Università degli Studi di Trieste

Facoltà di Economia

Introduzione all'Analisi Esplorativa dei Dati mediante R

Nicola Torelli e Matilde Trevisani

Versione 1.0 - 4 novembre 2003

Prefazione

In questa dispensa, che riprende alcuni materiali preparati nell'ambito del corsi di Statistica di base e di Analisi esplorativa dei dati tenuti rispettivamente presso l'Università di Padova (con il sostanziale apporto del dott. Claudio Agostinelli) e l'Università di Trieste, l'uso di **R** viene illustrato con riferimento agli strumenti elementari per condurre analisi descrittive di dati statistici e per l'analisi di regressione semplice. Questo consentirà anche di rivedere alcuni concetti di base di statistica attraverso il ricorso ad applicazioni delle tecniche studiate ad alcuni piccoli insiemi di dati reali.

I dati usati nel testo sono reperibili alla pagina *web* di indirizzo *http://www.econ.units.it/didattica/didattica.asp* nell'area dedicata al *download* del materiale didattico assegnata al corso di Analisi esplorativa dei dati della Facoltà di Economia dell'Università di Trieste. Inoltre, per i dati tratti dalla letteratura, la fonte è comunque sempre indicata.

A integrazione e approfondimento degli argomenti trattati si consigliano le letture citate in fondo al testo. In particolare, segnaliamo: il manuale *An Introduction to R* (disponibile assieme al *software*), per quanto concerne gli aspetti informatici; *Analisi dei dati statistici - I osservazioni in una o due dimensioni* di S. Zani, per la parte statistica.

Notazione

Nel testo i comandi di **R** sono indicati con lo stesso carattere usato dal *software*, ovvero con il carattere `typewriter`.
Le "parole chiave" (concetti, comandi, *etc.*) delle varie sezioni sono incorniciate al modo seguente: parola chiave.

Indice

Capitolo 1

Introduzione al pacchetto R

1.1 Il pacchetto R

1.1.1 Breve presentazione

R è un pacchetto per la grafica e l'analisi statistica.

R è uno sviluppo del linguaggio S. Un altro pacchetto che sviluppa lo stesso linguaggio è S-Plus, ma mentre S-Plus è un prodotto commerciale (e costoso), **R** è *gratis*.

> **R** *is 'GNU S'* : così inizia la *homepage* di R (all'indirizzo *http://www.r-project.org/*).
> Nel 1984 fu lanciato il, cosiddetto, GNU *Project* per sviluppare un sistema completo, operante come quello Unix (GNU è un acronimo ricorsivo per *GNU's Not Unix!*), che fosse *free software*. *Free* va inteso nel senso di 'libertà di usare, copiare, distribuire, studiare, adattare e sviluppare' il *software*, da cui discende anche il concetto più ristretto di gratuito. In Figura 1.1 è riportato il simbolo dello GNU *project*: (banalmente) la testa di uno gnu!

Figura 1.1: Il simbolo dello GNU *project*

R è molto simile a S-Plus, ma utenti di entrambi i pacchetti giudicano **R** migliore sotto certi aspetti. Ad esempio, **R** fa un utilizzo migliore della memoria: risulta, infatti, meno pesante per il sistema ed, in genere, più stabile.

1.1.2 Installazione

Si può ottenere una copia del programma **R** da *Internet* all'indirizzo:

http://cran.r-project.org

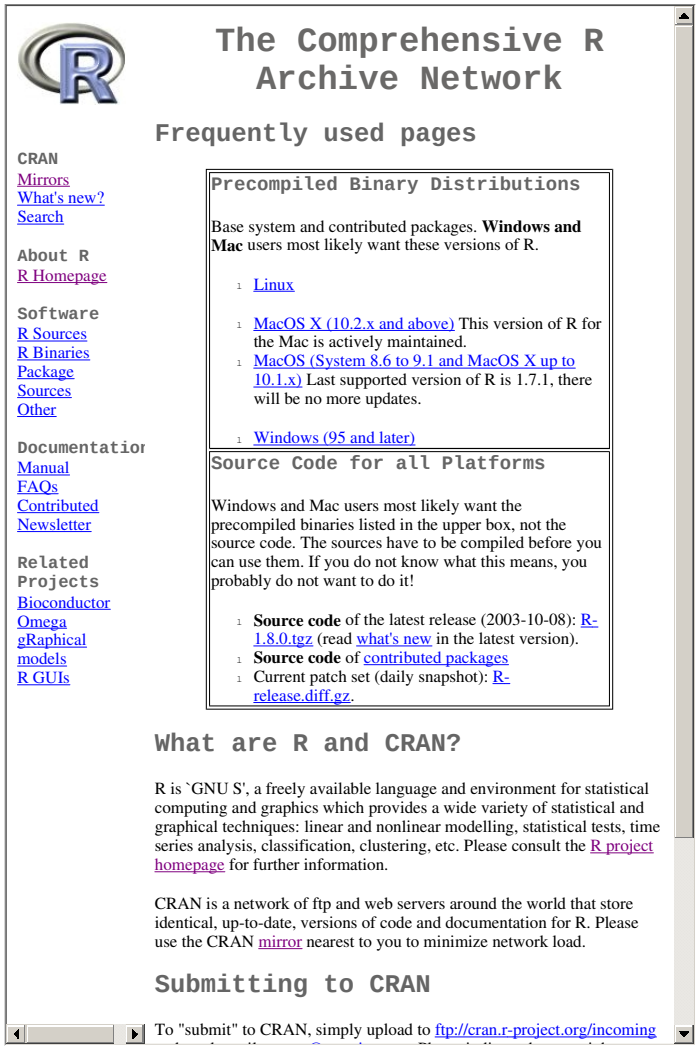
La Figura 1.2 mostra la pagina *web* suindicata.
Nel riquadro centrale ci sono i *links* per scaricare le distribuzioni di **R** per diverse piattaforme (tra cui *Unix*, *Linux* e *Windows*).

> Nella metà superiore si possono scaricare i codici binari (cioè precompilati o direttamente eseguibili). Nella metà inferiore sono accessibili i codici sorgenti (cioè i programmi non precompilati). La disponibilità dei sorgenti è in rispetto ai principi dello GNU *project*.

Alla data di oggi (4 novembre 2003) l'ultima versione è R 1.8.0. Per installare **R** per *Windows*, ad esempio, si seguano i seguenti passi:

1. Da *http://cran.r-project.org* aprire `Windows (95 and later)`;
2. nella pagina aperta aprire `base`;
3. dalla pagina aperta scaricare `Rw1080.exe`;
4. installare **R** eseguendo `Rw1080.exe` (facendo 'doppio click') e seguendo le istruzioni che di volta in volta vi saranno date.

Lungo il margine sinistro della pagina sono riportati i collegamenti principali tra cui anche il *link* `R Homepage` da quale discendono tutte le pagine del sito di **R** .
Tra i collegamenti, possono risultare utili quelli relativi alla `Documentazione` da cui si può accedere ai manuali e alle *Frequently Asked Questions* su **R** .



The Comprehensive R Archive Network

Frequently used pages

CRAN
Mirrors
What's new?
Search

About R
R Homepage

Software
R Sources
R Binaries
Package Sources
Other

Documentation
Manual
FAQs
Contributed
Newsletter

Related Projects
Bioconductor
Omega
gRaphical models
R GUIs

Precompiled Binary Distributions

Base system and contributed packages. **Windows and Mac** users most likely want these versions of R.

- Linux
- MacOS X (10.2.x and above) This version of R for the Mac is actively maintained.
- MacOS (System 8.6 to 9.1 and MacOS X up to 10.1.x) Last supported version of R is 1.7.1, there will be no more updates.
- Windows (95 and later)

Source Code for all Platforms

Windows and Mac users most likely want the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- **Source code** of the latest release (2003-10-08): R-1.8.0.tgz (read what's new in the latest version).
- **Source code** of contributed packages
- Current patch set (daily snapshot): R-release.diff.gz.

What are R and CRAN?

R is `GNU S', a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc. Please consult the R project homepage for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN mirror nearest to you to minimize network load.

Submitting to CRAN

To "submit" to CRAN, simply upload to ftp://cran.r-project.org/incoming

Figura 1.2: La pagina *http://cran.r-project.org*

1.2 Strumenti di lavoro, aiuto e salvataggio

In fase di installazione di **R** , o in un secondo momento, può essere creata sulla finestra principale di *Windows* un' icona, raffigurante il logo di **R**

Facendo doppio 'click' - con il tasto sinistro del *mouse* - sull'icona, si apre la *console*, o quadro di comando, di **R** . Il doppio 'click' sull'icona equivale ad eseguire l'applicazione `Rgui.exe` che è archiviata nel *folder* contenente il pacchetto **R** . Il percorso a `Rgui.exe` è *percorso a rw1080* `\rw1080\bin\Rgui.exe`. Nella finestra di **R** i menù `File`, `Edit`, `Misc` e `Help` sono utili fin dalla prima sessione di lavoro; nel seguito si indicheranno le principali funzioni. Il menù `Packages` si rivela essere uno strumento pratico di lavoro in una fase successiva: esso permette di aggiungere al pacchetto base di **R** altri pacchetti per analisi più specifiche.
Al *prompt* > si può iniziare a lavorare interattivamente in **R** eseguendo i comandi desiderati. Tutti i comandi sono funzioni del tipo `nomecomando()`. Se si omettono le parentesi, **R** stamperà il contenuto della funzione invece di eseguirla.
La *working directory* è la *directory* dove **R** assume siano collocati i vari *files* esterni, usati durante la sessione di lavoro per importare o esportare dati funzioni etc., e dove i *files* `.Rdata` e `.Rhistory` di cui diremo dopo sono creati. Per sistemare la *working directory* è sufficiente aprire la finestra di dialogo `Change directory` dal menù `File` e ivi scrivere il percorso scelto.
Inoltre, per una gestione efficiente del lavoro, si possono creare tante *working directories* quante sono le distinte aree di analisi nelle quali siamo di volta in volta impegnati.

Per avere informazioni su di un comando si può utilizzare l'*help*

1. in formato HTML, digitando

   ```
   > help.start()
   ```

   (a cui **R** risponderà `updating HTML package listing If nothing happens, you have to open ‘ C:\PROGRA~1\R\rw1080\doc\html\`

`rwin.html ' by yourself)`,
o, dal menù `help`, aprendo `R language (html)`;

2. oppure, se si conosce il nome del comando (ad esempio `hist`), si può digitare direttamente

```
> help(hist)
```

e leggere le istruzioni contenute nella finestra che **R** aprirà.

Tutto quello che è prodotto nella sessione di lavoro viene automaticamente archiviato. Per vedere quali sono gli oggetti archiviati - durante la sessione corrente come anche in sessioni precedenti che sono state salvate - si possono utilizzare i comandi `objects()` o `ls()`.
Per vedere la *storia* dei comandi - eseguiti nella sessione corrente oppure in sessioni passate e salvate - si può utilizzare l'istruzione `history()`. Di *default* sono mostrati gli ultimi 25 comandi; per cambiarne il numero, ad esempio per vedere gli ultimi 100 comandi, basta digitare `history(100)`. Inoltre, le istruzioni `savehistory()` e `loadhistory()` sono utili per salvare su o, rispettivamente, caricare da, un *file* esterno la storia dei comandi. Ad esempio, il comando

```
> savehistory(file="storia.txt")
```

salva i comandi dati durante la sessione corrente, oltre a quelli di sessioni passate e salvate, nel *file* di testo `storia.txt`. In una qualsiasi sessione successiva, questi comandi possono essere ricaricati nella `history()` della sessione stessa, digitando

```
> loadhistory(file="storia.txt")
```

Per concludere una sessione di lavoro digitiamo il comando `q()` (oppure semplicemente si può 'cliccare' sulla × in alto a destra della finestra) dato il quale ci verrà chiesto se vogliamo salvare il lavoro eseguito durante la sessione corrente. A risposta affermativa, **R** salverà gli oggetti creati e la storia dei comandi nel *file* `.Rdata` e, rispettivamente, nel *file* `.Rhistory`. In una sessione successiva, potremo quindi riprendere il lavoro da dove l'avevamo lasciato.
Durante la sessione di lavoro si consiglia di eseguire di tanto in tanto il comando `save.image()` per evitare eventuali perdite del lavoro a seguito di. un'irregolare chiusura di **R**

1.3 Operazioni e vettori

1.3.1 Operazioni numeriche e logiche

Per le operazioni di somma, sottrazione, moltiplicazione e divisione si utilizzano gli usuali simboli: `+ - / *`. Per l'elevazione a potenza si usa ** oppure `^`.
Ad esempio:

```
> 18*2
[1] 36
```

L'*output* viene stampato sul video preceduto da un indice contenuto tra parentesi quadre.
È necessario prestare attenzione all'uso delle parentesi tonde, ad esempio:

```
> 2+3*4
[1] 14
```

non è equivalente a

```
> (2+3)*4
[1] 20
```

Così come

```
> 4*3^2
[1] 36
```

non equivale a

```
> (4*3)^2
[1] 144
```

Bisogna quindi tener presente che l'elevamento a potenza precede la moltiplicazione (e la divisione) che a sua volta precede l'addizione (e la sottrazione).

Gli operatori logici corrispondono invece ai seguenti simboli:

<	minore di
>	maggiore di
<=	minore o uguale a
>=	maggiore o uguale a
==	uguale a
!=	diverso da
&	AND
|	OR
!	NOT

Quindi si ottiene:

```
> 4!=4
[1] FALSE
```

Che indica che l'affermazione è falsa. Come falsa risulta anche l'uguaglianza:

```
> (4*3)^2==4*3^2
[1] FALSE
```

mentre invece:

```
> (4*3)^2==4^2*3^2
[1] TRUE
```

Da notare che risulta:

```
> TRUE==1
[1] TRUE
> FALSE==0
[1] TRUE
```

1.3.2 Vettori

Per assegnare il valore 21 alla variabile `x` si digita:

```
> x<-21
```

oppure, alternativamente, si può usare la seguente sintassi:

```
> x=21
```


Digitando semplicemente il nome della variabile si può vedere il contenuto:

```
> x
[1] 21
```

mentre con il comando `rm()` possiamo rimuovere il contenuto di una variabile

```
> rm(x)
> x
Error: Object "x" not found
```

Si noti che **R** è sensibile ai caratteri maiuscoli e minuscoli così le variabili `x` e `X` non sono la stessa cosa.

