Soluzioni Esercizi di Applicazioni Insegnamento Modelli Statistici II - Biostatistica -

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

Ottobre 2022

Contents

Soluzioni esercizi "Richiami sintassi di R e R Markdown"

SOLUZIONI ESERCIZIO 1

La mediana è una misura di tendenza centrale che viene calcolata per caratteri quantitativi. Nel caso di distribuzioni di frequenza il calcolo si esegue in base al fatto che il numero delle unità campionate sia pari o dispari. Essendo, nel nostro caso, il numero complessivo delle unità pari a 15, e denotando la mediana con M, si ha:

$$M = x_{\left(\frac{n_x+1}{2}\right)} = x_{\left(\frac{15+1}{2}\right)} = x_{(8)} = 1.73.$$

Con l'operatore `c` definiamo il vettore dei valori (unici):

```
x <- c(1.7, 1.72, 1.73, 1.75, 1.78, 1.79)
```

ed il corrispondente vettore delle frequenze:

```
freq <- c(3, 3, 2, 3, 2, 2)
```

Utilizzando la funzione `rep`, si genera quindi il vettore contenente tutti i valori osservati

```
valori <- rep(x, freq); valori
```

```
#>  [1] 1.70 1.70 1.70 1.72 1.72 1.72 1.73 1.73 1.75 1.75 1.75 1.78 1.78 1.79 1.79
```

Si calcola infine il valore della mediana con la funzione `quantile`. Si noti che la funzione `quantile` ha come argomenti, oltre al vettore dei valori, `probs` che denota l'ordine del quantile da calcolare, e `type` che identifica il tipo di algoritmo utilizzato: quello corrispondente al numero 2 utilizza l'inversa della funzione di ripartizione empirica.

```
M <- quantile(valori,
              probs = 0.5,
              type = 2); M
```

```
#>  50%
#> 1.73
```

La metà dei pazienti presenta una concentrazione di teofillina dopo tre minuti dalla somministrazione minore o uguale a 1.73 mg/l.

SOLUZIONI ESERCIZIO 2

Per generare sequenze di valori si utilizza la funzione `seq`, specificando i valori iniziale e finale della successione (in questo caso `pi` e `-pi`), e, a seconda della richiesta, la lunghezza del vettore (`length.out`) o il passo (`by`, ovvero la distanza tra due elementi successivi). Nel caso in esame chiediamo `length.out = 200`.

```
x <- seq(-pi, pi,
         length.out = 200); x
```

```
#>   [1] -3.14159265 -3.11001886 -3.07844506 -3.04687127 -3.01529747 -2.98372368
#>   [7] -2.95214988 -2.92057608 -2.88900229 -2.85742849 -2.82585470 -2.79428090
#>  [13] -2.76270711 -2.73113331 -2.69955952 -2.66798572 -2.63641193 -2.60483813
#>  [19] -2.57326433 -2.54169054 -2.51011674 -2.47854295 -2.44696915 -2.41539536
#>  [25] -2.38382156 -2.35224777 -2.32067397 -2.28910017 -2.25752638 -2.22595258
#>  [31] -2.19437879 -2.16280499 -2.13123120 -2.09965740 -2.06808361 -2.03650981
#>  [37] -2.00493602 -1.97336222 -1.94178842 -1.91021463 -1.87864083 -1.84706704
#>  [43] -1.81549324 -1.78391945 -1.75234565 -1.72077186 -1.68919806 -1.65762426
#>  [49] -1.62605047 -1.59447667 -1.56290288 -1.53132908 -1.49975529 -1.46818149
#>  [55] -1.43660770 -1.40503390 -1.37346010 -1.34188631 -1.31031251 -1.27873872
#>  [61] -1.24716492 -1.21559113 -1.18401733 -1.15244354 -1.12086974 -1.08929595
#>  [67] -1.05772215 -1.02614835 -0.99457456 -0.96300076 -0.93142697 -0.89985317
#>  [73] -0.86827938 -0.83670558 -0.80513179 -0.77355799 -0.74198419 -0.71041040
#>  [79] -0.67883660 -0.64726281 -0.61568901 -0.58411522 -0.55254142 -0.52096763
#>  [85] -0.48939383 -0.45782003 -0.42624624 -0.39467244 -0.36309865 -0.33152485
#>  [91] -0.29995106 -0.26837726 -0.23680347 -0.20522967 -0.17365588 -0.14208208
#>  [97] -0.11050828 -0.07893449 -0.04736069 -0.01578690  0.01578690  0.04736069
#> [103]  0.07893449  0.11050828  0.14208208  0.17365588  0.20522967  0.23680347
#> [109]  0.26837726  0.29995106  0.33152485  0.36309865  0.39467244  0.42624624
#> [115]  0.45782003  0.48939383  0.52096763  0.55254142  0.58411522  0.61568901
#> [121]  0.64726281  0.67883660  0.71041040  0.74198419  0.77355799  0.80513179
#> [127]  0.83670558  0.86827938  0.89985317  0.93142697  0.96300076  0.99457456
#> [133]  1.02614835  1.05772215  1.08929595  1.12086974  1.15244354  1.18401733
#> [139]  1.21559113  1.24716492  1.27873872  1.31031251  1.34188631  1.37346010
#> [145]  1.40503390  1.43660770  1.46818149  1.49975529  1.53132908  1.56290288
#> [151]  1.59447667  1.62605047  1.65762426  1.68919806  1.72077186  1.75234565
#> [157]  1.78391945  1.81549324  1.84706704  1.87864083  1.91021463  1.94178842
#> [163]  1.97336222  2.00493602  2.03650981  2.06808361  2.09965740  2.13123120
#> [169]  2.16280499  2.19437879  2.22595258  2.25752638  2.28910017  2.32067397
#> [175]  2.35224777  2.38382156  2.41539536  2.44696915  2.47854295  2.51011674
#> [181]  2.54169054  2.57326433  2.60483813  2.63641193  2.66798572  2.69955952
#> [187]  2.73113331  2.76270711  2.79428090  2.82585470  2.85742849  2.88900229
#> [193]  2.92057608  2.95214988  2.98372368  3.01529747  3.04687127  3.07844506
#> [199]  3.11001886  3.14159265
```

SOLUZIONE ESERCIZIO 3

3.1

Si caricano i dati, salvati in formato .RData, utilizzando la funzione `load`.

```
load("psalev.Rdata")
```

La funzione `summary` restituisce alcune tra le principali statistiche descrittive (indici di posizione) per ciascuna delle variabili del dataframe passato in argomento (in questo caso una sola variabile).

```
summary(psalev)
```

```
#>      psalev
#>  Min.   :0.01200
#>  1st Qu.:0.07667
#>  Median :0.15600
#>  Mean   :0.26897
#>  3rd Qu.:0.27800
#>  Max.   :4.55333
```

Coerentemente con la natura della variabile **psalev** (misura il levllo di antigene nella prostata), tutte le osservazioni oggetto di studio presentano valori positivi, da un minimo di 0.01 ad un massimo di 4.55. Osserviamo che il valore massimo si discosta in modo estramamente marcato dal terzo quartile (pari a 0.28); gli altri quartili sono assumono valori abbastanza simili tra di loro. La media assume approssimativamente lo stesso valore del terzo quartile. Il risultato è una distribuzione asimmetrica, con una coda molto pesante verso destra.

Aggiungiamo poi alcuni indici di dispersione che non vengono calcolati dalla funzione `summary` (deviazione standard, scarti interquartile e coefficiente di variazione). La deviazione standard si calcola con la funzione `sd`. Si noti che questa (così come la funzione `var` per il calcolo della varianza) fornisce il calcolo della quantità campionaria, ovvero ottenuta dividendo per $(n-1)$. Se necessario ci si può riportare ai valori dello scarto quadratico medio della popolazione moltiplicando per $(n-1)/n$ in modo che risulti

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \mu)^2}{n}}.$$

```
sd(psalev$psalev)
```

```
#> [1] 0.4876327
```

Dal valore della deviazione standard ricaviamo che lo scostamento medio delle osservazioni rispetto alla loro media è pari a 0.49. Osserviamo che il valore della deviazione standard è superiore a quello assunto dalla media, sintomo di una variabilità delle osservazioni particolarmente elevata. Questo è confermato dal calcolo del coefficiente di variazione (cv: permette di adimensionare il valore della deviazione standard, che diventa così quantificabile e confrontabile).

```
cv <- function(x) {sd(x)/mean(x)}
cv(psalev$psalev)
```

```
#> [1] 1.812999
```

Ricaviamo infatti che la variabilità del fenomeno è dell'ordine del 180% del valore della media. Calcoliamo infine il valore dello scarto interquartile, ottenuto come differenza tra il primo ed il terzo quartile.

```
IQR(psalev$psalev)
```

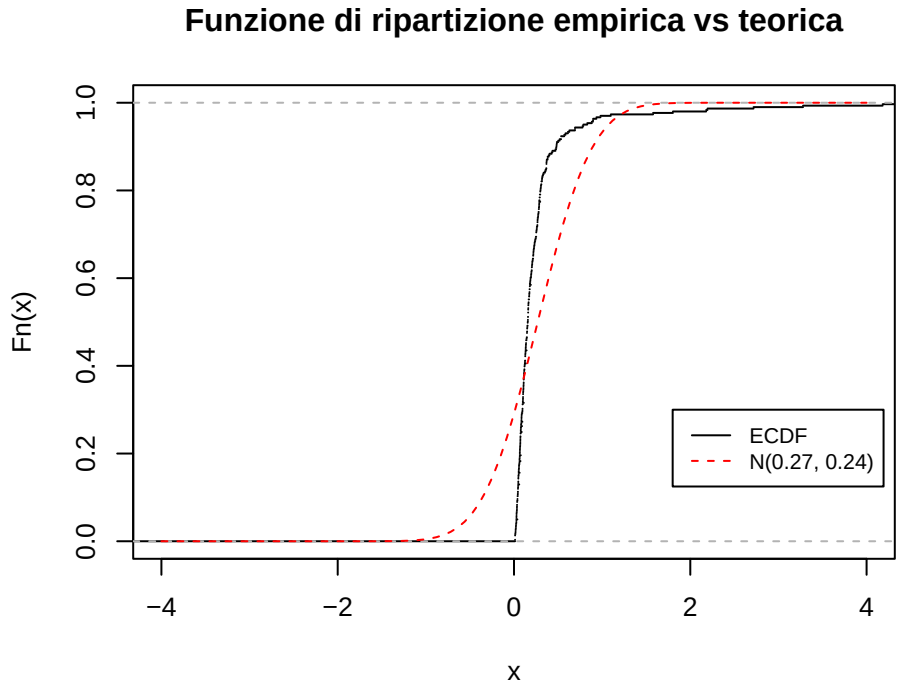
```
#> [1] 0.2013333
```

In questo caso il valore dell'indice è piuttosto basso, in quanto la maggior parte della variabilità si concentra nella parte finale della distribuzione.

3.2

Riportiamo dapprima il grafico della funzione di ripartizione empirica (calcolata tramnite il comando `ecdf`), aggiungendo la curva della funzione di ripartizione teorica (assumendo distribuzione normale con valori campionari dei parametri) e la relativa legenda.

```
plot(ecdf(psalev$psalev),
     do.points = FALSE,
     main = "Funzione di ripartizione empirica vs teorica",
     col = "black",
     xlim = c(-4, 4))
m <- mean(psalev$psalev)
sds <- sd(psalev$psalev)

curve(pnorm(x, mean = m, sd = sds),
      add = T,
      lty = 2,
      col = "red")

legend(1.8, 0.3,
       col = c("black", "red"),
       c("ECDF", "N(0.27, 0.24)"),
       lty = c(1, 2),
       cex = 0.8)
```



Risulta piuttosto evidente come le due funzioni di ripartizione (empirica e teorica) non seguano lo stesso un andamento. In particolare, notiamo che la distanza verticale tra le due curve è piuttosto pronunciato (si ricorda che il test di Kolmogorov-Smirnov per la verifica della normalità è basato su questa distanza). Questo porta a sostenere la non normalità della distribuzione della variabile **psalev**. Del resto, era già stata evidenzaiata l'assenza di simmetria in seguito all'analisi delle statistiche descrittive.

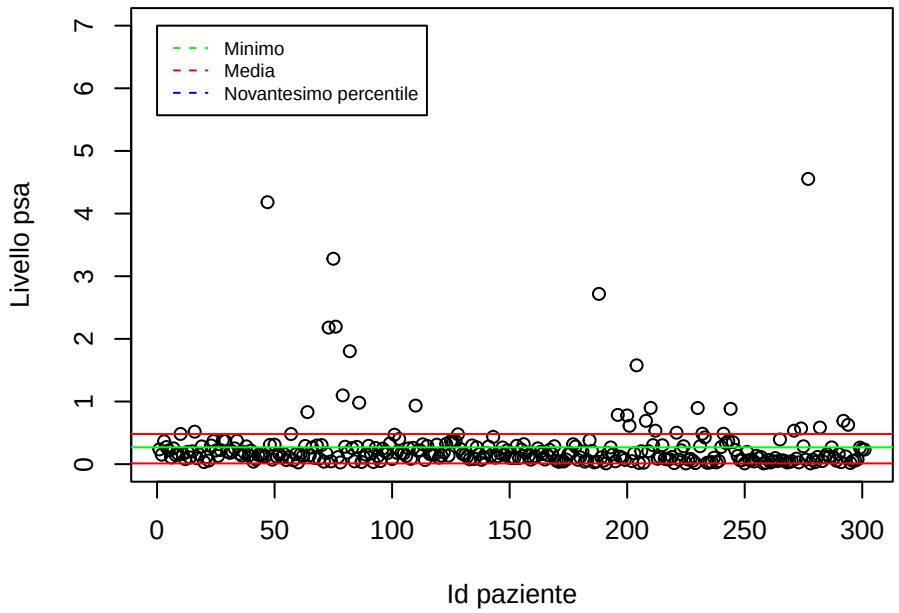
Esaminiamo poi il grafico a dispersione dei punti nel piano cartesiano (rispetto agli id dei pazienti). Aggiungiamo poi nel grafico i valori del minimo, della media e del novantesimo percentile.

```
plot(psalev$psalev,
     xlab = "Id paziente",
     ylab = "Livello psa",
     ylim = c(0,7))
q <- quantile(psalev$psalev, c(0, 0.90)); q
```

```
#>    0%   90%
#> 0.012 0.481
```

```
abline(h = c(m, q[1], q[2]),
       col = c("green", "red", "red"))

legend(0, 7,
       c("Minimo", "Media", "Novantesimo percentile"),
       col = c("green", "red", "blue"),
```

```
       lty = c(2, 2, 2),
       cex = 0.7)
```



Coerentemente con i risultati delle statistiche descrittive analizzanti in precedenza, notiamo che la maggior parte (90%) delle osservazioni assumono valori piuttosto bassi, e presentano una variabilità molto limitata. Osserviamo però la presenza di un ristretto numero di soggetti con un livello di antigene sensibilmente più elevato. Non si rileva alcuna dipendenza del valore della variabile **psalev** rispetto agli id dei pazienti.

SOLUZIONE ESERCIZIO 4

4.1

Si richiede la libreria `ISwR` (Introductory Statistics with `R`).

```
require(ISwR)
# help('thuesen')
```

Come riportato nell'help, il dataset contiene le rilevazioni effettuate su 24 pazienti con il diabete di tipo 1. Vengono riportati la **velocità ventricolare** (unità di misura: %/s) ed il **livello di glucosio** presente nel sangue (unità di misura: mmol/l).

La seguente funzione (contenuta nel pacchetto `skimr`[1]) permette di avere un quadro complessivo dei dati e delle statistiche di base a seconda della tipologia di carattere: fattore, numerico o carattere ed, eventulamente, in base alle diverse categorie.

```
skimr::skim_without_charts(thuesen)
```

Table 1: Data summary

Name	thuesen
Number of rows	24
Number of columns	2
_____	
Column type frequency:	
numeric	2
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
blood.glucose	0	1.00	10.30	4.34	4.20	7.08	9.40	12.70	19.50
short.velocity	1	0.96	1.33	0.23	1.03	1.19	1.27	1.42	1.95

```
which(is.na(thuesen$short.velocity))
```

```
#> [1] 16
```

Notiamo innanzitutto che è presente un valore mancante per la velocità; si tratta in particolare del valore della velocità per l'unità 16. La velocità ventricolare media, calcolata sui restanti 23 pazienti, è 1.326 (unità di misura %/s). Dal valore della mediana si nota inoltre che per metà dei pazienti la velocità ventricolare risulta minore o uguale a 1.27 (%/s). Un quarto dei pazienti ha invece una velocità ventricolare superiore a 1.42 %/s (terzo quartile).

La variabilità media della velocità intorno alla media aritmetica è di 0.23 (%/s). Tale valore è sensibilmente inferiore rispetto alla media, pertanto la variabilità della velocità ventricolare è piuttosto limitata (circa 18% della media).

Si noti che, a differenza della funzione `skim` che gestisce in modo automatico gli eventuali valori mancanti, le singole funzioni per il calcolo degli inidici (`mean`, `quantile`, `sd`, …) richiedono di specificare `na.rm = TRUE` per il calcolo in presenza di valori mancanti. Ad esempio, questo è necessario per il calcolo dello scarto interquatile (descrive la dispersione delle osservazioni centrali della distribuzione empirica).

[1] La funzione "base" del paccheto è `skim` che, per ciascuna variabile numerica, stampa anche l'istogramma della distribuzione. Tuttavia, la presenza dell'istogramma impedisce la corretta compilazione del file in formato pdf. Si utilizza pertanto la funzione `skim_without_charts` che non include l'istogramma (mantenendo invece inalterate tutte le altre informazioni).

```
IQR(thuesen$short.velocity, na.rm = TRUE)
```

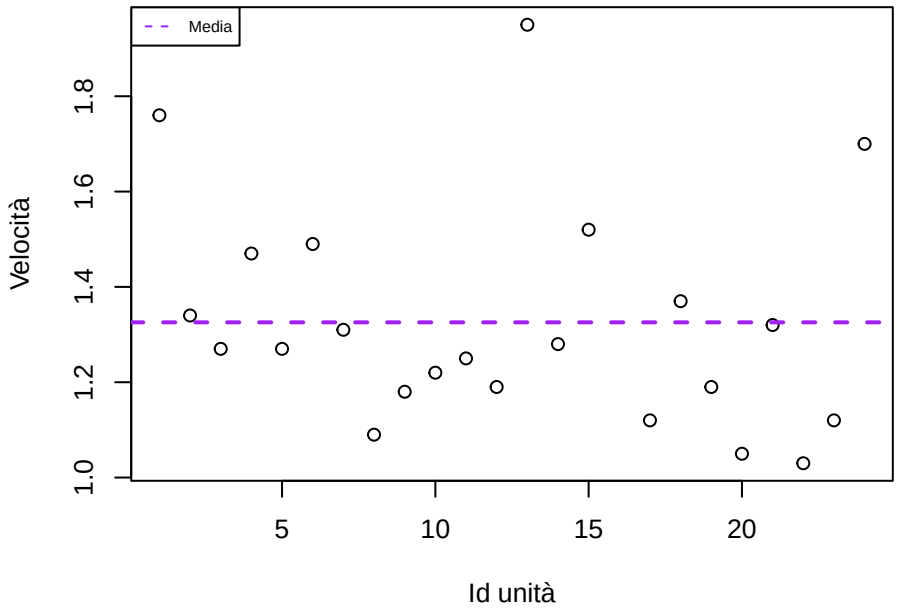
```
#> [1] 0.235
```

La parte centrale dei valori della velocità ventricolare campionaria si trova in un intervallo di ampiezza 0.235.

4.2

Rappresentiamo innanzitutto i valori osservati per la velocità ventricolari attraverso un grafico a dispersione (rispetto all'id dei pazienti). Il diagramma a dispersione viene disegnato modificando le etichette degli assi e aggiungendo una retta tratteggiata per evidenziare il valore centrale della distribuzione empirica.

```
plot(thuesen$short.velocity,
     xlab = "Id unità",
     ylab = "Velocità")

tm <- mean(thuesen$short.velocity, na.rm = TRUE)

abline(h = tm,
       col = 'purple',
       lty = 2,
       lwd = 2)

legend("topleft",
       col = c("purple"),
       c("Media"),
       lty = c(2),
       cex = 0.6)
```



Evidenziamo la presenza di tre osservazioni con valore superiore a 1.6; queste risultano particolarmente elevate rispetto alla media e discostate rispetto alle rimanenti osservazioni.

