

La Statistica applicata attraverso l'uso del programma R

Francesca Parpinel

febbraio 2000

1 Introduzione

È difficile pensare allo studio della statistica dei fenomeni reali non associato all'uso del computer, ma spesso, quando si insegna la statistica di base, rivolgiamo la nostra attenzione sostanzialmente alle costruzioni logiche, alle definizioni e ad esercizi teorici spesso slegati dalla realtà fenomenica. Pur riconoscendo l'estrema importanza per uno studente dello studio teorico della Statistica, che non è nostra intenzione perdere di vista ritenendola indispensabile, ci rendiamo conto che l'insegnamento della Statistica non può più prescindere dall'uso di una adeguata strumentazione informatica. Questo anche per permettere a chi ci sta ascoltando di non perdersi nel momento in cui si dovesse trovare di fronte a dei problemi pratici.

Ricordiamo anche che la nostra Università sta vivendo un momento di cambiamento che non deve essere solo nella forma ma soprattutto nell'offerta didattica agli studenti, ai quali deve essere fornita anche una preparazione tecnico-pratica con la quale procedere e da sfruttare come professionalizzante. Purtroppo però, spesso, l'uso del computer, o meglio l'abuso, senza un'appropriata conoscenza critica dello strumento, non consente un'apertura alle effettive capacità di integrare un corso di Statistica per mezzo di applicazioni al computer.

In questo rapporto vengono riportati alcuni esperimenti di applicazione della Statistica teorica condotti in un corso di Statistica di base in un Diploma Universitario (a tale proposito vogliamo ricordare che anche le future lauree triennali necessariamente avranno da imparare riguardo a questi corsi) mediante l'utilizzo di un programma che sta prendendo piede nell'ambito della ricerca statistica la cui caratteristica più interessante è quella di essere di pubblico dominio e quindi completamente gratis, da cui facilmente accessibile anche per gli studenti universitari. Stiamo facendo riferimento al programma R che può essere considerato una riscrittura, con ovvie caratterizzazioni di diversità, del linguaggio S (S è sviluppato dalla AT&T da Rick Becker, John Chambers e Alan Wilks). Chiariamo

sin d'ora che, in accordo con quanto già affermano gli stessi appartenenti al gruppo di sviluppo di R, il programma non è solo un programma statistico, ma un ambiente in cui le applicazioni statistiche hanno avuto maggiore sviluppo per motivi storici (non abbiamo infatti ancora citato i nomi dei padri di R, Ross Ihaka e Robert Gentleman del Dipartimento di Statistica dell'Università di Auckland, ai quali, dal '97, si sono aggiunti molti altri tra cui, ad esempio Brian Ripley, ma anche Guido Masarotto che cura lo sviluppo della versione di R in ambiente Windows).

In effetti, nelle pagine che seguono, alcuni esempi sono delle simulazioni che potrebbero essere effettuate mediante un qualsiasi altro programma. La scelta di R è il risultato di due diverse considerazioni: da una parte l'immediata disponibilità a costo nullo per gli studenti dei nostri corsi, in secondo luogo, l'approccio verso un programma che ha già implementate diverse applicazioni statistiche, pur permettendo ampio margine di operabilità.

2 Il programma R

R è un programma di pubblico dominio per varie piattaforme quali Windows, Linux e Macintosh, le cui informazioni e le versioni più aggiornate si ritrovano al sito:

www.ci.tuwien.ac.at/R¹

Esistono comunque dei *mirror* che spesso però abbiamo notato presentare dei ritardi negli aggiornamenti. Se inoltre un lettore è interessato a scaricare il programma completo nella versione sotto Windows potete considerare l'indirizzo

<http://merlot.dtv.unive.it/~parpinel/>

Ricordiamo che poiché alcune specifiche che riguardano spesso le operazioni di *input* e *output* dipendono dal sistema operativo su cui si sta lavorando (ad esempio nel salvataggio di file grafici) noi qui facciamo esplicito riferimento all'ambiente Windows, la cui installazione è notevolmente semplificata dalla presenza nella distribuzione di un file eseguibile che guida l'aggiornamento del programma. Si può infatti scegliere di installare il programma nella versione completa o solo alcune sue componenti.

Una volta aperto il programma ci si trova di fronte ad una finestra grafica, tipica dell'ambiente Windows, come illustra la figura 1, in cui è aperta un'altra finestra che corrisponde alla *console* di R, in cui, oltre ad alcune informazioni di R rigorosamente in inglese, presenta un *prompt* di sistema del tipo

>

I comandi vengono dati nella forma:

> nome.comando(arg1, arg2, . . .)

¹La versione aggiornata al 13-3-2000 è la 1.0.0

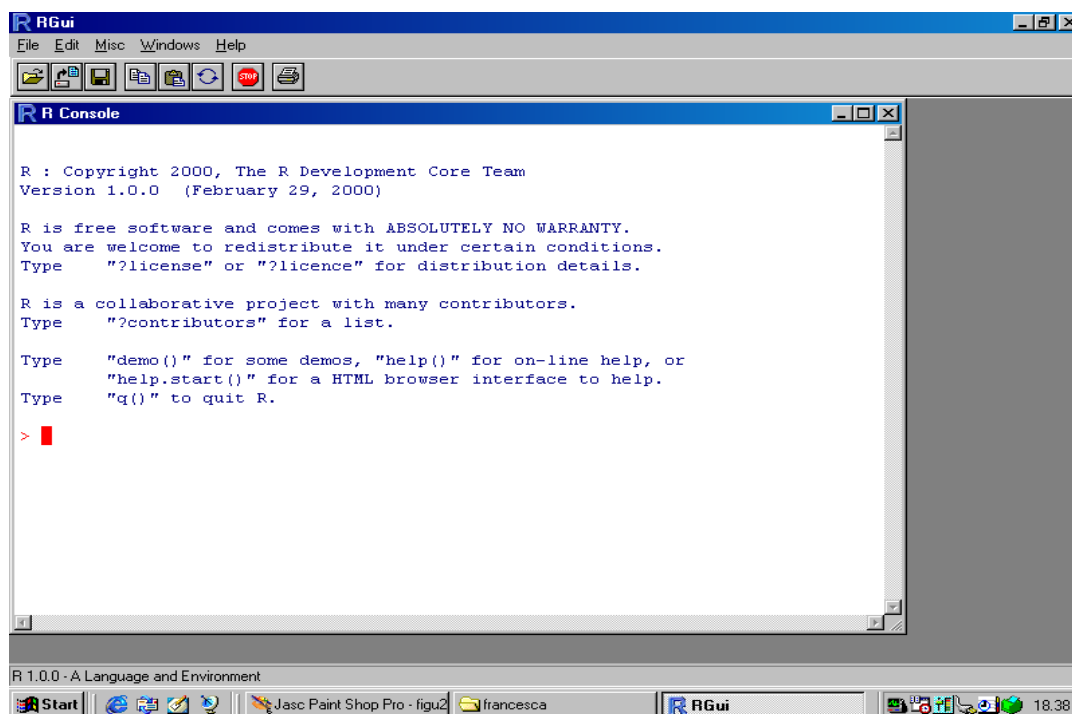


Figura 1: Programma R in ambiente Windows

dove nome . comando è il comando che vogliamo richiamare e arg1 , arg2 , ecc. sono le specificazioni delle variabili.