È possibile creare variabili contenenti più valori, cioè vettori, con il comando `c()`:

```
> vett<-c(1,2,3,4)
> vett
[1] 1 2 3 4
```

Abbiamo quindi creato un vettore di nome `vett` di quattro elementi.
Attenzione a non usare per il nome di una variabile i nomi riservati a funzioni di **R** .

È possibile estrarre valori dai vettori ponendo l'indice di posizione dell'elemento nel vettore tra parentesi quadre. Ad esempio:

```
> vett[2]
[1] 2
> vett[2:4]
[1] 2 3 4
> vett[c(1,3)]
[1] 1 3
> vett[-c(2,3)]
[1] 1 4
```

L'ultimo esempio mostra come sia possibile utilizzare il segno di sottrazione per escludere valori.
L'estrazione di elementi da un vettore può essere effettuato altresì con gli operatori logici. Costruiamo, ad esempio, un nuovo vettore:

```
> vett<-c(100,18,41,21,53)
```

Se vogliamo ad esempio i valori più grandi di 50:

```
> vett>50
[1]  TRUE FALSE FALSE FALSE  TRUE
> vett[vett>50]
[1] 100  53
```

Se vogliamo i valori compresi tra 20 e 80:

```
> vett[vett>20 & vett<80]
[1] 41 21 53
```

Se moltiplichiamo un vettore per uno scalare si ottiene che ogni elemento del vettore viene moltiplicato per lo scalare:

```
> vett*3
[1] 300  54 123  63 159
```

Così per tutte le altre operazioni. In generale quindi le operazioni eseguite sul vettore vengono effettuate su ciascuno dei suoi elementi.
Se costruiamo un altro vettore:

```
> vett2<-c(27,28,29,30,31)
```

e lo sommiamo al vettore `vett` si ottiene la somma elemento per elemento (i vettori devono avere lo stesso numero di elementi).

```
> vett+vett2
[1] 127  46  70  51  84
```

Analogamente per le altre operazioni.

Possiamo aggiungere elementi ad un vettore. Ad esempio, uniamo il vettore `vett` e il vettore `vett2` in un unico vettore che chiamiamo `vett1`:

```
> vett1<-c(vett,vett2)
> vett1
[1] 100  18  41  21  53 127  46  70  51  84
```


1.3.3 Prime funzioni utili allo statistico

La funzione `length()` fornisce il numero di elementi di un vettore:

```
> length(vett)
[1] 5
```

La funzione `sum()` consente invece di ottenere la somma degli elementi contenuti in un vettore:

```
> sum(vett)
[1] 233
```

La funzione `cumsum()` consente invece di avere la somma cumulata:

```
> cumsum(vett)
[1] 100 118 159 180 233
```

La media del vettore `vett` può essere semplicemente ottenuta nel seguente modo:

```
> sum(vett)/length(vett)
[1] 46.6
```

Esiste comunque la funzione `mean()` che fornisce direttamente la media aritmetica:

```
> mean(vett)
[1] 46.6
```

Per ordinare i valori si può utilizzare la funzione `sort()`

```
> svett<-sort(vett)
> svett
[1]  18  21  41  53 100
```

Dato che la dimensione di `vett` è dispari allora la mediana è il valore che occupa la posizione $5/2 + 1 = 3$ del vettore ordinato `svett`, quindi:

```
> svett[5/2+1]
[1] 41
```

Oppure più semplicemente usando la funzione `median()`:

```
> median(vett)
[1] 41
```

La mediana può essere ottenuta anche dalla funzione `quantile()` che calcola i quantili:

```
> quantile(vett,probs=0.5)
50%
 41
```

Dove `probs=0.5` indica che si desidera il quantile corrispondente al livello 0.5. Cosìper avere il primo e terzo quartile:

```
> quantile(vett,probs=c(0.25,0.75))
25% 75%
 21  53
```

Per calcolare lo scarto interquartile si può sfruttare la funzione `diff()`:

```
> diff(quantile(vett,probs=c(0.25,0.75)))
75%
 32
```

Infine la funzione `summary()` fornisce con un unico comando vari indicatori di sintesi:

```
> summary(vett)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   18.0    21.0    41.0    46.6    53.0   100.0
```

Per il calcolo della varianza possiamo utilizzare la definizione:

```
> sum((vett-mean(vett))^2)/length(vett)
[1] 879.44
```

mentre la funzione `var()` fornisce la varianza campionaria:

```
> var(vett)
[1] 1099.3
> var(vett)*(length(vett)-1)/length(vett)
[1] 879.44
```

Si ricorda che il denominatore della varianza campionaria è pari alla numerosità meno 1.

Capitolo 2

Due esempi per cominciare a usare R

2.1 Quanto pesano i neonati?

Come primo esempio consideriamo dei dati sui 177 nati nel 1968 presso l'ospedale di Camposampiero, un comune in provincia di Padova. Per ogni neonato sono registrati mese di nascita, sesso, peso (in grammi), età (in anni), durata gestazionale (in giorni) e, infine, ordine di nascita.

Lettura dei dati Il testo del *file* `neonati.txt` contenente i dati si presenta come segue:

```
mese    sesso   peso    eta   gest    ordine
5       1       3250    31    286     4
10      2       2800    31    272     4
9       1       3050    20    269     1
11      1       3850    25    278     2
7       2       2700    26    264     2
. . .
.
.
```

I dati, in formato testo, sono nella forma classica di una matrice: una riga di valori (assunti dalle variabili) per ogni unità e una colonna di valori (assunti dalle unità) per ogni variabile. Le variabili, inoltre, sono separate da spazi

bianchi. In tal caso, come anche nel caso in cui le variabili non appaiano come colonne ordinate e siano separate da ',' o ';', la matrice dei dati può essere importata in **R** con il comando `read.table()`. L'opzione `header=T` va specificata nel caso il *file* dei dati abbia i nomi delle variabili nell'intestazione. Ad esempio, l'assegnazione:

```
neonati<-read.table(file="neonati.txt",header=T)
```

crea un oggetto di nome `neonati` la cui struttura è quella di un *data frame*

```
> class(neonati)
[1] "data.frame"
```

ossia una struttura a matrice con le variabili per colonna, e per riga le osservazioni rilevate per neonato $1, 2, 3, 4, 5, \ldots$.

```
> neonati[1:5,]
  mese sesso peso eta gest ordine
1    5     1 3250  31  286      4
2   10     2 2800  31  272      4
3    9     1 3050  20  269      1
4   11     1 3850  25  278      2
5    7     2 2700  26  264      2
```

2.1.1 L'analisi delle singole variabili

Analizziamo la distribuzione delle variabili prese singolarmente.
Per poter richiamare le variabili del *data frame* mediante il nome a loro assegnato, è sufficiente dare il comando:

```
> attach(neonati)
```

La funzione `attach()` colloca il *data frame* nella posizione 2 del *percorso di ricerca* di **R**

```
> search()
[1] ".GlobalEnv"    "neonati"       "package:ctest" "Autoloads"
[5] "package:base"
```

rendendo così accessibili per nome le variabili `mese, sesso, peso, eta, gest, ordine`, per tutta la durata della sessione corrente - a meno di dare il comando inverso `detach(neonati)`.

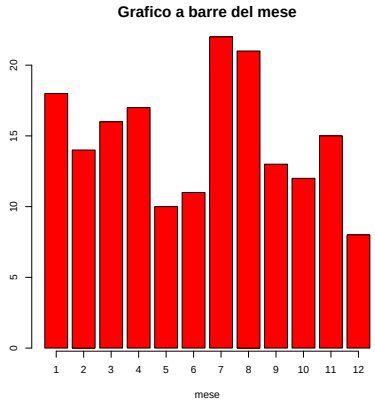


Figura 2.1: Diagramma a rettangoli separati.

Variabili qualitative Prendiamo la prima variabile: `mese di nascita`. É una variabile qualitativa le cui modalità, i mesi, sono codificate con i numeri da 1 a 12.
La distribuzione di frequenze (assolute) si può ottenere tramite la funzione `table()`:

```
> table(mese)
mese
 1  2  3  4  5  6  7  8  9 10 11 12
18 14 16 17 10 11 22 21 13 12 15  8
```

La funzione `barplot()` permette di visualizzare graficamente la distribuzione di frequenza attraverso un diagramma a rettangoli separati o grafico a barre.

```
> barplot(table(mese),main="Grafico a barre del mese",cex.main=1.5)
```

Mentre il grafico a barre rappresenta le frequenze con le altezze delle barre, il diagramma a torta, disegnato dalla funzione `pie()`, le rappresenta con le aree degli spicchi della torta.

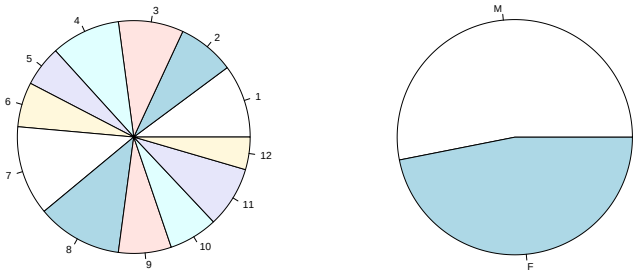


Figura 2.2: Diagrammi a torta.

```
> pie(table(mese))
```

Dalle figure 2.1 e 2.2 (pannello sinistro) il lettore giudichi quale rappresentazione sia più appropriata per immediatezza e chiarezza.

`sesso` è una variabile qualitativa (nominale) dicotomica in cui M=1 e F=2. Per calcolare la distribuzione di frequenze relative usiamo nuovamente la funzione `table()` assieme alla funzione `length()` nel modo seguente:

```
> table(sesso)/length(sesso)
sesso
        1         2
0.5310734 0.4689266
```

Lo stesso risultato si ottiene mediante le assegnazioni:

```
> r.M<-sum(sesso==1)/length(sesso)
> r.F<-sum(sesso==2)/length(sesso)
```

Usiamo dinuovo la funzione `pie()` per rappresentare graficamente le frequenze relative.

```
> pie(c(r.M,r.F),labels=c("M","F"))
```

Come si può notare dalla Figura 2.2 (pannello destro), quando le modalità di una variabile qualitativa sono poche, il diagramma a torta è una rappresentazione altrettanto efficace del diagramma a rettangoli separati.

Variabili quantitative La terza variabile, `peso`, è una variabile quantitativa. Varie rappresentazioni grafiche sono possibili.
Il diagramma ramo e foglia o *steam-and-leaf* è una forma utile di rappresentazione quando, oltre che dare un'informazione visuale della sua distribuzione, sia importante riportare i valori numerici della variabile.
In **R** il diagramma *steam-and-leaf* è prodotto dalla funzione `stem()`.

```
> stem(peso,scale=1)
```

`stem()` divide il *range* o campo di variazione della variabile in intervalli di ampiezza regolata dal numero di righe desiderato per il diagramma (controllato dall'opzione `scale=`). Quindi, le prime cifre (i *rami*) degli estremi degli intervalli sono poste a sinistra della linea verticale (ad esempio, le prime 2 cifre degli estremi degli intervalli $[2000, 2200), [2200, 2400), \ldots$ del *range* del `peso`), mentre le cifre nelle posizioni decimali successive (le *foglie*), dei valori osservati per ciascun intervallo, sono poste a destra di questa (ad esempio, le cifre $5, 8, \ldots$ dei pesi $2050, 2180, \ldots$ osservati nell'intervallo $[2000, 2200)$).
La lunghezza delle righe risulta pertanto proporzionale alla frequenza delle rispettive classi, dando un'informazione equivalente a quella di un istogramma (orizzontale). Si veda la figura 2.3 per un confronto tra il diagramma ramo e foglie con `scale=1` (sopra) e `scale=2` (sotto).
La prima riga informa sulla collocazione della virgola decimale. Il peso è in migliaia di grammi, da cui la virgola è dopo 2 cifre alla destra della linea verticale. Si noti tuttavia che l'utilità del diagramma ramo e foglie è limitata al caso in cui sia possibile conservare il dettaglio sui singoli valori (la seconda figura nel nostro caso) e, soprattutto, nel caso in cui si tratti di rappresentare dati quantitativi relativamente ad un numero ridotto di unità (molte meno, di solito, di quelle usate nell'esempio proposto).

Nel caso delle variabili quantitative la rappresentazione grafica generalmente più usata è l'istogramma, ottenuto in **R** con la funzione `hist()`. Ad esempio, con l'istruzione:


```
The decimal point is 2 digit(s) to the right of the |
20 | 58
22 | 0258
24 | 030
26 | 070055
28 | 0000001355700000555558
30 | 000000000034555579000000135557778
32 | 00000033455677800000002255555568
34 | 0000025555558000002235
36 | 000000025555550012455558
38 | 000144555000344556
40 | 00000355
42 | 00
44 |
46 | 08
```

```
The decimal point is 2 digit(s) to the right of the |
20 | 5
21 | 8
22 | 0
23 | 258
24 | 03
25 | 0
26 | 07
27 | 0055
28 | 00000013557
29 | 00000555558
30 | 000000000034555579
31 | 00000135557778
32 | 000000334556778
33 | 00000002255555568
34 | 0000025555558
35 | 000002235
36 | 00000002555555
37 | 0012455558
38 | 000144555
39 | 000344556
40 | 000
41 | 00355
42 |
43 | 00
44 |
45 |
46 | 08
```

Figura 2.3: I diagrammi ramo-e-foglia.