Può essere interessante verificare se i tre pazienti che presentano valori alti per la velocità ventricolare hanno anche livelli particolarmente elevati di glucosio. Selezioniamo quindi i pazienti che hanno valori superiori a 1.6 per la velocità.

```
which(thuesen$short.velocity > 1.6)
```

```
#> [1]  1 13 24
```

```
skimr::skim_without_charts(thuesen$blood.glucose)
```

Table 3: Data summary

Name	thuesen$blood.glucose
Number of rows	24
Number of columns	1
_____	
Column type frequency:	
numeric	1
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	10.3	4.34	4.2	7.08	9.4	12.7	19.5

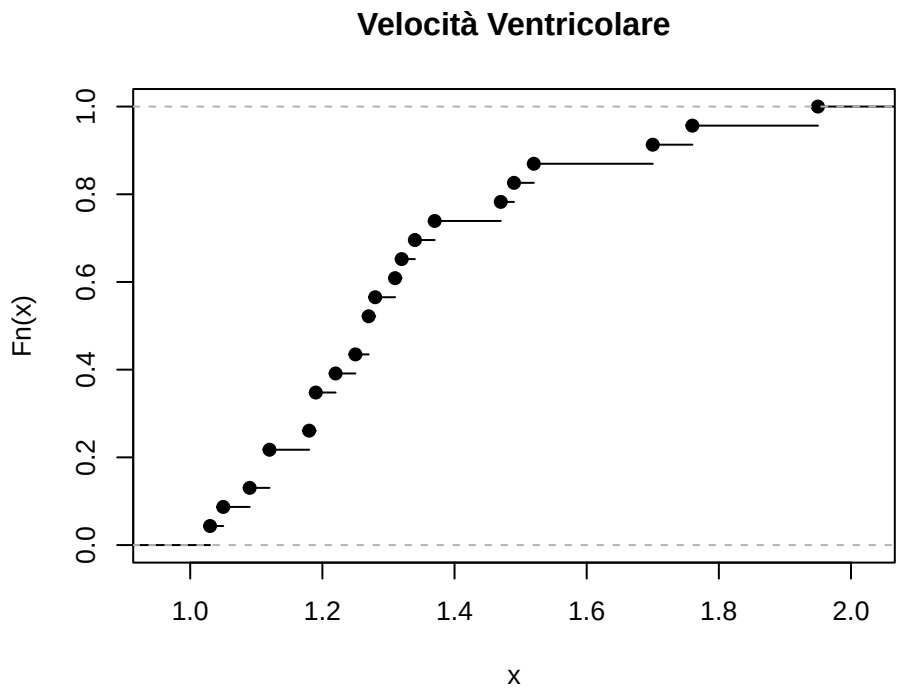
```
thuesen$blood.glucose[c(1, 13, 24)]
```

```
#> [1] 15.3 19.0  9.5
```

I pazienti con indice 1 e 13 hanno valori elevati di velocità ventricolare e presentano valori del glucosio superiori alla media. Al contrario, il paziente con id 24, pur avendo una velocità ventricolare particolarmente alta, presenta un livello di glucosio di poco inferiore al valor medio.

Disegnamo ora la funzione di riparzione empirica della velocità ventricolare: si tratta di una funzione costante a tratti e descrive l'andamento delle frequenze relative cumulate.

```
plot(ecdf(thuesen$short.velocity),
     main = "Velocità Ventricolare")
```



Nel grafico vengono rappresentati i punti corrispondenti ai diversi pazienti osservati (non è stata specificata l'opzione `do.points = FALSE`). Osserviamo, a titolo puramente esemplificativo, che (sulla base del campione osservato) la probabilità che il valore della velocità ventricolare sia minore o uguale di (circa) 1.1 %/s è approssimativamente pari a 0.2. Possiamo anche notare che il ritmo di accrescimento è maggiore per i valori più bassi: questi appaiono infatti molto ravvicinati tra loro. Al contrario i valori più elevati (in particolare i 3 segnalati in precedenza) si distanziano molto tra loro e rispetto a tutti gli altri.

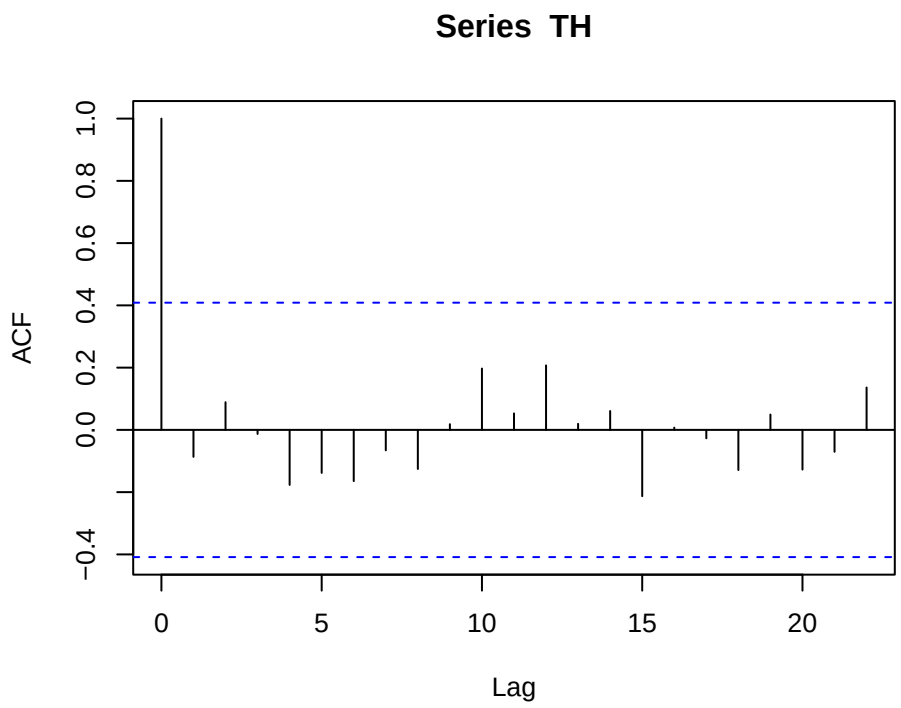
4.3

Come richiesto si disegna la funzione di autocorrelazione. Osserviamo però che le osservazioni sono rilevate su pazienti diversi e pertanto si suppone assenza di autocorrelazione. La funzione di autocorrelazione si calcola e rappresenta graficamente con il comando `acf`. E' possibile specificare il valore massimo del lag a cui effettuare il calcolo (argomento `lag.max`). Se la variabile presenta dei valori mancanti (come avviene nel nostro caso), il comando restituisce un errore (linea commentata nel codice seguente); è quindi necessario eliminare preliminarmente le osservazioni corrispondenti. Questo può essere fatto o attraverso il comando `which(is.na())`, come visto in precedenza, o utilizzando la funzione `complete.cases` che seleziona in modo automatico soltanto i valori completi.

```
# acf(thuesen$short.velocity)
TH <- thuesen$short.velocity[complete.cases(thuesen$short.velocity)]
TH <- thuesen$short.velocity[-which(is.na(thuesen$short.velocity))]
```

Possiamo quindi disegnare la funzione di autocorrelazione rispetto ai dati Come valore di `lag.max` selezioniamo 23, pari al numero di osservazioni rimanenti (ovviamente questo valore è il massimo possibile).

```
acf(TH, lag.max = 23)
```



Come previsto, si nota che tutti i valori delle autocorrelazioni si collocano approssimativamente nell'intervallo (-0.2, 0.2). In base all'ipotesi nulla (di assenza di autocorrelazione) ci si attende che in media al più 1 osservazione ogni 20 sia oltre queste due linee. Nel nostro caso non si riscontrano pertanto particolari autocorrelazioni.

SOLUZIONI ESERCIZIO 5

5.1

Si utilizza la funzione `rep` per creare i 4 vettori di 15 elementi per il gruppo sanguigno

```
A <- rep("A", 15)
B <- rep("B",  15)
AB <- rep("AB", 15)
O <- rep("O",  15)
```

ed i vettori di 60 elementi per il genere

```
MM <- rep("M",  60)
FF <- rep("F",  60)
```

Con la funzione `cbind` si aggregano per colonna i vettori creati e con la funzione `rbind` si ottiene un data frame con due colonne indicanti il gruppo sanguigno ed il genere. Si noti che è possibile assegnare i nomi delle colonne con il comando `names` (o `colnames`).

```
SM <- cbind(c(A, B, AB, O), MM)
SF <- cbind(c(A, B, AB, O), FF)
dat <- data.frame(rbind(SM, SF))
head(dat)
```

V1	MM
A	M
A	M
A	M
A	M
A	M
A	M

```
names(dat) <- c("GR", "GEN")
head(dat)
```

GR	GEN
A	M
A	M
A	M

GR	GEN
A	M
A	M
A	M

5.2

La funzione `table` permette di ricavare la tabella a doppia entrata a partire dal dataframe. Possiamo così verificare il bilanciamento per ogni gruppo sanguigno e per entrambi i generi.

```
table(dat)
```

```
#>     GEN
#> GR    F  M
#>   0  15 15
#>   A  15 15
#>   AB 15 15
#>   B  15 15
```

Osserviamo che per ciascun gruppo sanguigno ci sono 30 soggetti (15 maschi e 15 femmine). Viceversa, per entrambi i generi abbiamo 60 osservazioni, 15 per ognuno dei 4 gruppi sanguigni.

5.3

Il numero identificativo per ogni unità si piò aggiungere in una colonna del data frame. E' sufficiente usare l'operatore `$` per definire la nuova colonna `ID`, e contemporaneamnte assegnare i valori. Di defalut ogni nuova colonna viene aggiunta come ultima colonna nel dataframe; è possibile riordinarle in seguito, per esempio spostando la colonna `ID` in prima posizione. In alternativa è possibile utilizzare nuovamente il comando `cbind`.

```
# n <- nrow(dat)
n <- dim(dat)[1]
# dat <- cbind(ID = 1:n, dat)
dat$ID <- 1:n
head(dat)
```

GR	GEN	ID
A	M	1
A	M	2
A	M	3
A	M	4
A	M	5
A	M	6

```r
dat <- dat[, c(3, 1, 2)]
head(dat)
```

ID	GR	GEN
1	A	M
2	A	M
3	A	M
4	A	M
5	A	M
6	A	M

SOLUZIONI ESERCIZIO 6

6.1

- L'evento di interesse oggetto dell'osservazione è l'assegnazione del punteggio al formaggio cheddar di diverse marche.
- La popolazione è formata dall'insieme di tutti i formaggi prodotti che ricadono nella tipologia cheddar, mentre ogni unità statistica è rappresentata dal singolo formaggio.
- I dati provengono da uno studio osservazionale ed i caratteri disponibili sono tutti quantitativi continui.

6.2

Carichiamo i dati utilizzando il comando `load` e stampiamo le dimensioni del dataframe corrispondente utilizzando la funzione `dim`.

```r
load("percezione.Rdata")
dim(percezione)
```

```
#> [1] 30  4
```

Il dataset è costituito da 30 righe, corrispondenti ai 30 formaggi presi in esame, e 4 colonne, corrispondenti alle 4 caratteristiche (variabili) analizzate: il punteggio e la concentrazione di 3 tipologie di acido. Con la funzione `head` è possibile inoltre stampare le prime 10 righe del dataframe in modo da esaminare in modo preliminare le caratteristiche dei dati.

```r
head(percezione)
```

sapore	acetico	solfidrico	lattico
12.3	4.543	3.135	0.86
20.9	5.159	5.043	1.53
39.0	5.366	5.438	1.57
47.9	5.759	7.496	1.81
5.6	4.663	3.807	0.99
25.9	5.697	7.601	1.09

L'osservazione della prima riga e prima colonna rappresenta il punteggio medio soggettivo assegnato dai giudici al formaggio della prima marca pari a 12.3. Per questa unità si osserva una concentrazione di acido solfidrico (misurata sulla scala del pH[2]) pari a circa 3, di acido lattico pari a 0.86 e di acido acetico pari a 4.5.

Tramite la funzione `skimr::skim` della libreria `skimr` è possibile esplorare i dati. (Le stesse statistiche descrittive possono essere stampate con la funzione `summary`.)

```
skimr::skim_without_charts(percezione)
```

Table 10: Data summary

Name	percezione
Number of rows	30
Number of columns	4
_____	
Column type frequency:	
numeric	4
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
sapore	0	1	24.53	16.26	0.70	13.55	20.95	36.70	57.20
acetico	0	1	5.50	0.57	4.48	5.24	5.42	5.88	6.46
solfidrico	0	1	5.94	2.13	3.00	3.98	5.33	7.57	10.20
lattico	0	1	1.44	0.30	0.86	1.25	1.45	1.67	2.01

Si osserva innanzitutto che non sono presenti dati mancanti ed è pertanto possibile ottenere le statistiche descrittive univariate sulla base di tutte le osservazioni.

Il punteggio assegnato dalla giuria varia da un minimo di 0.7 ad un massimo di 57.2; il campo di variazione è quindi piuttosto ampio. Il punteggio medio assegnato è pari a 24.5. Si nota che la

[2]La scala pH assume solitamente valori tra 0 e 14; la sostanza si dice acida se pH minore di 7, basica se pH maggiore di 7 e neutra se pH uguale a 7. La Food and Drug Administration riporta per il formaggio cheddar un pH medio di 5.9.

metà dei formaggi considerati ha ricevuto un punteggio minore o uguale a 21 (mediana). Un quarto dei formaggi ha ricevuto un punteggio superiore a 36.7 (terzo quartile). Questi valori indicano la presenza di una leggera asimmetria, con una coda più pesante a destra.

E' anche interessante valutare quali sono i valori delle altre variabili corrispondenti al massimo punteggio assegnato.

```
ind <- which.max(percezione$sapore)
percezione[ind, ]
```

	sapore	acetico	solfidrico	lattico
12	57.2	6.446	7.908	1.9

Si nota che il formaggio che ha ricevuto maggior gradimento presenta valori superiori alla media per tutti e tre i tipi di acido.

Le tre variabili riguardanti la concentrazione di acido presentano campi di variazione piuttosto diversi tra loro; quello più ampio è riferito all'acido solfitrico che varia da 3 a 10. Al contrario acido acetico e acido lattico hanno campi di variazione molto più ristretti, da 4.5 a 6.5 e da 0.8 a 2 rispettivamente. In tutti e tre i casi i valori di media e mediana sono quasi coincidenti: pertanto metà dei formaggi hanno concentrazione di acido superiore alla media (e l'altra metà inferiore).

Per quanto riguarda la variabilità, la funzione `skim` riporta il valore dello scarto quadratico medio (deviazione standard); questo è molto basso per acido acetico e acido lattico, molto più elevato per l'acido solfitrico: questo ha una variabilità media intorno alla sua media di circa 2.1.

Per il coefficiente di variazione ricorriamo invece alla funzione `apply` che permette di calcolare una quantità per tutte le righe o le colonne del dataframe. La funzione richiede come argomenti:

- il nome assegnato al dataframe;
- il numero indicante la dimensione dei dati rispetto a cui si vuole eseguire il calcolo (2 per indicare le colonne, 1 per le righe);
- il nome della funzione da applicare.

```
cv <- function(x) {sd(x)/mean(x)}
apply(percezione, 2, cv)
```

```
#>     sapore    acetico solfidrico    lattico
#>  0.6625835  0.1038332  0.3579540  0.2104647
```

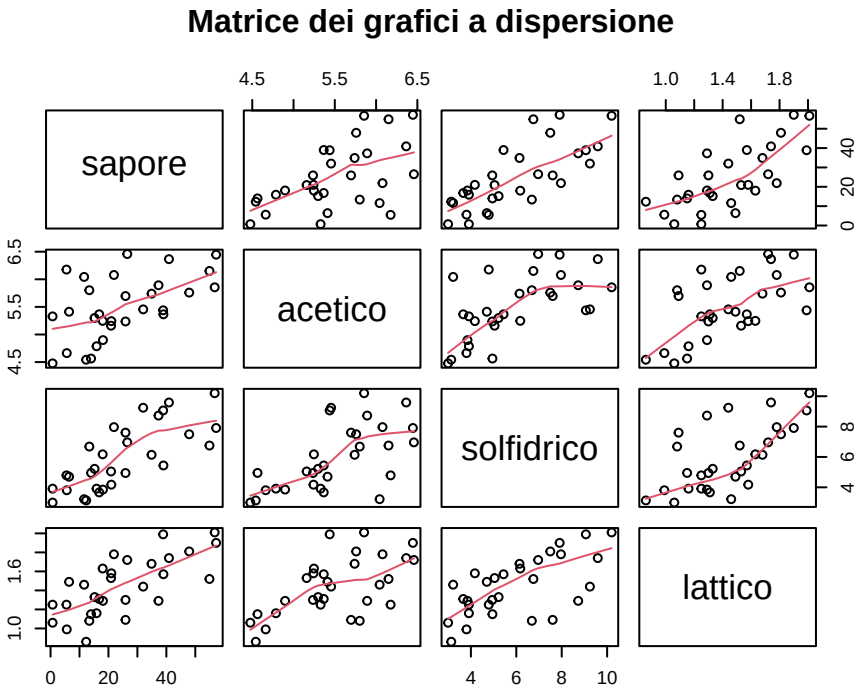
Osserviamo che, per tutte e tre le variabili, il valore del coefficiente di variazione è piuttosto contenuto, mai superiore al 35% della media. Solo nel caso della variabile punteggio si riscontra un valore più elevato.

6.3

La matrice dei diagrammi di dispersione si ottiene con la funzione `pairs` che richiede i seguenti argomenti:

- il nome del dataset;
- l'opzione `panel.smooth` per aggiungere la visualizzazione della linea di tendenza dell'associazione;
- l'opzione `main` per assegnare il titolo al grafico.

```
pairs(percezione,
      panel = panel.smooth,
      main = "Matrice dei grafici a dispersione")
```



Notiamo innanzitutto che, per ragioni di simmetria, è sufficiente analizzare solamente i grafici posti sopra (o equivalentemente sotto) la diagonale.

Nel secondo grafico della prima riga si nota che la linea di tendenza è lineare crescente e che il campo di variazione del punteggio (asse y) è ampio, mentre quello dell'acido acetico (asse x) non è molto elevato (4.5-6.5). C'è maggiore variabilità dei punti dell'acido acetico per i valori superiori a 5.5. I punti nei grafici della prima riga terza e quarta colonna (relativi al punteggio vs concentrazione di acido solfidrico e lattico) si dispongono in modo lineare crescente, i punti sono sparsi nel campo di variazione e non si notano punti particolarmente distanti dalla linea di tendenza. L'andamento dei punti del grafico della prima riga e terza colonna è sempre crescente anche se è meno lineare rispetto a quello degli altri due grafici precedenti. Non si notano anche in questo caso punti particolarmente distanti dalla linea di tendenza.

Si nota infine una tendenza lineare crescente anche tra le coppie delle variabili che rappresentano le concentrazioni di acido acetico e solfidrico, e lattico.

6.4

Per compilare il file sia in formato HTML sia in formato Word è sufficiente specificare nel preambolo (sotto le indicazioni di titolo, autore e data) come segue:

```
output:
  word_document: default
  html_document: default
```

Per estrarre il codice `R` presente nel file si utilizza il comando `purl` presente nella libreria `knitr`, e specificando come unico argomento il nome del file .Rmd; per esempio: `knitr::purl("Es1.Rmd")`.

SOLUZIONI ESERCIZIO 7

7.1

Si riportano i valori nei rispettivi vettori utilizzando il comando `c`:

```
spesa <- c(50, 51, 60, 60, 54, 52, 45, 66)
epa <- c(15, 20, 25, 23, 18, 19, 23, 21)
giorni <- c(245, 200, 400, 400, 250, 280, 352, 360)
```

7.2

Consideriamo ora il vettore relativo alla riduzione della spesa sanitaria; utilizziamo le funzioni `mean` e `median` per calcolare media e mediana rispettivamente.

```
mean(spesa)
```

```
#> [1] 54.75
```

```
median(spesa)
```

```
#> [1] 53
```

Risulta che in media, negli 8 paesi considerati, la spesa sanitaria ha subito una riduzione di 54,750 euro. Si osserva inoltre che metà dei paesi osservati presenta una riduzione della spesa superiore a 53 euro. Media e mediana assumono infatti valori piuttosto simili tra loro.

7.3

Si crea il data frame utilizzando l'omonima funzione di `R`:

```
data <- data.frame(spesa, epa, giorni)
```

Per ottenere le principali statistiche descrittive utilizziamo la funzione `skim` del pacchetto `skimr`.

```
skimr::skim_without_charts(data)
```

Table 13: Data summary

Name	data
Number of rows	8
Number of columns	3
_____	
Column type frequency:	
numeric	3
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
spesa	0	1	54.75	6.78	45	50.75	53.0	60	66
epa	0	1	20.50	3.21	15	18.75	20.5	23	25
giorni	0	1	310.88	76.80	200	248.75	316.0	370	400

Partendo dalla variabile **spesa**, di cui abbiamo già analizzato media e mediana, osserviamo che il campo di variazione è piuttosto limitato, compreso tra un minimo di 45 ed un massimo di 66. Un quarto dei paesi ha una riduzione della spesa inferiore a 50000 euro, un quarto superiore a 60000 euro (notiamo che primo e terzo quartile sono approssimativamente equidistanti rispetto ai valori centrali). Lo scarto quadratico medio per la riduzione di spesa mostra che la distanza media tra i termini della distribuzione e la loro media aritmetica è pari a circa 7000 euro (esprime la dispersione media rispetto al valore medio della distribuzione).