Per analizzare ciò che è rimasto in memoria del programma è sufficiente digitare

```
> ls()
```

oppure

```
> objects()
```

L'assegnazione di valori alle variabili e di definizione delle variabili numeriche può essere condotta in modi diversi:

```
> variabile<-valore
```

oppure

```
> variabile_valore
```

o ancora

```
> assign(valore,variabile)
```

Per richiamare l'aiuto in linea rispetto a qualche funzione o comando è sufficiente digitare

```
> help(nome.comando)
```

Se si vuole richiamare l'aiuto in formato html, la cui efficienza è stata notevolmente aumentata, è sufficiente richiamare

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

Questo comando necessita la presenza di un *browser*: la nostra esperienza con Netscape (versione 4.7) ha permesso di apprezzare questo tipo di aiuto. Si noti in R l'uso delle parentesi tonde, dimenticandole viene riportata la struttura della funzione richiamata, ad esempio:

```
> help.start
```

produce

```
function(gui = "irrelevant", browser = "irrelevant")
{
  a <- system.file("rwin.html", pkg="doc/html", lib=R.home())
  if (a == "")
    a <- system.file("rwin.htm", pkg="doc/html", lib=R.home())
  if (a == "")
    stop("I can't find the html help\n")
  else {
    a <- gsub("/", "\\\\" , a)
    cat("If nothing happens, you have to open '",a,"' yourself\n")
    .Internal(help.start());
  }
  invisible("")
}
```

Le operazioni elementari seguono le regole del calcolo matematico. Inoltre il programma non fa calcolo simbolico; esistono comunque delle funzioni particolari che permettono di disegnare le funzioni ad una variabile in un sistema di assi cartesiani a due dimensioni: il comando `curve()` è un comando parametrizzato in x e permette di disegnare qualsiasi funzione descritta in x . Ad esempio

```
> curve(x^3*(1-x)^7, 0, 1)
```

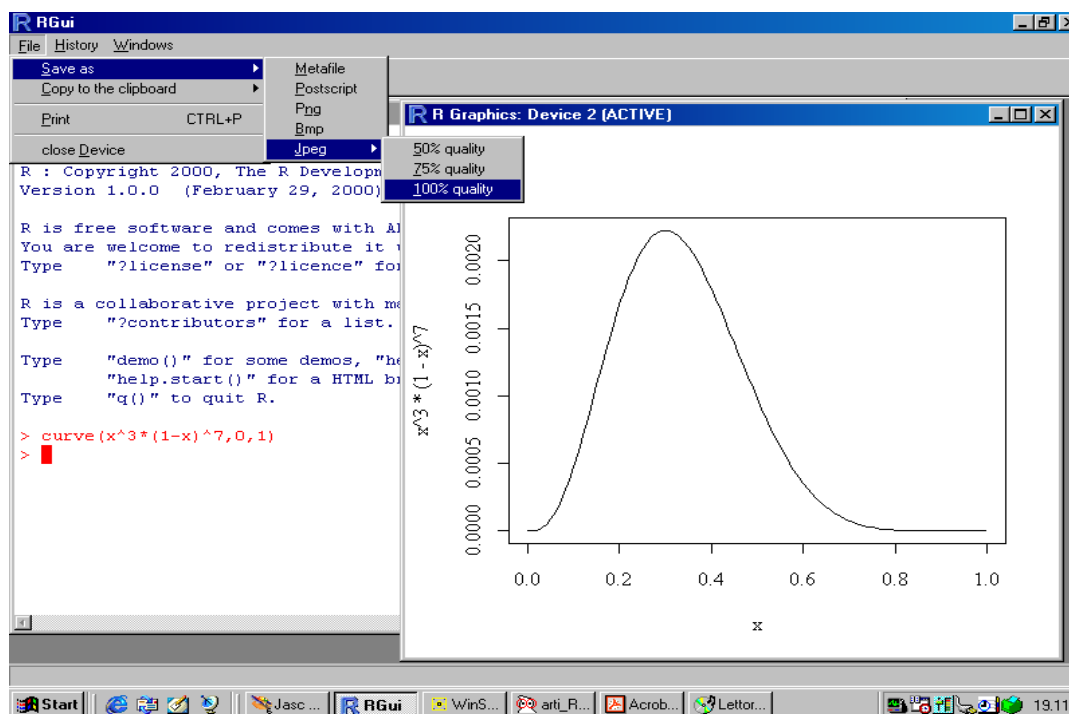


Figura 2: Finestra grafica di R

disegna la funzione di verosimiglianza $y = x^3(1-x)^7$ nell'intervallo $[0; 1]$, aprendo un'ulteriore finestra grafica (così come illustrato in figura 2) per la quale la linea comandi della finestra principale prevede il salvataggio in formati diversi. Alternativamente, esiste la possibilità di creare file in formato postscript cambiando l'uscita grafica mediante il comando

```
> postscript(file="percorso/nome.file",args)
```

prima di richiamare il grafico. Si noti che il percorso mantiene la struttura Unix, ad esempio per registrare nel file

```
C:\Documenti\Francesca\prova.ps
```

bisogna digitare

```
> postscript(file="C:/Documenti/Francesca/prova.ps")
```

Al fine di ripristinare le condizioni iniziali e quindi l'uscita grafica nella finestra Windows si digita:

```
> dev.off()
```

Volendo registrare i comandi in un file in formato PicTeX di LATEX

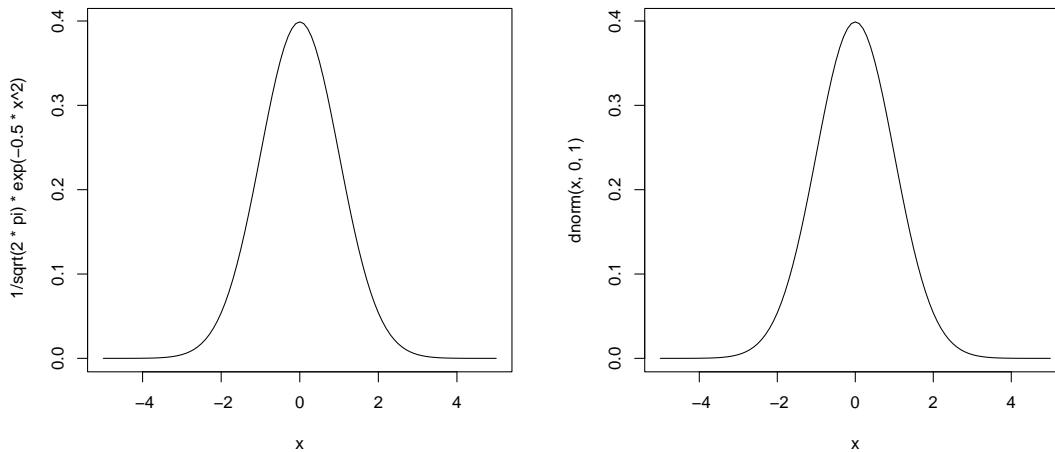


Figura 3: Curva normale disegnata con due comandi diversi

```
> pictex(file="percorso/nome.file",args)
```

Vediamo ora di disegnare la funzione di densità di una normale standard

```
> curve(1/sqrt(2*pi)*exp(-0.5*x^2),-5,5)
```

notiamo che otteniamo lo stesso grafico con il comando

```
> curve(dnorm(x,0,1),-5,5)
```

così come indica la figura 3.

Infatti molte distribuzioni di probabilità notevoli sono inserite come funzioni di R; le possibili opzioni per ciascuna di queste riguardano il calcolo della funzione di densità (o di probabilità, nel caso discreto) e della funzione di ripartizione in un punto x , il calcolo dei quantili per un valore di probabilità p e la generazione di n numeri casuali dalla variabile in considerazione. Per una distribuzione normale standard tali comandi sono, rispettivamente,

```
> dnorm(x,mean=0,sd=1)
```

```
> pnorm(x,mean=0,sd=1)
```

```
> qnorm(p,mean=0,sd=1)
```

```
> rnorm(n,mean=0,sd=1)
```

in cui x e p possono essere dei vettori.