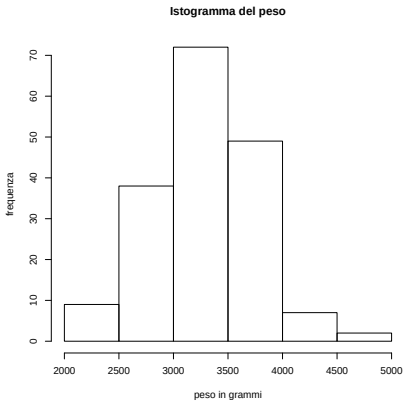


Figura 2.4: Istogramma del peso.

```
> hist(peso,main="Istogramma del peso",
+ xlab="peso in grammi",ylab="frequenza")
```

il *range* dei dati è suddiviso di *default* in classi intervallari di uguale ampiezza, e rettangoli di altezza pari alla frequenza assoluta per classe sono quindi riprodotti. L'istogramma per il peso è riportato in Figura 2.4.
R decide automaticamente il numero di classi (in questo caso ne ha generate 6), altrimenti se ne può fissare il numero usando l'opzione `breaks=`. Ad esempio, specifichiamo lo stesso numero di classi del grafico ramo e foglia prodotto con `scale=2` e, per rendere l'istogramma confrontabile con quest'ultimo, poniamo `right=F` affinchè le classi siano aperte a destra (e chiuse a sinistra).

```
> hist(temperatura,main="Istogramma del Peso",
+ xlab="peso in grammi","ylab="frequenza",
+ breaks=27,right=F)
```

Come si vede dalla Figura 2.5, coll'aumentare del numero delle classi l'istogramma diviene più instabile ma allo stesso tempo presenta un maggiore dettaglio. Inoltre, l'istogramma è stato volutamente ruotato per facilitare il

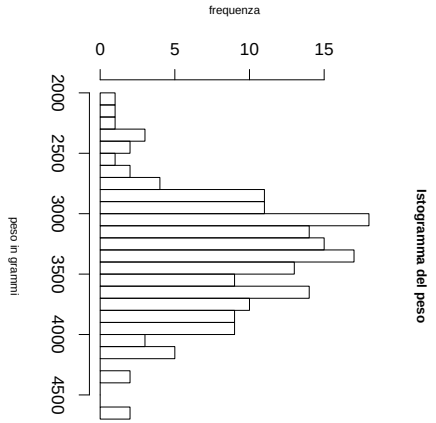


Figura 2.5: Istogramma del peso con 14 classi.

confronto con il grafico ramo e foglia in Figura 2.3 (in basso). I due diagrammi appaiono, com'è ovvio, perfettamente equivalenti nella forma. La differenza risiede essenzialmente nella visualizzazione, nel diagramma ramo e foglia, degli esatti valori numerici dei dati.
L'opzione `breaks=` può essere specificata anche tramite un vettore di valori che, in questo caso, definiscono gli estremi delle classi. Così facendo, si precisa sia l'ampiezza delle classi (che può risultare diversa) che il numero di classi.
Con l'opzione `freq=F`, le altezze dei rettangoli, anzichè dalle frequenze assolute, sono date dalle densità, ossia dalle frequenze relative su base unitaria. Se le classi non sono di uguale ampiezza, necessariamente `freq=F`. Le frequenze relative risultano quindi proporzionali alle aree dei rettangoli, diversamente dai grafici a barre in cui sono proporzionali alle altezze.
Se usiamo l'opzione `plot=F`, l'istogramma non appare nella finestra grafica. In cambio, tutte le informazioni relative alla sua costruzione sono stampate a schermo.

```
>  hist(peso,plot=F)
$breaks
[1] 1999.999 2500.001 3000.001 3500.001 4000.001 4500.001 5000.001
```

```
$counts
[1]  9 38 72 49  7  2

$intensities
[1] 1.016947e-04 4.293785e-04 8.135593e-04 5.536723e-04 7.909605e-05
[6] 2.259887e-05

$density
[1] 1.016947e-04 4.293785e-04 8.135593e-04 5.536723e-04 7.909605e-05
[6] 2.259887e-05

$mids
[1] 2250.000 2750.001 3250.001 3750.001 4250.001 4750.001

$xname
[1] "peso"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"
```

`breaks` sono gli estremi delle classi, `counts` sono le frequenze assolute per ogni classe, `density` sono le frequenze relative su base unitaria di ogni classe (coincidono con `intensities`: quest'ultime sono inserite unicamente per rendere la funzione compatibile con **S**) e `mids` sono i valori centrali delle classi.
Si noti che queste stesse informazioni sono allo stesso modo disponibili con l'assegnazione

```
inf.hist<-hist(peso)
```

L'oggetto `inf.hist` le contiene. In questo caso anche il grafico verrà visualizzato.

Un'altra rappresentazione grafica che descrive la distribuzione dei dati attraverso alcune caratteristiche di sintesi è il diagramma a scatola. Si confronti infatti la Figura 2.6 generata dall'istruzione `boxplot()` con l'istruzione `summary()` già nota. Inoltre, la funzione `boxplot.stats()` fornisce esplicitamente le informazioni con cui **R** costruisce il diagramma a scatola.


```
> boxplot(peso,main="Diagramma a scatola",cex.main=1.5)
> boxplot.stats(peso)
$stats
[1] 2050 3000 3300 3650 4600

$n
[1] 177

...

$out
[1] 4680
> summary(peso)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2050    3000    3300    3323    3650    4680
```

La componente `stats` riporta l'estremo inferiore dei baffi (2050), l'estremo inferiore della scatola (3000), il valore centrale (3300), l'estremo superiore della scatola (3650) e l'estremo superiore dei baffi (4600). L'estremo inferiore della scatola, il valore centrale e l'estremo superiore della scatola corrispondono, rispettivamente, a primo quartile, mediana e terzo quartile. Per definire gli estremi dei baffi occorre innanzitutto calcolare la differenza (o *range*) interquartile IQR, i.e.,

$$IQR = Q(.75) - Q(.25)$$

dove Q(.25) e Q(.75) denotano il primo e terzo quartile. Quindi, l'estremo inferiore (superiore) dei baffi è definito come la minima (massima) osservazione che è maggiore (minore) o uguale il primo (terzo) quartile meno (più) 1.5×IQR. Nel nostro esempio l'estremo inferiore dei baffi coincide con il valore minimo dei dati (2050), mentre l'estremo superiore è minore del valore massimo (4680).
Ancora, `n` è il numero di osservazioni, mentre `out` comprende il valore di tutte le osservazioni che sono al di fuori dei baffi, altrimenti detti valori *outside* tra cui si possono individuare eventuali *outliers*.

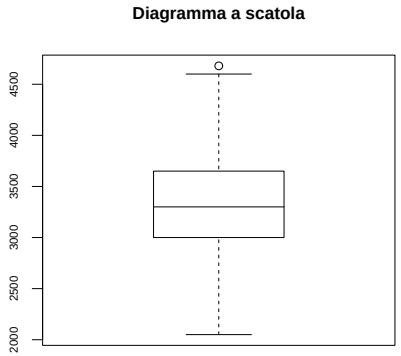


Figura 2.6: Boxplot del peso.

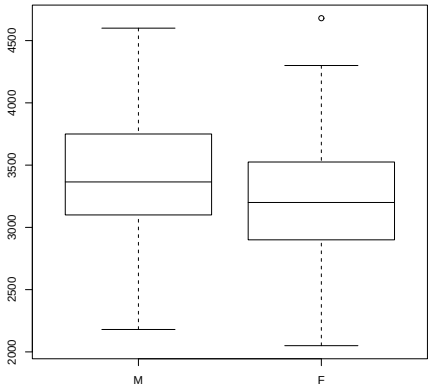


Figura 2.7: Boxplot del peso separatamente per maschi e femmine.

Un'altra rappresentazione grafica utile nel caso di variabili continue è il diagramma dei quantili o funzione di ripartizione empirica.
Si ricorda che per *quantile* corrispondente alla frazione p dei dati, si intende quel valore che divide i dati in modo che una frazione p di questi abbia valore inferiore ad esso, e una frazione 1-p abbia valore superiore. Indicheremo questo valore con $Q(p)$; naturalmente, $0 \leq p \leq 1$. In modo equivalente, ci si può riferire a un quantile associato alla frazione p come al $(100 \cdot p)^o$ percentile. Ad esempio, il quantile corrispondente alla frazione 0.95 dei dati è anche detto 95^o percentile.
Tale definizione, tuttavia, può comportare qualche problema quando ci si accinga a calcolare dei quantili da un insieme di dati. Ad esempio, si consideri un campione di 10 dati. Nel caso volessimo calcolare il 25^o percentile dei dati, non potremmo individuare nessun valore osservato che divida il 25% dei dati dal restante 75%. I valori osservati, infatti, dividono i dati in percentuali che sono multipli del 10%. Inoltre, anche nel caso in cui ad un quantile potesse essere associato un valore effettivamente osservato, ci si troverebbe comunque nell'imbarazzo di dover decidere in quale frazione dei dati, se inferiore o superiore, includere il valore stesso.
Per ovviare a queste difficoltà, si usa generalmente nel calcolo una definizione operativa di quantile. Ovvero, dopo aver ordinato i dati y_i, per $i = 1, \ldots, n$, dal più piccolo al più grande, così ottenendo i dati ordinati $y_{(i)}$, $i = 1, \ldots, n$, si definiscono quantili $Q(p_i)$, per $p_i = (i - 0.5)/n$, $i = 1, \ldots, n$, i valori $y_{(i)}$ stessi.
Se volessi quindi calcolare i valori p_i che corrispondano ai quantili empirici dati da un vettore di n dati ordinati, mi basterebbe eseguire il comando:

```
> p.quantili<-(1:n-0.5)/n
```

Si ricorda che in **R** si possono definire funzioni proprie, il che peraltro risulta spesso molto comodo. Ad esempio, in questo caso, potremmo definire la funzione `p.quantili(n)` come segue

```
> p.quantili<-function(n){
 p<-(1:n -.5)/n
 p
 }
```

e ogniqualvolta volessimo ottenere le frazioni p_i per un campione di numerosità n, basterebbe richiamare la funzione nel modo seguente:

```
> p.quantili(length(campione))
```

Se `campione` è un insieme di 10 dati, la funzione restituisce i seguenti p_i:

```
> p.quantili(10)
 [1] 0.05 0.15 0.25 0.35 0.45 0.55 0.65 0.75 0.85 0.95
```

La funzione appena scritta, disponibile per tutta la sessione di lavoro, potrebbe anche essere salvata e passata ad un amico che potrebbe volerla utilizzare. Questo discorso, tuttavia, è un po' più complesso e potrà essere ripreso più avanti.
Ritornando alla variabile `peso`, costruiamo il diagramma dei quantili.

```
> plot(sort(peso),p.quantili(length(peso)),
+ xlab="quantili del peso", ylab="frazione dei dati",
+ main="Diagramma dei quantili")
```

Esso risulta come da Figura 2.8.

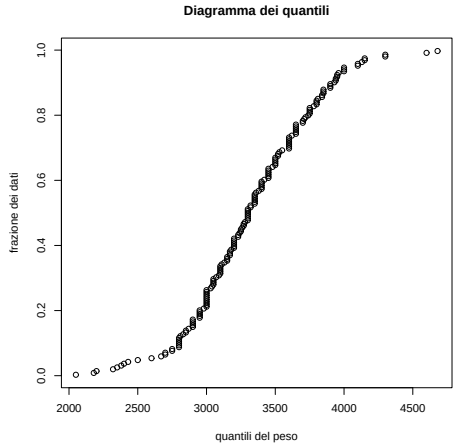


Figura 2.8: Grafico dei quantili del peso.

La funzione plot(a,b) è utile in molti casi. Essa serve per rappresentare su un diagramma un insieme di punti le cui coordinate x e y sono, rispettivamente, nei due vettori a e b.
Riassumendo, la funzione di ripartizione empirica, oltre a fornire i quantili, presenta altre caratteristiche utili.

- ◇ La pendenza locale della curva dei quantili dà un'indicazione della densità locale dei dati: in corrispondenza di punti in cui la pendenza è più ripida la densità è più elevata.
- ◇ Il grafico, in questo caso, non dà solo una sintesi della distribuzione dei dati, ma riporta tutti i singoli dati.
- ◇ Inoltre, ogni punto è disegnato distintamente, anche in presenza di duplicazioni dei dati.