Anche per quanto riguarda il numero di persone malate di epateite c, il campo di variazione è ristertto; si va infatti da un minimo di 15 malati, ad un massimo di 25. L'indice di variabilità rispetto al numero medio di persone malate è pari a circa 3, che se confrontato alla media pari a 20.5, si traduce in una variabilità molto bassa.

Il valore della deviazione standard è invece molto più alto per il numero di giorni in cui non è stata obbligatoria a profilassi, pari a circa 67 giorni. In particolare si va da un minimo di 200 giorni ad un massimo di 400. Solo un quarto dei paesi ha un numero di giorni superiore a 370, approssimativamente un anno. In media il numero di giorni in cui la profilassi non è stata obbligatoria è di circa 310.

Per calcolare il coefficiente di variazione delle tre variabili si utilizza la funzione `apply`, specificando come argomenti il nome del dataframe, il numero 2 per indicare le colonne, e il nome della funzione da applicare.

```
cv <- function(x) {sd(x)/mean(x)}
apply(data, 2, cv)
```

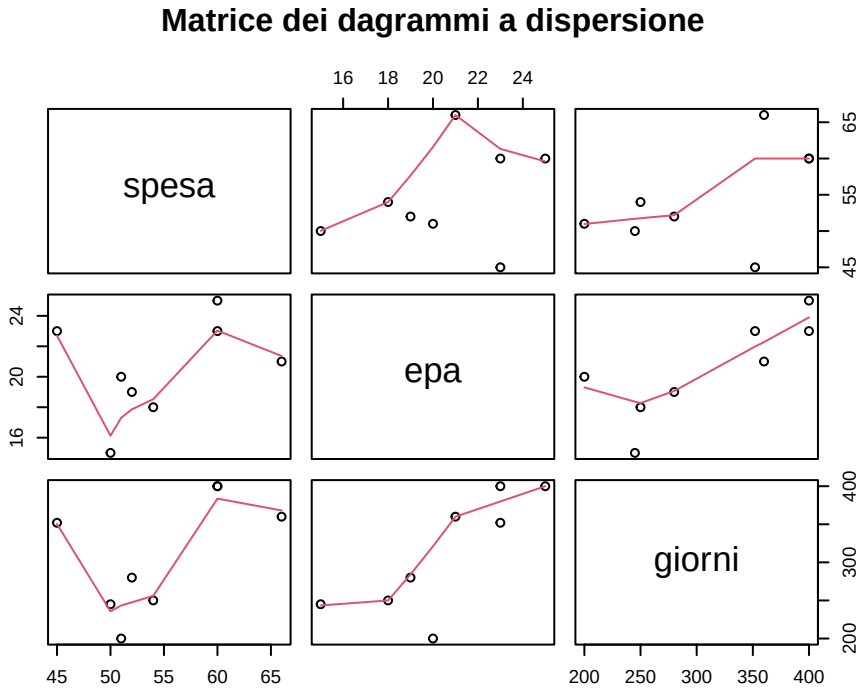
```
#>     spesa       epa    giorni
#> 0.1237820 0.1564456 0.2470303
```

Osserviamo che per tutte e tre le variabili il coefficiente di variazione è piuttosto basso, in ogni caso inferiore al 25% delle rispettive medie.

7.4

Si utilizza la funzione `pairs` per visualizzare la matrice dei diagrammi a dispersione. L'opzione `panel = panel.smooth` consente di aggiungere la visualizzazione della linea di tendenza dell'associazione.

```
pairs(data,
      panel = panel.smooth,
      main = "Matrice dei dagrammi a dispersione")
```



Consideriamo dapprima il grafico posto in prima riga e seconda colonna della matrice (associazione tra variabili **spesa** ed **epa**):

- Si nota che la riduzione di spesa ha un campo di variazione piuttosto ampio.
- Le 8 osservazioni sono distribuite nell'intero piano. Evidenziano una tendenza lineare crescente per l'associazione della riduzione della spesa rispetto al numero di malati.
- Si nota la presenza di un unico punto particolarmente distante dalla linea di tendenza centrale: si tratta di un paese in cui, ad un alto numero di persone con la patologia, corrisponde una lieve riduzione della spesa sanitaria.

Per quanto riguarda il grafico in prima riga e terza colonna della matrice dei diagrammi (variabili **spesa** e **giorni**), si nota invece che i punti sono distribuiti nel piano e la tendenza dell'associazione non è ben chiara. Si evidenzia inoltre la presenza di due punti posti ai margini del piano.

Infine, nel grafico in terza riga e seconda colonna (variabili **epa** e **giorni**) è interessante notare l'associazione lineare crescente evidenziata dalla linea di tendenza tra numero dei malati e giorni di riduzione delle misure di profilassi.

SOLUZIONI ESERCIZIO A

A.1

Carichiamo i dati usando la funzione `load`.

```
load("school.RData")
```

Per l'analisi esplorativa dei dati, possiamo utilizzara, come già visto in precedenza. la funzione `skim` del pacchetto `skimr`.

```
skimr::skim_without_charts(school)
```

Table 15: Data summary

Name	school
Number of rows	388
Number of columns	5
_____	
Column type frequency:	
numeric	5
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
students	0	1	2611.06	3801.45	81.00	373.00	909.50	3142.25	25151.00
calworks	0	1	13.43	10.79	0.00	5.26	10.75	19.14	71.71
expenditure	0	1	5267.55	602.56	3926.07	4894.33	5186.03	5523.18	7711.51
income	0	1	14.08	6.26	0.46	10.03	13.33	17.51	36.17
score	0	1	5.89	1.15	3.32	5.04	5.78	6.60	10.38

La funzione restituisce, in un unico visualizzazione, diverse informazioni sulla struttura oltre alle principali statistiche descrittive univariate del dataset. In particolare:

- In una prima parte dello schema riassuntivo, sono riportate le informazioni generali sul dataset e sulla sua struttura: vediamo quindi che sono elencati il nome dell'oggetto, il numero di righe, corrispondente al numero di osservazioni, e il numero di colonne, che coincide con il numero di variabili.
- In seguito viene poi specificato il tipo delle variabili presenti e l'eventuale presenza di variabili di gruppo.
- L'ultima parte è invece dedicata ad un'analisi descrittiva (univariata) delle diverse variabili presenti; per ciascuna di queste sono quindi riportati: il numero di (eventuali) missing values, il conseguente tasso di completezza delle variabili, le principali statistiche descrittive relative alle singole colonne (media, deviazione standard, minimo e massimo, quartili), e una bozza di istogramma.

Nel caso che stiamo esaminando, in particolare, il dataset è costituito da 388 osservazioni, per un totale di 5 variabili; tutte le variabili sono numeriche. Osserviamo poi che nessuna delle variabili presenta dati mancanti, e dunque il tasso di completezza è pari a 1 per tutte.

- Per quanto riguarda le statistiche descrittive univariate, ci concentriamo dapprima sulla variabile **score**; rileviamo che il campo di variazione è abbastanza ristretto, essendo il valore minimo pari a 3.32 e il valore massimo a 10.38. Dal valore della mediana, possiamo osservare che la metà dei distretti scolastici considerati ha ottenuto un punteggio minore o uguale a 5.78. La media assume un valore molto simile (approssimativamente 5.89), e questo può portare a ipotizzare preventivamente una discreta simmetria della distribuzione. Dall'analisi dei quartili, otteniamo che un quarto dei distretti esaminati ha ottenuto un punteggio inferiore a 3.32, e un altro quarto superiore a 6.60. Anche l'osservazione dei quartili porta a confermare l'ipotesi di una buona simmetria della distribuzione. Come indice di variabilità è riportata la deviazione standard (scarto quadratico medio): la variabile **score** ha una variabilità piuttosto bassa, presentando un valore pari a 1.15; possiamo interpretare questo risultato affermando che il punteggio ottenuto dai distretti scolastici presenta una variabilità media intorno alla media è di circa ± 1.15.
- Passando alle altre variabili, possiamo osservare per esempio che le varaibili **students** ed **expenditure** presentano una variabilità estremamente elevata. Il numero totale di studenti iscritti, in particolare, ha una variabilità media intorno alla media pari a circa ± 3800, e anche il range riferito a questa variabile è molto ampio; è facile interpretare questi dati affermando che le dimensioni dei diversi distretti variano notevolmente: ci sono distretti scolastici più piccoli, con pochissimi studenti iscritti (81, come valore minimo), e distretti scolastici di dimensioni molto maggiori, con un numero assai superiore di studenti iscritti (fino a oltre 25 mila). A

tal proposito, la grossa differenza tra i valori di mediana e media mette in evidenza una forte asimmetria (coda molto pesante verso destra) della distribuzione.
- Le due rimanenti variabili **calworks** e **income** presentano invece valori dello scarto quadratico medio molto più contenuti, seppure non bassissimi. Per quanto riguarda la variabile **calworks**, inoltre, possiamo osservare che è l'unica che assume un valore minimo pari a zero (tutte le altre assumono solo valori strettamente positivi): ci sono quindi alcuni distretti scolastici che non hanno alcuno studente idoneo a beneficiare di assistenza economica. Metà dei distretti scolastici hanno invece una quota di studenti idonei ad avere assistenza economica che è superiore al 10%, un quarto dei distretti addirittura una quota superiore approssimativamente al 20%. Il massimo infine è pari all'incirca al 70%, ma trattandosi di una distribuzione fortemente asimmetrica, è plausibile che si tratti di un caso abbastanza isolato.

Selezioniamo ora la riga corrispondente all'osservazione richiesta; per selezionare il distretto che ha ottenuto il punteggio meno elevato utilizziamo la funzione `which.min()`:

```
i <- which.min(school$score)
school[i, ]
```

	students	calworks	expenditure	income	score
88	146	25.3425	6231.602	7.105	3.316468

Confrontando i risultati ottenuti con le statistiche univariate appena commentate, osserviamo che tale distretto scolastico è caratterizzato da un basso numero di studenti (appena 146), da una percentuale di studenti con assistenza economica superiore alla media (25%, la media è intorno al 13%), da un reddito al di sotto della media (posizionandosi inoltre prima del primo quartile, possiamo afferamre che appartiene al 25% dei distretti con un reddito più basso), e da una spesa per studente lievemente al di sopra della media (superiore anche al terzo quartile).

A.2

Per il calcolo della matrice delle correlazioni, utilizziamo la funzione `cor()`; dal momento che le variabili disponibili sono tutte numeriche, possiamo applicare direttamente la funzione al dataframe dei dati; nei casi in cui siano presenti anche variabili categoriali, è necessario escludere le colonne corrispondenti da questo studio, in quanto l'indice di correlazione di Pearson si applica solamente a variabili quantitative.

```
cor(school)
```

```
#>                students    calworks  expenditure      income        score
#> students     1.00000000  0.05118819 -0.125909903  0.04575825  0.366929825
#> calworks     0.05118819  1.00000000  0.176402279 -0.39484545 -0.271006087
#> expenditure -0.12590990  0.17640228  1.000000000  0.10307919 -0.002633605
#> income       0.04575825 -0.39484545  0.103079194  1.00000000  0.511776306
#> score        0.36692982 -0.27100609 -0.002633605  0.51177631  1.000000000
```

La matrice di correlazione (campionaria) che otteniamo è una matrice (ovviamente) quadrata e simmetrica; possiamo quindi considerare solamente la parte triangolare superiore o inferiore (sulla diagonale sono presenti i valori delle correlazioni di ciascuna variabile con se stessa, ovvero le varianze e pertanto pari a 1).

Per quanto riguarda la variabile **score**, osserviamo intanto un valore approssimativamente nullo in corrispondenza della correlazione con la variabile **expenditure**: non rileviamo quindi alcuna associazione tra il punteggio ottenuto dal distretto e la spesa per studente.

Osserviamo invece un valore abbastanza elevato ad esempio tra le variabili **score** e **income**, pari a 0.51; si osserva quindi un'associazione positiva tra il profitto medio di un distretto scolastico e il punteggio da esso ottenuto (all'aumentare del profitto, si incrementa il punteggio). Analogamente per le variabili **score** e **students** (coefficiente di correlazioni positivo: 0.36)

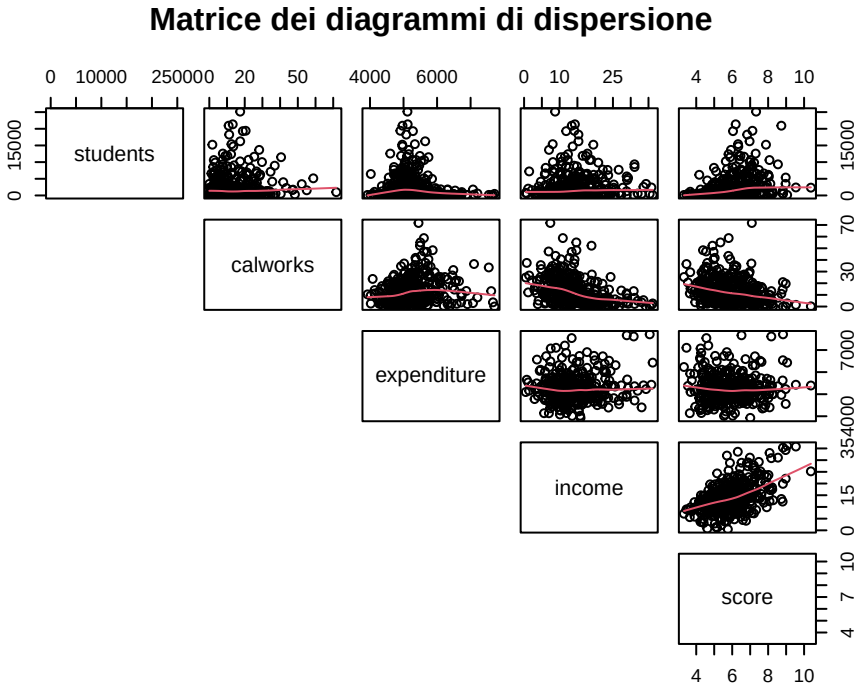
Viceversa si riscontra un'associazione negativa tra la percentuale di studenti idonei a beneficiare di un sostegno economico e il punteggio (all'aumentare del numero di studenti con assistenza economica, il punteggio ha un decremento): il valore del coefficiente di correlazione tra queste due variabili è infatti pari a -0.27. Nuovamente, è analoga la situazione considerando la coppia di variabili **calworks** e **income**, che presentano un valore pari a -0.39.

Osserviamo infine che tutte le rimanenti coppie di variabili presentano tra di loro coefficienti di correlazione piuttosto bassi.

A.3

Per ottenere la matrice dei diagrammi a dispersione, utilizziamo la funzione `pairs()`; l'opzione `panel.smooth` consente di aggiungere al grafico anche le linee di tendenza dell'associazione (stimata non parametricamente, tramite interpolazione); l'opzione `lower.panel = NULL` consente di plottare solo la parte triangolare superiore della matrice (che è simmetrica, dunque i grafici si ripetono).

```
pairs(school,
      panel = panel.smooth,
      lower.panel = NULL,
      main = "Matrice dei diagrammi di dispersione")
```



Attraverso l'esame dei grafici riportati, è utile valutare la presenza di punti particolarmente significativi o anomali, o di pattern particolari. Per esempio, nel grafico tra le variabili **score** e **income**, evidenziamo la presenza di un piccolo gruppo di osservazioni che si discostano in modo significativo dal resto della nuvola di punti: si tratta di un ristretto numero di distretti scolastici che hanno ottenuto un punteggio generalmente superiore rispetto agli altri, e hanno un profitto medio notevolmente più elevato. Un altra osservazione interessante riguarda invece la prima riga di grafici, relativi alla variabile **students**; qui, in tutti i casi, si nota una nuvola di punti molto concentrata nella parte bassa, corrispondente a distretti scolastici con un numero di studenti iscritti non estremamente elevato, con punti molto più dispersi in tutta la parte superiore; questo conferma una serie di constatazioni già effettuate precedentemente: il numero di studenti iscritti è caratterizzato da una variabilità particolarmente elevata e da una distribuzione molto asimmetrica, con una coda molto pesante verso destra (valori alti della variabile). O ancora, in tutti i grafici della colonna relativa alla variabile **income** è presente un piccolo raggruppamento di punti lievemente discostato rispetto alla nuvola principale; si tratta di distretti scolastici con un profitto particolarmente elevato.

Soluzioni esercizi "Generazione e valutazione di una serie di numeri pseudo-casuali"

SOLUZIONI ESERCIZIO 8

8.1

Si tratta di un generatore di tipo congruenziale misto, in cui moltiplicatore, incremento e modulo sono pari rispettivamente a 26, 5 e 106. Il periodo può essere ottenuto dai valori dei parametri del generatore; è noto che il valore del periodo è *al più*, m, ma può anche essere minore. E' comunque possibile affermare che i valori generati diversi tra loro sono al più m, quindi al più 106 in questo caso.

8.2

Si implementa il codice per ottenere numeri pseudo-casuali dal generatore in esame.

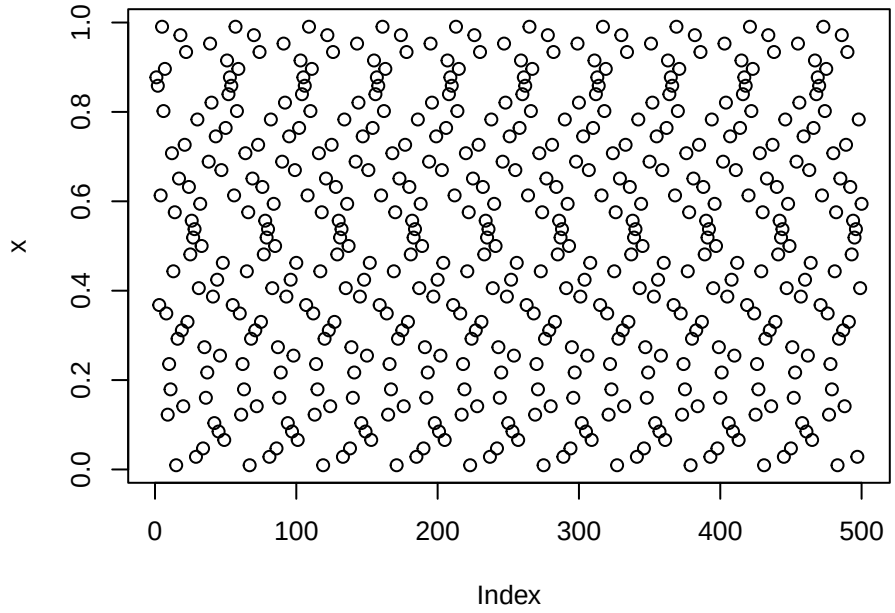
```
gen <- function(a, c, m, x1, niter) {
  x <- numeric(length = niter)
  for (i in 1:niter) {
    x1 <- (a*x1 + c) %% m
    x[i] <- x1 / m
  }
  return(x)
}

gen(a = 26, c = 5, m = 106, x1 = 36, niter = 10)
```

```
#>  [1] 0.8773585 0.8584906 0.3679245 0.6132075 0.9905660 0.8018868 0.8962264
#>  [8] 0.3490566 0.1226415 0.2358491
```

8.3

Il valore del modulo ($m = 106$) suggerisce già che la principale criticità del generatore sia legata al numero limitato di diversi valori che possono essere generati (al più 106). Proviamo quindi ad aumentare il numero di generazioni (per esempio, da 10 a 500) e li rappresentiamo graficamente per valutarne la periodicità.

```
x <- gen(a = 26, c = 5, m = 106, x1 = 36, niter = 500)
plot(x)
```



Il grafico ottenuto mostra un'evidente struttura periodica, per altro con periodo inferiore rispetto ad $m = 106$. Calcolando il numero di valori *unici* ottenuti, possiamo dedurre che in questo caso il periodo è pari a 52 (molto basso).

```
length(unique(x))
```

```
#> [1] 52
```

E' possibile modificare il generatore proposto, per esempio aumentando il valore di m (e.g. $m = 2^{32}$) e considerare $a = 25$ anziché $a = 26$.

```
x <- gen(a = 25, c = 5, m = 2^32, x1 = 36, niter = 500)
length(unique(x))
```

```
#> [1] 500
```

SOLUZIONI ESERCIZIO 9

9.1

Si generano 1000 realizzazioni dalle due variabili casuali uniformi richieste:

```r
a1 <- 0
b1 <- 1
set.seed(163)
ran1 <- runif(1000, a1, b1)

a2 <- 2
b2 <- 3
set.seed(163)
ran2 <- runif(1000, a2, b2)
```

Vlautiamo se le realizzazioni ottenute sono coerenti con la distribuzione di riferimento calcolando i valori di media e varianza campionarie:

```r
mean(ran1)
```

```
#> [1] 0.4993001
```

```r
var(ran1)
```

```
#> [1] 0.08412021
```

```r
#
mean(ran2)
```

```
#> [1] 2.4993
```

```r
var(ran2)
```

```
#> [1] 0.08412021
```

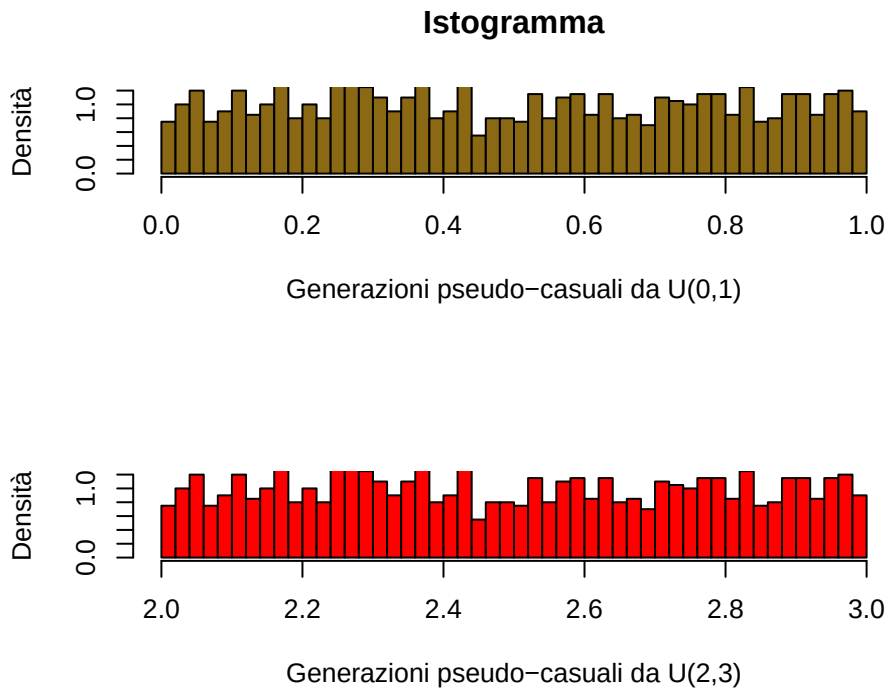
Infatti i valori attesi sono $\mathrm{E}(X) = \frac{1}{2}(a+b)$ ovvero $1/2$ e $5/2$ rispettivamente, mentre la varianza è $\mathrm{V}(X) = \frac{1}{12}(b-a)^2$ ovvero $1/12$.