Altre distribuzioni possono essere richiamate sostituendo `norm(arg,mean,sd)` nei comandi appena citati con quelli descritti in tabella 1. Oltre a queste distribuzioni ci sono anche le funzioni `ptukey()` e `qtukey()` per la distribuzione del rango studentizzato di campioni tratti dalla distribuzione normale.

	per la variabile
exp(arg,par)	esponenziale di media 1/par
chisq(arg1,par)	chi-quadrato con par gradi di libertà
f(arg,par1,par2,par3)	F di Snedecor con par1,par2 gradi di libertà e parametro di non centralità par3
pois(arg,par)	Poisson di media par
unif(arg,par1,par2)	uniforme continua sull'intervallo (par1,par2)
gamma(arg,par1,par2)	gamma con parametro di forma par1 e di scala par2
binom(arg,par1,par2)	binomiale di numerosità par1 e probabilità di estrazione par2
hyper(arg,par1,par2,par3)	ipergeometrica da una popolazione con par1 palline bianche, par2 palline nere dalle quali si estrae un campione di numerosità par3
beta(arg,par1,par2,par3)	beta con parametri di forma par1 e par2 e parametro di non centralità par3
cauchy(arg,par1,par2)	Cauchy con parametro di posizione par1 e di scala par2
geom(arg,par)	geometrica di probabilità par
lnorm(arg,par1,par2)	lognormale con parametri par1 e par2
logis(arg,par1,par2)	logistica con parametro di posizione par1 e di scala par2
nbinom(arg,par1,par2)	binomiale negativa di numerosità par1 e probabilità di estrazione par2
weibull(arg,par1,par2)	Weibull di parametro di forma par1 e di scala par2
wilcox(arg,par1,par2)	Wilcoxon con parametri par1 e par2

Tabella 1: Distribuzioni di probabilità

3 Applicazione: simulazione del teorema del limite centrale

Una prima applicazione alla teoria statistica di un programma come R, potrebbe essere quella della simulazione del teorema del limite centrale. Certamente come qualsiasi esercizio di simulazione non vuole essere una dimostrazione del teorema ma solo una applicazione per un più facile apprendimento della teoria sottostante.

Sappiamo che il teorema del limite centrale fornisce una distribuzione limite per la media campionaria da una popolazione di cui siano note solo media e varianza

Teorema 3.1 *Sia X_1, \dots, X_n un campione bernoulliano di n elementi tratto da una popolazione X di media μ e varianza σ^2 . La media campionaria $\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i$ opportunamente standardizzata ha distribuzione che asintoticamente converge a quella di una normale standardizzata. Indicando infatti con*

$$W_n = \frac{\bar{X}_n - \mu}{\sigma} \sqrt{n}$$

e con $F_{W_n}(\cdot)$ la funzione di ripartizione di W_n , si ha che

$$\lim_{n \rightarrow \infty} F_{W_n}(z) = \Phi(z)$$

per ogni z reale, dove $\Phi(\cdot)$ indica la funzione di ripartizione di una variabile normale standard.

Proviamo a generare dati da una variabile casuale uniforme sull'intervallo $(-5, 5)$. Formiamo 10000 campioni ciascuno di numerosità 50 e ne calcoliamo la media; i comandi sono i seguenti (si noti che comunque si deve inizializzare la variabile da utilizzare, ad esempio `dd`)

```
> dd<-0
> for (i in 1:10000) dd[i]<-mean(runif(50,-5,5))
```

disegniamo il corrispondente istogramma di frequenze al quale sovrapponiamo anche la funzione di densità teorica di una distribuzione normale di media 0 e varianza $100/(12*50)$:

```
> hist(dd,breaks=100,xlab="Medie",ylab="Frequenze relative",
+ main="Istogramma", freq=F)
> curve(dnorm(x,0,1/sqrt(6)),add=T)
```

otteniamo il grafico in figura 4.

Per fornire un'ulteriore specificazione possiamo a questo punto provare a considerare il teorema limite centrale per una qualsiasi distribuzione di partenza, ad esempio una popolazione esponenziale di media 2. In questo caso,

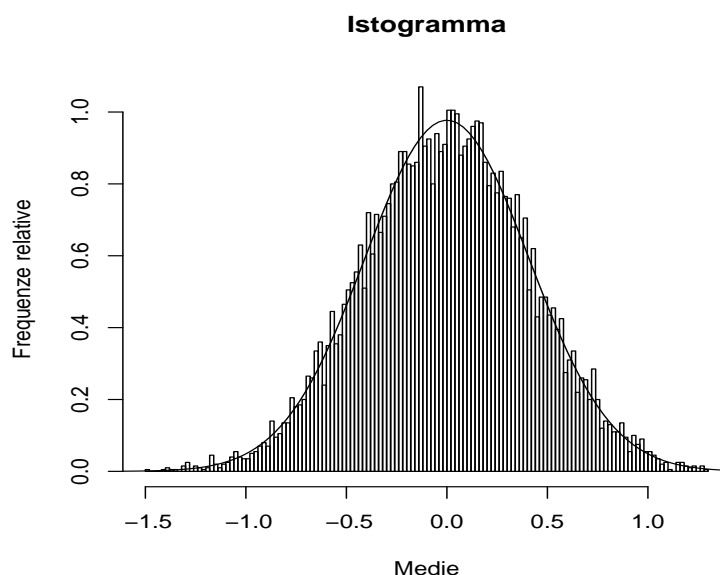


Figura 4: Distribuzione delle medie campionarie

```
> dd1<-0
> for (i in 1:10000) dd1[i]<-mean(rexp(50,0.5))
```

disegniamo il corrispondente istogramma di frequenze al quale sovrapponiamo anche la funzione di densità teorica di una distribuzione normale di media 2 e varianza 4/50:

```
> hist(dd1,breaks=100,xlab="Medie",ylab="Frequenze relative",
+ main="Istogramma", freq=F)
> curve(dnorm(x,2,2/sqrt(50)),add=T)
```

otteniamo il grafico in figura 5.

4 Rappresentazioni grafiche

Vediamo ora un modo per analizzare la distribuzione di un insieme di dati. È possibile, ad esempio, dopo aver letto un insieme di dati, considerarne le statistiche campionarie di posizione attraverso le due funzioni `summary(x)` e `fivenum(x)`. Quest'ultima fornisce in sequenza il valore minimo, il primo quartile, la mediana, il terzo quartile e il valore massimo dell'insieme di dati cui è applicata. La funzione `summary(x)`, oltre a queste informazioni fornisce anche la media dei valori.

Questi valori possono essere efficacemente rappresentati graficamente attraverso il diagramma **scatola-baffi** che si ottiene con il comando