2.1.2 Confronto tra gruppi di dati

Per confrontare diverse distribuzioni relative alla medesima variabile osservata per due diversi gruppi di unità, un metodo semplice può essere quello di riportare l'uno vicino all'altro i grafici costruiti per i singoli gruppi, in modo da avere un confronto visuale efficace.
Ad esempio, nel caso volessimo individuare eventuali differenze nella distribuzione del peso dovute al sesso del neonato, potremmo dare il comando

```
> boxplot(peso[sesso==1],peso[sesso==2],names=c("M","F"))
```

che mostra i diagrammi a scatola distinti per maschi e femmine con l'asse del peso in comune (Figura 2.7).
Si ricordi che il vettore definito da `peso[sesso==1]` è quello dei pesi riferiti alle unità per cui la condizione di uguaglianza data in parentesi risulta vera.
Allo stesso modo potremmo confrontare la distribuzione dell'età della madre per ordine di nascita. La Figura 2.9 fa un confronto dell'età al 1^o, 2^o, fino al 6^o figlio, tralasciando il 7^o, massimo ordine di nascita per i dati, contando solo un'osservazione per l'età della madre.

```
> table(ordine)
ordine
 1  2  3  4  5  6  7
82 57 21 10  3  3  1
```

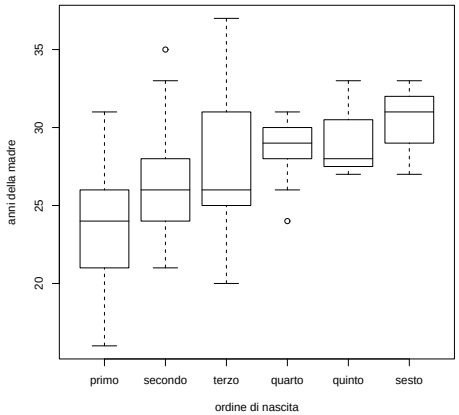


Figura 2.9: Boxplots dell'età della madre rispetto l'ordine di nascita.

```
>  boxplot(eta[ordine==1],eta[ordine==2],eta[ordine==3],
+ eta[ordine==4],eta[ordine==5],eta[ordine==6],
+ names=c("primo","secondo","terzo","quarto","quinto",
+ "sesto"),xlab="ordine di nascita", ylab="anni della madre")
```

Il confronto del peso per maschi e femmine come l'analisi dell'età per ordine di nascita, sono esempi di studi sulla distribuzione di una variabile per diversi livelli di un fattore. In questi casi si usa fare la scomposizione della varianza per analizzare quanta parte della variabilità totale della variabile sia dovuta alle differenze esistenti in media tra i gruppi e quanta, invece, sia dovuta alla variabilità accidentale che i gruppi hanno in media al loro interno.
Dopo aver calcolato le medie condizionate, i.e. le medie per ogni gruppo,

```
> cond.mean<-c(mean(peso[sesso==1]),mean(peso[sesso==2]))
> cond.mean
[1] 3417.128 3216.747
```

così come le dimensioni campionarie di ogni gruppo,


```
> cond.length<-c(length(peso[sesso==1]),length(peso[sesso==2]))
> cond.length
[1] 94 83
```

si ottiene la varianza tra i gruppi o *between-variance*, come varianza pesata (con la numerosità dei gruppi) delle medie condizionate:

```
> between.var<-(cond.length[1]*(cond.mean[1]-mean(peso))^2+
+ cond.length[2]*(cond.mean[2]-mean(peso))^2)/length(peso)
> betweeen.var
[1] 9999.334
```

La varianza intra-gruppo o *within-variance* si ottiene come media pesata (con la numerosità dei gruppi) delle varianze specifiche per gruppo:

```
> within.va<-(cond.length[1]*var(peso[sesso==1])*93/94+
+ cond.length[2]*var(peso[sesso==2])*82/83)/length(peso)
> within.var
[1] 202343.2
```

Si noti come in **R** si usi il comando `var()`, già visto, ma aggiustato per ottenere la varianza non campionaria della variabile.
Come è noto, la varianza totale risulta essere la somma dalla varianza tra i gruppi e della varianza intra-gruppo.

```
> total.var<-var(peso)*(length(peso)-1)/length(peso)
> total.var
[1] 212342.5

> within.var+between.var
[1] 212342.5
> total.var
[1] 212342.5
```

Finalmente, il rapporto della varianza tra i gruppi sulla varianza totale dà la frazione della variabilità spiegata dalle differenze che i diversi livelli del fattore inducono in media sulla variabile studiata.

```
between.var/total.var
[1] 0.04709058
```

Nel caso in esame, la variabilità indotta dal fattore sesso risulta una componente trascurabile della variabilità complessiva del peso.

2.1.3 Relazione tra due variabili

Per analizzare la relazione che intercorre tra due variabili, misurate sullo stesso gruppo di unità, lo strumento più potente è il diagramma di dispersione o *scatter plot*.
L'istruzione in **R** per produrre una 'nuvola di punti' - un terzo termine equivalente - è `plot()`, già precedentemente incontrata.
Si esplorino le 'nuvole' i cui comandi sono sotto riportati.

```
plot(peso,eta)
plot(peso,gest)
```

La dipendenza lineare tra due variabili è espressa dalla covarianza, o, meglio, da un indice detto di correlazione.
Si confrontino i risultati delle funzioni interne `cov()` e `cor()` con quelli ottenuti dalle formule analitiche note.

```
cov(peso,eta)
```

divide per $(n-1)$, infatti:

```
cov(peso,eta)*(n-1)/n
```

e' uguale a:

```
mean((peso-mean(peso))*(eta-mean(eta)))
```

`cor()`, invece, non deve essere corretto.

```
cor(peso,eta)
```

e' uguale al noto rapporto:

```
cov(peso,eta)/(sd(peso)* sd(eta))
```

Si noti che `sd()` è lo scarto quadratico medio calcolato come radice quadrata della funzione `var()`, ossia della varianza campionaria $(/(n-1))$

2.1.4 Esercizi

1. Completare la sezione delle *variabili una ad una* descrivendo le variabili `età`, `gest` e `ord`, mediante opportuni grafici e tabelle numeriche.
2. Si calcoli la scomposizione dela varianza per la variabile `età` rispetto all'ordine di nascita.

2.2 Inseminazione di nuvole

In un esperimento nel Sud della Florida, tra il 1968 e il 1972, 26 nuvole furono inseminate con ioduro di argento. Vennero poi misurate (in *piedi*$\times$*acri*) le precipitazioni provenienti da queste nuvole e da altre 26 nuvole, non trattate, che dovevano agire come unità di controllo nell'esperimento.

Innanzitutto importiamo i dati (tratti da *Graphical methods for data analysis*, Chambers *et al.*, 1983) contenuti in `nuvole.txt`:

```
> nuvole<-read.table("nuvole.txt",he=T)
> nuvole
    Controllo Trattamento
1      1202.6      2745.6
2       830.1      1697.8
3       372.4      1656.0
4       345.5       978.0
5       321.2       703.4
6       244.3       489.1
7       163.0       430.0
8       147.8       334.1
9        95.0       302.8
10       87.0       274.7
11       81.2       274.7
12       68.5       255.0
13       47.3       242.5
14       41.1       200.7
15       36.6       198.6
16       29.0       129.6
17       28.6       119.0
18       26.3       118.3
19       26.1       115.3
20       24.4        92.4
21       21.7        40.6
22       17.3        32.7
23       11.5        31.4
24        4.9        17.5
25        4.9         7.7
26        1.0         4.1
```

Si noti che i valori delle precipitazioni sono disposti in ordine decrescente: è questa la forma originaria in cui i dati sono riportati in letteratura. Ma, si presti attenzione che si tratta di due gruppi di dati e quindi di unità diverse. Pertanto, i dati potrebbero presentarsi anche con ordine crescente per le 26 nuvole trattate:

```
   Controllo Trattamento
1     1202.6         4.1
2      830.1         7.7
3      372.4        17.5
4      345.5        31.4
5      321.2        32.7
.        .           .
. .          .
.        .           .
24       4.9      1656.0
25       4.9      1697.8
26       1.0      2745.6
```

o in qualsiasi altro ordine, senza con ciò cambiare nulla nell'informazione da essi fornita.
Quindi, carichiamo il *dataframe* `nuvole`:

```
> attach(nuvole)
```

2.2.1 Un primo sguardo ai dati

La Figura 2.10 mostra nella metà superiore del riquadro i diagrammi dei quantili per le precipitazioni, sia da nuvole trattate che da quelle di controllo. Si noti, in entrambi i grafici, come l'80% dei dati riporta valori bassi per le precipitazioni ($<$500, per le nuvole trattate, $<$250 per le nuvole di controllo), mentre il restante 20% riguardi valori relativamente molto più elevati e dispersi. Quando ci si trova di fronte ad un comportamento dei dati di questo tipo (comune in dati che misurano quantità che possono assumere solo valori positivi), torna utile trasformare i dati utilizzando trasformazioni degli stessi che possano consentire di evidenziare meglio caratteristiche salienti dei dati. Nel caso in esame, ad esempio, si potrebbe provare a considerare la trasformata logaritmica, ovvero a passare ai logaritmi dei dati. Le funzioni di ripartizione empirica per i logaritmi delle precipitazioni nella metà inferiore del riquadro, ottenute con i comandi:


```
> logTrattate<-log(Trattamento)
> logControllo<-log(Controllo)
> par(mfrow=c(1,2))
> plot(sort(logTrattate),p.quantili(26),
+ xlab="quantili delle log-precipitazioni",ylab="frazione dei dati")
> plot(sort(logControllo),p.quantili(26),
+ xlab="quantili delle log-precipitazioni",ylab="frazione dei dati")
```

mostrano infatti una distribuzione molto più regolare dei valori nel rispettivo *range* di definizione.
Si noti che nell'esempio abbiamo usato `log()` che in **R** corriponde al logaritmo in base e. In modo equivalente, ai fini dell'esercizio, avremmo potuto usare un logaritmo in altra base, ad esempio `log10` che permette di calcolare il logaritmo in base 10 (si veda a proposito l'`help(log)`).

Per confrontare la distribuzione delle precipitazioni provenienti dalle nuvole inseminate con quelle provenienti dalle nuvole di controllo, si possono riportare 'vicini' i rispettivi diagrammi a scatola, come già visto nella sezione precedente. La Figura 2.11 mostra la coppia dei *boxplots* per i dati originari, a sinistra, e per i dati (*log*-)trasformati, a destra. I comandi sono i seguenti:

```
> par(mfrow=c(1,2),pty='s')
> boxplot(Trattamento,Controllo,labels=c("Trattamento",
+ "Controllo"),main="Precipitazioni a confronto")
> boxplot(logTrattate,logControllo,names=c("Trattamento",
+ "Controllo"),main="Log-Precipitazioni a confronto")
```

Perchè usiamo una trasformazione logaritmica?
Nella Figura 2.10, in alto, si era notato come i dati originari si concentravano prevalentemente intorno a valori bassi del *range* delle precipitazioni, mentre i restanti pochi valori erano relativamente alti e dispersi. Tale comportamento si traduceva, tra l'altro, in una pendenza molto marcata della curva dei quantili in corrispondenza ai valori bassi, con un conseguente appiattimento in corrispondenza ai valori più elevati.
Parallelamente, spostando la nostra attenzione alla rappresentazione coi diagrammi a scatola, a sinistra della Figura 2.11, risulta dinuovo evidente l'accentuata asimmetria positiva della distribuzione (i.e. a coda destra lunga). Infatti, sia per il gruppo trattato che per quello di controllo, tanto la sezione

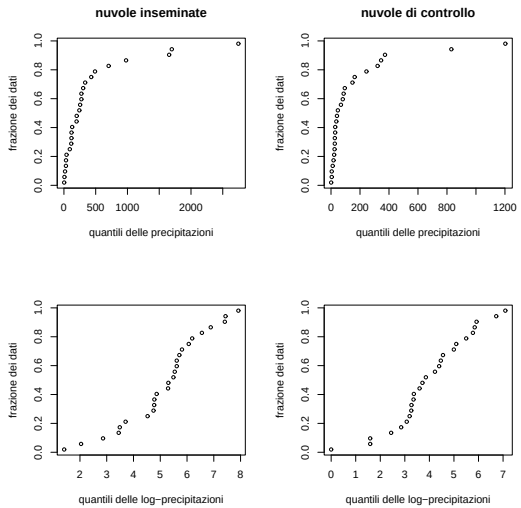


Figura 2.10: Diagrammi dei quantili per i dati delle nuvole.

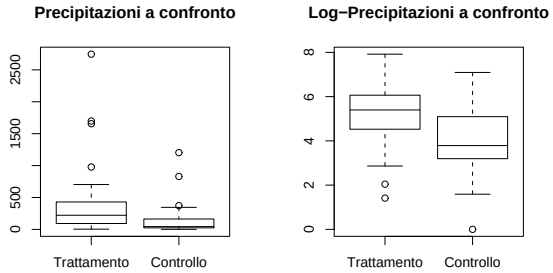


Figura 2.11: Boxplots per i dati delle nuvole.

della scatola quanto il baffo superiori sono più estesi in lunghezza delle analoghe parti inferiori. In aggiunta, possiamo notare come ad una media delle precipitazioni più alta nel gruppo trattato corrisponda anche una maggiore dispersione rispetto al gruppo di controllo.
Allora, per rendere la distribuzione dei dati più simmetrica, e con una dispersione simile tra i gruppi, si applica generalmente la trasformazione logaritmica. Il logaritmo ha l'effetto di ridurre i numeri osservati a numeri molto più piccoli; tuttavia l'azione di ridimensionamento delle cifre operata dal logaritmo non è omogenea ed è tanto più marcata quanto più grande è il valore a cui essa è applicata (riduce di poco i volori piccoli e di tantissimo i valori molto grandi). Si usa quindi tutte le volte che su un grafico si vogliano rappresentare valori che hanno ordini di grandezza molto diversi tra loro.