9.2

Si riportano gli istogrammi aprendo un'unica finestra grafica con 2 righe e 1 colonna tramite il comando `par`

```r
par(mfrow=c(2,1))
hist(ran1, col = "goldenrod4",
     breaks = 50,
     freq = FALSE,
     ylim = c(0,1.2),
```

```
     main = "Istogramma",
     xlab = "Generazioni pseudo-casuali da U(0,1)",
     ylab = "Densità")
#
hist(ran2, col = "red",
     breaks = 50,
     freq = FALSE,
     ylim = c(0,1.2),
     main = "",
     xlab = "Generazioni pseudo-casuali da U(2,3)",
     ylab = "Densità")
```

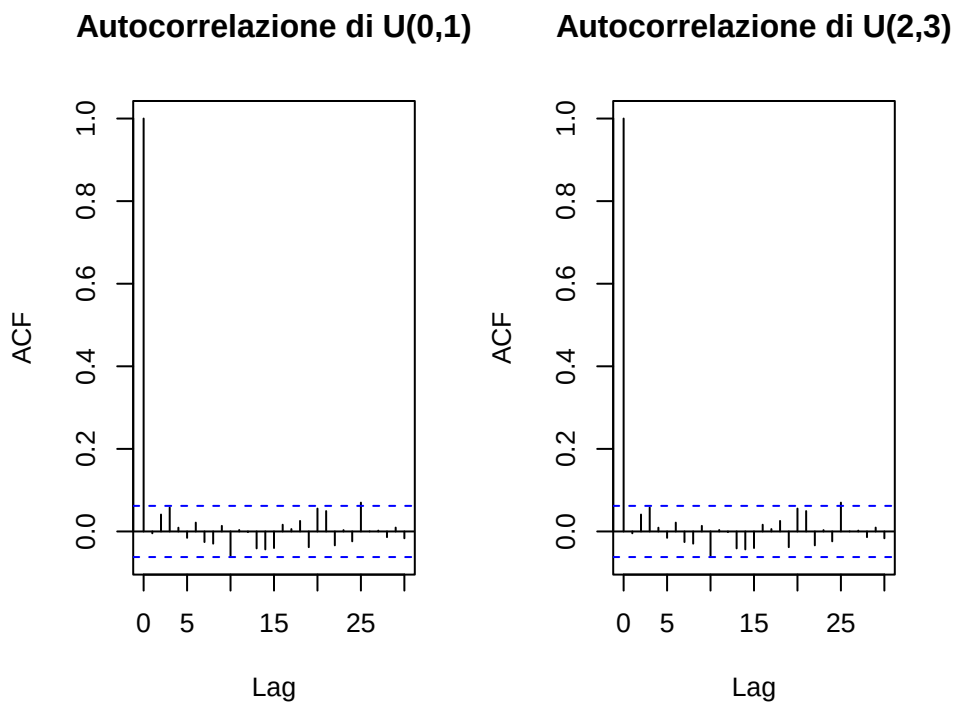


Entrambe le distribuzioni si distribuiscono in modo omogeneo nei rispettivi campi di variazione (0-1 e 2-3 rispettivamente). Non si evidenziano scostamenti particolarmente significativi dalla forma di una distribuzione uniforme teorica. Notiamo infine che le due distribuzioni sono identiche (solamente traslate l'una rispetto all'altra lungo l'asse x); questo è effetto dell'aver definito lo stesso valore per il seme prima di entrambe le generazioni pseudo-casuali (dimostra effettivamente la *non* casualità, ma solo pseudo-casualità del generatore).

9.3

Si disegnano le due funzioni di autocorrelazione in un'unica finestra grafica di 1 riga e 2 colonne usando il lag di default della funzione.

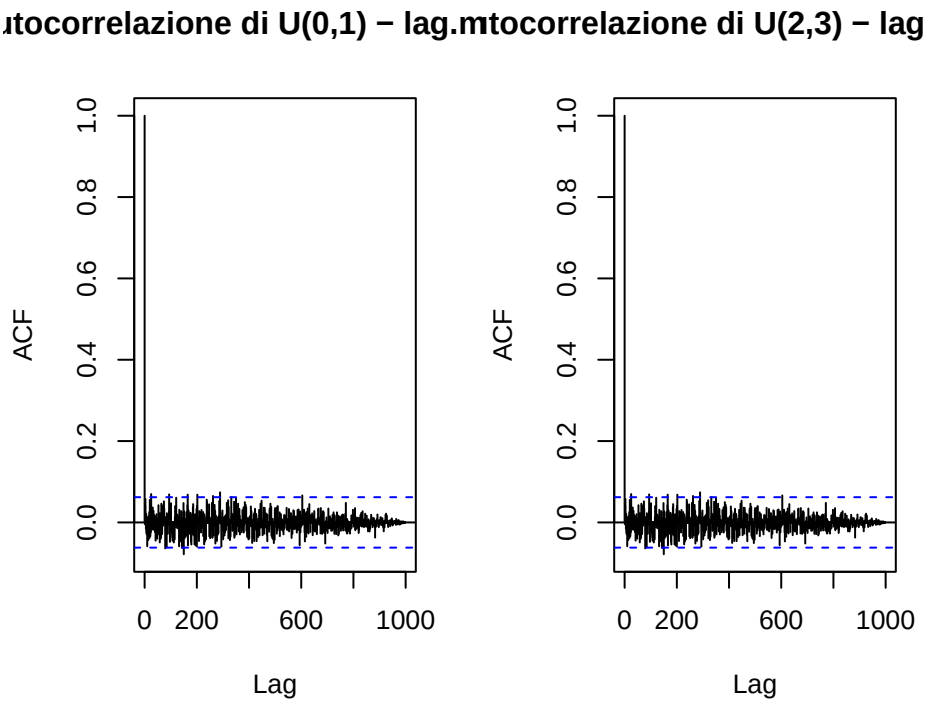
```
par(mfrow=c(1,2))
acf(ran1,
    main = "Autocorrelazione di U(0,1)")
acf(ran2,
    main = "Autocorrelazione di U(2,3)")
```



Osserviamo che il lag massimo è di default posto pari a 30; non si evidenzia la presenza di autocorrelazione, in quanto tutte le barre verticali sono contenute all'interno della zona di riferimento delimitata dalle barre tratteggiate. Anche in questo caso notiamo che i due grafici sono identici.

Si disegnano poi le stesse autocorrelazioni utilizzando n come valore del lag massimo.

```
par(mfrow=c(1,2))
acf(ran1,
    lag.max = 1000,
    main = "Autocorrelazione di U(0,1) - lag.max = n")
acf(ran2,
    lag.max = 1000,
    main = "Autocorrelazione di U(2,3) - lag.max = n ")
```



La situazione non è diversa rispetto al caso precedente, ed in entrambi i grafici si nota che non c'è evidenza di autocorrezione.

SOLUZIONI ESERCIZIO 10

Per leggere il file si utilizza la funzione `read.csv` specificando che non ci sono etichette di colonna con `header=FALSE`.[3] Si noti che nel file csv letto in `R` viene assegnato il nome `V1` all'unica colonna presente.

```
rand1 <- read.csv("rand1.csv", header  = FALSE)
```

10.1

Si analizzano i dati utilizzando la funzione `skim_without_charts`.

```
require(skimr)
skim_without_charts(rand1)
```

[3] E' disponibile anche il dataset in formato .Rdata, in questo caso si usa la funzione 'load' per leggere i dati

Table 18: Data summary

Name	rand1
Number of rows	100
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
V1	0	1	0.57	0.27	0.02	0.4	0.6	0.78	1

Osserviamo che, rispetto all'ipotesi di unifromità dei dati nell'intervallo $[0, 1]$, il valore della media aritmetica (0.57) è leggermente superiore rispetto a quello atteso. Viceversa, si nota che la varianza campionaria (pari a $0.27^2 = 0.0729$) è inferiore a quella attesa (0.083).

10.2

Per la frequenza relativa dei numeri entro un certo intervallo si calcola inizialmente la frequenze assoluta (conteggio) di tali valoiri utilizzando il comando `which` in combinazione con gli operatori logici. Si ottiene poi facilmente la frequenza relativa richiesta dividendo per 100:

```
ind <- which(rand1$V1 > 0.3 & rand1$V1 < 0.7)
length(ind)/100
```

```
#> [1] 0.47
```

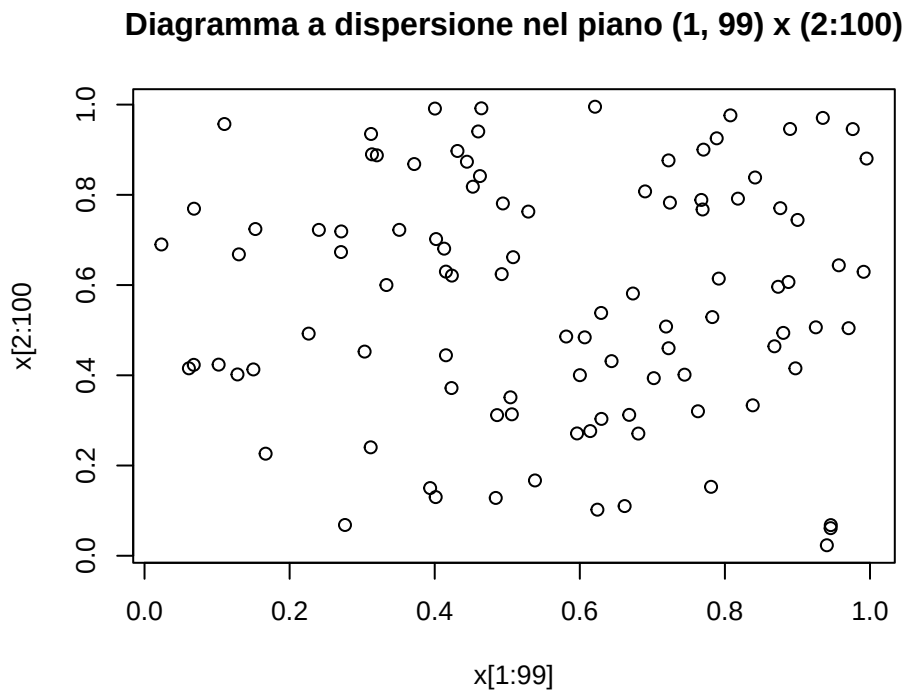
Si ottiene che il 47% dei valori sono superiori a 0.3 e inferiori a 0.7. Questo risultato è superiore a quello atteso sotto l'ipotesi di uniformità della ditsribuzione in [0.1] (il valore atteso sarebbe pari a $0.7 - 0.3 = 0.4$).

10.3

a) Test grafici

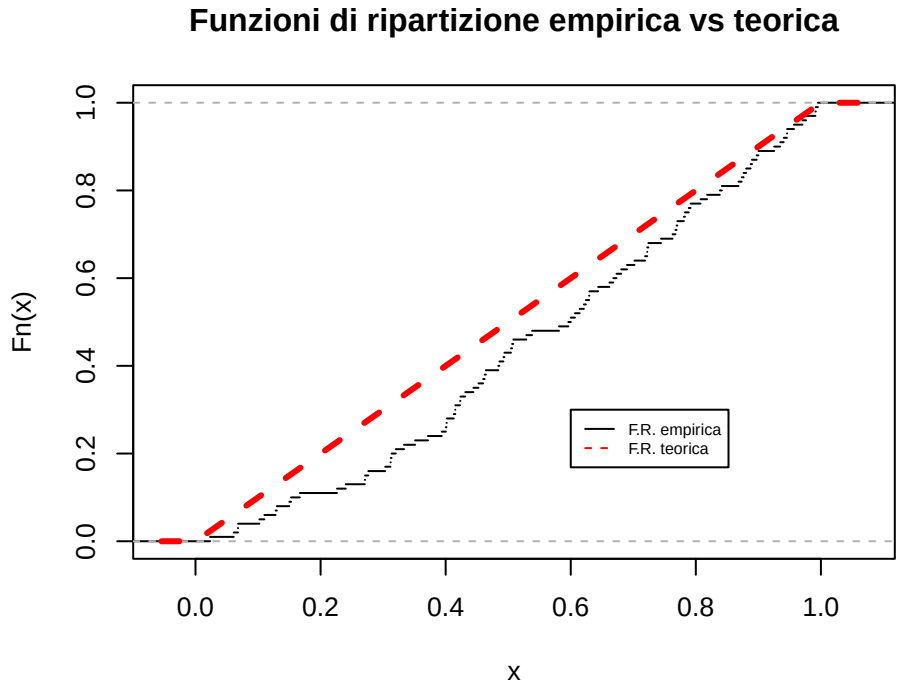
Eseguiamo dapprima i test grafici richiesti per valutare la pseudo-casualità dei numeri della serie.

```
plot(rand1$V1[1:99],
     rand1$V1[2:100],
     main = "Diagramma a dispersione nel piano (1, 99) x (2:100)",
     xlab = "x[1:99]", ylab = "x[2:100")
```



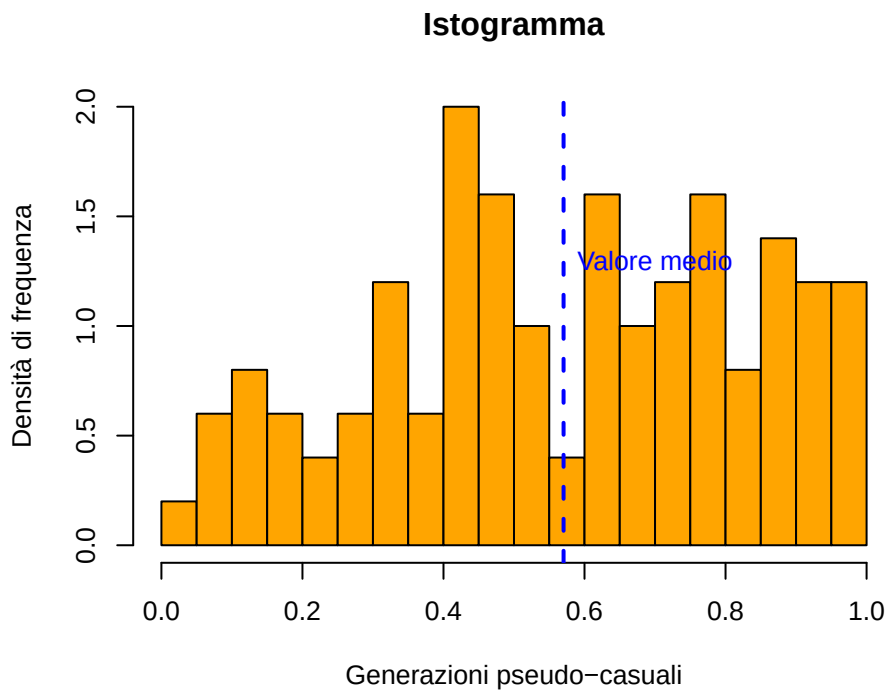
Il primo test grafico è il diagramma a dispersione nel piano $(1, n-1) \times (2, n)$.. In questo caso i punti si dispongono in modo casuale nel piano cartesiano, senza che si evidenzi la presenza di pattern o periodicità particolari. Questo grafico sembra suggerire che vi è indipendenza tra i punti generati.

```
plot(ecdf(rand1$V1),
     do.points=FALSE,
     main = 'Funzioni di ripartizione empirica vs teorica')
curve(punif(x, 0, 1),
      lty = 'dashed',
      col = 'red',
      lwd = 3,
      add = TRUE)
legend(0.6, 0.3,
       col = c("black","red"),
       c("F.R. empirica", "F.R. teorica"),
       lty = c(1, 2),
       cex = 0.6)
```



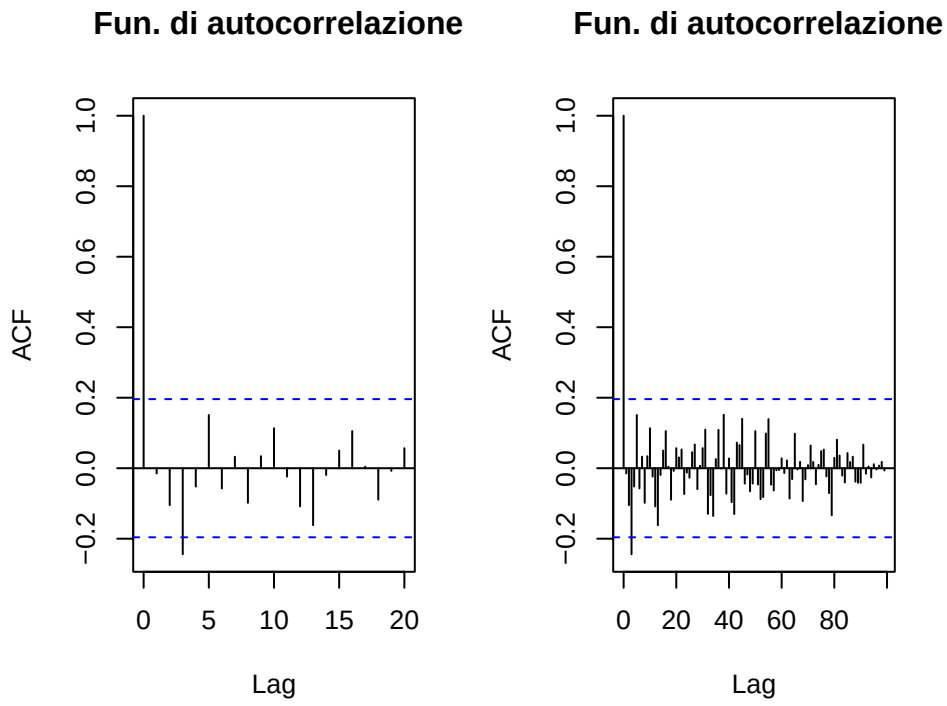
Riportiamo poi il grafico della funzione di ripartizione empirica, confrontandola con quella teorica sotto l'ipotesi di uniformità dei punti. Aggiungiamo poi una legenda al grafico. In questo caso, al contrario del grafico precedente, l'analisi delle funzioni di ripartizione non consente di accettare l'ipotesi che i punti generati provengano da una variabile casuale uniforme sull'intervallo $[0, 1]$. Si nota infatti che la curva relativa alla funzione di ripartizione empirica si discosta in modo sensibile dalla linea tratteggiata relativa alla funzione teorica.

```
hist(rand1$V1, col = 'orange',
     breaks = 20,
     freq = FALSE,
     ylim = c(0, 2),
     main = 'Istogramma',
     xlab = 'Generazioni pseudo-casuali',
     ylab = 'Densità di frequenza')
abline(v=mean(rand1$V1),
       col = 'blue',
       lwd=2, lty=2)
text(0.7, 1.3,
     c("Valore medio"),
     col = "blue")
```



Anche l'istogramma della distribuzione mostra in modo chiaro la non uniformità dei punti generati. Questi si distribuiscono infatti in modo molto più accentuato in alcune regioni dell'intervallo $[0, 1]$. Al contrario nella prima parte dell'intervallo la concentrazione dei punti è molto inferiore a quanto atteso. (Si potrebbe aggiungere la linea della funzione di densità per valutare meglio questo aspetto; in questo caso è costante e pari a 1 su tutto l'intervallo).

```
par(mfrow = c(1, 2))
acf(rand1$V1, main = " Fun. di autocorrelazione")
acf(rand1$V1, main = " Fun. di autocorrelazione", lag.max = length(rand1$V1))
```



Infine rappresentiamo la funzione di autocorrelazione, utilizzando due diversi valori per l'argomento `lag.max` (quello di default ed n). In entrambi i casi si osserva che solo una delle barre verticali finisce (di poco) oltre la regione delimitata dalle linee tratteggaite, segno che non vi è dipendenza tra i punti generati.