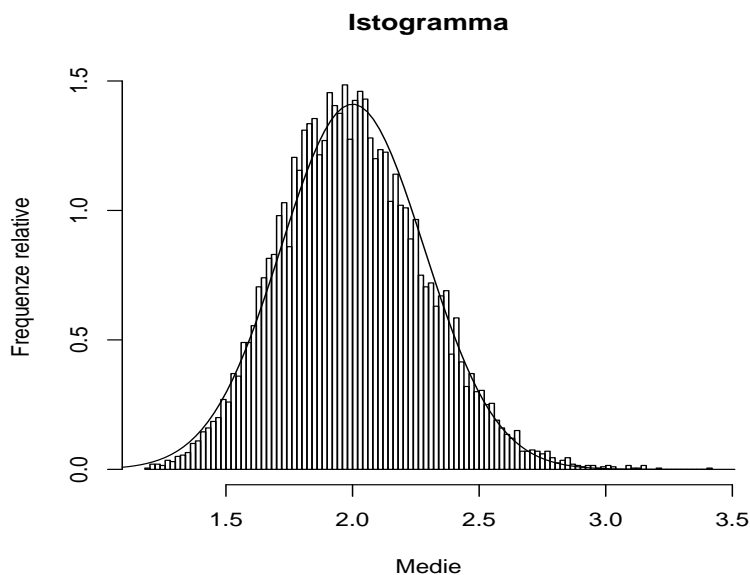


Figura 5: Distribuzione delle medie campionarie

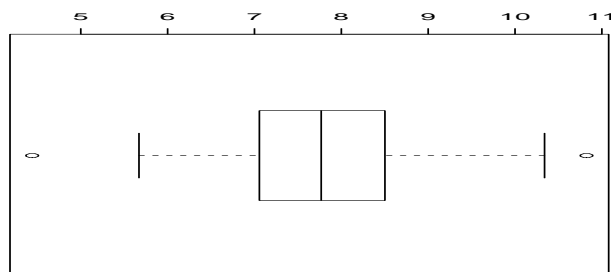


Figura 6: Diagramma scatola-baffi.

```
> boxplot(x)
```

il quale presenta già le barriere calcolate sul modello normale che risulta ad esempio nella forma in figura 6. Molte delle opzioni dei comandi sono modificabili, ad esempio si possono eliminare o modificare le barriere.

Per avere un'idea grezza della distribuzione di frequenza e quindi eventualmente della densità sottostante il fenomeno, si può fornire la rappresentazione **ramo-foglia**. In R comunque il comando

```
> stem(x)
```

non fornisce una specificazione molto raffinata. Consideriamo l'insieme di dati che rappresentano la lunghezza in miglia dei 141 maggiori fiumi del Nord America, dati che sono presenti nella distribuzione di base di R e che possono essere richiamati con

```
> data(rivers)
> stem(rivers)
```

quest'ultimo comando fornisce la rappresentazione ramo-foglia in figura 7.

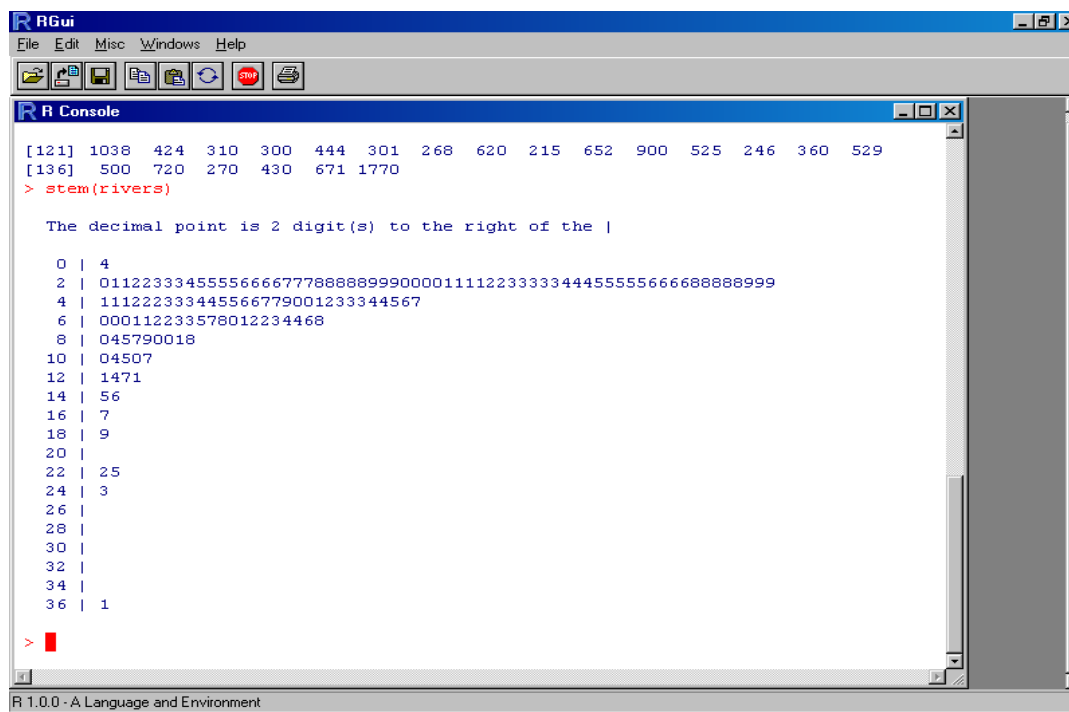


Figura 7: Ramo-foglia in R

Altrimenti è possibile usare l'istogramma di frequenza (si veda figura 8):

```
> lim_c(100,200,300,400,500,600,700,800,900,1000,
+ 1200,1500,3000,4000)
> hist(rivers,breaks=lim,xlab="Lunghezza dei fiumi Americani",
+ ylab="Frequenze relative",main="Istogramma")
```

Si noti che è implementato nella distribuzione di R anche una procedura non parametrica della stima della densità, attraverso la tecnica nucleo, che si richiama con il comando

```
> density(x)
```

Nell'esempio appena descritto, per la rappresentazione grafica è sufficiente, dopo aver disegnato l'istogramma,

```
> lines(density(rivers),col="red")
```

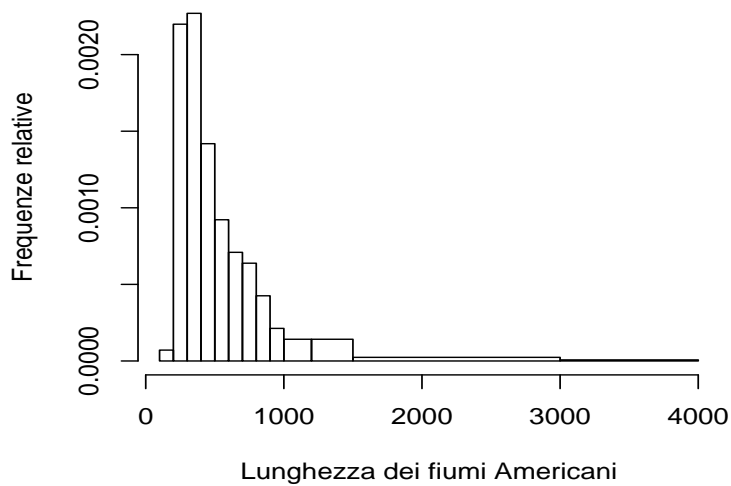
Istogramma

Figura 8: Istogramma di frequenza

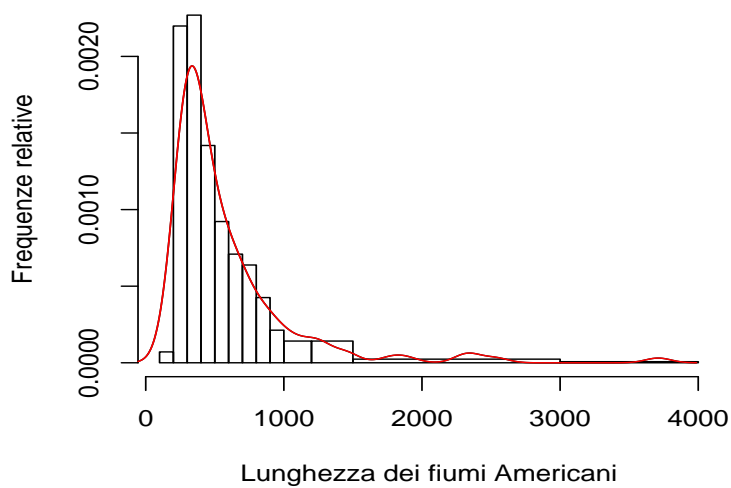
Istogramma

Figura 9: Stima della densità

e si ottiene, sovrapposto all'istogramma, la stima della densità. Un'ulteriore rappresentazione, utili a fini descrittivi e per le distribuzioni di fenomeni quantitativi, è il diagramma a **torta** che viene richiamato con il comando

```
> piechart(x)
```

Chiaramente ciascuna rappresentazione grafica avrà significato solo se opportunamente applicata.

