43333
> sd(tempi)/sqrt(25)
[1] 4.665259
```

Data la dimensione non particolarmente elevata del campione possono esserci dubbi sulla validità dell'ultima stima. Per controllare la sua accuratezza si può ricorrere a una procedura bootstrap schematizzabile nel modo seguente:

1. A partire dai dati campionari si genera un nuovo campione di taglia $n = 25$ tramite procedura di campionamento con riposizionamento (quindi lo stesso dato può entrare più volte nel campione così generato).
2. Si calcola la media di questo nuovo campione e la si inserisce in un vettore.
3. Si ripete il procedimento un numero sufficientemente grande di volte.
4. Si valuta la media del vettore delle medie campione (stima della media della popolazione) e la sua deviazione standard (stima dell'errore standard della media).

In R tale procedura può essere implementata mediante l'uso di un semplice ciclo, come il seguente:

```
> m <- NULL;
> for(i in 1:200) {x<- sample(tempi, replace=TRUE); m <- c(m, mean(x))}
```

Il vettore m contiene, alla fine del ciclo, 200 stime della media campionaria. La media e la deviazione standard di m sono:

```
> mean(m)
[1] 41.546
> sd(m)
[1] 4.017688
```

Si nota che l'errore standard della media campione risulta più piccolo di quanto ottenuto usando l'approssimazione asintotica.

I valori ottenuti tramite simulazione risentono dell'incertezza insita nel processo aleatorio: ripetendo il procedimento di ricampionamento si arriverà a delle stime leggermente diverse. In effetti, una questione lasciata volutamente in sospeso è quanto deve essere grande il numero R di ripetizioni affinché le stime ottenute siano affidabili. In realtà non esiste una risposta univoca a questa domanda. Un numero di ripetizioni dell'ordine di 100-300 è sicuramente sufficiente per stimare una media e il suo errore standard, mentre per valutazioni più ambiziose, come identificarne l'intervallo di confidenza al 95% o addirittura dare una stima della sua funzione di distribuzione è necessario avere $R \geq 1000$.

12.2 Intervallo di confidenza di un parametro

Per sviluppare la teoria degli intervalli di confidenza all'interno delle tecniche bootstrap è necessario introdurre una breve trattazione matematica. Sia F la distribuzione generatrice (ignota) da cui si estrae un campione di taglia n e $\hat{F}$ la distribuzione empirica calcolata a partire dai dati campionari. Si è interessati a valutare un parametro θ sulla popolazione tramite una funzione statistica t tale che $\theta = t(F)$. In pratica t è un algoritmo da applicare alla popolazione per ottenere il valore del parametro di interesse. Si consideri anche T, la variabile casuale di cui t è il valore campionario. La distorsione (bias) e la varianza di T sono date da:

$$\begin{aligned} b(F) &= E[T|F] - \theta = E[T|F] - t(F) \\ v(F) &= Var(T|F). \end{aligned}$$

Per campioni sufficientemente grandi ci si può attendere che:

$$Z = \frac{T - \theta - b(F)}{v(F)^{1/2}} \sim N(0,1)$$

e che quindi un intervallo di confidenza bilaterale per θ di livello $1-\alpha$ abbia estremi:

$$t - b(F) - z_{1-\alpha/2}\, v(F)^{1/2} \quad , \quad t - b(F) - z_{\alpha/2}\, v(F)^{1/2} \tag{12.1}$$

dove z_α è il quantile α-esimo della distribuzione normale standard. Tuttavia l'approssimazione utilizzata dipende da F, n e T e non vi è garanzia che essa sia effettivamente accurata.

In ogni caso l'Eq. (12.1) dipende dalla distribuzione incognita F. L'idea di base del bootstrap entra qui in gioco sotto forma del principio di sostituzione (plug-in principle): a F si sostituisce la sua stima $\hat{F}$ e, analogamente, anche il bias e la varianza di T vengono rimpiazzate dalle loro stime bootstrap $b(\hat{F})$ e $v(\hat{F})$. Per far ciò si generano R campioni bootstrap e si valutano le corrispondenti stime della variabile casuale $T_1^*, \ldots, T_r^*$. La notazione di indicare con il simbolo * le stime bootstrap di una statistica è largamente diffusa in letteratura da rappresentare uno standard universale. Quindi si suppone che sia:

$$b(F) = b(\hat{F}) = \frac{1}{R}\sum_{r=1}^{R} T_r^* - t = \bar{T}^* - t$$

$$v(F) = v(\hat{F}) = \frac{1}{R-1}\sum_{r=1}^{R} (T_r^* - \bar{T}^*)^2.$$

Queste equazioni risentono di due differenti errori. Il primo è di tipo statistico, dovuto alla sostituzione di F con $\hat{F}$, il secondo di tipo stocastico, dovuto alla aleatoreità delle simulazioni. Scegliento R sufficientemente grande si può ridurre l'impatto di questa fonte d'errore.

Se la distribuzione di $T - \theta$ è troppo lontana dalla normalità per giustificare l'uso di intervalli di confidenza gaussiani, si può costruire un più accurato intervallo di confidenza sfruttando l'idea che

$T^* - \theta$ e $T - \theta$ hanno approssimativamente la stessa distribuzione e che quindi i quantili della seconda possono essere stimati da quelli della prima. Si ha in questo caso il cosidetto *basic bootstrap confidence interval* di livello $1 - \alpha$, che ha come estremi:

$$t - \left(T^*_{((R+1)(1-\alpha/2))} - t\right) \quad , \quad t - \left(T^*_{((R+1)\alpha/2)} - t\right) \tag{12.2}$$

dove $T^*_{(1)}, \ldots, T^*_{(R)}$ sono i valori ordinati di T^*_r.

Una ulteriore stima, a prima vista attraente, dell'intervallo di confidenza di livello $1-\alpha$ è l'intervallo di confidenza ai percentili che ha come estremi:

$$T^*_{((R+1)\alpha/2)} \quad , \quad T^*_{((R+1)(1-\alpha/2))} \tag{12.3}$$

ossia proprio i percentili simulati. Tuttavia tale intervallo risente pesantemente della distorsione dovuta al fatto che il campionamento viene fatto non da F ma dalla sua stima $\hat{F}$ e la sua copertura può essere molto lontana da quella nominale.

Si considerano infine due modifiche dei metodi precedenti, che hanno un comportamento sostanzialmente migliore riguardo la concordanza tra copertura nominale e copertura effettiva. Il primo di questi due metodi fa uso dell'approssimazione normale, ma sostituisce all'approssimazione $N(0,1)$ per $Z = (T - \theta)/v(F)^{1/2}$ la sua approssimazione bootstrap. Per ogni campione generato si calcola cioè la quantità $z^*_r = (T^*_r - t)/v^{*1/2}$. I valori così ottenuti vengono ordinati e usati per stimare i quantili di Z. Il problema è avere una stima di quanto valga, per ogni campione generato, la quantità v^* (ossia la varianza di T^*_r). In mancanza di stime analitiche dalla teoria asintotica è possibile reiterare la tecnica bootstrap per ottenere, per ogni campione, una stima della variabilità della statistica T^*_r. Il peso computazionale in questo caso può però essere troppo elevato e rendere il metodo inutilizzabile nella pratica. Alla fine della procedura si possono valutare i limiti del cosidetto intervallo di confidenza studentizzato di livello $1 - \alpha$:

$$t - v(\hat{F})^{1/2}\, z^*_{((R+1)(1-\alpha/2))} \quad , \quad t - v(\hat{F})^{1/2}\, z^*_{((R+1)\alpha/2)} \tag{12.4}$$

Si noti che in questo caso, a differenza di quanto si ha in Eq. (12.1), non si corregge esplicitamente per il bias, dato che questo effetto viene automaticamente considerato nel calcolo dei quantili della distribuzione bootstrap.

Il secondo metodo porta al cosidetto *bias-corrected accelerated* (BC_a) *confidence interval* di livello $1 - \alpha$:

$$T^*_{((R+1)\alpha'/2)} \quad , \quad T^*_{((R+1)(1-\alpha''/2))} \tag{12.5}$$

dove i valori α' e α'' vengono scelti in modo tale da ottimizzare le proprietà dell'intervallo correggendo l'effetto del bias (si veda [18] per i dettagli matematici). Date le buone proprietà, sia di copertura che di semplicità di calcolo, questo tipo di intervallo bootstrap è particolarmente utilizzato in letteratura.

In R le stime di questi cinque intervalli di confidenza possono essere ottenute mediante le funzioni *boot* e *boot.ci*, della libreria standard *boot*, come nell'esempio seguente.

Esempio

Riprendendo l'esempio trattato in Sec. 12.1, si vuole trovare un intervallo di confidenza bilaterale di livello 95% per la media dei tempi di intervento del servizio tecnico della compagnia telefonica.

Prima di procedere al ricampionamento mediante la funzione *boot* si deve definire una funzione che calcoli le statistiche da sottoporre a indagine bootstrap. Tale funzione accetta sempre un minimo di due argomenti, ossia il vettore o il data frame che contiene i dati originali e un vettore che contiene gli indici (o i pesi, per tecniche bootstrap pesate) che servono per estrarre gli elementi dal campione originario. Nel nostro caso si vuole far uso di una tecnica standard (con pesi uguali per ogni dato) e le statistiche da estrarre altro non sono che la media e la sua varianza calcolati sui diversi campioni bootstrap. Come detto nella sezione precedente, il valore della varianza della media è necessario per

calcolare anche l'intervallo di confidenza studentizzato; in questo caso la teoria asintotica ci permette di stimare direttamente questa quantità senza bisogno di reiterare la tecnica bootstrap su ogni campione. La funzione può quindi essere scritta come:

```
> boot.f <- function(d, i) { c(mean(d[i]), var(d[i])/length(d[i])) }
```

L'argomento d è usato per passare il data frame contenente le misure originarie, mentre il vettore i per passare il vettore degli indici che volta volta vengono usati per selezionare un campione diverso a partire dagli elementi del vettore. La funzione ritorna un vettore contenete la media del campione e la sua varianza. A questo punto si può usare la funzione *boot* nel modo seguente:

```
> library(boot)
> bt <- boot(tempi, boot.f, R=999)
```

Gli argomenti sono nell'ordine: il vettore contenente i dati di misura reali; la funzione da usare per il ricampionamento; il numero di ripetizioni da effettuare. Il valore di *bt* può quindi essere passato alla funzione *boot.ci* per il calcolo degli intervalli di confidenza:

```
> boot.ci(bt)
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = bt)

Intervals :
Level      Normal              Basic             Studentized
95%   (33.15, 49.85 )   (32.40, 49.27 )   (33.42, 52.98 )

Level     Percentile            BCa
95%   (33.60, 50.47 )   (34.50, 52.25 )
Calculations and Intervals on Original Scale
```

In output si ottengono le cinque stime dell'intervallo di confidenza bootstrap descritte precedentemente. Si noti che le stime studentizzata e BCa sono molto simili tra loro, ma abbastanza diverse dalle altre tre.

12.3 Applicazione: regressione resistente

Come detto in Sec. 3.7.2, una delle applicazioni delle tecniche di ricampionamento bootstrap si ha nel caso di regressione resistente. Si consideri l'esempio di Sec. 3.7.2. Per determinare gli errori sui due parametri di regressione si può usare una tecnica standard di ricampionamento bootstrap non parametrico. Il primo passo è caricare la libreria *boot* e unire i vettori numerici usati nel corso dell'esempio (contenenti temperatura e produzione dell'impianto) in un unico da data frame:

```
> library(boot)
> prd <- data.frame(temp, prod)
```

Si deve quindi definire la funzione che calcola le statistiche da sottoporre a indagine bootstrap, cioè il vettore dei coefficienti fittati sui diversi campioni dalla funzione di regressione resistente. La funzione può quindi essere scritta come:

```
> bts <- function(d, i) {
+      mod.lts <- ltsreg(prod ~ temp, data=d[i,]);
+      mod.lts$coeff }
```

In questo caso l'argomento d è usato per passare il data frame contenente le misure originarie, mentre il vettore i per passare il vettore degli indici che volta volta vengono usati per selezionare un campione diverso di righe del data frame. L'ultima riga della funzione fa in modo che essa ritorni il valore dei due coefficienti di regressione fittati su ognuno dei campioni generati. A questo punto si può usare la funzione *boot* nel modo seguente:

```
> boot(prd, bts, R=999)
```

L'output della funzione è:

```
ORDINARY NONPARAMETRIC BOOTSTRAP
[...]