In base ai test grafici effettuati, possiamo concludere che, nonostante non vi sia dipendenza tra le osservazioni, la distribuzione empirica non è simile a quella teorica: i dati generati sono indipendenti ma non provengono da una variabile casuale Uniforme sull'intervallo $[0, 1]$.

b) Test statistici

Effettuiamo ora i test statistici per valuatre l'ipotesi di uniformità dei punti generati.

```
ks.test(rand1$V1, "punif")
```

```
#>
#>  Asymptotic one-sample Kolmogorov-Smirnov test
#>
#> data:  rand1$V1
#> D = 0.15366, p-value = 0.01779
#> alternative hypothesis: two-sided
```

Il test di Kolmogorov-Smirnov (utilizzabile attraverso il comando `ks.test` in `R`), è un test non parametrico che consente di confrontare la funzione di ripartizione empirica dei dati con una di

riferimento (quella uniforme in questo caso). La statistica test è calcolata sulla base della massima distanza tra le due funzioni di ripartizione; il valore osservato è pari a 0.15, e il p-value corrispondente 0.02. Questo ci porta a rigettare l'ipotesi nulla di uniformità della distribuzione (il p-value è sufficientemente piccolo: e.g., p-value < 0.05).

Il test Chi quadrato ("Goodness of Fit CHi Squared Test") consente di verificare l'ipotesi nulla che i dati seguano una determinata distribuzione specificata attraverso un vettore di probabilità. Richiede di assegnare ciascuno dei valori osservati ad un determinato sotto-intervallo; si ricorda che nel caso in cui in un determinato sotto-intervallo abbiamo un numero troppo basso di osservazioni (generalmente si considera 5 osservazioni come caso limite), i risultati del test non risultano attendibili. Nel nostro caso suddividiamo l'intervallo $[0, 1]$ in 10 sottointervalli di uguale ampiezza (ampiezza 0.1) e assegnamo le osservazioni a ciascuno di questi. Specifichiamo poi la distribuzione uniforme di riferimento attraverso il vettore che assegna a ciasun sotto-intervallo la stessa probabilità (0.1).

```
foss <- table(cut(rand1$V1, seq(0, 1, by = 0.1)))
foss
```

```
#>
#>   (0,0.1] (0.1,0.2] (0.2,0.3] (0.3,0.4] (0.4,0.5] (0.5,0.6] (0.6,0.7] (0.7,0.8]
#>         4         7         5         9        18         7        13        14
#> (0.8,0.9]   (0.9,1]
#>        11        12
```

```
p <- rep(0.1, 10)
chisq.test(foss, p = p)
```

```
#>
#>  Chi-squared test for given probabilities
#>
#> data:  foss
#> X-squared = 17.4, df = 9, p-value = 0.04281
```

```
qchisq(0.95, df = 9)
```

```
#> [1] 16.91898
```

Osserviamo che in questo caso abbiamo una delle classi con un numero basso di osservazioni (4). Essendo l'unica, ed essendo molto vicino al caso limite, scegliamo comunque di proseguire l'analisi con questo numero di gruppi. Il test Chi Quadrato restituisce un valore della statistica test pari a 17.4, con 9 gradi di libertà (il numero di gradi di libertà è sempre pari al numero di grappi meno 1). Per valutare il risultato del test (fissato un livello di significatività, per esempio $\alpha = 0.05$) ricaviamo il valore critico dalla distribuzione Chi Quadrato a 9 gradi di libertà. Questo valore, risulta pari a 16.92. Il valore osservato della statistica test è maggiore e porta quindi a rigettare l'ipotesi nulla che le osservazioni provengano da una distribuzione uniforme su $[0, 1]$.

SOLUZIONI ESERCIZIO 11

11.1

Si caricano i dati specificando il nome del dataset ed il nome del pacchetto in cui sono contenuti. L'help relativo al dataset, riporta che i valori sono riferiti ad una **stessa persona** e sono rilevati ad intervalli regolari ogni 10 minuti, per 48 volte.

```
data(lh, package = "datasets")
lh <- as.numeric(lh)
head(lh)
```

```
#> [1] 2.4 2.4 2.4 2.2 2.1 1.5
```

Otteniamo le principali statistiche descrittive univariate utilizzando la funzione `skim_without_charts`.

```
skim_without_charts(lh)
```

Table 20: Data summary

Name	lh
Number of rows	48
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

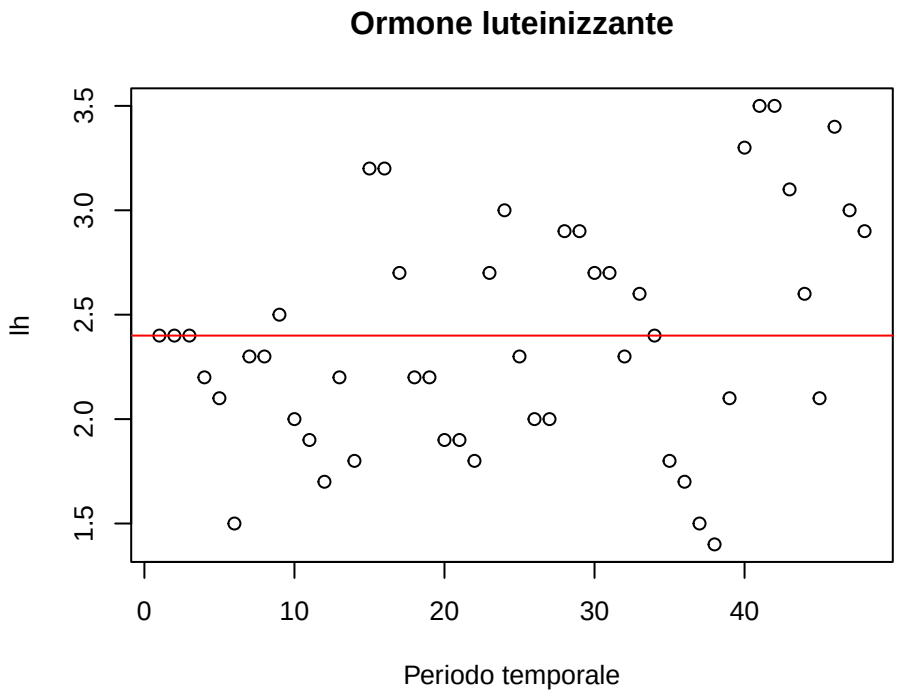
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	2.4	0.55	1.4	2	2.3	2.75	3.5

Il dataset continee un'unica variabile, il livello ormonale, che varia da un minimo di 1.40 ad un massimo di 3.50. La variabilità media intorno al valore medio è 0.55, che confrontata con il valore della media, pari a 2.4, evidenzia una distribuzione poco variabile. Media e mediana sono simili tra loro, per cui la distribuzione empirica sembra essere abbastanza simmetrica.

Possiamo anche ottenere una rappresentazione grafica (grafico a dispersione) dei valori osservati.

```
plot(lh,
     xlab = "Periodo temporale",
     main = "Ormone luteinizzante")
abline(h = mean(lh), col = "red")
```

```
legend(0.6, 0.3,
       col = c("black", "red"),
       c("F.R. empirica", "F.R. teorica"),
       lty = c(1, 2),
       cex = 0.6)
```

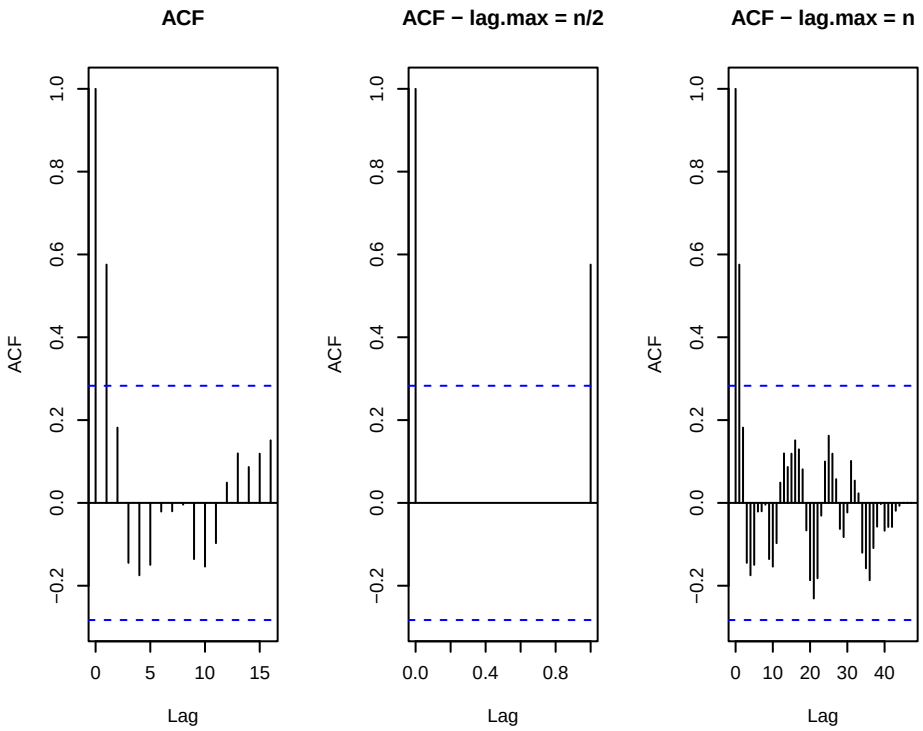


Notiamo che le prime 14 rilevazioni sono tutte inferiori alla media. La variabilità aumenta con le ultime rilevazioni.

11.2

Si disegna la funzione di autocorrelazione usando tre diversi valori per l'argomento `lag.max`.

```
n <- length(lh)
par(mfrow = c(1, 3))
acf(lh,
    main = "ACF")
acf(lh, lag.max = 1,
    main = "ACF - lag.max = n/2")
acf(lh, lag.max = n,
    main = "ACF - lag.max = n")
```



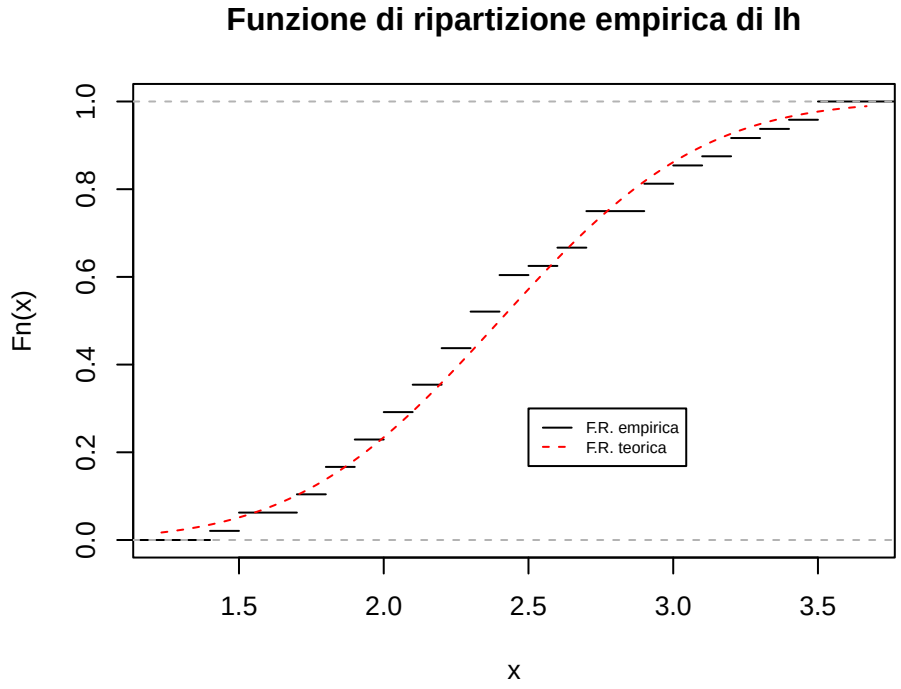
Notiamo che, in tutti e tre i grafici, il primo valore è superiore al valore atteso sotto l'ipotesi di assenza di autocorrelazione. Si deduce quindi che il livello di ormone ha una debole autocorrelazione che cala abbastanza velocemente nel tempo (già al secondo lag rientra all'interno dei valori soglia). Gli altri valori risultano infatti in linea con quelli attesi sotto l'ipotesi assenza di correlazione seriale. Si nota inoltre un andamento a **sinusoide** della funzione nel tempo, con una tendenza a diminuire l'ampiezza (convergenza verso 0).

11.3

Si disegna la funzione di ripartizione empirica, confrontandola con quella di una variabile casuale con distribuzione di Gauss con media e varianza uguali a quelle osservate nei dati campionari.

```
plot(ecdf(lh),
     do.points = FALSE,
     main = 'Funzione di ripartizione empirica di lh')
lm <- mean(lh)
ls <- sd(lh)
curve(pnorm(x, mean = lm, sd = ls),
      lty = 'dashed',
      col = 'red',
      add = TRUE)
legend(2.5, 0.3,
       col = c("black", "red"),
       c("F.R. empirica", "F.R. teorica"),
```

```
       lty = c(1, 2),
       cex = 0.6)
```



Si nota che la funzione di ripartizione empirica ricalca quella teorica; questa rappresentazione sembra quindi confermare che le osservazioni provengono da una distribuzione Gaussiana (univariata).

11.4

Il test non parametrico di Kolmogorov-Smirnov consente di valutare l'adattamento tra le due funzioni tra le due funzioni di ripartizione. Si noti che occorre specificare la distribuzione sotto l'ipotesi nulla (Gaussiana) ed i relativi parametri (calcolati al punto precedente).

```
ks.test(lh,
        "pnorm",
        lm,
        ls, exact = FALSE)
```

```
#>
#>  Asymptotic one-sample Kolmogorov-Smirnov test
#>
#> data:  lh
#> D = 0.10417, p-value = 0.6749
#> alternative hypothesis: two-sided
```

Il valore della statistica test è 0.10 e il livello di significatività osservato (p-value) è elevato, pari a 0.68. Non vi è quindi sufficiente evidenza statistica per rigettare l'ipotesi nulla che le osservazioni provengano da una distribuzione Gaussiana. In base a questo test vediamo che c'è aderenza tra la funzione di ripartizione empirica e quella teorica di una variabile casuale di Gauss con media e varianza uguali a quelle campionarie.

Il warning ottenuto si riferisce al fatto che questo test è sensibile alla presenza di valori identici (ties). Tramite la tabella di frequenza vediamo infatti che quasi tutti i valori si ripetono.

```
table(lh)
```

```
#> lh
#> 1.4 1.5 1.7 1.8 1.9   2 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.9   3 3.1 3.2 3.3 3.4 3.5
#>   1   2   2   3   3   3   3   4   4   4   1   2   4   3   2   1   2   1   1   2
```

Pertanto il valore della statistica test ed il conseguente $p - value$ sono riferiti ad un numero di osservazioni minori rispetto a quelle osservate.

SOLUZIONI ESERCIZIO 12

Si lascia allo studente la descrizione del metodo ed i commenti relativi alla parte teorica.

SOLUZIONI ESERCIZIO 13

13.1

Si descrivono le osservazioni riferite alle ultime tre colonne essendo la prima la colonna riferita all'identificativo del paziente.

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

V1	V2	V3	V4
1	57.7	36.9	38.5
2	51.3	53.0	44.8
3	21.8	17.8	20.6
4	43.5	48.9	51.4
5	10.5	10.1	11.8
6	11.6	9.2	10.0

```
require(skimr)
skim_without_charts(dataeNo[,2:4])
```

Table 23: Data summary

Name	dataeNo[, 2:4]
Number of rows	38
Number of columns	3
_____	
Column type frequency:	
numeric	3
_____	
Group variables	None

Variable type: numeric

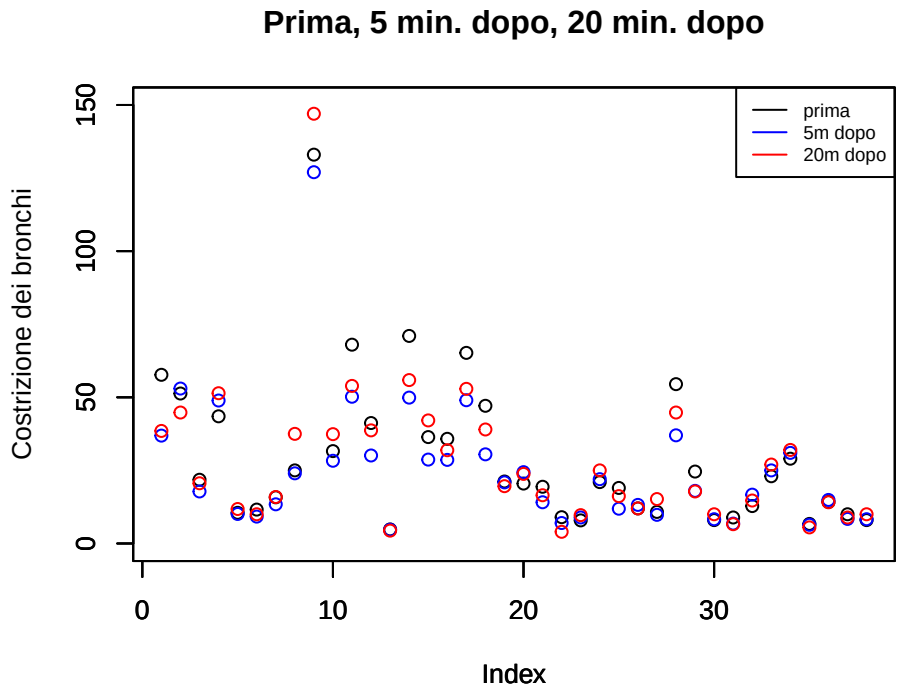
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
V2	0	1	29.26	25.57	4.8	11.00	21.10	40.00	133
V3	0	1	25.09	22.10	4.7	9.88	19.45	30.40	127
V4	0	1	28.08	25.25	4.0	11.85	20.10	38.65	147

Si nota che le tre serie di dati seguono approssimativamente lo stesso andamento, presentando un massimo che si discosta molto da tutti gli altri indici; questo comportamento denota distribuzioni fortemente asimmetriche con una lunga coda a destra. Confrontando le tre diverse serie, si osserva poi che quella relativa alle osservazioni rilevate 5 minuti dopo gli esercizi fisici presenta i valori più bassi per tutti gli indici. L'esercizio fisico sembra quindi ridurre la costrizione dei bronchi, anche se l'effetto è a breve termine. Già 20 minuti dopo, si nota infatti che i valori delle statistiche descrittive tornano ad essere molto vicine a quelle calcolate sui dati rilevati prima degli esercizi fisici (il valore del massimo è addirittura maggiore). Anche il valore della deviazione standard è inferiore sui dati rilevati 5 minuti dopo lo sforzo fisico rispetto alle altre due serie.