5 Descrizione di un file di dati

Per leggere un file di tipo ASCII è sufficiente dare il comando `scan(file="nome-file")`, ad esempio

```
> catastrofi<-scan(file="//Bach/Docenti/Statistica/d1.txt")
```

carica nella variabile `catastrofi` i seguenti 50 dati

1	2	2	3	3	4	4	5	5	5
5	6	7	7	9	9	9	10	11	12
22	24	28	29	31	33	36	38	38	38
39	41	48	49	53	55	74	82	117	134
192	207	224	225	236	280	301	308	351	527

che riguardano le spese per eventi catastrofici sostenute da una Compagnia di Assicurazioni degli Stati Uniti. Poiché si tratta di dati tipicamente di tipo continuo e quindi raggruppabili in classi, possiamo raggrupparli in una tabella di frequenze. Scegliamo di suddividere l'asse reale in intervalli delimitati dai seguenti valori

0.5	12.5	20.5	35.5	44.5	59.5	320.5	560
-----	------	------	------	------	------	-------	-----

che assegnamo alla variabile **limiti**

```
> limiti<-c(0.5, 12.5, 20.5, 35.5, 44.5, 59.5, 320.5,560)
```

Per la costruzione in tabella di frequenze è necessario convertire la variabile numerica in una variabile di tipo qualitativo in cui i fattori corrispondono alle classi di modalità mediante il comando `cut(x,breaks=...)`

```
> cata<-factor(cut(catastrofi,breaks=limiti))
```

In tal modo abbiamo un fenomeno(**cata**) in cui gli elementi osservati sono dei fattori. Per costruire la tabella, a questo punto è sufficiente dare il comando `table(x)`

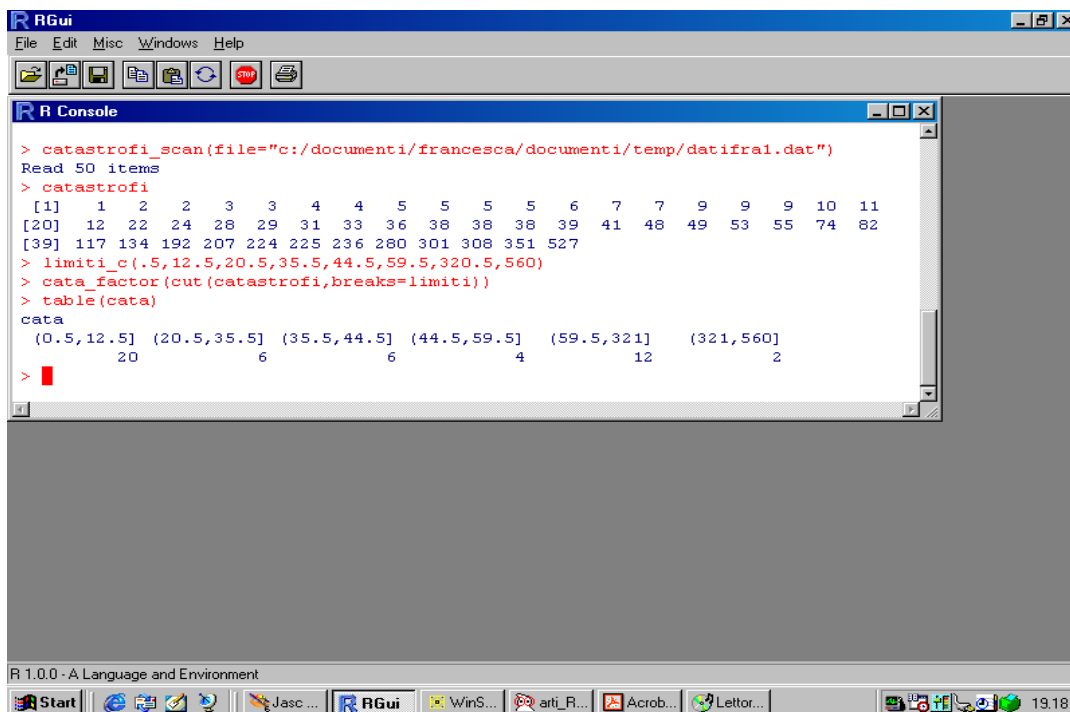


Figura 10: Tabella di frequenze per modalità suddivise in classi

```
> table(cata)
```

Per la distribuzione di frequenze relative

```
> table(cata)/length(cata)
```

e per controllare se effettivamente la somma delle frequenze relative è pari ad 1

```
> sum(table(cata)/length(cata))
```

Per costruire la funzione di frequenze cumulate si può operare in due modi: o si utilizza la funzione `cumsum(x)` (che in questo caso però fornisce la distribuzione sulle singole classi) oppure se vogliamo costruire la distribuzione sui dati grezzi è sufficiente richiamare la libreria `stepfun`

```
> library(stepfun)
```

e considerare la funzione `ecdf` il cui grafico è riportato in figura 11.

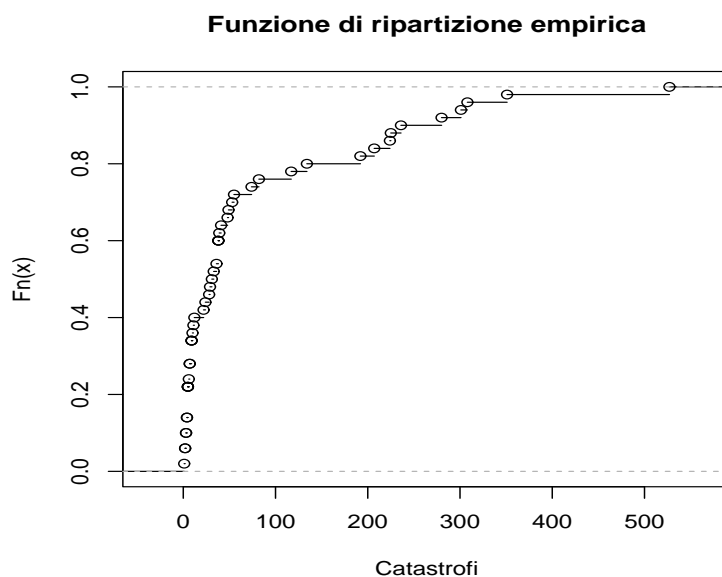


Figura 11: Funzione di frequenze cumulate

6 Rappresentazione di serie storiche

Un insieme di dati rilevati per lo stesso fenomeno in sequenza temporale viene detto serie storica e può essere rappresentato in modi appropriati. Consideriamo ad esempio il file “airline.dat” che contiene i dati mensili dei passeggeri dei voli internazionali dal gennaio ’49 al dicembre ’60 (i dati sono in tabella 2) organizzati su 12 righe per 13 colonne, delle quali la prima contiene l’anno di riferimento e di seguito i dati di ciascun mese dell’anno):

Anno	gen.	feb.	mar.	apr.	mag.	giu.	lug.	ago.	set.	ott.	nov.	dic.
1949	112	118	132	129	121	135	148	148	136	119	104	118
1950	115	126	141	135	125	149	170	170	158	133	114	140
1951	145	150	178	163	172	178	199	199	184	162	146	166
1952	171	180	193	181	183	218	230	242	209	191	172	194
1953	196	196	236	235	229	243	264	272	237	211	180	201
1954	204	188	235	227	234	264	302	293	259	229	203	229
1955	242	233	267	269	270	315	364	347	312	274	237	278
1956	284	277	317	313	318	374	413	405	355	306	271	306
1957	315	301	356	348	355	422	465	467	404	347	305	336
1958	340	318	362	348	363	435	491	505	404	359	310	337
1959	360	342	406	396	420	472	548	559	463	407	362	405
1960	417	391	419	461	472	535	622	606	508	461	390	432