Bootstrap Statistics :
    original      bias    std. error
t1* 2.132231 -1.23820689   3.8537880
t2* 1.309091  0.05006522   0.1413159
```

Dalla tabella si leggono i valori dei parametri di regressione valutati sul campione originario; la correzione per bias come risulta dalle 999 simulazioni e l'errore standard sul parametro di regressione. Si nota che la correzione per bias è molto grande per il parametro $t1$, e meno accentuata per $t2$. Confrontando i valori dei parametri corretti per il bias con quanto si otteneva nel caso di regressione robusta (Sec. 3.7.1) si vede che essi sono praticamente coincidenti.

12.4 Gibbs sampling

Il campionatore di Gibbs, o Gibbs sampling, è un algoritmo che consente di generare un campione dalla distribuzione di probabilità congiunta di due o più variabili casuali. Lo scopo della tecnica può essere quella di approssimare la distribuzione di probabilità congiunta o calcolare un suo integrale (valore di aspettazione, densità marginali). Il Gibbs sampling è la più semplice e di più immediata implementazione di una catena di Markov Monte Carlo (MCMC). La tecnica è applicabile a patto che si conosca la distribuzione di probabilità condizionata di ognuna delle variabili casuali considerate.

Per comprenderne il funzionamento si supponga di avere un problema dipendente da k parametri $\theta_1, \ldots, \theta_k$ su cui si vuole fare una inferenza. Siano D i dati. Si supponga che la distribuzione congiunta dei parametri $f(\theta_1, \ldots, \theta_k|D)$ non sia nota o sia troppo complessa per essere utilizzata direttamente. Se è di interesse l'inferenza su uno dei parametri o se si vuole calcolarne la densità marginale:

$$f(\theta_i|D) = \int f(\theta_j)_{j\neq i}|D)d(\theta_j)_{j\neq i}$$

può essere troppo complesso o impossibile eseguire esplicitamente l'integrazione. Può essere invece che siano disponibili le k densità condizionate:

$$f(\theta_i|(\theta_j)_{j\neq i}, D) \quad i, j = 1, \ldots, k$$

e che sia semplice campionare valori da esse. In questo caso si può dimostrare che il seguente algoritmo iterativo permette di campionare valori dalla distribuzione congiunta (e di conseguenza dalle distibuzioni marginali), senza doverla specificare esplicitamente.

Per fissare le idee assumiamo che vi siano solo due parametri θ_1, θ_2. Si sceglie, all'interno del suo dominio, un valore di partenza per $\theta_1 = \theta_1(0)$. Quindi si campiona un valore di partenza di θ_2 a partire dalla sua distribuzione marginale:

$$\theta_2(0) \sim f(\theta_2|\theta_1(0), D)$$

. A questo punto, all aumentare del passo t, si ripete il seguente schema ciclico:

$$\theta_1(t) \sim f(\theta_1|\theta_2(t-1), D)$$

$$\theta_2(t) \sim f(\theta_2|\theta_1(t), D)$$

Lo schema implementato è una catena di Markov dato che il passo t dipende solo dai valori dei parametri al passo $t-1$ ed è indipendente dai valori precedenti. Tale catena tende, sotto ipotesi deboli, a una distribuzione stazionaria, che è proprio la distribuzione $P(\theta_1, \theta_2|D)$ [10]. Per avere un campione di vettori di parametri di lunghezza n, su cui calcolare le quantità di interesse (valore atteso, varianza, percentili, ecc.), si procede quindi eseguendo un numero $R+n$ di cicli dell'algoritmo. I primi R passi sono necessari per portare l'algoritmo in convergenza; si scartano quindi i valori campionati in tali passi e si considerano solo i valori ottenuti negli n cicli successivi.

La procedura può essere facilmente estesa a un numero qualsiasi di parametri. Data la sua semplicitaà, la tecnica si presta a rapide implementazioni ed è quindi largamente impiegata in pratica.

Esempio

Si consideri il seguente esempio tratto da [10]. Si abbiano due variabili casuali X e Y la cui densità congiunta sia:

$$f(x,y) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \left(\begin{array}{c} n \\ x \end{array} \right) y^{x+\alpha-1}(1-y)^{n-x+\beta-1} \quad , \quad x = 0, \ldots, n \quad 0 \leq y \leq 1$$

e si voglia approssimare la densità marginale $f(x)$. Trascurando le dipendenze generali da n, α, β, per le densità condizionate si ha:

$$f(x|y) \sim Binomial(x, n, y)$$

$$f(y|x) \sim Beta(y, x+\alpha, n-x+\beta)$$

Si noti per inciso che in questo caso il problema è risolvibile analiticamente. Integrando la distribuzione congiunta si ha:

$$f(x) = \left(\begin{array}{c} n \\ x \end{array} \right) \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(x+\alpha)\Gamma(n-x+\beta)}{\Gamma(\alpha+\beta+n)} \quad , \quad x = 0, \ldots, n$$

che altro non è che l'espressione della distribuzione beta-binomiale.

Assumendo ad esempio $n = 12$, $alpha = 4$, $beta = 3$ si possono confrontare l'approssimazione della distribuzione $f(x)$ ottenuta tramite campionamento MCMC. Per far ciò si può definire in R la funzione di distribuzione teorica da approssimare:

```
betabinom <- function(x, a, b, n) {
 choose(n,x) * gamma(a+b)/(gamma(a)*gamma(b)) *
 gamma(x+a)*gamma(n-x+b)/gamma(a+b+n) }
```

e si utilizza tale funzione per generare le frequenze teoriche, nell'intervallo $x = 0, \ldots, 12$:

```
> f.teorica <- betabinom(0:12, 4, 3, 12)
```

L'algoritmo di campionamento di Gibbs è di facile implementazione, ed è definito nella funzione seguente:

```
gibbs <- function(rep, a, b, n, R=500) {
y <- runif(1, 0, 1);                # valore di partenza casuale tra 0 e 1
res <- matrix(ncol=2, nrow=rep);    # matrice dei valori simulati
for(i in 1:rep) {
    x <- rbinom(1, n, y);
    y <- rbeta(1, x+a, n-x+b);
    res[i,] <- c(x,y)
  }
res[-(1:R),]
}
```

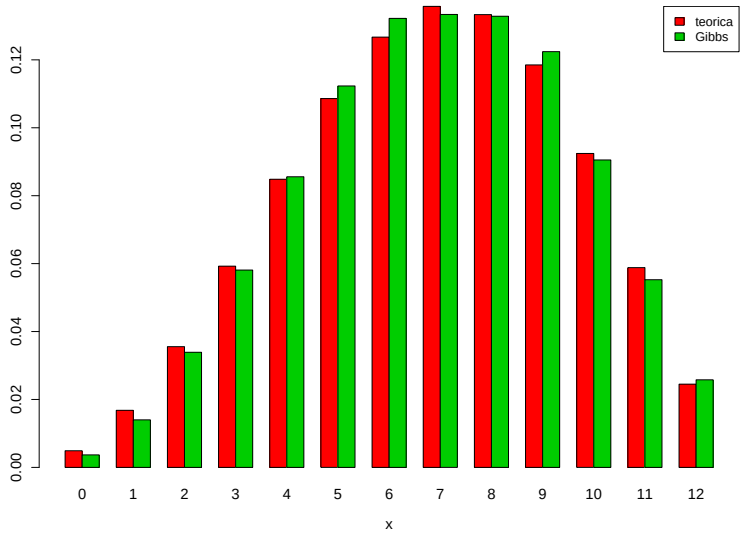

Figura 12.1: Confronto tra la densità di probabilità marginale della variabile X dell'esempio e della sua approssimazione mediante Gibbs sampling. Si osserva l'accuratezza con cui il metodo riproduce il risultato atteso.

Il valore di R permette di specificare quanti passi iniziali si reputano necessari per portare l'algoritmo in convergenza (se non si specifica nessun valore viene assunto di default $R = 500$). La funzione viene chiamata nel modo seguente:

```
> f.gibbs <- gibbs(10000, 4, 3, 12)
```

L'oggetto restituito è una matrice di 9500 righe e due colonne. Nella prima di esse vi sono i valori simulati della variabile X, nella seconda quelli della variabile Y. La distribuzione marginale di X è facilmente ottenibile:

```
> f.g <- table(f.gibbs[,1])/9500
```

Il paragone grafico tra i due risultati (Fig. 12.1) si ottiene con le chiamate:

```
> barplot(rbind(f.teorica, f.g), beside=TRUE, col=c(2,3), xlab="x")
> legend("topright", c("teorica", "Gibbs"), fill=c(2,3))
```

Si evidenzia l'eccellente corrispondenza tra la distribuzione teorica e la sua approssimazione mediante Gibbs sampling.

12.4.1 Gibbs sampling e Statistica bayesiana

Nell'ambito della Statistica bayesiana il campionatore di Gibbs è frequentemente impiegato per il calcolo della distribuzione a posteriori di un parametro.

Si supponga di condurre uno studio per dare una stima intervallare di un qualche parametro di interesse θ. Nel contesto bayesiano si ipotizza che tale parametro abbia nella popolazione campionata una qualche distribuzione *a priori* $f(\theta)$, che riassume l'informazione in possesso dello sperimentatore prima di condurre lo studio. Le informazioni desunte da dati storici riportati in letteratura possono essere d'aiuto in questa fase. La corretta specificazione di una distribuzione a priori sensata e rispondente alla realtà è una operazione che si rivela critica, soprattutto se i campioni selezionati per lo studio sono di piccola taglia. Talvolta sono assunte ipotesi molto lasche sulla distribuzione a priori;

ad esempio una distribuzione uniforme all'interno del range di valori che può assumere il parametro in esame riflette la completa ignoranza dello sperimentatore in materia. In tal caso si parla di distribuzione a priori non informativa.

Dopo aver esaminato il campione D lo sperimentatore è in grado di dare una valutazione della distribuzione *a posteriori* $f(\theta|D)$ del parametro di interesse, mediante il calcolo della funzione di likelihood $f(D|\theta)$ e del teorema di Bayes:

$$f(\theta|D) = \frac{f(D|\theta)\ f(\theta)}{\int_\Theta f(D|\theta)\ f(\theta)\ d\theta}$$

dove Θ è il dominio di θ. A seconda della forma funzionale scelta per modellizzare la distribuzione a priori e dalla funzione di likelihood, l'integrale può non essere valutabile in forma chiusa.

Se la funzione di likelihood dipende da dei parametri latenti, fuori dal controllo dello sperimentatore, è necessario far uso di una tecnica Monte Carlo per valutare la forma della distribuzione a posteriori del parametro in esame.

Esempio

Si vuole stimare la prevalenza di una malattia avendo a disposizione un campione di n soggetti e un test diagnostico rapido di sensibilita s e specificità s' note. Se il test diagnostico non è perfetto sono possibili risultati falsi positivi o falsi negativi.

Inizialmente lo sperimentatore non ha nessuna informazione sulla prevalenza della malattia nella zona dove il campione è stato selezionato. A questa situazione si adatta una distribuzione a priori non informativa; nel caso di una prevalenza è possibile scegliere una distribuzione uniforme in [0, 1], rappresentata convenientemente dalla distribuzione beta con parametri $\alpha = 1$ e $\beta = 1$:

$$f(p) = \frac{1}{B(\alpha, \beta)} p^{\alpha-1}(1-p)^{\beta-1}$$

con:

$$B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$$

Sottoponendo al test gli n individui, si trova che tutti i test danno esito negativo. Alla luce di tale risultato si vuole dare una stima dell'intervallo di confidenza al 95% della prevalenza della malattia nella popolazione, informazione che può essere ottenuta dai quantili della distribuzione a posteriori di p. Il campione in esame presenta quindi zero veri positivi, ma un numero ignoto y di falsi negativi. La funzione di likelihood necessita della specificazione del numero y, ma ciò è al di fuori delle possibilità dello sperimentatore.

Il problema si può risolvere mediante una procedura MCMC. Infatti, il numero ignoto di falsi negativi y è il valore campionario di una statistica Y la cui distribuzione, condizionata al valore di p, è binomiale:

$$f(y|p) = Binomial(y, n, \frac{p\ (1-s)}{p\ (1-s) + (1-p)\ s'})$$

dove si è fatto uso del teorema di Bayes per calcolare la probabilità di un risultato falso negativo, date la prevalenza p della malattia, la sensibilità s e la specificità s' del test diagnostico.

La distribuzione $f(p|y)$ si può ottenere come:

$$f(p|y) = \frac{f(y,p)}{f(y)} = \frac{f(y|p)\ f(p)}{f(y)}$$

Svolgendo l'integrale necessario per il calcolo di $f(y)$ si ottiene:

$$f(p|y) = Beta(p, \alpha + y, n - y + \beta)$$

Come si può notare la scelta di una distribuzione beta per modellizzare la probabilità a priori si riflette in una identica forma funzionale nella probabilità condizionata. Si dice che, per osservazioni di tipo binomiale, la distribuzione beta è *coniugata*.

Il problema può essere trattato computazionalmente in maniera piuttosto semplice ricorrendo a un campionatore di Gibbs. Siano $y(t)$ $(0 \leq y_i \leq n)$ il numero ignoto di falsi negativi e $p(t)$ la prevalenza stimata al passo t-esimo del campionatore di Gibbs. Inizialmente si ha:

$$p(0) \sim Beta(p, \alpha, \beta)$$

$$y(0) \sim Binomial(y, n, \frac{p(0)\ (1-s)}{p(0)\ (1-s) + (1-p(0))\ s'})$$

La catena è quindi descritta dallo schema ciclico:

$$p(t) \sim Beta(p, \alpha + y(t-1), n - y(t-1) + \beta)$$

$$y(t) \sim Binomial(y, n, \frac{p(t)\ (1-s)}{p(t)\ (1-s) + (1-p(t))\ s'})$$

L'algoritmo di campionamento di Gibbs è definito nella funzione seguente:

```
gibbs <- function(rep, n, alpha, beta, s, s1, R=500)
{
p <- rbeta(1, alpha, beta);
y <- rbinom(1, n, p*(1-s)/(p*(1-s)+(1-p)*s1));
pv <- vector();
yv <- vector();

for(i in 1:rep) {
  p <- rbeta(1, y + alpha, n - y + beta);
  y <- rbinom(1, n, p*(1-s)/(p*(1-s)+(1-p)*s1));
  pv[i] <- p;
  yv[i] <- y;
  }
 pv[-(1:R)]
}
```

Come in precedenza, il valore di R permette di specificare quanti passi iniziali si reputano necessari per portare l'algoritmo in convergenza. Nella funzione con s si indica il valore della sensibilità del test diagnostico e con $s1$ la sua specificità. Se si sono testati $n = 30$ soggetti, con un test di sensibilità 0.9 e specificità 0.95, ottenendo tutti risultati negativi, la stima Monte Carlo della distribuzione a posteriori si ottiene con la chiamata:

```
> f.gibbs <- gibbs(10000, 30, 1, 1, 0.9, 0.95)
```

dove si è scelto di eseguire 10000 passi della catena di Markov. Gli estremi dell'intervallo di confidenza bilaterale al 95% per la prevalenza a posteriori della malattia si possono valutare mediante la funzione *quantile*:

```
> quantile(f.gibbs, c(0.025, 0.975))
        2.5%        97.5%
0.0008954415 0.1244885564
```

quindi tra lo 0.09% e il 12% circa.

Si noti, per inciso, che il metodo descritto si adatta anche al caso in cui si abbia una stima informativa della prevalenza a priori della malattia. In questi casi si utilizzerà una distribuzione beta con parametri opportuni, senza alterare la trattazione teorica presentata sopra.

Appendice A

Una breve introduzione ai comandi di R

A.1 Perché usare R

- È free software, scaricabile da *www.r-project.org*
- È distribuito per molte piattaforme: Windows, Macintosh, Linux e altre versioni di UNIX
- È robusto, completo e costantemente sviluppato
- È versatile perché consente di programmare nuove procedure

A.2 ... e perché non usarlo

- Interfaccia grafica carente
- Difficoltà nell'apprendere la sintassi

A.3 Le basi di R: l'help

Per ottenere aiuto si usa la funzione *help*:

```
> help(cbind)
```

fornisce informazioni riguardo alla funzione *cbind*.

L'help è disponibile anche in formato HTML:

```
> help.start()
```

A.4 Le basi di R: l'assegnamento

L'operatore di assegnamento è '$<-$' e non '=':

```
> x <- 5
```

assegna alla variabile x il valore 5 e non fornisce nessun output.

Il valore di x è visualizzabile dietro richiesta:

```
>  x
[1] 5
```

A.5 Le basi di R: operatori e funzioni

Gli operatori aritmetici sono $+ \; - \; * \; /$