Nel seguente chunck si aggiunge allo stesso grafico la seconda e la terza serie con il comando `par(new=TRUE)` e si cambiano i colori.

```
plot(dataeNo[,2],
     type = "p",
     main = "Prima, 5 min. dopo, 20 min. dopo",
     ylim=c(0,150),
     ylab = "Costrizione dei bronchi")
par(new=TRUE)
plot(dataeNo[,3],
     type="p",
     col="blue",
     main=" ",
     ylim=c(0,150),
     ylab = "")
par(new=TRUE)
plot(dataeNo[,4],
     type="p",
```

```
     col="red",
     main=" ",
     ylim=c(0,150),
     ylab = "")

legend("topright",
       col=c("black","blue", "red"),
       c("prima","5m dopo", "20m dopo"),
       lty=c(1,1,1),
       cex=0.7)
```



13.2

Si calcola la differenza interquartile per tutte e tre le variabili direttamente con la funzione `apply`.

```
apply(dataeNo[, 2:4], 2, IQR)
```

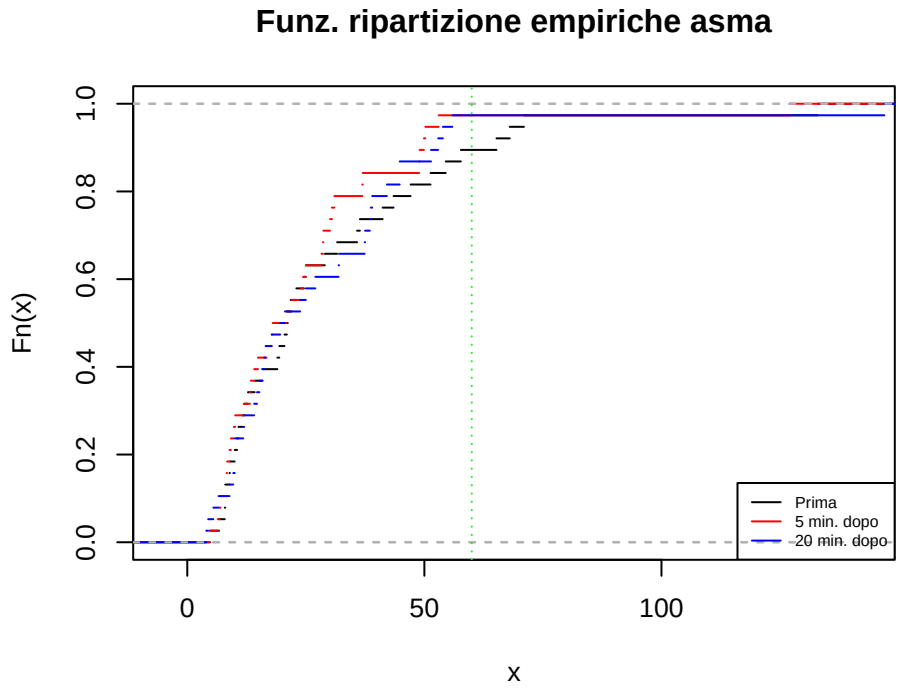
```
#>     V2     V3     V4
#> 29.000 20.525 26.800
```

Come già rilevato per il valore della deviazione standard, si osserva che lo scarto interqiartile è inferiore per i dati raccolti dopo 5 minuti dalla fine dell'esercizio fisico. Lo sforzo fisico, pertanto, oltre a ridurre la misura di costrizione dei bronchi, ne diminuisce anche la variabilità; questa tende subito ad aumentare nuovamente all'allontanarsi dalla fine degli esercizi (dopo 20 minuti).

13.3

Si disegnano le tre funzioni di ripartizione empirche nella stessa finestra grafica (è sufficientemente aggiungere l'opzione `add = TRUE`). Aggiungiamo poi la legenda per consentire di distinguere tra le funzioni delle tre diverse serie.

```
plot(ecdf(dataeNo[, 2]),
     col = "black",
     do.points = FALSE,
     main = "Funz. ripartizione empiriche asma")
plot(ecdf(dataeNo[, 3]),
     col = "red",
     do.points = FALSE,
     add = TRUE)
plot(ecdf(dataeNo[, 4]),
     col = "blue",
     do.points = FALSE,
     add = TRUE)

legend("bottomright",
       col = c("black", "red", "blue"),
       c("Prima", "5 min. dopo", "20 min. dopo"),
       lty = c(1, 1, 1),
       cex = 0.6)

abline(v = 60, lty = "dotted", col = "green")
```



Si osserva che nella prima parte delle distribuzioni (all'incirca per valori inferiori a 30) le tre funzioni seguono lo stesso andamento. Al contrario, per valori superiori a 30 le tre funzioni si differenziano in modo sensibile. Per esempio, la probabilità che la misura di costrizione dei bronchi sia minore di 60 prima dello sforzo fisico è di poco superiore a 0.8 (curva nera), mentre è prossimo a 1 dopo gli esercizi (curva blu e curva rossa).

SOLUZIONI ESERCIZIO B

Si rimanda alle dispense di teoria, alle applicazioni e alle soluzioni degli esercizi precedenti.

Soluzioni esercizi "Generazione di realizzazioni da variabili casuali"

SOLUZIONI ESERCIZIO 14

14.1

Si utilizza la funzione `rexp()` per generare le osservazioni richieste. La funzione richiede solo il numero di punti da generare e il valore di λ (`rate`).

```
x <- rexp(1000, rate = 0.4)
```

Per valutare le osservazioni generate calcoliamo i valori di media e varianza campionarie, confrontandole poi con i valori teorici noti per le variabile casuale esponenziale ($E[X] = \frac{1}{\lambda} = 2.5$ e $V(X) = \frac{1}{\lambda^2} = 6.25$).

```
summary(x)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>  0.0007  0.8434  1.9945  2.7529  3.8826 16.5900
```

```
var(x)
```

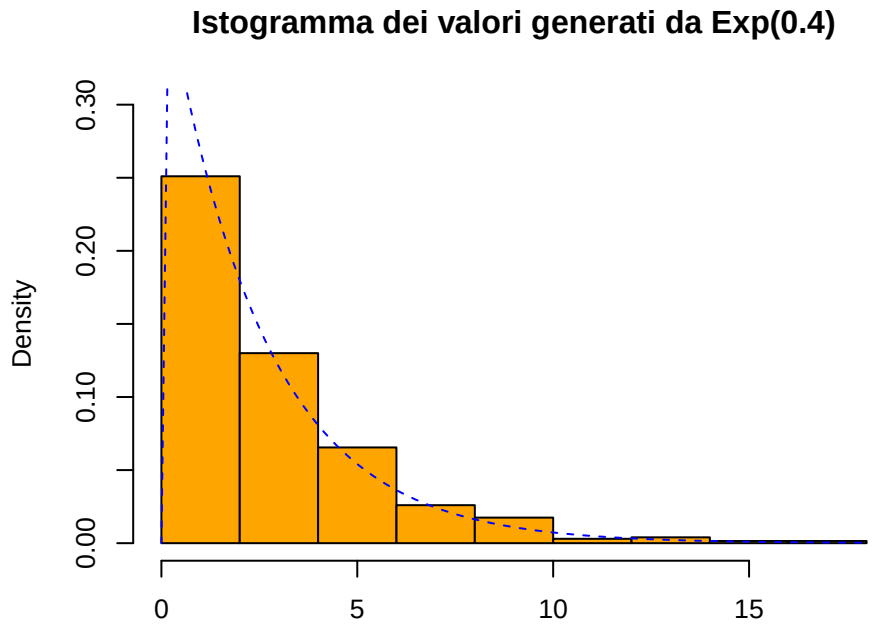
```
#> [1] 6.907431
```

Il valore della media campionaria è 2.75, molto vicino alla media teorica di 2.5 per una variabile Esponenziale con $\lambda = 0.4$. Il valore campionario della varianza (6.91) si discosta invece leggermente da quello teorico (6.25): le osservazioni generate hanno una variabilità leggermente superiore all'atteso. Osserviamo poi che i valori sono tutti positivi (coerentemente con i risultati teorici noti), e che il massimo è superiore a 16 (lunga coda a destra).

14.2

Rappresentiamo la distribuzione empirica sotto forma di istogramma. Sovrapponiamo poi al grafico la curva della distribuzione teorica.

```
hist(x, freq = FALSE,
     xlab = "", main = "Istogramma dei valori generati da Exp(0.4)",
     ylim = c(0, 0.3),
     col = "orange")
curve(dexp(x, rate = 0.4),
      lty = 2, col = "blue", add = TRUE)
```



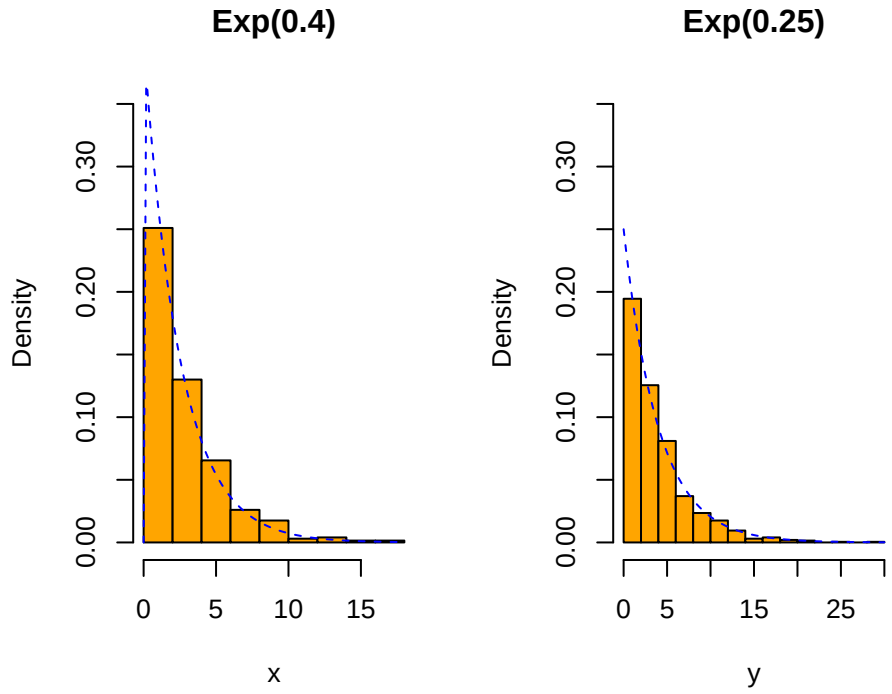
L'istogramma della distribuzione empirica aderisce alla curva della funzione di densità teorica.

14.3

Aggiungiamo ora al grafico precedente la rappresentazione delle nuove osservazioni generate da una variabile Esponenziale con $\lambda = 0.25$.

```
y <- rexp(1000, rate = 0.25)

par(mfrow = c(1, 2))
hist(x, freq = FALSE,
     col = "orange",
```

```
     ylim = c(0, 0.35),
     main = "Exp(0.4)")
curve(dexp(x, rate = 0.4),
      col = "blue", lty = 2,
      add = TRUE)

hist(y, freq = FALSE,
     col = "orange",
     ylim = c(0, 0.35),
     main = "Exp(0.25)")
curve(dexp(x, rate = 0.25),
      col = "blue", lty = 2,
      add = TRUE)
```

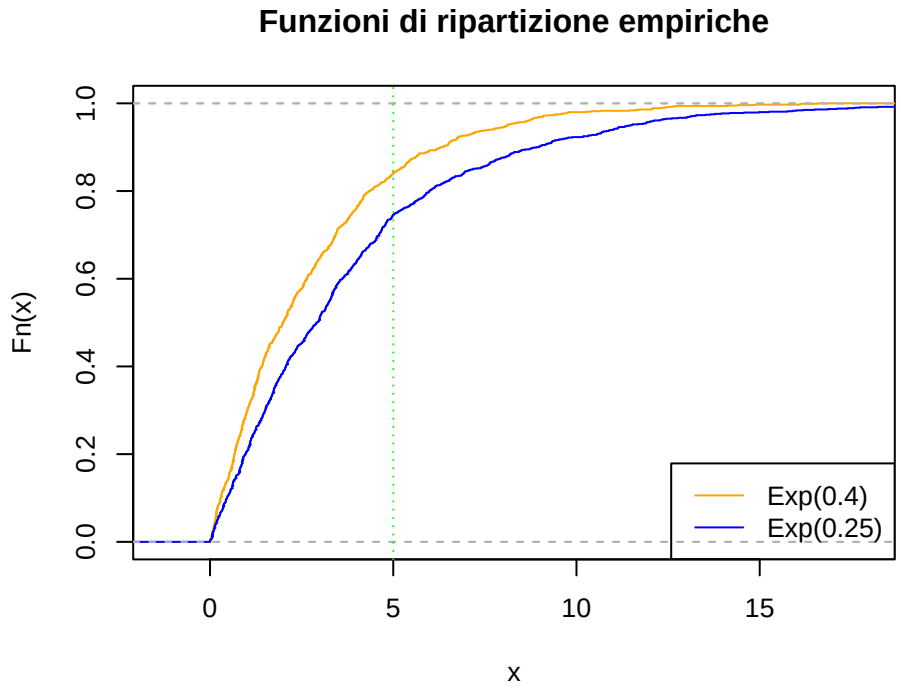


Fissando il range dell'asse Y in modo da poter confrontare le due distribuzioni più facilmente, osserviamo che quella con parametro $\lambda = 0.25$ ha una variabilità molto più elevata (la varianza è del resto inversamente proporzianale al valore di *lambda*). il campo di variazione della prima varia infatti approssimativamente tra 0 e 15, quello della seconda tra 0 e 30. Entrambe gli istogrammi aderiscono alle rispettive curve teoriche.

14.4

Infine rappresentiamo in un unico grafico le due funzioni di ripartizioni empiriche.

```r
plot(ecdf(x),
     col = "orange",
     do.points = FALSE,
     main = "Funzioni di ripartizione empiriche")
plot(ecdf(y),
     col = "blue",
     do.points = FALSE, add = TRUE)

legend("bottomright",
       c("Exp(0.4)", "Exp(0.25)"),
       col = c("orange", "blue"),
       lty = c(1, 1))

abline(v = 5, col = "green", lty = "dotted")
```



Le due funzioni hanno lo stesso andamento; quella relativa alla variabile aleatoria Exp(0.4) presenta probabilità più elevate a parità del valore di x. Per esempio $P(Exp(0.4) \leq 5) \approxeq 0.8$, mentre $P(Exp(0.25) \leq 5) \approxeq 0.7$.

SOLUZIONI ESERCIZIO 15

15.1

Dopo aver fissato il valore del seme, si generano 1000 determinazioni dalla V.C. di Gauss utilizzando la funzione `rnorm`. Si noti che questo comando richiede, oltre al numero di realizzazioni da generare, i valori di media e deviazione standard (**non** varianza).

```
set.seed(1734)
n <- 1000
X <- rnorm(n, mean = 175.6, sd = 7.1)
```

Per illustrare i valori ottenuti calcoliamo le principali statistiche univariate.

```
summary(X)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   154.8   171.0   175.4   175.6   180.3   199.6
```

```
sd(X)
```

```
#> [1] 7.227039
```

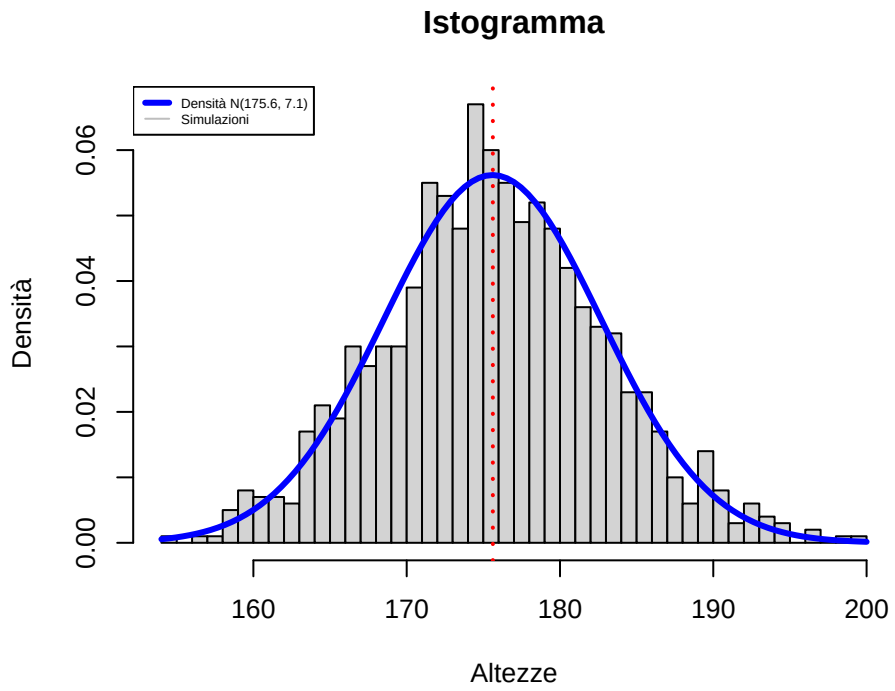
Osserviamo in particolare che il valore della media campionaria, pari a 175.6, coincide con quello della parametro teorico utilizzato per la realizzazione dei punti. Anche la deviazione standard campionaria si avvicina molto al valore teorico suggerito dal testo dell'esercizio. Media e mediana coincidono, suggerendo la simmetria della distribuzione. Minimo e massimo (anch'essi disposti in modo simmetrico rispetto al centro della distribuzione) indicano un range pari a circa 45. Ricordando che per la V.C. la maggior parte delle osservazioni deve trovarsi tra i valori $\mu \pm 3\sigma = 175.6 \pm 3 \cdot 7.1$, ovvero tra 154.3 e 196.9, è ulteriormente confermata la corretta normalità delle osservazioni generate.

15.2

Si rappresentano i dati generati sotto forma di istogramma e si confronta il risultato con la curva della densità teorica.

```
hist(X,
     main = "Istogramma",
     freq = FALSE,
     breaks = 60,
     xlab = "Altezze",
     ylab = "Densità")

abline(v = mean(X), lty = 3, col = "red", lwd = 2)
curve(dnorm(x, 175.6, 7.1),
```

```
      add = TRUE,
      col = "blue",
      lwd = 3)
legend("topleft",
       c("Densità N(175.6, 7.1)", "Simulazioni"),
       col= c("blue", "gray"),
       lty = c(1, 1),
       lwd = c(3, 1),
       cex = 0.5)
```



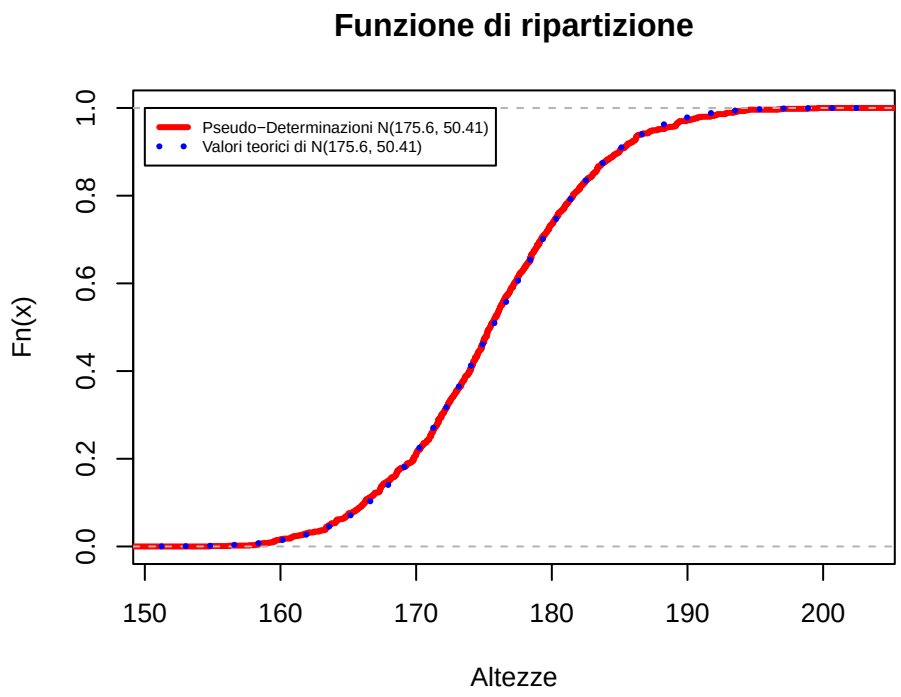
Si nota innanzitutto che la curva teorica si sovrappone quasi perfettamente alla forma definita dalle barre dell'istogramma, segno che le 1000 generazioni ottenute al punto precedente provengono effettivamente da una V.C. di Gauss con i parametri specificati. Si nota poi, come già osservato in precedenza, la simmetria della distribuzione. La media, infine, si dispone in corrispondenza del picco centrale della distribuzione.

15.3

Si rappresentano le funzioni di ripartizione empirica e teorica nello stesso grafico.

```
plot(ecdf(X),
     main = "Funzione di ripartizione",
     col = "red",
     lty = 1,
```

```
     lwd = 3,
     xlab = "Altezze")
curve(pnorm(x, 175.6, 7.1),
      add = TRUE,
      col = "blue",
      lty = 3,
      lwd = 3)
legend(150, 1,
       c("Pseudo-Determinazioni N(175.6, 50.41)",
         "Valori teorici di N(175.6, 50.41)"),
       col = c("red", "blue"),
       lty = c(1, 3),
       lwd = c(3, 3),
       cex = 0.6)
```



Nuovamente, le due curve coincidono in modo perfetto. Si osservi che gli ottimi risultati ottenuti diepndono anche dall'elevato numero di realizzazioni; con valori pià bassi ($n = 50$) i risultati sarebbero in generale meno precisi.