Tabella 2: Numero passeggeri

Le istruzioni per caricare la matrice di dati sono

```
> aer0<-matrix(scan(file="airline.dat"),ncol=13,byrow=T)
```

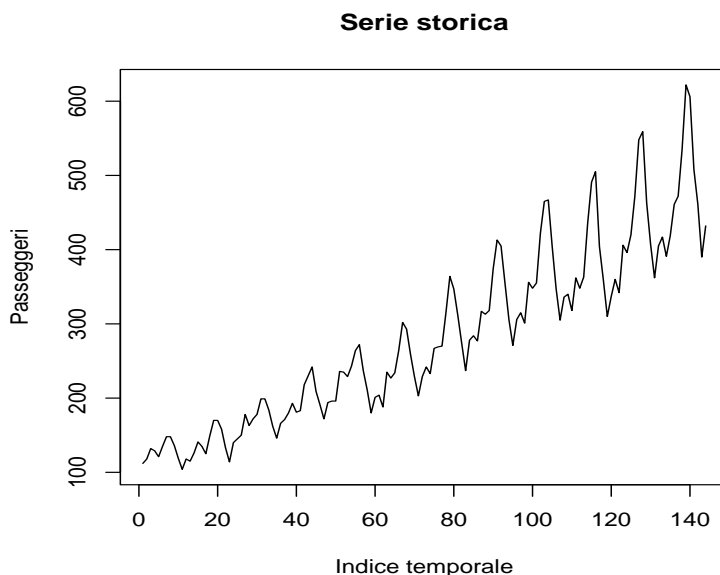


Figura 12: Serie storica

ma a noi interessa solo la serie di dati successivi per cui ne prendiamo solo 12 colonne escludendo la prima

```
> aer1<-aer0[,2:13]
```

e poi consideriamo un vettore

```
> aer2<-matrix(t(aer1))
```

Per poterla rappresentare adeguatamente, mantenendo la caratteristica di continuità dell'indice temporale, in ascissa mettiamo la successione dell'indice temporale e in ordinata l'intensità del fenomeno rilevato, e colleghiamo ciascun punto successivo con un segmento, che brevemente viene realizzato in R con

```
> plot(aer2,type="l")
```

e porta al grafico rappresentato in figura 12. Da questo grafico si nota subito la presenza di una componente stagionale e una di trend. Il pacchetto `ts` contiene strumentazioni per l'analisi delle serie storiche; ricordiamo che per caricare il pacchetto è necessario scrivere

```
> library(ts)
```

Prima di procedere si deve definire l'oggetto considerato come serie storica e quindi

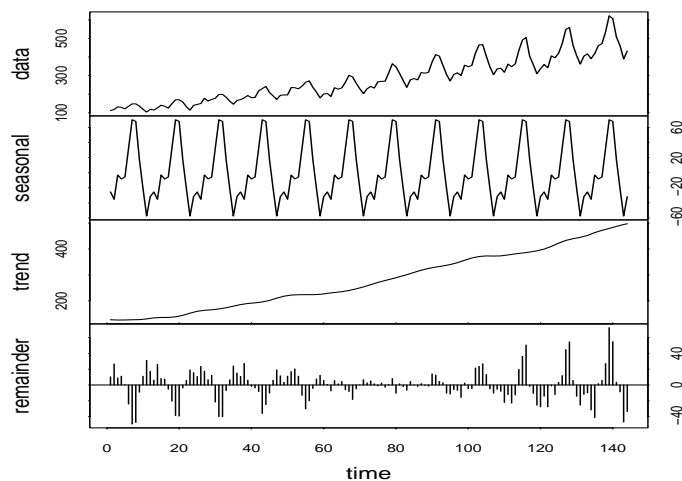


Figura 13: Serie storica destagionalizzata

```
> tae_ts(ae2, start=c(1949, 1), frequency=12, class="ts")
> passeggeri_tae[, 1]
```

In seguito si può ad esempio usare la *routine* per ottenere la decomposizione stagionale

```
> stl(passeggeri, s.window="per")
```

che disegnata mediante

```
> plot(stl(passeggeri, s.window="per"))
```

porta al disegno in figura 13.

7 Simulazione di intervalli di confidenza

Un'altro esperimento utile all'apprendimento della statistica inferenziale può essere quello riferito agli intervalli di confidenza. Sono stati simulati intervalli di confidenza di livello $1 - \alpha$ a partire da una distribuzione nota per confermare che la frequenza relativa degli intervalli che contengono il vero parametro tende a $1 - \alpha$. Considerando di disporre di n osservazioni bernoulliane da una popolazione normale di varianza nota, un intervallo di confidenza a livello $1 - \alpha$ per la media della popolazione è pari a

$$\left[\bar{x}_n - z_{\alpha/2} \frac{\sigma}{\sqrt{n}}; \bar{x}_n + z_{\alpha/2} \frac{\sigma}{\sqrt{n}} \right]$$

in cui $z_{\alpha/2}$ indica il percentile di ordine $1 - \alpha/2$ in una normale standard. Dalla teoria è noto che questo intervallo non è altro che una delle possibili determinazioni dell'intervallo casuale

$$I_{\alpha} = \left[\bar{X}_n - z_{\alpha/2} \frac{\sigma}{\sqrt{n}}; \bar{X}_n + z_{\alpha/2} \frac{\sigma}{\sqrt{n}} \right]$$

la cui probabilità è esattamente $1 - \alpha$

$$P(I_{\alpha}) = 1 - \alpha$$

Di conseguenza, se disponiamo di un numero altissimo di intervalli di confidenza dalla stessa popolazione per lo stesso parametro, ad esempio pari a N , ci aspettiamo che solo una porzione di questi contenga il vero e ignoto parametro da stimare, il cui corrisponde ad un numero che si aggira attorno a $N(1 - \alpha)$.

Un esercizio di questo tipo permette un'introduzione diretta alla tecnica della simulazione secondo la quale si deve considerare completamente nota la popolazione di partenza (altrimenti cosa potremmo simulare e controllare?): prendiamo una popolazione normale di media 4 e varianza 1 e da questa generiamo 100 sequenze campionarie ciascuna di numerosità pari a 20 e sulla quale calcoliamo la media campionaria:

```
> nN<-100
> nn<-20
> m.sim<-0
> for (i in 1:nN) {sim<-rnorm(nn,4,1); m.sim[i]<-mean(sim)}
```

Visualizziamo la distribuzione dei dati ottenuti attraverso un istogramma (figura 14):

```
> hist(m.sim,10,xlab="medie campionarie",
+ ylab="Frequenze relative",main="",freq=F)
```

la cui media è

```
> mean(m.sim)
```

Quel che ora dobbiamo fare è costruire gli intervalli di confidenza corrispondenti a ciascuna realizzazione campionaria. Fissiamo un livello di confidenza pari al 95%:

```
> zeta<-qnorm(0.975,0,1)
```

e inizializziamo le variabili che contengono i limiti degli intervalli:

```
> lim<-matrix(0,2,nN)
> for (i in 1:nN) {lim[1,i]<-m.sim[i]-zeta/sqrt(nn);
+ lim[2,i]<-m.sim[i]+zeta/sqrt(nn)}
```