2.2.2 Il grafico dei quantili contro quantili per confrontare gruppi di dati

Un altro metodo efficace per confrontare la distribuzione di due insiemi di dati è il grafico dei quantili contro quantili detto anche *Q-Q plot* o diagramma quantile-quantile. Esso si costruisce contrapponendo ai quantili $Q_x(p)$ di un insieme di dati x_i, $i = 1, \ldots, n$, i quantili corrispondenti $Q_y(p)$ dell'insieme di dati y_i, $i = 1, \ldots, m$, da confrontare.
Nell'esempio delle nuvole, il gruppo di controllo ha lo stesso numero di osservazioni del gruppo trattato ($n = m$). Quindi, il *Q-Q plot* non è altro che il grafico dei dati ordinati del primo gruppo contro i dati ordinati del secondo. Si ricordi, infatti, che i quantili $Q(p_i)$ per $p_i = (i - 0.5)/n$, sono gli stessi dati ordinati. La Figura 2.12 riporta il grafico ed è prodotta dai comandi seguenti:

```
> plot(sort(Controllo),sort(Trattamento),
+ xlab="Precipitazioni da nuvole di controllo",
+ ylab="Precipitazioni da nuvole trattate",
+ main="Diagramma Quantile-Quantile")
> abline(a=0,b=1)
```

Il comando `abline(a=0, b=1)` aggiunge ad un grafico già presente nella finestra grafica attiva, una linea retta con intercetta e coefficiente angolare dato. Si è così aggiunta al grafico la bisettrice del $I^o - III^o$ quadrante ($y = x$) per meglio analizzare in che cosa le due distribuzioni sono differenti. Infatti,

se le due distribuzioni fossero identiche, tutti i punti starebbero su questa retta. Invece, in Figura 2.12, quasi tutti i punti sono sopra la retta $y = x$, e quanto più elevati sono i valori tanto più distanti sono i punti da questa. Tale configurazione suggerisce dinuovo di considerare i logaritmi delle misure. La Figura 2.13 mostra il *Q-Q plot* dei dati trasformati ed è prodotta dai comandi seguenti:

```
> plot(sort(logControllo),sort(logTrattate),
+ xlab="log-Precipitazioni da nuvole di controllo",
+ ylab="log-Precipitazioni da nuvole trattate")
> abline(a=0,b=1)
```

I punti sono distribuiti ora, diagonalmente, in modo molto più regolare. In effetti, si può immaginare una retta, al di sopra della bisettrice, attorno a cui essi si distribuiscono. Proviamo a disegnarla: ciò che risulta è una retta quasi parallela a $y = x$ (in effetti la pendenza è uguale circa a 0.95) e spostata rispetto ad essa di circa 1.34 unità.
Le argomentazioni sopra ci fanno riflettere che le due distribuzioni (su scala logaritmica) sono in effetti simili nella forma, anche se differiscono per una costante additiva e - in modo trascurabile - per una costante moltiplicativa. Ma cosa significa tutto ciò per i dati su scala originale?
Se, su scala logaritmica, la relazione tra i dati è approssimativamente:

$$log\, y_{(i)} = k \cdot log\, x_{(i)} + c,$$

allora, su scala originale, la relazione diventa:

$$y_{(i)} = x_{(i)}^{k} \cdot e^{c}.$$

La Figura 2.14 sovrappone alla Figura 2.12 la retta di riferimento $y_{(i)} = x_{(i)}^{0.95} \cdot e^{1.342}$.
Il comando in **R** per aggiungere la retta sul grafico quantile-quantile su scala logaritmica è il seguente:

```
> abline(a=1.34,b=.95,lty=2)
```

quello per aggiungere la retta sul grafico in scala originaria è invece:

```
> x<-seq(0,1300,by=1)
> lines(x,x^.95*exp(1.342),lty=2)
```

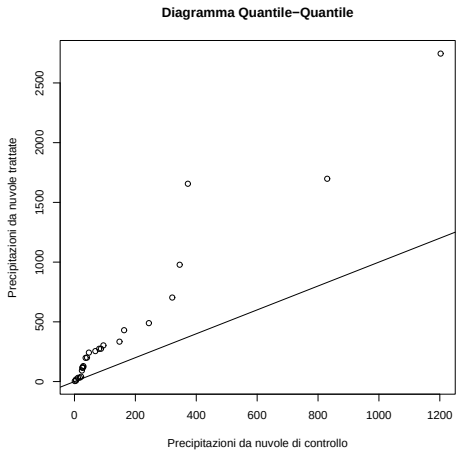



Figura 2.12: Grafico quantile-quantile delle nuvole.

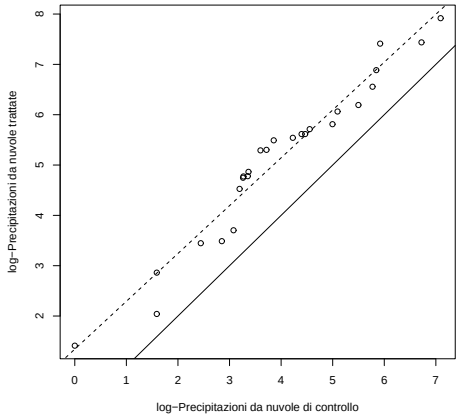


Figura 2.13: Grafico quantile-quantile delle (log-)precipitazioni.

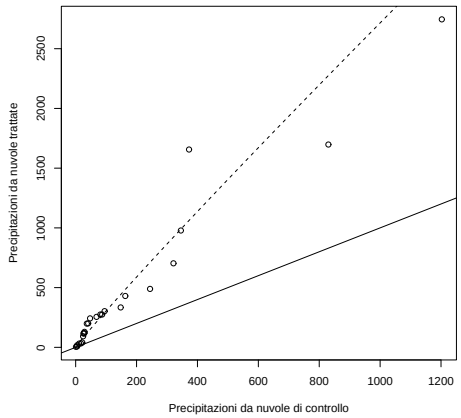


Figura 2.14: La retta di riferimento per i punti nella scala originale.

2.2.3 Il grafico dei quantili contro quantili teorici per confrontare i dati osservati con un modello teorico

La rappresentazione grafica appena vista è utile anche nel caso si voglia confrontare la distribuzione di un insieme di dati osservati con quella di un modello teorico di riferimento. In particolare, se la distribuzione teorica che interessa è la Normale, la domanda che ci si pone è: 'i dati presentano le caratteristiche distributive che sono conformi al modello di riferimento Normale?', ovvero, 'la distribuzione empirica dei dati può essere approssimata abbastanza bene da una distribuzione normale (per gli obiettivi della nostra analisi)?'
Per disegnare un *Normal Q-Q plot* (o *Normal probability plot*), basta pensare di contrapporre i quantili empirici, $Q_e(p_i) = y_{(i)}$, ai quantili teorici della Normale, $Q_t(p_i) = F_N^{-1}(p_i)$, rispetto alle frazioni p_i dei dati.
Di seguito sono riportati i comandi per generare (i) il - già noto - diagramma dei quantili empirici delle log-precipitazioni (dalle nuvole inseminate), (ii) il diagramma dei quantili teorici di una Normale standard mediante la funzione qnorm che calcola i valori $Q_t(p_i) = F_N^{-1}(p_i)$ cercati.

Si veda l'*help* di (dnorm), pnorm, qnorm, (rnorm) per avere maggiori dettagli sui calcoli che **R** permette di fare internamente con la distribuzione Normale. In particolare, si confrontino i valori sulle tavole della Normale (che si trovano nei testi di statistica) con quelli calcolati in **R** .

```
> n<-nrow(nuvole)
> par(mfrow=c(2,1),pty='s')
# (i)
> plot(p.quantili(n),sort(logTrattate),xlab='Frazione p',
+ ylab='Quantile empirico delle log-precipitazioni')
# (ii)
> plot(qnorm(p.quantili(n)),type='l',xlab='Probabilita p',
+ ylab='Quantile teorico della Normale standard')
```

Il *Normal Q-Q plot* si ottiene quindi dando il comando:

```
> plot(qnorm(p.quantili(n)),sort(logTrattate),
+ xlab='Quantile teorico della Normale standard',
+ ylab='Quantile empirico delle log-precipitazioni',
+ main='Diagramma quantile-quantile Normale')
```

Per verificare 'se i dati si distribuiscano normalmente', il metodo comunemente usato consiste nel tracciare la retta passante per la coppia di punti dati dai quartili della distribuzione teorica e di quella osservata, (I^t,I^o) e (III^t,III^o), e vedere come i punti del *Normal Q-Q plot* si distribuiscano attorno ad essa. Se i punti del grafico non si discostano in modo sensibile da tale retta allora abbiamo un'indicazione che i dati sono conformi al modello teorico. Si tratta di un metodo per valutare graficamente alcune caratteristiche dei dati, e la valutazione di scostamenti da un modello teorico non è agevole (dobbiamo comunque aspettarci qualche scostamento se i dati sono pochi) e richiede qualche esperienza. La statistica fornisce anche metodi formali per valutare l'adattamento ad un modello teorico (ancch'essi basati sulla differenza fra quantili empiri e teorici). Lo studio di questi va ben oltre gli scopi di questa trattazione.
I comandi in **R** per trovare i coefficienti della retta passante per i due punti individuati dai primi e terzi quartili sono i seguenti:

```
quantile(logTrattate,prob=c(.25,.75))->q.nu
quantile(qnorm(p.quantili(n)),prob=c(.25,.75))->q.no
```

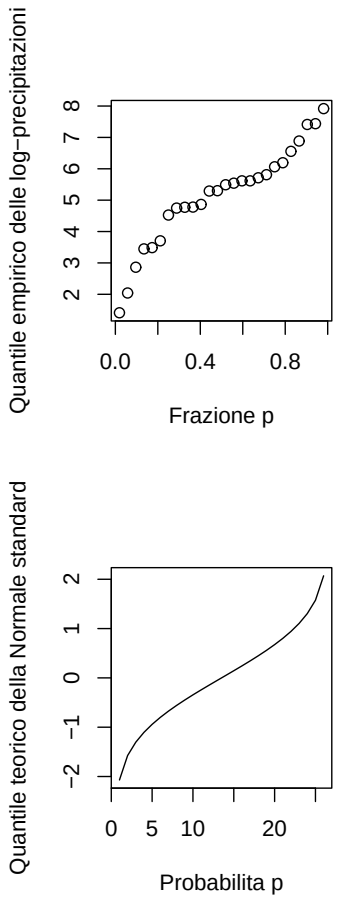


Figura 2.15: Quantili empirici e quantili teorici a confronto.

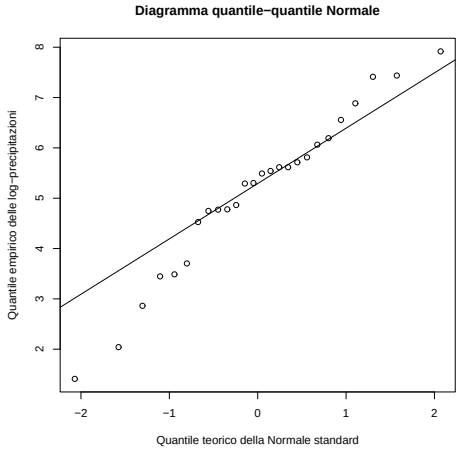


Figura 2.16: Diagramma quantile-quantile Normale.

```
>
b.qn<-diff(q.nu)/diff(q.no)
a.qn<-q.nu[1]-b.qn*q.no[1]
# e per tracciare la retta:
abline(a.qn,b.qn)
```

La Figura 2.16 mostra il diagramma così ottenuto. Lo stesso grafico si poteva produrre con le *routines* qqnorm e qqline già definite in **R**

```
qqnorm(logTrattate)
qqline(logTrattate)
```

2.2.4 Esercizi

1. Disegnare i diagrammi quantili-quantili per il `peso` distinto per maschi e femmine dei dati sui neonati alla sezione precedente, e commentarli. Nel caso i dati da confrontare non abbiano uguale numerosità, si può usare la *routine* qqplot già contenuta in **R** (per il suo utilizzo, si veda l'*help* relativo).

Capitolo 3

L'analisi di regressione semplice

3.1 Le sigarette più forti sono anche più dannose?

Il primo insieme di dati che utilizziamo riguarda le caratteristiche di alcuni tipi di sigarette. L'interesse dell'analisi consiste nello stabilire se sigarette più forti, ovvero con maggiore quantità di condensato (catrame), producano più monossido di carbonio (è infatti questa la sostanza che fa male alla salute quando fumiamo). Cioè se sul pacchetto troviamo scritto che c'è più catrame dobbiamo essere più preoccupati per la salute?

Carichiamo i dati contenuti nel *file* `sigar.txt`. Esso è costituito da cinque colonne, e quelle dalla seconda alla quinta sono relative rispettivamente alle seguenti variabili: il condensato (contenuto di catrame in mg), il contenuto di nicotina, il peso (gr) ed il contenuto di monossido di carbonio (mg) per sigaretta, per 25 diverse marche di sigarette (il nome della marca è nella prima colonna).

```
> sigar<-read.table("sigar.txt")
> sigar
                 V1   V2   V3     V4   V5
1            Alpine 14.1 0.86 0.9853 13.6
2     Benson&Hedges 16.0 1.06 1.0938 16.6
3        BullDurham 29.8 2.03 1.1650 23.5
4       CamelLights  8.0 0.67 0.9280 10.2
```



```
5           Carlton  4.1 0.40 0.9462  5.4
6        Chesterfield 15.0 1.04 0.8885 15.0
7        GoldenLights  8.8 0.76 1.0267  9.0
8                Kent 12.4 0.95 0.9225 12.3
9                Kool 16.6 1.12 0.9372 16.3
10                L&M 14.9 1.02 0.8858 15.4
11         LarkLights 13.7 1.01 0.9643 13.0
12           Marlboro 15.1 0.90 0.9316 14.4
13              Merit  7.8 0.57 0.9705 10.0
14        MultiFilter 11.4 0.78 1.1240 10.2
15      NewportLights  9.0 0.74 0.8517  9.5
16                Now  1.0 0.13 0.7851  1.5
17           OldGold 17.0 1.26 0.9186 18.5
18      PallMallLight 12.8 1.08 1.0395 12.6
19            Raleigh 15.8 0.96 0.9573 17.5
20         SalemUltra  4.5 0.42 0.9106  4.9
21           Tareyton 14.5 1.01 1.0070 15.9
22               True  7.3 0.61 0.9806  8.5
23 ViceroyRichLight  8.6 0.69 0.9693 10.6
24      VirginiaSlims 15.2 1.02 0.9496 13.9
25      WinstonLights 12.0 0.82 1.1184 14.9
```

Vogliamo verificare se è possibile esprimere sinteticamente e in modo efficace la relazione che lega i valori medi del monossido di carbonio (variabile dipendente) ed il condensato (variabile esplicativa). Per semplificare le successive espressioni assegniamo i dati sul condensato e quelli sul monossido di carbonio a due vettori distinti.

```
> condens<-sigar[,2]
> monoss<-sigar[,5]
```

Per dare un primo sguardo alla natura della relazione tra le due variabili, costruiamo un diagramma di dispersione o nuvola di punti.

```
> plot(condens,monoss,xlab="Condensato",ylab="Monossido",
+ main="Grafico Monossido Condensato")
```

Come si vede dal grafico in Figura 3.1, tra le variabili sembra esserci una relazione lineare abbastanza marcata. Si noti anche la presenza di una unità

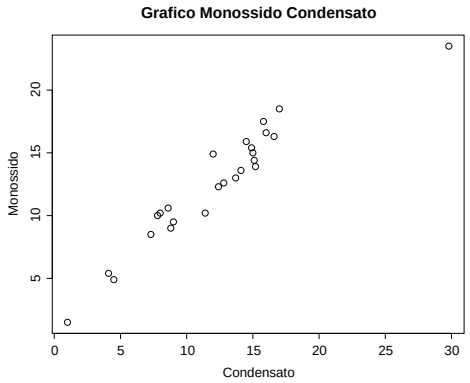


Figura 3.1: Diagramma di dispersione per le variabili monossido e condensato.