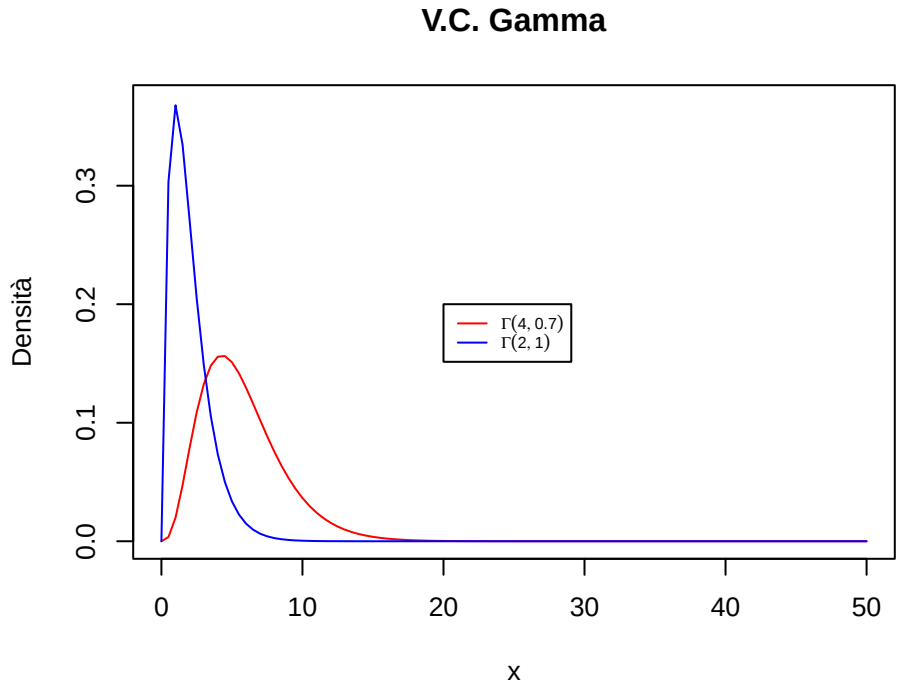
SOLUZIONI ESERCIZIO 16

16.1

Per la variabile casuale Gamma analogamente alla variabile casuale di Gauss, si dispone delle funzioni `dgamma`, `pgamma`, `qgamma` per il calcolo della funzione di densità, della funzione di ripartizione e dei quantili, e la funzione `rgamma` per la generazione di punti casuali secondo qquesta distribuzione.

Rappresentiamo la densità della variabile casuale Gamma al variare dei parametri di forma α e di scala $1/\beta$ nello stesso grafico. Si noti l'utilizzo del comando `expression` nella legenda per ottenere la lettera greca Gamma.

```
curve(dgamma(x, shape = 4, rate = 0.7),
      col = "red",
      ylim = c(0, 0.37),
      xlim = c(0, 50),
      ylab = "Densità",
      main = "V.C. Gamma")

curve(dgamma(x, shape = 2, rate = 1),
      col = "blue",
      add = TRUE)

names <- expression(Gamma(alpha = 4, beta = 0.7), Gamma(alpha = 2, beta = 1))

legend(20, 0.2,
       legend = names,
       col = c("red", "blue"),
       lty = c(1, 1),
       cex = 0.6)
```



Si ricorda che, come confermato dal grafico, la densità di una V.C. ha sempre supporto sui reali positivi. Si nota che due curve seguono lo stesso andamento, ma risultano molto diverse tra loro; in particolare si osserva che la curva rossa (relativa alla V.C. con parametri 4 e 0.7) presenta maggiore variabilità intorno alla media. La curva blue presenta invece un picco più elevato e traslato verso sinistra lungo l'asse delle ascisse. Da un punto di vista teorico, la varianza di una V.C. Gamma si ottiene come α/β^2, ovvero approssimativamente 8 e 2 per le due V.C. rispettivamente; si conferma la maggiore variabilità della prima.

16.2

Si generano 1000 realizzazioni dalla V.C. Gamma($\alpha = 4, \beta = 0.7$).

```
n <- 1000
set.seed(172)
G <- rgamma(n, shape = 4, rate = 0.7)
```

Riportiamo le principali statistiche descrittive ed il confronto con i corrispondenti valori teorici.

```
summary(G)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   0.510   3.651   5.335   5.781   7.328  17.558
```

```
var(G)
```

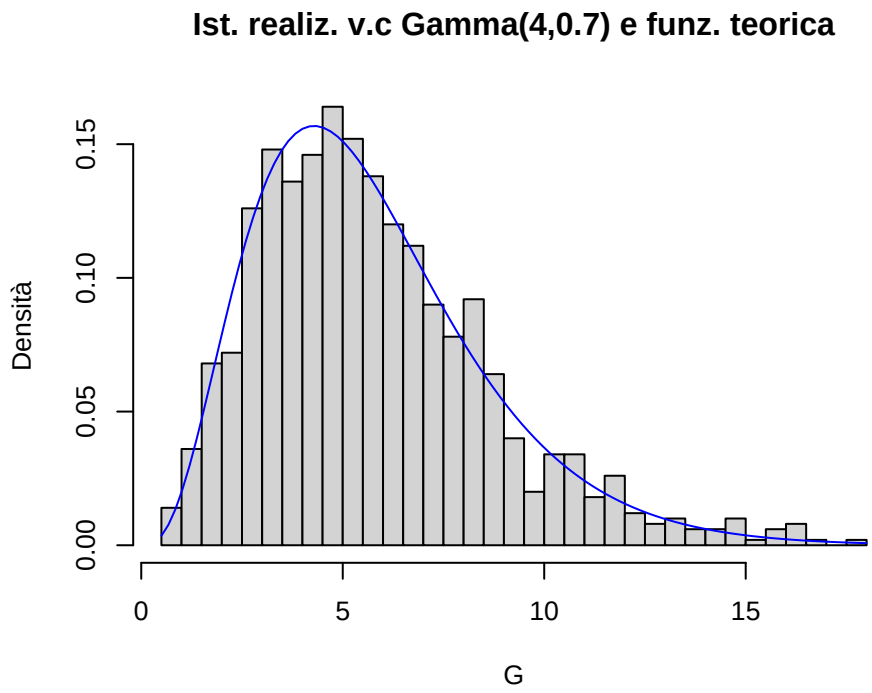
```
#> [1] 8.480012
```

Ricordiamo dapprima che il valore atteso di una V.C. Gamma è α/β (pari a 5.7 in questo caso) mentre la varianza teorica è $\alpha/(\beta)^2$ (uguale a 8.2). Confrontando queste quantità con quelle campionarie, si nota che i valori empirici di media e varianza campionarie sono molto simili ai valori teorici (media campionaria pari a 5.8, varianza campionaria pari a 8.5). Le osservazioni ottenute, inoltre assumono correttamente solo valori positivi, andando da un minimo di 0.5 ad un massimo di 17.6. Il valore della mediana è abbastanza simile a quello della media, nonostante sia noto che la V.C. Gamma non è simmetrica. Dall'analisi delle statistiche ottenute (oltre che dal grafico del punto precedente) si osserva infatti la presenza di una coda destra molto più lunga di quella sinistra (distanza tra terzo quartile e massimo superiore alla distanza tra minimo e terzo quartile).

16.3

Si riporta l'istogramma delle osservazioni generate al punto precedente.

```
hist(G,
     breaks = 60,
     freq = FALSE,
     main = "Ist. realiz. v.c Gamma(4,0.7) e funz. teorica",
     ylab = "Densità")
curve(dgamma(x, shape = 4, rate = 0.7),
      col = "blue",
      add = TRUE)
```



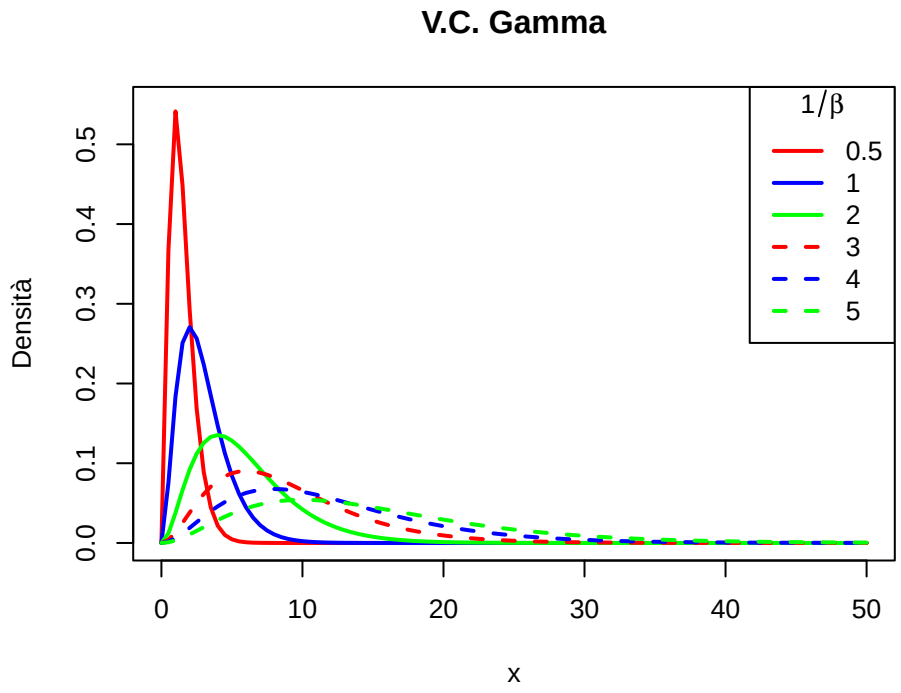
Anche in quasto caso, si verifica che la curva teorica si sovrappone in modo molto accurato alla forma definita dall'istogramma. La V.C. Gamma si utilizza per modellizzare fenomeni che assumono solo realizzazioni continue e positive. Un esempio specifico (si vedano le dispense di teoria) riguardano l'ammontare di pioggia caduta.

SOLUZIONI ESERCIZIO 16.1

Si riporta di seguito il codice per lo svolgimento dell'esercizio. Per i commenti si rimanda all'Esercizio 16 e alle dispense. Si noti che in questo caso viene fornito il valore del parametro di scala (pari a $1/\beta$), pertanto utilizzando il comando `dgamma` non si utilizza l'opzione `rate`, ma direttamente `scale`.

```
scala <- c(0.5, 1, 2, 3, 4, 5)
colori <- c("red", "blue", "green", "red", "blue", "green")
forme <- c(1, 1, 1, 2, 2, 2)

curve(dgamma(x, shape = 3, scale = scala[1]),
      col = colori[1],
      lty = forme[1],
      lwd = 2,
      ylim = c(0, 0.55),
      xlim = c(0, 50),
      ylab = "Densità",
      main = "V.C. Gamma")
```

```r
for (i in 2:length(scala)) {
  curve(dgamma(x, shape = 3, scale = scala[i]),
      col = colori[i],
      lty = forme[i],
      lwd = 2,
      add = TRUE)
}

legend("topright",
       legend = scala,
       title = expression(1/ beta),
       lty = forme,
       col = colori,
       lwd = 2)
```

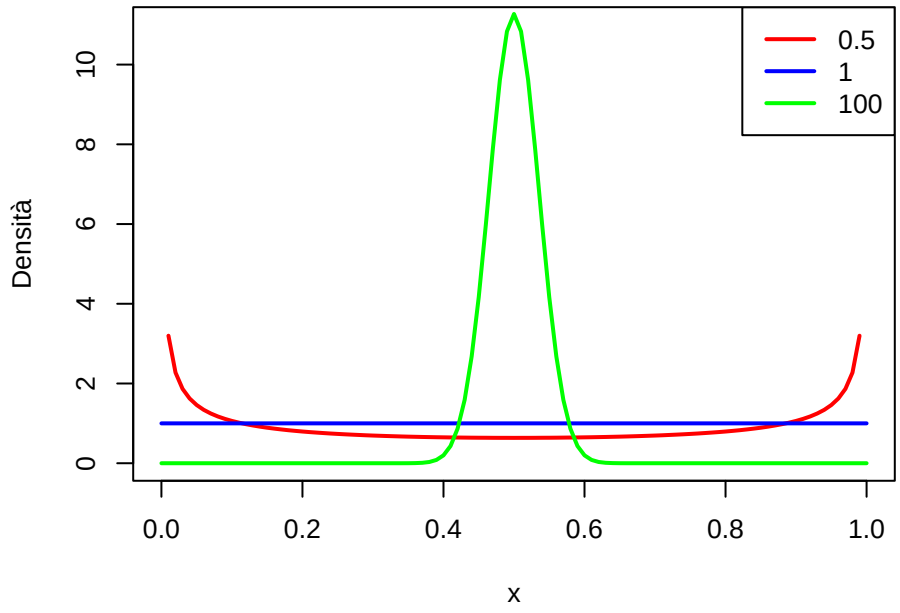


Si nota che al crescrere dei valori del parametro di scala sia la media sia la deviazione standard crescono.

SOLUZIONI ESERCIZIO 16.2

Per la V.C. si utilizzano i comandi `dbeta`, `pbeta`, `qbeta`, `rbeta`. Si noti che la V.C. Beta richiede due parametri α e β, che nelle funzioni di `R` sono rappresentati dalle opzioni `shape1` e `shape2` rispettivamente.

```r
curve(dbeta(x, shape1 = 0.5, shape2 = 0.5),
      col = "red",
      ylim = c(0, 11),
      ylab = "Densità",
      lwd = 2)
curve(dbeta(x, shape1 = 1, shape2 = 1),
      col = "blue",
      lwd = 2,
      add = TRUE)
curve(dbeta(x, shape1 = 100, shape2 = 100),
      col = "green",
      lwd = 2,
      add = TRUE)

legend("topright",
       legend = c(0.5, 1, 100),
       lty = c(1, 1, 1),
       col = c("red", "blue", "green"),
       lwd = 2)
```



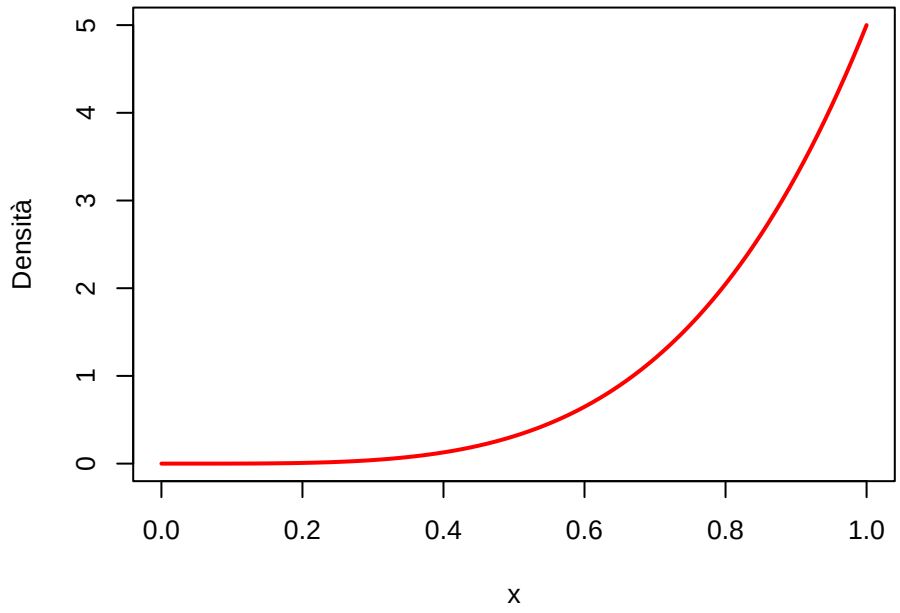
Osserviamo che tutte e tre le curve tracciate sono simmetriche (rispetto ad uno stesso punto $1/2$)[4]. Spicca in particolare la forma della densità relativa ai parametri $\alpha = \beta = 1$, lineare e costante (a

[4] A livello teorico questo è giustificato dal fatto che l'indice di asimmetria contiene a numeratore un termine $\beta - \alpha$ che azzera l'indice nel caso in cui $\alpha = \beta$.

quota 1). Con valori dei parametri pari a 0.5 la funzione è convessa, mentre diverta concava per valori superiori a 1 (100, in questo caso).

Consdieriamo ora una coppia di parametri tali che $\alpha > \beta$, per esempio $\alpha = 5$, $\beta = 1$. Per quanto detto in precedenza, ci aspettiamo che in questo caso, la forma della funzione di densità non sarà simmetrica.

```
curve(dbeta(x, shape1 = 5, shape2 = 1),
      col = "red",
      ylab = "Densità",
      lwd = 2)
```



Si verifica infattio la non simmetria della curva tracciata, oltre al fatto che l'andamento è completamente diverso rispetto a quello del grafico precedente. Questo testimonia la grande flessibilità della V.C. Beta. Si noti che invertendo i valori dei parametri (i.e., $\alpha = 1$, $\beta = 5$),la forma della densità risulta ancora diversa.

SOLUZIONI ESERCIZIO 17

17.1

Per generare realizzazioni da una V.C. Binomiale, si utilizza la funzione `rbinom` che richiede come argomenti, oltre al numero di realizzazioni `n=1000`, il numero di prove (`size=41`) e il valore della probabilità di successo nella singola prova (`prob`); questa deve essere calcolata come $p = 31/41 = 0.76$.

```r
set.seed(123)
prob <- 31/41
X <- rbinom(1000, size = 41, prob = prob)
head(X, 3)
```

```
#> [1] 33 29 32
```

Le prime 3 realizzazioni sono 33, 29 e 32 conteggi (si sono verificati, rispettivamente 33, 29 e 32 successi, in ciascuno caso su un totale di 41 prove).

Utilizziamo la funzione `summary` per determinare le statistiche descrittive.

```r
summary(X)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   21.00   29.00   31.00   31.04   33.00   39.00
```

Il numero minimo e quello massimo di successi ottenuti sulle 41 prove sono rispettivamente 21 e 39. In media sono stati simulati 31 successi su 41 prove totali, valore molto simile quello della mediana: in metà delle simulaizoni si sono rilevtai più di 31 successi.

Si può anche effettuare il confronto tra i valori di media e varianza realizzati ed i corrispondenti valori teorici relativi alla V.C. Binomiale.

```r
mean(X)
```

```
#> [1] 31.038
```

```r
41*prob
```

```
#> [1] 31
```

```r
var(X)
```

```
#> [1] 7.401958
```

```r
41*prob*(1-prob)
```

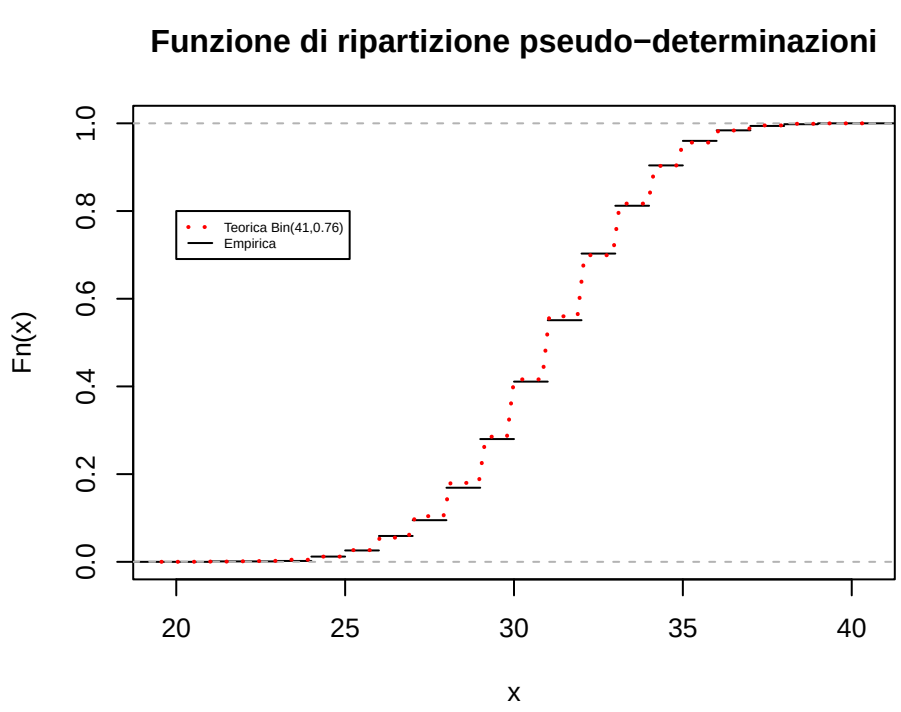
```
#> [1] 7.560976
```

Si osserva che i valori ottenuti sono molti simili a quelli attesi.

17.2

Si rappresenta la funzione di ripartizione empirica sulla base delle osservazioni realizzate, e la si confronta con la curva della funzione di ripartizione teorica.

```r
plot(ecdf(X),
     do.points = FALSE,
     main = "Funzione di ripartizione pseudo-determinazioni")

curve(pbinom(x, 41, prob),
     add = TRUE,
     col = "red",
     lty = 3,
     lwd = 2)

legend(20, 0.8,
       c("Teorica Bin(41,0.76)", "Empirica"),
       col = c("red", "black"),
       lty = c(3, 1),
       lwd = c(2, 1),
       cex = 0.5)
```



Si nota che, nuovamente, la funzione di ripartizione empirica è molto simile a quella. teorica. Si ricorda che la Binomiale tende alla Normale al crescere della numerosità; si può provare a sovrapporre il grafico della V.C. Normale (con valori dei parametri pari a quelli campionari delle realizzazioni ottenute) per verificare l'aderenza delle curve.