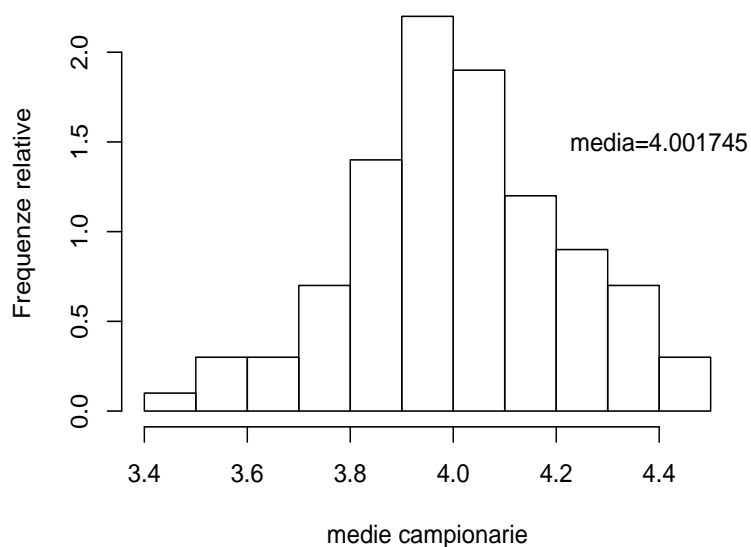


Figura 14: Istogramma delle medie campionarie

Visualizziamo tali intervalli utilizzando una variabile ausiliaria **y** che permetta di costruire intervalli successivi, di seguito riportiamo le istruzioni per ottenere il grafico in figura 15:

```
> y<-seq(1:nN)
> y<-matrix(y,nN,2)
> lim2<-t(lim)
> plot(y,lim2)
> for (i in 1:nN) lines(y[i,], lim2[i,])
```

Per contare quanti intervalli contengono la vera media, che sappiamo per simulazione essere pari a 4, utilizziamo una variabile logica **vero**:

```
> vero<-matrix(0,nN,2)
> vero<-c(lim2[,1]>4,lim2[,2]<4)
```

la matrice **vero**: contiene a questo punto solo degli zero e degli uno, il totale degli uno corrisponde al numero di intervalli che non contengono il vero valore della media

```
> sum(vero)
```

che nel nostro caso è pari a 3. Ciò significa che il 30% degli intervalli non contiene la media della popolazione, che naturalmente è perfettamente concorde con i risultati teorici.

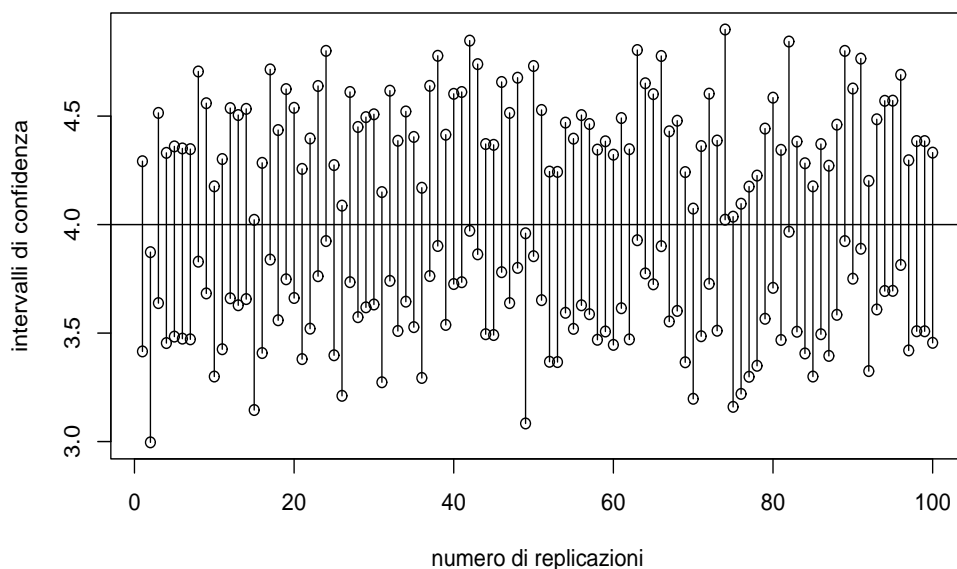


Figura 15: Intervalli di confidenza

8 Analisi di fenomeni bivariati: dipendenza lineare.

Con un programma grafico è possibile condurre un'efficace introduzione all'analisi descrittiva dei fenomeni bivariati. Infatti poiché per uno studio congiunto di due fenomeni, che indichiamo con X e Y , generalmente vengono rilevati su ciascuna unità statistica di osservazione i due fenomeni ottenendo le coppie di osservazioni

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$$

la prima cosa da fare è si analizza se è possibile stabilire una qualche forma di indipendenza fra i due fenomeni e, se i fenomeni sono quantitativi, si rappresentano le coppie di valori osservati in un *diagramma si dispersione*.

Prendiamo come esempio i seguenti fenomeni: X cavalli di potenza di un'automobile e Y litri di carburante consumati ogni 100 km. In tabella 3 vengono riportate le osservazioni su 15 unità osservate.

La prima rappresentazione grafica attraverso diagramma di dispersione è fornita in figura 16. Le istruzioni per ottenere tale rappresentazione sono molto semplici, come si può notare in seguito:

```
> auto<-matrix(0,15,2)
```

Unità	x_i	y_i	Unità	x_i	y_i	Unità	x_i	y_i
1	52	6.7	6	80	9.1	11	103	11.6
2	65	7.1	7	84	8.8	12	105	12.2
3	70	7.4	8	95	9.9	13	110	11.8
4	71	8.6	9	98	9.4	14	139	13.0
5	74	9.7	10	100	8.5	15	145	13.3

Tabella 3: Osservazioni.

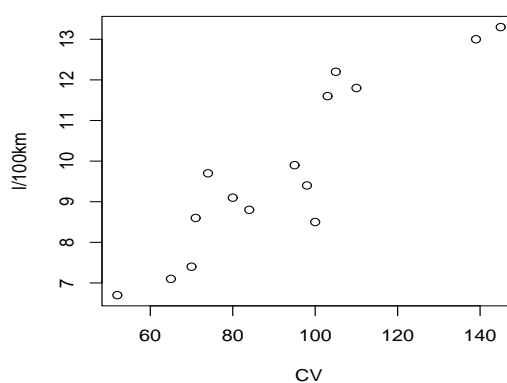


Figura 16: Dati di consumo per automobili

```
> auto[,1]<-c(52,65,70,71,74,80,84,95,98,100,103,105,
+ 110,139,145)
> auto[,2]<-c(6.7,7.1,7.4,8.6,9.7,9.1,8.8,9.9,9.4,8.5,
+ 11.6,12.2,11.8,13.0,13.3)
> plot(auto,xlab="CV",ylab="l/100km")
```

Quest'ultimo comando disegna, in un sistema di assi cartesiani, i punti che hanno come ascissa la prima colonna della matrice **auto** e come ordinata la seconda colonna. Per adattare un modello lineare a questa serie di dati è sufficiente richiamare la funzione **lm** che costruisce un modello lineare. L'andamento lineare della dipendenza fra i due fenomeni rappresentata dal grafico in figura 17, viene prodotta con il comando:

```
> abline(coef(lm(auto[,2]~auto[,1])))
```

Un utile esercizio consiste nel considerare un insieme di dati che presenta un'alto valore di correlazione. La correlazione si richiama brevemente con la funzione **cor**, nell'esempio appena visto produce già la matrice di correlazione

```
> cor(auto)
      [,1]      [,2]
```