che presenta valori delle due variabili che la collocano al di fuori della nuvola in cui stanno invece la maggior parte dei punti.
Una retta dovrebbe quindi rappresentare bene la relazione tra le due variabili. Occorre pertanto determinare i parametri a e b della retta di regressione:

$$monossido = a + b\ condensato$$

È noto che i parametri possono essere determinati utilizzando il metodo dei minimi quadrati. Se chiamiamo x la variabile esplicativa ed y la variabile dipendente i coefficienti $\hat{a}$ e $\hat{b}$ della retta dei minimi quadrati risultano pari a

$$\hat{b} = \frac{\mathrm{Cov}(x,y)}{\mathrm{Var}(x)} \qquad \text{e} \qquad \hat{a} = \mathrm{M}(y) - \hat{b}\ M(x) \tag{3.1}$$

Nota Nella presente dispensa, a scopo didattico, verranno eseguiti i singoli passaggi che in **R** consentono di ottenere le quantità d'interesse. In realtà, **R** contiene procedure sofisticate e complete per condurre analisi di regressione, come ad esempio la funzione `lm()`; tuttavia l'approfondimento di tale funzione va al di là dei nostri scopi. Il lettore interessato può consultare, per iniziare, il manuale contenuto nel pacchetto: *An Introduction to R. Notes on*

R *: A Programming Environment for Data Analysis and Graphics.*

È necessario quindi calcolare le medie delle due variabili, la varianza della variabile indipendente e la covarianza fra le due variabili.
Calcoliamo dapprima le due medie:

```
> n<-length(condens)
> media.condens<-sum(condens)/n
> media.condens
[1] 12.216
>
> media.monoss<-sum(monoss)/n
> media.monoss
[1] 12.528
```

Si noti che è comunque più agevole utilizzare la funzione `mean()` per il calcolo della media di un vettore di dati.
Calcoliamo ora le varianze di entrambe le variabili:

```
> var.condens<-sum(condens^2)/n - media.condens^2
> var.condens
[1] 30.81734
>
> var.monoss<-sum(monoss^2)/n - media.monoss^2
> var.monoss
[1] 21.56602
```

Potrebbe inoltre risultare più agevole calcolare le varianze mediante la funzione `var()`, tuttavia occorre ricordare che tale funzione calcola la somma dei quadrati degli scarti dalla media, divisa per $n-1$ e quindi tenerne, se necessario, conto.
L'ultima cosa che resta da calcolare è la covarianza:

```
> covar<-sum((condens-media.condens)*(monoss-media.monoss))/n
> covar
[1] 24.68395
```

Anche per calcolare la covarianza fra due variabile è possibile usare la funzione `cov(variabile1,variabile2)`, ma anche in questo caso il denominatore della covarianza, è pari a $n-1$.
Facendo adesso riferimento alla equazione 3.1, troviamo:

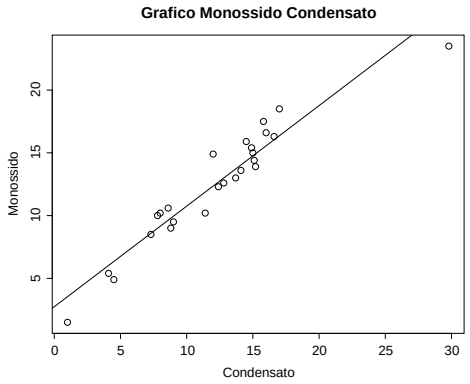


Figura 3.2: Retta di regressione del monossido sul condensato.

```
> b<-covar/var.condens
> b
[1] 0.800976
>
> a<-media.monoss - b * media.condens
> a
[1] 2.743278
```

La funzione `abline(a,b)` aggiunge al grafico corrente una retta con intercetta `a` e coefficiente angolare `b`.

```
> abline(a,b)
```

Il risultato è mostrato in Figura 3.2.
Calcoliamo adesso i residui della regressione, cioè la differenza tra i valori osservati e i valori previsti dalla retta (visualizziamo, per brevità solo i primi 5).

```
> prev.monoss<-a + b * condens
> residui<-monoss - prev.monoss
>
```



```
> residui[1:5]
[1] -0.4370387  1.0411069 -3.1123615  1.0489147 -0.6272790
```

Come si vede, i residui assumono valori positivi e negativi (si ricordi infatti che la somma degli stessi nel caso della retta dei minimi quadrati è pari a 0). Si noti anche che il residuo corrispondente alla terza osservazione è, in valore assoluto, piuttosto elevato. Questo residuo è riferito al punto che avevamo già visto essere al di fuori della nuvola.
In **R** è possibile anche utilizzare dei grafici dinamici. Appartiene a questa classe il comando `identify()`. Esso permette in corrispondenza al diagramma di dispersione sulla finestra attiva di evidenziare a quale elemento del vettore di dati si riferisca un determinato punto sul grafico. Se, inoltre, ad ogni unità è associato un nome questo può essere visualizzato. Lo studente provi ad eseguire, nel caso in esame, i seguenti comandi:

```
> labels<-sigar[,1]
> identify(condens,monoss,labels)
```

a questo punto 'cliccando' con il tasto sinistro del *mouse* a fianco di un punto apparirà il nome della marca cui si riferisce. Per uscire da tale processo si 'clicchi' il tasto destro e si scelga `stop`. Sulla *console* apparirà l'elenco dei punti selezionati nella fase dinamica.
Per valutare la qualità della rappresentazione offerta dalla retta di regressione può essere utile tracciare il grafico dei residui contro la variabile esplicativa.

```
> plot(condens,residui,xlab="Condensato",ylab="Residui",
+ main="Grafico Monossido Residui")
```

Il grafico è riportato in Figura 3.3 e ancora una volta si nota il punto in alto a destra nel diagramma di dispersione cui corrispondere un valore elevato (e negativo) del residuo. L'andamento dei residui non è in defintiva tale da non consigliare ulteriori riflessioni ed analisi: si noti che per valori bassi del condensato si ottengono residui in prevalenza negativi, mentre valori del condensato attorno alla media hanno in prevalenza residui alti. Insomma si può proporre un approfondimento dell'analisi di regressione che tenti di cogliere quelle regolarità ancora presenti nell'andamento dei residui (se l'analisi di regressione lineare fosse del tutto soddisfacente i residui non dovrebbero mostrare 'regolarità' degne di interesse).

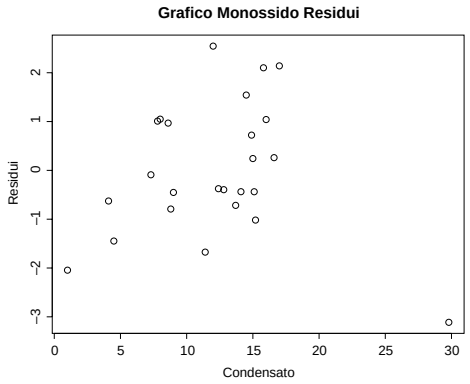


Figura 3.3: Residui della regressione lineare contro i valori della variabile esplicativa

Esercizi

1. Lo studente calcoli il valore di R^2 e le varie componenti della varianza della variabile dipendente.

2. Se R^2 risulta elevato (ad esempio, attorno a 0,9) posso senz'altro affermare che la regressione lineare è del tutto soddisfacente e che ulteriori analisi sono inutili?

3. Si provi a proporre la stessa analisi utilizzando come esplicativa la nicotina.

4. È vero che le sigarette che hanno maggior contenuto di nicotina hanno anche più catrame?

5. Spesso risulta utile verificare se i residui della regressione si conformino a qualche modello distributivo. In particolare di frequente è ragionevole attendersi che essi, potendo essere interpretati come variazioni accidentali attorno ad un andamento medio, siano ben descritti dalla classica curva gaussiana. Si valuti con un qq-plot se in questo caso i residui siano conformi ad un modello gaussiano.

◇

3.2 Che rendimento hanno le centrali eoliche?

Consideriamo ora un secondo esempio (tratto da *Introduction to linear regression analysis*, Montgomery e Peck, 1992) che si riferisce al caso in cui la curva di regressione (che immaginiamo descriva come varia la media della variabile dipendente condizionatamente ai valori della variabile esplicativa) non possa essere efficacemente rappresentata da una retta. Tuttavia nel caso in questione vedremo che è possibile trasformare mediante funzioni non lineari le variabili coinvolte in modo che la funzione di regressione per le variabili trasformate sia ben approssimata da una retta.
Questo equivale a dire che il legame fra la variabile dipendente (o meglio fra le medie condizionate della variabile dipendente) y e la variabile esplicativa x si possa scrivere

$$g(y) = a + bh(x)$$

per opportune trasformazioni $g(\cdot)$ e $h(\cdot)$.

L'insieme di dati che ci accingiamo a studiare riguarda l'energia elettrica prodotta Y (misurata in un'opportuna unità di misura) da un mulino a vento e supponiamo che questa possa essere prevista o spiegata dalla variabile esplicativa X velocità del vento (in nodi orari). Ovviamente, ci si aspetta che quando c'è più vento si abbia anche una maggiore produzione di energia elettrica. Sarebbe interessante rispondere tuttavia a quesiti più precisi, ad esempio: (a) se il vento aumenta la sua velocità di 1 nodo all'ora, di quanto aumenterà in media la produzione di energia elettrica? (b) Se fosse vero che al crescere della velocità aumenta l'energia prodotta, allora Trieste, che può contare occasionalmente sulla bora sarebbe un posto idoneo in cui posizionare centrali di produzione eolica?
Immaginiamo che i dati siano nel *file* `vento.txt`.

```
> vento<-read.table("vento.txt")
> vento
      V1    V2
1   5.00 1.582
2   6.00 1.822
3   3.40 1.057
```

```
4   2.70 0.500
5  10.00 2.236
6   9.70 2.386
7   9.55 2.294
8   3.05 0.558
9   8.15 2.166
10  6.20 1.866
11  2.90 0.653
12  6.35 1.930
13  4.60 1.562
14  5.80 1.737
15  7.40 2.088
16  3.60 1.137
17  7.85 2.179
18  8.80 2.112
19  7.00 1.800
20  5.45 1.501
21  9.10 2.303
22 10.20 2.310
23  4.10 1.194
24  3.95 1.144
25  2.45 0.123
```

Salviamo su due vettori distinti la velocità e la corrente prodotta.

```
> vel<-vento[,1]
> dc<-vento[,2]
```

Per renderci conto del tipo di relazione tra le due variabili, costruiamo, al solito, il diagramma di dispersione della corrente prodotta contro la velocità del vento.

```
> plot(vel,dc,xlab="Velocita' del vento",
+ ylab="Intensita' della corrente",
+ main="Grafico Velocita' Intensita'")
```

Il grafico è riportato nella Figura 3.4. La figura mostra chiaramente che in questo caso non risulterebbe opportuno ipotizzare una relazione lineare tra le due variabili. Tuttavia è sempre possibile calcolare i parametri a e b della retta di regressione dei minimi quadrati, corrispondente alla seguente forma:

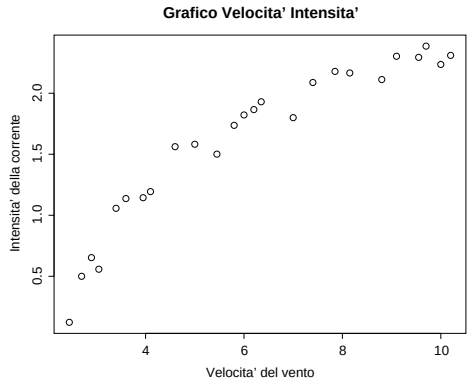


Figura 3.4: Diagramma di dispersione della corrente prodotta e della velocità del vento.

corrente = a + b velocià

Calcoliamo le medie delle due variabili:

```
> media.vel<-mean(vel)
> media.vel
[1] 6.132
>
> media.dc<-mean(dc)
> media.dc
[1] 1.6096
```

Le varianze si possono determinare nel modo che segue (utilizzando stavolta la funzione `var()`):

```
> n<-length(vel)
> n
[1] 25
>
> var.vel<-var(vel)*(n-1)/n
> var.vel
```

```
[1] 6.142176
>
> var.dc<-var(dc)*(n-1)/n
> var.dc
[1] 0.4084475
```

La covarianza tra la variabile dipendente e la variabile esplicativa è:

```
> covar<-sum((vel-media.vel)*(dc-media.dc))/n
> covar
[1] 1.481179
```

Siamo ora in grado di calcolare i due parametri della retta di regressione 3.2.

```
> b<-covar/var.vel
> b
[1] 0.2411489
>
> a<-media.dc - b * media.vel
> a
[1] 0.1308751
```

Calcoliamo ora il valore di R^2 per la retta di regressione. Un modo per calcolarlo è quello di calcolare la varianza spiegata e per questo sarà necessario ottenere i valori previsti dalla retta di regressione.

```
> prev.dc<-a + b * vel
```

La varianza spiegata è la varianza dei valori previsti e, essendo la media dei valori previsti pari alla media della variabile dipendente, abbiamo:

```
> var.spiegata<-sum((prev.dc - media.dc)^2)/n
> var.spiegata
[1] 0.3571846
```

L'indice di determinazione R^2 è il rapporto tra la varianza spiegata e la varianza totale, dove quest'ultima è la varianza della variabile dipendente y:

```
> R2<-var.spiegata/var.dc
> R2
[1] 0.8744932
```


Lo studente può provare a verificare che il calcolo di R^2 utilizzando altre espressioni alternative conduce allo stesso risultato. Si provi a tal fine a calcolare direttamente la varianza dei residui, oppure a calcolare la varianza spiegata (dalla regressione lineare) come $cov(x, y)^2/var(x)$.
Si noti infine che nel caso di regressione lineare semplice l'indice di determinazione R^2 coincide con il coefficiente di correlazione al quadrato:

```
> r<-covar/sqrt(var.dc * var.vel)
> r^2
[1] 0.8744932
```