```
> x <- y + 6
```

assegna a x il valore di y aumentato di 6.

Le funzioni accettano argomenti in parentesi tonda:

```
> x <- sqrt(y)
> x <- rnorm(100, m=2, sd=3)   # 100 numeri distribuiti N(2, 3^2)
```

A.6 Le basi di R: i vettori

Per definire un vettore si usa l'operatore di concatenamento $c(\ldots)$:

```
> x <- c(1, 3, 5, 8, 2)
```

crea un vettore x di lunghezza 5 assegnando le componenti.

La chiamata:

```
> x + 3
[1]  4  6  8 11  5
```

somma 3 a ogni componente del vettore x.

Per accedere a una componente di un vettore si usano le parentesi quadre:

```
> x <- c(1, 3, 5, 8, 2)
> x[3]
[1] 5
```

A.7 Le basi di R: le matrici

Le matrici si costruiscono con la funzione *matrix*:

```
> A <- matrix( c(1, 3, 5, 6, 8, 11), nr=3)
> A
     [,1] [,2]
[1,]    1    6
[2,]    3    8
[3,]    5   11
```

- $nr = 3$ specifica il numero di righe
- La matrice A viene riempita per colonne

Le matrici possono essere costruite mettendo assieme dei vettori con le funzioni *rbind* (unisce vettori riga) e *cbind* (unisce vettori colonna):

```
> x <- c(1, 4, 6, 2)
> y <- c(5, 1, 2, 9)
> rbind(x, y)
  [,1] [,2] [,3] [,4]
x    1    4    6    2
y    5    1    2    9
```

Per accedere a un elemento di una matrice si usano le parentesi quadre:

```
> A <- matrix( c(1, 3, 5, 6, 8, 11), nr=3)
> A[2, 1]  # seconda riga, prima colonna
[1] 3
```

per selezionare una riga o una colonna si usa la sintassi:

```
> A[3, ]   # seleziona la terza riga
> A[ ,2]   # seleziona la seconda colonna
```

A.8 Le basi di R: le liste

Una lista è un contenitore di oggetti di diverso tipo:

```
> ls <- list(colore="rosso", altezza=120)
```

ls contiene gli oggetti *colore* (carattere) e *altezza* (numerico).

Per accedere agli elementi della lista si usa l'operatore '$':

```
> ls$colore
[1] "rosso"
> ls$altezza
[1] 120
```

A.9 Le basi di R: importare dati da un file

Per leggere un file si può usare la funzione *read.table*:

```
> read.table("file.dat", head=TRUE)
```

L'opzione $head = TRUE$ è necessaria se le tabella contiene una riga di intestazione con i nomi delle variabili.

A.10 Importare e modificare una tabella di dati

Se i dati importati provengono da un foglio di calcolo (es: Excel) è spesso necessario modificarne il formato. Nei fogli di calcolo è comune classificare i dati in tabelle a due entrate in cui le righe corrispondono a diversi valori di una variabile $A1$ e le colonne a diversi valori di una variabile $A2$, come nella tabella seguente.

sogg	T1	T2	T3
1	1.0	2.1	3.3
2	2.5	2.7	1.3
3	1.1	2.2	1.8
4	4.4	3.2	2.6
5	2.1	3.9	2.5

In R è spesso necessario avere gli stessi dati in un formato differente con le misure disposte in record separati, accompagnati da due variabili in cui viene tenuta traccia della loro provenienza (cioè della classificazione rispetto a $A1$ e $A2$). La funzione *reshape* può essere impiegata a tal fine:

```
> d <- read.table("dati", head=TRUE)   # leggo la tabella di dati da file
> reshape(d, direction="long", idvar="sogg", varying=list(names(d[,2:4])),
+ timevar="T", v.names="val")
    sogg T val
1.1    1 1 1.0
2.1    2 1 2.5
3.1    3 1 1.1
4.1    4 1 4.4
5.1    5 1 2.1
1.2    1 2 2.1
2.2    2 2 2.7
3.2    3 2 2.2
4.2    4 2 3.2
5.2    5 2 3.9
1.3    1 3 3.3
2.3    2 3 1.3
3.3    3 3 1.8
4.3    4 3 2.6
5.3    5 3 2.5
```

idvar è la variabile che identifica i soggetti per i quali si hanno le misure ripetute (la chiave di classificazione $A1$), *varying* le variabili corrispondenti ai livelli del fattore $A2$ (in questo caso le colonne dalla 2 alla 4 della tabella), *timevar* e *v.names* servono solamente ad attribuire il nome di variabile alle colonne relative al fattore $A2$ e alle misure.

A.11 Le sequenze numeriche

Si possono creare semplici sequenze di numeri interi usando l'operatore ':'

```
> 4:9
[1] 4 5 6 7 8 9
```

Per sequenze più complicate si usa la funzione *seq*:

```
> seq(1, 23, by=3)
[1]  1  4  7 10 13 16 19 22
```

A.12 I fattori

Una variabile categoriale si definisce con la funzione *factor*:

```
> a <- factor(c(1,1,1,2,2,3,4,4,4))
> a
[1] 1 1 1 2 2 3 4 4 4
Levels: 1 2 3 4
```

La funzione *gl* è d'aiuto per generare un fattore specificandone i livelli. Essa accetta di base 3 argomenti:

```
gl(n, k, length = n*k)
```

dove n è il numero di livelli, k il numero di repliche *length* la lunghezza del vettore da generare (se omesso vale $n * k$):

```
> gl(3, 2)        # fattore a 3 livelli con due ripetizioni l'uno
[1] 1 1 2 2 3 3
Levels: 1 2 3
> gl (2, 4, 16)   # fattore a 2 livelli con 4 ripetizioni.
                  # l'intero vettore ripetuto 2 volte: 16 = 2*(2*4)
[1] 1 1 1 1 2 2 2 2 1 1 1 1 2 2 2 2
Levels: 1 2
```

A.13 Estrarre e selezionare dati

Usando una condizione logica:

```
> x <- c(1,2,3,5,7,11)
> x[x > 5]
[1]  7 11
> x[x > 5 & x < 10]
[1] 7

> y <- c("a","b","c","d","e","f")
> y[x != 5]         # il simbolo != significa "diverso da"
[1] "a" "b" "c" "e" "f"
```

Usando un vettore di indici:

```
> x <- c(1,2,3,5,7,11)
> x[ c(2,4,5) ]
[1] 2 5 7
```

Si usa un '-' per escludere dalla selezione:

```
> x <- c(1,2,3,5,7,11)
> x[ -c(2,3) ]
[1]  1  5  7 11
```

Appendice B

GNU Free Documentation License

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "**Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "**Invariant Sections**" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does

not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "**Cover Texts**" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "**Transparent**" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "**Opaque**".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "**Title Page**" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "**publisher**" means any person or entity that distributes copies of the Document to the public.

A section "**Entitled XYZ**" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "**Acknowledgements**", "**Dedications**", "**Endorsements**", or "**History**".) To "**Preserve the Title**" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their

copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with ... Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Appendice C

History

- v. 1.0.0 (gennaio 2006):
  versione base
- v. 1.0.1 (maggio 2006):
  aggiunta MANOVA
- v. 1.1.0 (luglio - ottobre 2006):
  aggiunti test multipli di Holm e Bonferroni per la correlazione
  diviso capitolo Analisi multivariata in due parti
  aggiunto PAM
  aggiunto MDS
  aggiunta CA
  modificate cluster analysis, PCA e LDA
  aggiunta distribuzione normale multivariata
  aggiunto shrunken centroid
  aggiunto Significance Analysis of Microarrays (SAM)
  aggiunto indice analitico
  revisione linguistica
- v. 1.1.1 (dicembre 2006):
  modificata kernel density
  modificato kernel smoothing
  aggiunti modelli additivi generali
  aggiunta projection pursuit regression
- v. 1.1.2 (aprile 2007):
  aggiunta sottosezione in LDA
- v. 1.1.3 (agosto 2007):
  modificata ANOVA per modelli random
- v. 1.1.4 (febbraio 2008):
  aggiunto esempio SAM
  modificato esempio Contrasti multipli
- v. 1.2.0 (aprile 2008):
  aggiunto capitolo Geostatistica
  aggiunto paragrafo Calcolo dei residui per regressione di Cox
- v. 1.2.1 (dicembre 2008):
  aggiunto paragrafo Simulazione Monte Carlo per il calcolo dei valori p
  aggiunto paragrafo Confronti multipli: test di Tukey e modelli mixed-effect
  modificato Disegni split-plot

- v. 1.2.2 (aprile 2009):
  aggiunto paragrafo Gibbs sampling
- v. 1.3.0 (marzo 2010):
  aggiunte sezioni di Geostatistica basata su modello
  aggiunto capitolo Analisi genomica
- v. 1.3.1 (novembre 2010):
  aggiunta sezione Tecniche di valutazione della bontà di un modello logistico
  modificata funzione covariate e aggiornati esempi
- v. 1.3.2 (gennaio 2011):
  aggiornata FDL alla versione 1.3
- v. 1.3.3 (agosto 2012):
  revisione capitolo Modelli lineari generalizzati (GLM)

Indice analitico

Bibliografia

[1] D.G. Altman and P.K. Andersen, *Bootstrap investigation of the stability of a Cox regression model*. Statistics in Medicine, 8:771-783, 1989.

[2] P. Armitage, G. Berry and J.N.S. Matthews, *Statistical Methods in Medical Research*. Blackwell Publishing, 2002 (4th ed.).

[3] R.J. Barnes, *The variogram sill and the sample variance*. Math. Geology. 23(4), 673, 1991.

[4] D. Bates, *Fitting linear mixed models in R*. R News, 5(1):27-30, 2005.

[5] C. Bishop, *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

[6] B. Boser, I. Guyon and V. Vapnik, *A training algorithm for optimal margin classifiers*. In Proceedings of Fifth Annual ACM Workshop on Computational Learning Theory, pp. 144-152, Pittsburgh, ACM Press, 1992.

[7] L. Breiman, *Random Forests*. Machine Learning, 45(1):5-32, 2001.

[8] L. Breiman, J.H. Friedman, R.A. Olshen and C.J. Stone, *Classification and Regression Trees*. Wadsworth, California, 1984.

[9] C.J.C. Burges, *A tutorial on support vector machines for pattern recognition*. Data Mining and Knowledge Discovery, 2:121-167, 1998.

[10] G. Casella and E.I. George, *Explaining the Gibbs Sampler*. The American Statistician, 46(3):167-174, 1992.

[11] C.C. Chang and C.J. Lin, *LIBSVM: a library for support vector machines*. 2001. http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[12] C. Chatfield, *Analysis of Time Series: an Introduction*. Chapman and Hall, London, 1989 (4th ed.).

[13] W.S. Cleveland, *LOWESS: A program for smoothing scatterplots by robust locally weighted regression*. The American Statistician, 35:54, 1981.

[14] W.S. Cleveland, E. Grosse and W.M. Shyu, *Local regression models*. Chapter 8 of *Statistical Models in S* eds J.M. Chambers and T.J. Hastie, Wadsworth & Brooks/Cole, 1992.

[15] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associated, Hillsdale (NJ), 1988.

[16] W.J. Conover, *Practical Nonparametric Statistics*. John Wiley & Sons, New York, 1999 (3th ed.).

[17] W.J. Conover, M.E. Johnson and M.M. Johnson, *A comparative study of tests for homogeneity of variances, with applications to the outer continental shelf bidding data*. Technometrics 23:351-361, 1981.