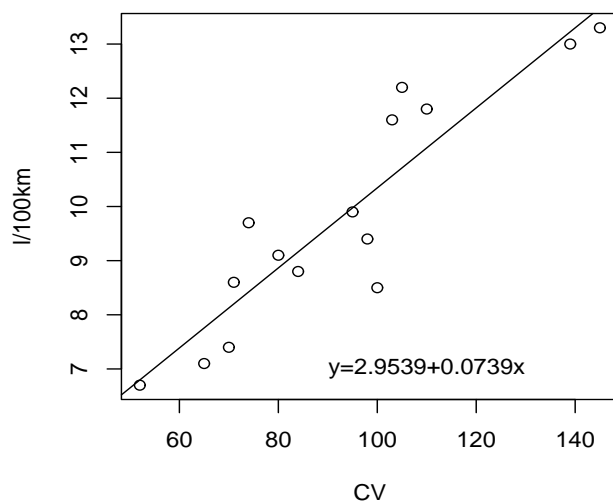


Figura 17: Retta di regressione

```
[1,] 1.000000 0.908718
[2,] 0.908718 1.000000
```

Vogliamo studiare la relazione fra la corrente prodotta dai mulini a vento e la velocità del vento:

```
> vento<-matrix(scan(file="vento.txt"),byrow=T,25,2)
> plot(vento)
```

La figura 18 riporta il diagramma appena disegnato. Poiché vi è un'alta correlazione fra le variabili

```
> cor(vento)
      [,1]      [,2]
[1,] 1.0000000 0.9351434
[2,] 0.9351434 1.0000000
```

Vogliamo interpolare la retta di regressione, o meglio un modello lineare a questa serie di dati

```
> line1<-lm(vento[,2]~vento[,1])
```

che produce come risultato:

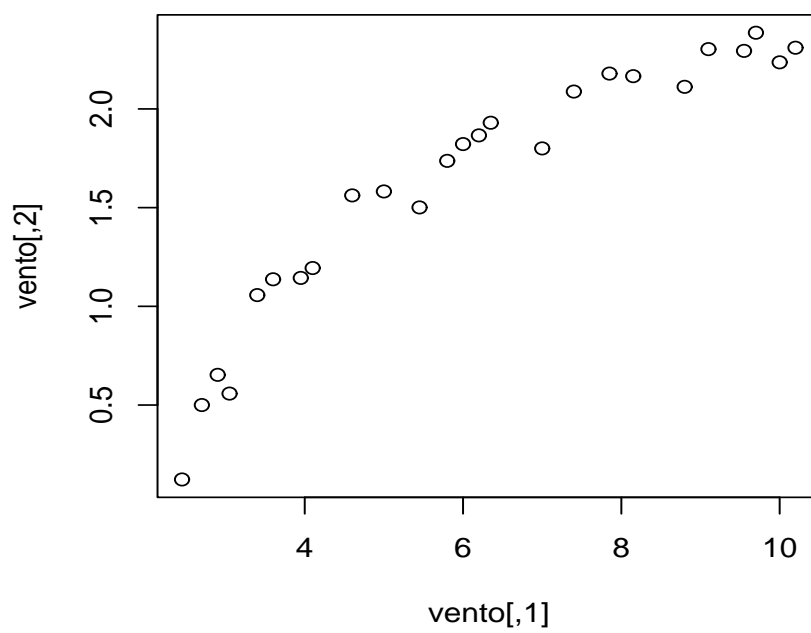


Figura 18: Diagramma di dispersione dei dati di “vento.txt”

```
> line1
```

```
Call:
```

```
lm(formula = vento[, 2] ~ vento[, 1])
```

```
Coefficients:
```

```
(Intercept)   vento[, 1]
      0.1309         0.2411
```

Se invece vogliamo stabilire i risultati dell'analisi di regressione:

```
> summary(line1)
```

che fornisce informazioni sulla significatività dei parametri del modello lineare e sulla bontà di adattamento del modello.

```
> summary(line1)
```

```
Call:
```

```
lm(formula = vento[, 2] ~ vento[, 1])
```

```
Residuals:
```

```
      Min       1Q   Median       3Q      Max
-0.59869 -0.14099  0.06059  0.17262  0.32184
```

```
Coefficients:
```

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.13088     0.12599   1.039    0.31
vento[, 1]   0.24115     0.01905  12.659 7.55e-12 ***
```

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.2361 on 23 degrees of freedom
```

```
Multiple R-Squared:  0.8745,    Adjusted R-squared:  0.869
```

```
F-statistic: 160.3 on 1 and 23 degrees of freedom, p-value: 7.546e-012
```

Per disegnare la retta di regressione:

```
> plot(vento)
```

```
> abline(coef(line1))
```

Se ci interessa analizzare i residui del modello innanzitutto li disegniamo:

```
> plot(vento[,1], resid(line1))
```

Dalla figura cui19 notiamo che c'è ancora qualcosa da spiegare nel modello. Ad esempio possiamo pensare che la dipendenza fra le variabili sia di tipo parabolico

$$Y = a + bX + cX^2$$

che risulta ancora una funzione lineare nei parametri a , b e c . Quindi creiamo la nuova variabile **ve2**:

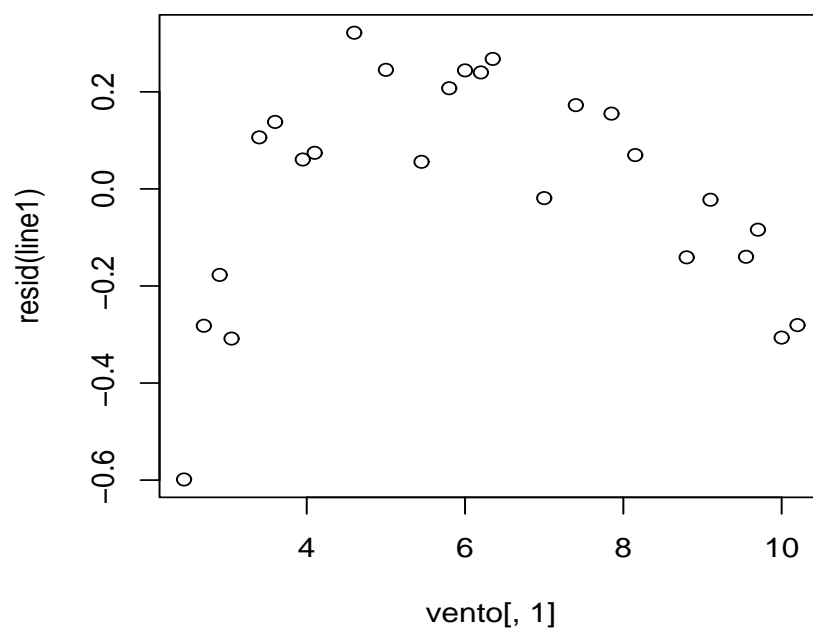


Figura 19: Residui

```
> ve2<-(vento[,1])^2
```

e consideriamo il nuovo modello lineare

```
> line2<-lm(vento[,2]~vento[,1]+ve2).
```

Oppure possiamo pensare che la dipendenza fra le variabili sia di tipo iperbolico

$$Y = a + b \frac{1}{X}$$

che risulta ancora una funzione lineare nei parametri a e b . Quindi creiamo la nuova variabile **ve3**:

```
> ve3<-1/vento[,1]
```

e consideriamo il nuovo modello lineare

```
> line2<-lm(ve3)
```

Riferimenti bibliografici

- [1] *An introduction to R: Notes on R a programming environment for data analysis and graphics*, Version 1.0.0 (2000 february 29). R Development Core Team.
- [2] *The R Reference index*, Version 1.0.0 (2000 february 29). R Core Team.

Indice

1	Introduzione	1
2	Il programma R	2
3	Applicazione: simulazione del teorema del limite centrale	8
4	Rappresentazioni grafiche	9
5	Descrizione di un file di dati	13
6	Rappresentazione di serie storiche	15
7	Simulazione di intervalli di confidenza	17
8	Analisi di fenomeni bivariati: dipendenza lineare.	20