La retta di regressione spiega l'87% della varianza totale e tale valore sembra piuttosto elevato, tuttavia si ricordi che non è sufficiente guardare al valore di R^2 per ritenere soddisfacente il risultato dell'analisi: è opportuno guardare quantomeno l'andamento dei residui. Calcoliamo quindi i residui e tracciamone un grafico contro la variabile esplicativa.

```
> res<-dc - prev.dc
>
> plot(vel,res,xlab="Velocita' del vento",
+ ylab="Residui sulla regr. lineare",
+ main="Grafico Velocita' Residui")
```

Il grafico è nella Figura 3.5.
Il modello di regressione considerato implicherebbe che vi sia in media una crescita dell'energia prodotta al crescere della velocità del vento. Sulla base delle analisi fin qui svolte si può dire di quanto aumenti in media l'energia prodotta se la velocità cresce di un nodo all'ora? E la risposta data è ragionevole?
I residui hanno tuttavia un andamento niente affatto 'casuale' e privo di sistematicità, e mettono in evidenza che la relazione tra l'intensità della corrente prodotta e la velocità del vento non è lineare (cosa che peraltro avevamo sospettato fin dall'inizio). Questo ci spinge a cercare un modello non lineare. Ritornando al diagramma di dispersione 3.4 si può notare che, al crescere della velocità del vento, l'effetto di variazioni nella velocità del vento su variazioni della corrente prodotta sembra ridursi e inoltre la variabile dipendente sembra raggiungere un asintoto. Se così fosse, utilizzare il generatore in presenza di venti estremamente forti non darebbe particolari vantaggi perchè oltre una certa velocità il rendimento del generatore decresce. Se a venti forti

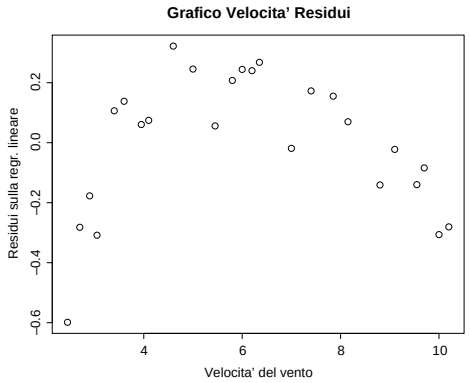


Figura 3.5: Grafico dei residui ottenuti usando un modello lineare contro la velocità del vento.

fosse, ad esempio, associato un aumento del rischio di rotture del generatore, potrebbe non esserci alcun vantaggio a posizionare la centrale in zone dove possono soffiare venti sono molto forti, poichè in quel caso il generatore andrebbe addirittura spento (Trieste con la bora potrebbe non essere il luogo ideale).
In questo caso la relazione fra y e x potrebbe essere resa lineare se utilizzassimo al posto della velocità del vento x il suo reciproco.
Per valutare se tale idea è ragionevole, calcoliamo quindi la nuova variabile reciproco della velocità e tracciamo un grafico di questa'ultima in corrisponedenza della quantità di corrente prodotta.

```
> recvel<-1/vel
>
> plot(recvel,dc,xlab="Reciproco della velocita' del vento",
+ ylab="Intensita' della corrente",
+ main="Grafico Reciproco velocita' Intensita'")
```

Il grafico è mostrato in Figura 3.6, dalla quale si vede come la relazione tra l'intensità della corrente prodotta e il reciproco della velocità del vento sia ben descritta da una retta.

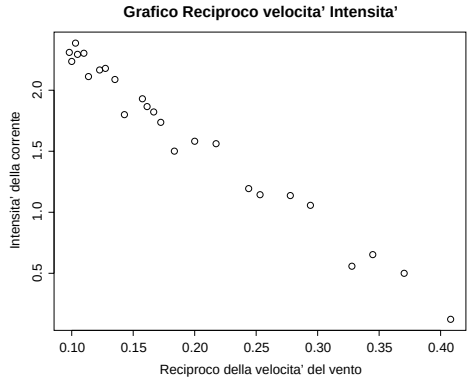


Figura 3.6: Grafico dell'intensità della corrente prodotta rispetto al reciproco della velocità del vento.

Quanto fatto equivale a ipotizzare il seguente modello non lineare:

$$\text{corrente} = a + b\frac{1}{\text{velocità}} \tag{3.2}$$

che è appunto lineare nel reciproco della velocità. Lo studente verifichi che tale modello implica che nella scala originale la corrente prodotta raggiunga un asintoto e valuti a quanto dovrebbe corrispondere quest'ultimo.
Calcoliamo con il solito procedimento i parametri del modello utilizzando la variabile trasformata (mediante reciproco).

```
> media.recvel<-mean(recvel)
> media.recvel
[1] 0.1974549
>
> var.recvel<-var(recvel)*(n-1)/n
> var.recvel
[1] 0.008324097
>
> covar<-sum((recvel-media.recvel)*(dc-media.dc))/n
```

```
> covar
[1] -0.05772384
>
> b<-covar/var.recvel
> b
[1] -6.934547
> a<-media.dc - b * media.recvel
> a
[1] 2.97886
>
```

Calcoliamo R^2 per il nuovo modello ricordando che per la regressione lineare semplice $R^2 = r^2(dc, recvel)$:

```
> r2<-covar^2/(var.dc*var.recvel)
> r2
[1] 0.9800249
```

Il valore di R^2 è adesso molto elevato e l'adattamento del modello sembra eccellente.
Tracciamo infine il grafico dei residui per il modello 3.2 rispetto alla variabile esplicativa di quel modello, cioè il reciproco della velocità del vento.

```
> res<-dc - a + b * recvel
>
> plot(recvel,dc,xlab="Reciproco della velocita' del vento",
+ ylab="Residui",main="Grafico Reciproco velocita' Residui")
```

Il grafico dei residui è presente in Figura 3.7. Questa volta i residui non sembrano avere alcun particolare *pattern* e siamo dunque soddisfatti.
Con le istruzioni che seguono viene costruito il grafico in Figura 3.8, che mostra l'intensità della corrente elettrica prodotta prevista per i valori della velocità del vento più grandi di quelli osservati:

```
> vettx<-seq(2,16,by=0.1)
>
> plot(vel,dc,xlab="Velocita' del vento",
+ ylab="Intensita' della corrente",
+ main="Grafico Reciproco velocita' Intensita'",
+ xlim=c(0,16),ylim=c(0,3))
```

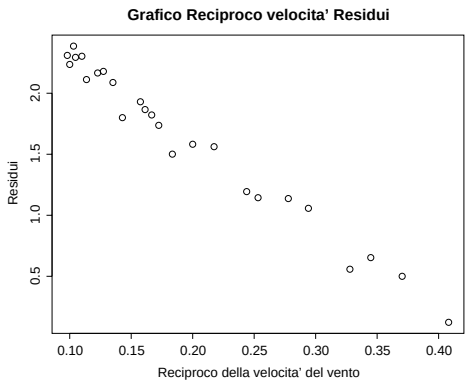



Figura 3.7: Grafico dei residui per il modello lineare nel reciproco della velocità del vento.

```
>
> lines(vettx,a+b*1/vettx)
```

In questo caso è evidente che il modello basato sul reciproco della velocità rispetta la tendenza dell'intensità della corrente elettrica generata a raggiungere un asintoto, proprio come volevamo accadesse.

◇

3.3 Vediamo chi ha più cervello

Introduciamo ora un terzo insieme di dati con l'obiettivo di studiare la relazione esistente tra il peso corporeo medio di un animale adulto e la sua massa cerebrale. Si vuole capire se per governare un peso corporeo elevato sia necessaria una maggiore massa cerebrale. La matrice di dati contiene il nome dell'animale, il peso corporeo (in chilogrammi) e la massa cerebrale (in grammi).
Con il comando `read.table` leggiamo il *file* di dati `animali.txt`:

```
> animali<-read.table("animali.txt")
```

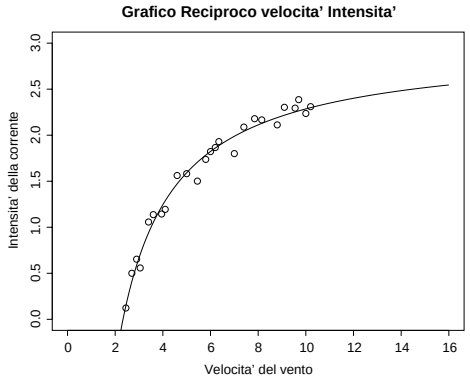


Figura 3.8: Valori previsti dal modello 3.2 per grandi velocità del vento.

```
> animali
                    V1       V2     V3
1  Castoro_di_montagna    1.350    8.1
2                Mucca  465.000  423.0
3          Lupo_grigio   36.300  119.5
4                Capra   27.660  115.0
5     Maiale_di_guinea    1.040    5.5
6    Elefante_asiatico 2547.000 4603.0
7                Asino  187.000  419.0
8              Cavallo  521.000  655.0
9     Scimmia_africana   10.000  115.0
10               Gatto    3.300   25.6
11             Giraffe  529.000  680.0
12             Gorilla  207.000  406.0
13                Uomo   62.000 1320.0
14   Elefante_africano 6654.000 5712.0
15     Scimmia_indiana    6.800  179.0
16             Canguro   35.000   56.0
17             Criceto    0.120    1.0
18                Topo    0.023    0.4
```



```
19            Coniglio    2.500   12.1
20              Pecora   55.500  175.0
21            Giaguaro  100.000  157.0
22           Scimpanze   52.160  440.0
23               Ratto    0.280    1.9
24               Talpa    0.122    3.0
25              Maiale  192.000  180.0
```

Costruiamo due vettori, il primo contenente il peso corporeo e il secondo la massa cerebrale:

```
> peso<-animali[,2]
> massa<-animali[,3]
```

Osserviamo il diagramma di dispersione fra le variabili per valutare quale possibile modello potrebbe descrivere la relazione fra le due variabili:

```
> plot(peso,massa,xlab="Peso corporeo", ylab="Massa
cerebrale")
```

Il grafico è riportato in Figura 3.9. Una regressione lineare del tipo:

$$massa = \alpha + \beta peso$$

apparirebbe a prima vista adeguata. Passiamo allora al calcolo dei parametri $\hat{\alpha}$ e $\hat{\beta}$ attraverso il metodo dei minimi quadrati.
Lo studente verifichi, utilizzando le usuali espressioni, che $\hat{\alpha}$ e $\hat{\beta}$ risultano pari a 191.2279 (l'intercetta) e 0.943165 (il coefficiente di regressione lineare). Aggiungiamo al primo grafico la retta di regressione dei minimi quadrati così ottenuta:

```
> plot(peso,massa,xlab="Peso corporeo",ylab="Massa cerebrale",
+ xlim=c(0,600),ylim=c(0,1500))
>
> abline(191.2279,0.943165)
```

La successiva Figura 3.10 rappresenta l'ingrandimento del grafico precedente, in particolare si ha uno zoom dell'angolo in basso a sinistra (in pratica risultano esclusi i due valori relativi agli animali con peso corporeo più elevato). Il valore di $\hat{\beta}$ è positivo e indicherebbe che un corpo più pesante presenta in media una massa cerebrale più elevata. L'intercetta ottenuta sembra tuttavia

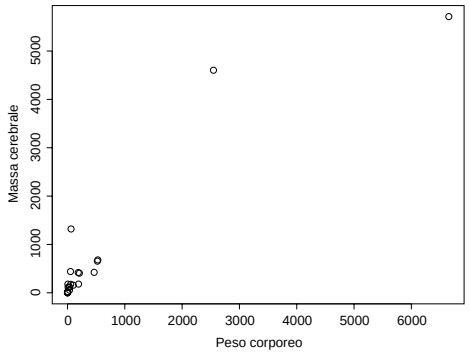


Figura 3.9: Diagramma di dispersione per le variabili peso corporeo e massa cerebrale.