[18] A.C. Davison and D.V. Hinkley, *Bootstrap Methods and Their Application*. Cambridge University Press, 1997.

[19] R.C. Deonier, S. Tavaré and M.S. Waterman, *Computational Genome Analysis*. Springer, 2005.

[20] S. Derksen and H.J. Keselman, *Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables*. British Journal of Mathematical and Statistical Psychology, 45:265-282, 1992.

[21] R. Diaz-Uriarte and S. Alvarez de Andres, *Variable selection from random forests: application to gene expression data*. Tech. report, 2005. http://ligarto.org/rdiaz/Papers/rfVS/randomForestVarSel.html.

[22] P.J. Diggle and P.J. Ribeiro jr., *Model-based Geostatistics*. Springer, 2007.

[23] A.J. Dobson, *An Introduction to Generalized Linear Models*. Chapman & Hall/CRC, 2002 (2nd ed.).

[24] B. Efron, *Bootstrap Methods: Another look at The Jackknife*. The Annals of Statistics, 7(1):1-26, 1979.

[25] B. Efron and R. Tibshirani, *An introduction to the Bootstrap*. Marcel and Decker, 1993.

[26] J.J. Faraway, *Practical Regression and ANOVA using R*. http://cran.r-project.org/other-docs.html.

[27] M.-J. Fortin and M. Dale, *Spatial Analysis. A Guide for Ecologists*. Cambridge University Press, 2005.

[28] R. Gentleman, V.J. Carey, W. Huber, R.A. Irizarry and S. Dudoit, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Springer, 2005.

[29] P. Grambsch and T.M. Therneau, *Proportional hazard tests and diagnostics based on weighted residuals*. Biometrika, 81:515-526, 1994.

[30] I. Guyon, J. Weston, S. Barnhill and V. Vapnik, *Gene Selection for Cancer Classification using Support Vector Machines*. Machine Learning, 46:389-422, 2002.

[31] W. Härdle and L. Simar, *Applied Multivariate Statistical Analysis*. Springer-Verlag, Berlin, 2003.

[32] D.P. Harrington and T.R. Fleming, *A class of rank test procedures for censored survival data*. Biometrika, 69:553-566, 1982.

[33] T. Hastie and R. Tibshirani, *Generalized Additive Models*. Chapman and Hall, 1990.

[34] D.W. Hosmer, T. Hosmer, S. le Cessie and S. Lemeshow, *A comparison of goodness-of-fit tests for the logistic regression model*. Statistics in Medicine, 16:965-980, 1997.

[35] D.W. Hosmer and S. Lemeshow, *Applied Logistic Regression*. John Wiley & Sons, New York, 2000 (2nd ed.).

[36] J.C. Hsu, *Multiple Comparisons. Theory and methods*. Chapman & Hall/CRC, 1996.

[37] P.J. Huber, *Robust Statistics*. John Wiley & Sons, New York, 1981.

[38] C.M. Hurvich and C.L. Tsai, *The impact of model selection on inference in linear regression*. The American Statistician, 44:214-217, 1990.

[39] E.H. Isaaks and R.M. Srivastava, *An Introduction to Applied Geostatistics*. Oxford University Press, Oxford, 1989.

[40] L. Kaufman and P.J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York, 1990.

[41] P. Legendre and L. Legendre, *Numerical Ecology*. Elsevier, Amsterdam, 1998.

[42] J.C. Pinheiro and D.M. Bates, *Mixed-Effects Models in S and S-PLUS*. Springer, 2000.

[43] R Development Core Team, *An Introduction to R*. http://cran.r-project.org/manuals.html.

[44] R Foundation, http://www.r-project.org.

[45] B.D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, 1996.

[46] E.B. Roecker, *Prediction error and its estimation for subset–selected models*. Technometrics, 33:459-468, 1991.

[47] P. Royston and D. Altman, *Regression using fractional polynomials of continuous covariates*. Applied Statistics, 3:429-467, 1994.

[48] W. Sauerbrei and P. Royston, *Building multivariable prognostic and diagnostic models: transformation of the predictors by using fractional polynomials*. Journal of the Royal Statistical Society (Series A), 162:71:94, 1999.

[49] R.H. Shumway and D.S. Stoffer, *Time Series Analysis and its Applications*. Springer-Verlag, New York, 2000.

[50] G.W. Snedecor and W.G. Cochran, *Statistical Methods*. Iowa State University Press, 1989 (8th ed.).

[51] R.L. Somorjai, B. Dolenko and R. Baumgartner, *Class prediction and discovery using gene microarray and proteomics mass spectroscopy data: curses, caveats, cautions*. Bioinformatics, 19:1484-1491, 2003.

[52] J.D. Storey, *A direct approach to false discovery rates*. Journal of the Royal Statistical Society B, 64:479-498, 2002.

[53] V. Svetnik, A. Liaw, C. Tong and T. Wang, *Application of Breiman's random forest to modeling structure-activity relationships of pharmaceutical molecules*. Lecture Notes in Computer Science, vol. 3077, pp. 334-343. F. Roli, J. Kittler, and T. Windeatt (eds.). Springer, Berlin, 2004.

[54] T.M. Therneau, *A Package for Survival Analysis in S*, 1999.

[55] V.G. Tusher, R. Tibshirani and G. Chu, *Significance analysis of microarrays applied to the ionizing radiation response*. PNAS, 98:5116-5121, 2001.

[56] R. Tibshirani, *Regression shrinkage and selection via the lasso*. Journal of the Royal Statistical Society B 58:267-288, 1996.

[57] R. Tibshirani, T. Hastie, B. Narasimhan and G. Chu, *Diagnosis of multiple cancer types by shrunken centroids of gene expression*. PNAS, 99:6567-6572, 2002.

[58] V.N. Vapnik, *Statistical Learning Theory*. John Wiley & Sons, New York, 1998.

[59] W.N. Venables and B.D. Ripley, *Modern Applied Statistics with S*. Springer-Verlag, New York, 2002 (4th ed.).

[60] S.N. Wood, *Generalized Additive Models: An Introduction with R*. CRC Press, 2006.

[61] S.N. Wood, *mgcv: GAMs and generalized ridge regression for R*. R News 1(2):20-25, 2001.