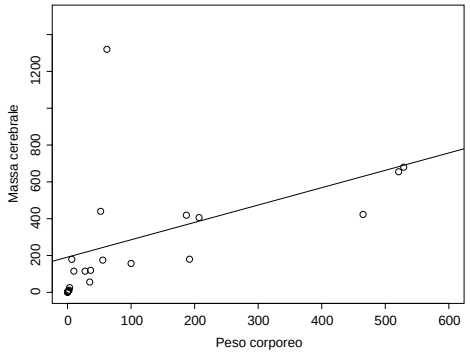


Figura 3.10: Diagramma di dispersione e retta di regressione dei minimi quadrati (esclusi i due punti in alto a destra) .

eccessivamente elevata e i piccoli animali sono rappresentati in modo inadeguato dal modello: infatti pesi corporei al di sotto del kilogrammo avrebbero secondo la curva di regressione ottenuta una massa cerebrale in media poco distante dai 200 grammi. Ad esempio, per il maiale della Guinea che pesa circa un kilogrammo ed ha una massa cerebrale pari 5.5 grammi, la retta ottenuta prevederebbe una massa cerebrale pari a 192 grammi. Questo accade per quasi tutti i valori con basso peso corporeo che infatti sono al disotto della retta di regressione.
Si lascia per esercizio il calcolo dell'indice di determinazione e delle componenti della varianza della variabile dipendente (la massa cerebrale): varianza residua e varianza di regressione (o spiegata).
Il valore di R^2 risulta pari a 0.8683447 e indica che la retta di regressione coglie buona parte della variabilità della massa cerebrale, Le considerazioni fatte sopra portano però a essere più critici nei confronti della retta di regressione ottenuta. Quanto già notato è ancora più evidente se si conduce un'analisi dei residui, con l'obiettivo di verificare che non esistano *patterns* interessanti che, portino a condurre nuove analisi che consentano di ottenere un adattamento ancora migliore.
Possiamo quindi calcolare le masse cerebrali previste

```
massa.prevista<-191.2279+0.943165*massa
> massa.prevista
 [1,]  192.5012
 [2,]  629.7996
 [3,]  225.4648
 [4,]  217.3158
 [5,]  192.2088
 [6,] 2593.4689
 [7,]  367.5997
 [8,]  682.6168
 [9,]  200.6595
[10,]  194.3403
[11,]  690.1621
[12,]  386.4630
[13,]  249.7041
[14,] 6467.0472
[15,]  197.6414
[16,]  224.2387
```

```
[17,]  191.3411
[18,]  191.2496
[19,]  193.5858
[20,]  243.5735
[21,]  285.5444
[22,]  240.4234
[23,]  191.4920
[24,]  191.3429
[25,]  372.3155
```

e quindi i residui (per brevità qui visualizziamo solo i primi 10):

```
> residui_massa-massa.prevista
> residui[1:10]
            [,1]
 [1,] -184.40115
 [2,] -206.79956
 [3,] -105.96477
 [4,] -102.31582
 [5,] -186.70877
 [6,] 2009.53110
 [7,]   51.40028
 [8,]  -27.61680
 [9,]  -85.65953
[10,] -168.74033
```

Il grafico dei residui è in Figura 3.11.

```
> plot(massa.prevista,residui,xlab="Massa cerebrale prevista",
+ ylab="Residui")
```

Il grafico dei residui contro i valori previsti evidenzia l'inadeguatezza del modello per i valori piccoli (dove è marcata la tendenza a sovrastimare la variabile dipendente) e la presenza di alcuni residui elevati (che invece indicano una sottostima), il primo in corrispondenza dell'elefante asiatico, il secondo riferito all'uomo dove, il valore previsto è pari a 250 grammi contro i 1320 osservati. Il grafico mostra quindi ancora che la funzione di regressione lineare non è del tutto soddisfacente.
Come già notato, gli animali con peso corporeo piccolo sono mal rappresentati, e da un'analisi dei grafici proposti si nota che anche gli animali con un peso

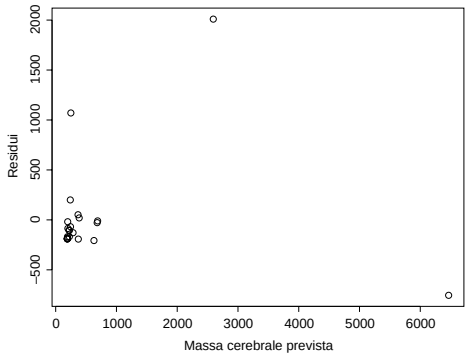


Figura 3.11: Diagramma di dispersione residui-valori previsti dallaa retta di regressione.

corporeo elevato non sono ben rappresentati dalla retta di regressione. Si può quindi desumere che la relazione fra massa e peso sarebbe colta meglio da una funzione non lineare (questo poteva essere già colto da un'attenta osservazione dello scatterplot iniziale, ma spesso aiuta molto a valutare la presenza di relazioni non lineari un altro grafico che mostri i residui in corrispondenza dei valori della variabile x, lo studenti provi a tracciarlo per esercizio). Si può anche osservare che le due variabili analizzate mostrano entrambe una forte asimmetria positiva (tanti valori piccoli e pochi molto elevati). Si può quindi tentare di valutare se l'analisi condotta con nuove variabili ottenute mediante trasformazioni non lineari delle masse e dei pesi possa essere soddisfacente.
Può essere a tal fine interessante provare ad analizzare i nuovi dati che si ottengono dopo una trasformazione di entrambe le variabili su scala logaritmica (utilizziamo in questo caso la base 10, ma nella sostanza potremmo ottenere risultati analoghi riguardo l'effetto linearizzante della trasformazione, se si utilizzasse il logaritmo in base e. Lo studente lo verifichi come esercizio). Calcoliamo le nuove variabili (ricordiamo che in **R** `log10` permette di calcolare il logaritmo in base 10 mentre `log` è il logaritmo in base e):

```
> logpeso<-log10(peso)
```

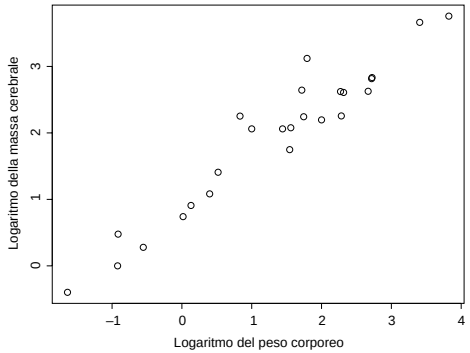


Figura 3.12: diagramma di dispersione per le due variabili trasformate mediante logaritmo

```
> logmassa<-log10(massa)
```

e rappresentiamole graficamente:

```
> plot(logpeso,logmassa,xlab="Logaritmo del peso corporeo",
+ ylab="Logaritmo della massa cerebrale")
```

Il grafico è riportato in Figura 3.12. Risulta ora evidente la possibilità di adattare ai dati una retta di regressione. Lo studente può verificare che i due parametri, intercetta e coefficiente di regressione, calcolati con il metodo dei minimi quadrati per le variabili trasformate, sono pari rispettivamente a 0,933923 e 0,752266.
Aggiungiamo ora al grafico precedente la retta di regressione:

```
> plot(logpeso,logmassa,xlab="Logaritmo del peso corporeo",
+ ylab="Logaritmo della massa cerebrale")
>
> abline(0.933923,0.752266)
```

Il grafico è riportato in Figura 3.13.
Calcoliamo ora i valori previsti:

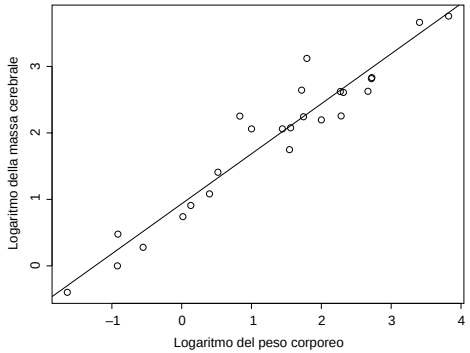


Figura 3.13: .

```
> logmassa.prevista<-0.933923+0.752266*logmassa
> logmassa.prevista
           [,1]
 [1,]  1.0319687
 [2,]  2.9405570
 [3,]  2.1073876
 [4,]  2.0185793
 [5,]  0.9467366
 [6,]  3.4961626
 [7,]  2.6429521
 [8,]  2.9777075
 [9,]  1.6861890
[10,]  1.3239834
[11,]  2.9826860
[12,]  2.6761486
[13,]  2.2822782
[14,]  3.8098980
[15,]  1.5601911
[16,]  2.0954728
[17,]  0.2412224
```

```
[18,] -0.2984934
[19,]  1.2332797
[20,]  2.2460952
[21,]  2.4384549
[22,]  2.2258176
[23,]  0.5180388
[24,]  0.2466226
[25,]  2.6515728
```

e i residui (visualizziamo solo i primi 10):

```
> logres<-logmassa-logmassa.prevista
> logres[1:10]
              [,1]
 [1,] -0.123483640
 [2,] -0.314216658
 [3,] -0.030019733
 [4,]  0.042118545
 [5,] -0.206373914
 [6,]  0.166878381
 [7,] -0.020738058
 [8,] -0.161466207
 [9,]  0.374508890
[10,]  0.084256582
```

Il grafico dei residui (riportato in Figura 3.14) che si ottiene evidenzia l'assenza di particolari strutture e non sono più presenti valori sovrarappresentati dalla retta per *logx* piccolo e valori dei residui molto elevati per *logx* grande:

```
> plot(logmassa.prevista,logres,
+ xlab="Logaritmo della Massa cerebrale prevista",
+ ylab="Residui")
```

Calcoliamo infine R^2 e le varianze:

```
> logvar.totale<-sum((logmassa-mean(logmassa))^2)/
+ length(logmassa)
>
> logvar.residua<-sum(logres^2)/length(logres)
> logvar.spiegata<-logvar.totale-logvar.residua
```

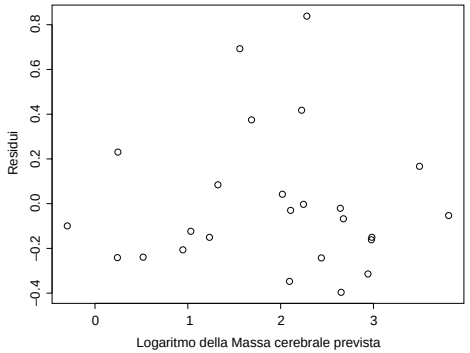



Figura 3.14: Grafico dei residui e dei valori previsti dalla regressione dopo la trasformazione logaritmica.

```
> logr2<-logvar.spiegata/logvar.totale
> logvar.totale
[1] 1.167536
> logvar.residua
[1] 0.09141816
> logvar.spiegata
[1] 1.076118
> logr2
[1] 0.9217
```

Si noti che la retta di regressione che abbiamo appena studiato è pari a:

$$\log_{10} massa = \gamma + \delta \log_{10} peso$$

Se torniamo alle variabili originali, essa corrisponde alla seguente relazione:

$$massa = 10^{\gamma} \times peso^{\delta}$$

che è non lineare ed è immediato notare che a un peso corporeo nullo fa corrispondere una massa cerebrale nulla. Conviene quindi utilizzare quest'ultima formula per calcolare le previsioni dal modello nelle scale delle variabili originali:

```
> massa.prevista.10<-10^logpar[1,] * peso^logpar[2,]
> massa.prevista.10
 [1]   10.763875  872.081404  128.052375  104.370868    8.845790 3134.459006
 [7]  439.493121  949.964786   48.549968   21.085475  960.917194  474.404313
[13]  191.548271 6455.026137   36.323786  124.587024    1.742699    0.502929
[19]   17.111171  176.236239  274.444731  168.196745    3.296392    1.764504
[25]  448.304156
```

I valori previsti per gli animali con un peso notevole e quelli per animali con un peso corporeo molto piccolo sono ora soddisfacenti. In particolare, il valore previsto per il maiale della guinea è pari ora a 8.84 grammi contro i 192.2 previsti dalla retta di regressione usata sulle variabili originali. Il grafico di questa curva di regressione (Figura 3.15) si ottiene come segue:

```
> linea<-seq(min(peso),max(peso),by=100)
> plot(peso,massa,xlab="Peso corporeo",
+ ylab="Massa cerebrale",
+ xlim=c(0,600),ylim=c(0,1500))
>
> lines(linea,10^logpar[1,] * linea^logpar[2,])
```

Si noti infine che la somma dei residui dalla seconda regressione espressi nelle variabili originarie non è zero (poichè le proprietà dei minimi quadrati, fra cui la nullità della somma dei residui, sono valide per i valori nella scala logarirtmica):

```
> residui.10<-massa-massa.prevista.10
sum(residui.10)>
[1] 760.027
```

Quest'ultimo esempio, come anche quello precedente, riguarda un caso in cui esiste una trasformazione linearizzante, ciò non deve indurre a credere che sia sempre possibile ricondurre una relazione non lineare ad una lineare: in numerosi casi le relazioni fra variabili sono intrinsecamente non lineari e occorrono strumenti più sofisticati per condurre adeguate analisi di regressione.

Esercizi

1. Ci si aspetta, in realtà, che l'uomo abbia un cervello con massa maggiore degli altri animali. Questo risulta dall'analisi? E se l'uomo ha veramente un cervello più grande rispetto agli altri mammiferi, di quanto è più grande?

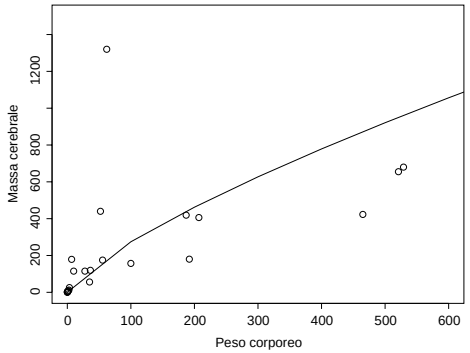


Figura 3.15: .

2. E' vero che i topi sono più intelligenti degli altri animali?

3. Qualcuno sostiene che vi siano sensibili differenze nella relazione massa cerebrale/peso a seconda che si tratti di animali selvatici o di animali addomesticati (tipo cavallo, mucca, capra, etc). Si dividano i dati in due gruppi (selvatici e addomesticati) e si svolgano analisi di regressione separate. Si traggano opportune conclusioni. (Si riportino le due rette di regressione sullo stesso grafico)

Riferimenti bibliografici

Alcuni riferimenti bibliografici utili per approfondire il linguaggio **S** e la sua implementazione **R** sono, oltre al manuale di **R** disponibile assieme al *software* o scaricabile alla pagina *http://www.r-project.org/*

Venables, W.N., Smith, D.M. and the R Development Core Team (2003), *An Introduction to R. Notes on* **R** *: A Programming Environment for Data Analysis and Graphics*, Version 1.8.0 (2003-10-08),

i testi di seguito elencati.

Iacus, S., Masarotto, G. (2003), *Laboratorio di statistica con R*, McGraw-Hill, Milano.

Venables, W.N. e Ripley, B.D. (2002), *Modern Applied Statistics with S-plus*, 4th ed., Springer-Verlag, London.

Venables, W.N. e Ripley, B.D. (2000), *S programming*, 3rd ed., Springer-Verlag, London.

Krause, A. e Olson, M. (2000), *The Basic of S and S-plus*, 2nd ed., Springer-Verlag, New York.

Everitt, B.S. (1994), *A Handbook of Statistical Analyses using S-plus*, Chapman & Hall, New York.

Per rivedere concetti di statistica si consigliano i testi di seguito elencati.

Zani, S. (1994), *Analisi dei dati statistici - I osservazioni in una e due dimensioni*, Giuffrè Editore, Milano.

Chambers, J.M., Cleveland, W.S., Kleiner, B., Tukey, P.A. (1983), *Graphical methods for data analysis*, Wadsworth & Brooks/Cole Publishing Company, Pacific Grove, California.

Montgomery, D.C. e Peck, E.A. (1992), *Introduction to linear regression analysis*, Wiley & Sons, New York.