SOLUZIONI ESERCIZIO 17.1

17.1.1

Bisogna innanzitutto calcolare la probabilità che lo studente, tirando ad indovinare, indovini la risposta corretta; questo valore è chiaramemente $1/5 = 0.2$ (5 categorie di risposta). Per valutare la probabilità che indovini tutte le risposte, o viceversa, nessuna risposta, si utilizza il comando `dbinom`, come segue.

```
p <- 1/5
dbinom(10, 10, p)
```

```
#> [1] 1.024e-07
```

```
dbinom(0, 10, p)
```

```
#> [1] 0.1073742
```

Si ottinee che la probabilità di non sbagliare nessuna risposta è 0; la probabilità di sbagliare tutte le risposte è molto più elevata è infatti 0.11.

17.1.2

La media e la deviazione standard sono le seguenti

```
10*p
```

```
#> [1] 2
```

```
sqrt(10*p*(1-p))
```

```
#> [1] 1.264911
```

In media, lo studente che tira ad indovinare la risposta di tutte le domande, ne indovina 2 su 10, con uno scostamento medio da questo valore pari a 1.2. (Quindi approssimativamente 2 ± 1).

SOLUZIONI ESERCIZIO 17.2

17.2.1

Si generano 1000 realizzazioni dalla V.C. di Poisson considerando 4 diversi valori per il parametro, e utilizzando sempre lo stesso seme (123). Si sistemano i risultati in un dataframe `P`.

```
n <- 1000
lambda1 <- 5
lambda2 <- 10
lambda3 <- 20
lambda4 <- 30
set.seed(123)
p1 <- rpois(n, lambda1)
set.seed(123)
p2 <- rpois(n, lambda2)
set.seed(123)
p3 <- rpois(n, lambda3)
set.seed(123)
p4 <- rpois(n, lambda4)

P <- data.frame(p1, p2, p3, p4)
apply(P, 2, summary)
```

```
#>             p1     p2     p3    p4
#> Min.     0.000  2.000  6.000 13.00
#> 1st Qu.  3.000  8.000 17.000 26.00
#> Median   5.000 10.000 20.000 30.00
#> Mean     4.981  9.983 19.916 29.91
#> 3rd Qu.  6.000 12.000 23.000 34.00
#> Max.    14.000 21.000 35.000 48.00
```

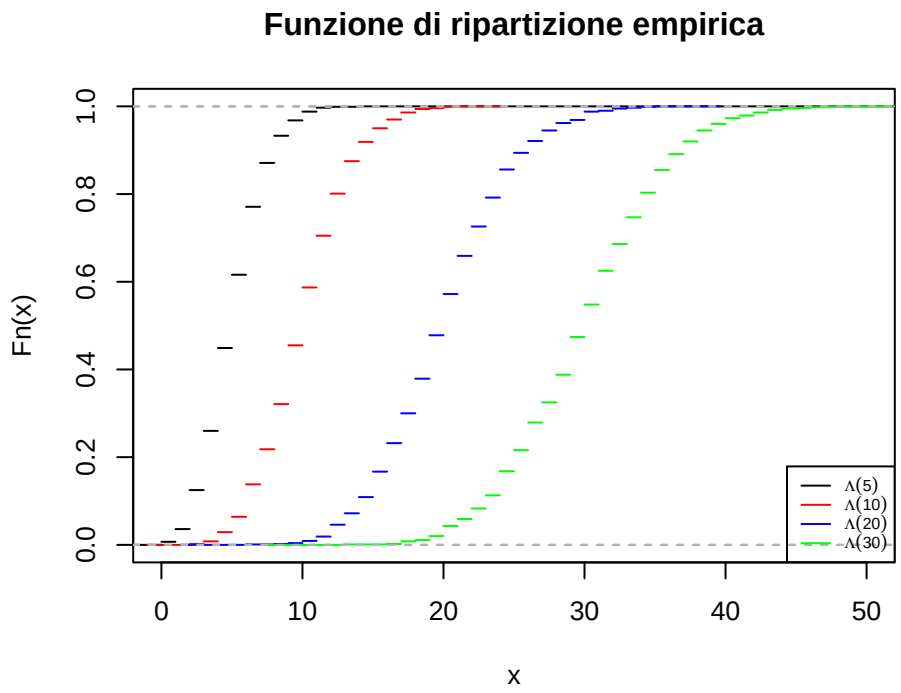
Si nota che i conteggi realizzati sono in media più elevati al crescere del parametro; questo è evidente analizzando tutte gli indici riportati. Si osserva poi che la distribuzione sembra presentare una lunga coda a destra (quindi asimmetria) per bassi valori di λ, mentre all'aumentare di questo valore la distribuzione appare sempre più simmetrica. Del resto è noto che per valori elevati di λ la Poisson può essere approssimata da una V.C. Normale.

17.2.2

Si disegnano le quattro funzioni di ripartizione empirche nella stessa finestra grafica

```
plot(ecdf(P[, 1]),
     col = "black",
     xlim = c(0,50),
     do.points = FALSE,
     main = "Funzione di ripartizione empirica")
plot(ecdf(P[, 2]),
     col = "red",
     do.points = FALSE,
     add = TRUE)
plot(ecdf(P[, 3]),
     do.points = FALSE,
```

```
     col = "blue",
     add = TRUE)
plot(ecdf(P[, 4]),
     do.points = FALSE,
     col = "green",
     add = TRUE)

names <- expression(Lambda(lambda = 5),
                    Lambda(lambda = 10),
                    Lambda(lambda = 20),
                    Lambda(lambda = 30))

legend("bottomright",
       legend = names,
       col = c("black", "red", "blue", "green"),
       lty = c(1, 1, 1, 1),
       cex = 0.6)
```



Si osserva che all'aumentare del valore di λ le curve delle funzioni di ripartizione empirica risultano sempre più traslate verso destra lungo l'asse delle ascisse. Questo è coerente con l'osservazione (si veda il punto precedente) che il valore della media cresce con λ (difatti è noto che la funzione di ripartizione passa sempre per il punto $(\hat{\mu}, 0.5)$, dove $\hat{\mu}$ denota la media delle osservazioni). ne consegue che, a titolo d'esempio, la probabilità che $\mathbb{P}(Pois(30) \leq 30) = 0.5$, mentre $\mathbb{P}(Pois(20) \leq 30) \approxeq 1$.

17.2.3

Si calcola la probabilità che $X = 20$ per ciascuna delle V.C. considerate; utilizzando la funzione `dpois`, passando coem parametri il punto $x = 20$ e il valore del parametro λ.

```
dpois(x = 20, lambda1)
```

```
#> [1] 2.641211e-07
```

```
dpois(x = 20, lambda2)
```

```
#> [1] 0.001866081
```

```
dpois(x = 20, lambda3)
```

```
#> [1] 0.08883532
```

```
dpois(x = 20, lambda4)
```

```
#> [1] 0.01341115
```

La probabilità cresce in al crescere di λ. Confrontiamo ora questo risultato con quello relativo ai dati empirici ottenuti al punto 1. Si analizza, a titolo d'esempio, il caso in cui $\lambda = 30$; è analogo per le altre V.C. Le frequenze relative delle realizzazioni si determina nel modo seguente.

```
table(p4)/1000
```

```
#> p4
#>    13    16    17    18    19    20    21    22    23    24    25    26    27
#> 0.001 0.001 0.006 0.003 0.009 0.023 0.016 0.024 0.030 0.055 0.048 0.063 0.046
#>    28    29    30    31    32    33    34    35    36    37    38    39    40
#> 0.063 0.086 0.074 0.077 0.061 0.061 0.056 0.052 0.036 0.029 0.025 0.015 0.013
#>    41    42    43    44    45    46    47    48
#> 0.006 0.007 0.006 0.003 0.001 0.002 0.001 0.001
```

Il valore della frequenza relativa corrispondente a 20 è 0.023, non troppo distante dal valore teorico ottenuto, pari a 0.013.

SOLUZIONI ESERCIZIO 18

Si leggono i dati (contenuti in un file .dat) utilizzando la funzione `read.table`, e specificando `header=TRUE` per indicare la presenza delle etichette delle colonne (variabili).

```
Afterlife <- read.table("http://stat4ds.rwth-aachen.de/data/Afterlife.dat",
                        header=TRUE)
```

Avendo verificato che i dati sono importati in formato numeric (pur essendo tutte variabili categoriali), trasformiamo le varie colonne utilizzando la funzione `as.factor`. Si descrivono quindi i dati con la funzione `skim_without_charts`.

```
Afterlife$postlife<-as.factor(Afterlife$postlife)
Afterlife$religion<-as.factor(Afterlife$religion)
Afterlife$gender<-as.factor(Afterlife$gender)
require(skimr)
skimr::skim_without_charts(Afterlife)
```

Table 25: Data summary

Name	Afterlife
Number of rows	1553
Number of columns	4
Column type frequency:	
factor	3
numeric	1
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
postlife	0	1	FALSE	2	1: 1357, 2: 196
religion	0	1	FALSE	3	1: 1070, 2: 449, 3: 34
gender	0	1	FALSE	2	2: 894, 1: 659

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
subject	0	1	777	448.46	1	389	777	1165	1553

Tra i soggetti intervistati si registra un'evidente superiorità numerica di coloro che credono nel "dopo vita" (1357), rispetto a coloro che non ci credono (196). Vi è poi un maggior numero di soggetti di religione protestante (1070), e un numero molto limitato di soggetti di religione ebraica (34). I rimanenti sono di religione cattolica (449).

Per determinare la tabella a doppia entrata si utilizzano le funzioni `table` e `proportions`. La prima permette di ricavare la tabella a doppia entrata dei conteggi; con l'aggiunta del comando `proportions`, si passa alle frequenze relative.

Ricaviamo ora:

- la distribuzione congiunta (coincidente con la tabella a doppia entrata);

```
proportions(table(Afterlife$religion, Afterlife$postlife))
```

```
#>
#>              1          2
#>   1 0.61558274 0.07340631
#>   2 0.24726336 0.04185448
#>   3 0.01094656 0.01094656
```

- le distribuzioni marginali;

```
proportions(table(Afterlife$postlife)); proportions(table(Afterlife$religion))
```

```
#>
#>         1         2
#> 0.8737927 0.1262073
```

```
#>
#>          1          2          3
#> 0.68898905 0.28911784 0.02189311
```

- la distribuzione condizionata della credenza in base alla religione.

```
proportions(table(Afterlife$religion, Afterlife$postlife), 1)
```

```
#>
#>             1         2
#>   1 0.8934579 0.1065421
#>   2 0.8552339 0.1447661
#>   3 0.5000000 0.5000000
```

I commenti sono lasciati allo studente.

Soluzioni esercizi "Modelli di Poisson e Binomiale Negativa per dati di coteggio"

SOLUZIONE ESERCIZIO 19

19.1

Si considerano i dati ufficiali dei conteggi dei pazienti affetti da COVID-19 che si trovano in **terapia intensiva**. Tali conteggi sono presenti nel repository https://raw.githubusercontent.com/pcm-

dpc/COVID-19/master/ ed aggionati quotidianamente a livello ufficiale dal **Dipartimento della Protezione Civile**.

Si caricano i dati specificando il nome del repository e dello specifico dataset. Tra le diverse colonne si selezionano solo quella relativa ai conteggi delle terapie intendive e quella della data; quest'ultima viene caricata in formato character, e deve essere pertanto convertita.

```
repository <- "https://raw.githubusercontent.com/pcm-dpc/COVID-19/master/"
overall.dataset <- "dati-andamento-nazionale/dpc-covid19-ita-andamento-nazionale.csv"
overall.filename <- paste(repository, overall.dataset,sep = "")
Italy <- read.csv(overall.filename)
df <- Italy[, c(1, 4)]
df$data <- as.Date(as.POSIXct(df$data, format = "%Y-%m-%dT %H:%M:%S"))
df <- subset(df, data <= as.Date('2022-10-26'))
colnames(df) <- c("Data", "Terapia")
rownames(df) <- 1:nrow(df)
```

Calcoliamo le statistiche descrittive della serie di dati utilizzando la funzione `skim_without_charts`.

```
require(skimr)
skim_without_charts(df)
```

Table 28: Data summary

Name	df
Number of rows	976
Number of columns	2
Column type frequency:	
Date	1
numeric	1
Group variables	None

Variable type: Date

skim_variable	n_missing	complete_rate	min	max	median	n_unique
Data	0	1	2020-02-24	2022-10-26	2021-06-25	976

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
Terapia	0	1	1084.98	1134.27	26	252	482	1748	4068

I dati risalgono ad un periodo compreso tra il 24 Febbraio 2020 (primo giorno di rilevazione) al 23 ottobre 2022 (giorno in cui sono state scritte le soluzioni di questo esercizio). Si tratta di misurazioni relative ad un totale di 973 giorni.

Dall'analisi delle statistiche descrittive relative al conteggio dei pazienti ricoverati in terapia intensiva emerge che durante il periodo considerato i reparti di terapia intensiva sono stati occupati, in media, da 1080 pazienti. Il minimo e il massimo sono pari a 26 e 4068 persone rispettivamente. Questo testimonia una variabilità molto elevata, confermata dal valore della deviazione standard, molto alto e pari a 1135.

Valutiamo anche le date in cui sono stati registrati il numero minimo e massimo di pazienti in terapia intensiva.

```
ind_max <- which.max(df$Terapia)
df[ind_max, ]
```

	Data	Terapia
40	2020-04-03	4068

```
ind_min <- which.min(df$Terapia)
df[ind_min, ]
```

Data	Terapia
2020-02-24	26

```
tail(df, 1)
```

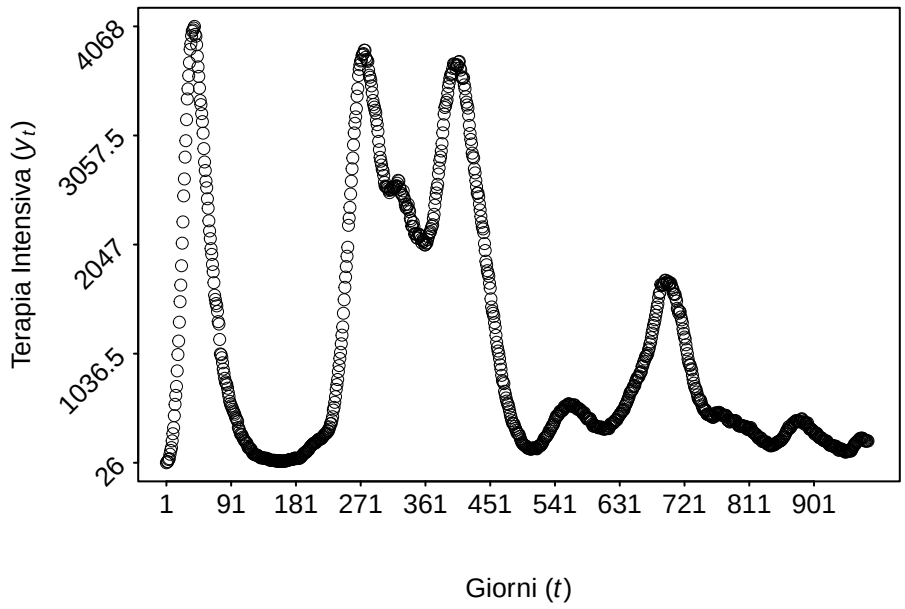
	Data	Terapia
976	2022-10-26	227

Il giorno di minima occupazione dei reparti di terapia intensiva risale all'inizio delle rilevazioni, il 24 febbraio 2022, con 26 pazienti ricoverati. Il massimo numero, pari a 4068, risale invece al 3 aprile 2020, drante la prima ondata. Attualmente (26 ottobre 2022) sono registrati 227 pazienti ricoverati.

Per finire le analisi preliminari dei dati, rappresentiamo graficamente l'andamento nel tempo del numero di pazienti ricoverati in terapia intensiva.

```
n <- nrow(df)
m1 <- min(df$Terapia)
m2 <- max(df$Terapia)
ytick <- seq(m1, m2, length.out = 5)
xtick <- seq(1, n, by = 90)
plot(df$Terapia,
```

```
     ylab = expression("Terapia Intensiva ("*italic(y[t])*")"),
     xlab = expression("Giorni ("*italic(t)*")"),
     yaxt = "n",
     xaxt = "n",
     xlim = c(1, n+5),
     ylim = c(m1-10, m2+10),
     lwd = 0.5,
     lty = 1,
     col = "black")

axis(side = 2, at = ytick, labels = FALSE,
     cex.lab = 0.5, padj = 2, tck = -0.009)
text(par("usr")[1],
     ytick,
     labels = ytick, srt = 45,
     pos = 2, xpd = TRUE, cex.lab = 0.1)
axis(side = 1, at = xtick,
     labels = FALSE, cex = 0.1, tck = -0.009)
text(x = xtick,
     par("usr")[3],
     labels = xtick,
     cex.lab = 0.5,
     pos = 1, xpd = TRUE)
```



19.2

Per stimare il modello di Poisson non omogeneo si definiscono dapprima le variabili esplicative in funzione del tempo, considerando un trend lineare, un trend quadratico ed un trend log-lineare. Si aggiunge inoltre una componente autoregressiva. Il modello è pertanto formulato come segue:

$$\log \lambda_t(y_{(t-1)}) = \beta_0 + t\beta_1 + (t^2/100)\beta_2 + \log(t)\beta_3 + z_{t-1}\rho,$$

dove con ρ si indica il coefficiente autoregressivo e z_{t-1} una trasformazione di y_{t-1}.

```
Regressors <- cbind(linearTrend = seq(along = df$Terapia),
                    quadTrend = seq(along = df$Terapia)^2/100,
                    linlogTrend = log(seq(along = df$Terapia)))
```

La stima del modello si ottiene utilizzando la funzione `tsglm` del pacchetto `tscount` che richiede i seguenti argomenti:

- `ts`: contiene la serie dei conteggi (dei pazienti in terapia intensiva);
- `link`: specifica il modello per λ_t: `link = "log"` indica il modello log-lineare;
- `model`: consente di specificare l'ordine dei ritardi per la componente autoregressiva: `model = list(past_obs=1)` indica ritardi del primo ordine;
- `xregr`: la matrice delle covariate (temporali, nel nostro caso);
- `distr`: la distribuzione (condizionata).

```
options(scipen = 100)
require(tscount)
M3Italy <- tsglm(ts = df$Terapia,
                 link = "log",
                 model = list(past_obs=1),
                 xreg = Regressors,
                 distr = "poisson")
summary(M3Italy)
```

```
#>
#> Call:
#> tsglm(ts = df$Terapia, model = list(past_obs = 1), xreg = Regressors,
#>     link = "log", distr = "poisson")
#>
#> Coefficients:
#>                 Estimate   Std.Error    CI(lower)   CI(upper)
#> (Intercept)   0.14320423  0.02180821   0.10046092   0.1859475
#> beta_1        0.97425368  0.00149028   0.97133279   0.9771746
#> linearTrend  -0.00018265  0.00004829  -0.00027730  -0.0000880
#> quadTrend     0.00000422  0.00000365  -0.00000294   0.0000114
#> linlogTrend   0.01957353  0.00515417   0.00947154   0.0296755
#> Standard errors and confidence intervals (level =  95 %) obtained
```

```
#> by normal approximation.
#>
#> Link function: log
#> Distribution family: poisson
#> Number of coefficients: 5
#> Log-likelihood: -4961.877
#> AIC: 9933.753
#> BIC: 9958.171
#> QIC: 9933.753
```

Il summary del modello contiene la stime di massima verosimiglianza dei parametri del modello (si noti che la notazione del pacchetto è differente rispetto a quella introdotta a lezione: il parametro `beta_1` del summary è la stima della componente autoregressiva ρ).

Il valore stimato per l'intercetta è pari a 0.14 e corrisponde al valore di $\log(\lambda_t)$ quando tutte le covariate sono pari a 0. I coefficienti relativi alle covariate temporali sono negativi per quanto riguarda il trend lineare e quello quardratico, positivo per il trend log-lineare. Riferendosi alla stessa covariata (il tempo), l'interpretazione deve necessariamente tenere conto dei tre coefficienti in modo congiunto. Di conseguenza, all'aumentare di 1 del valore del tempo, i trend lineare e quadratico apportano un contributo negativo al valore di $\log(\lambda_t)$, mentre un contributo positivo arriva dal trend log-lineare. Per quantificare il valore complessivo dei contributi è necessario considerarli in modo congiunto. Il valore del coefficiente relativo al termine autoregressivo (`beta_1` nel summary) è positivo: il termine z_{t-1} che presenta un ritardo del primo ordine fornisce un contributo positivo al valore di $\log(\lambda_t)$.

Le stime dei coefficienti di regressione sono fornite insieme all'**intervallo di confidenza** calcolato in base alla distribuzione asintotica per gli stimatori di massima verosimiglianza al livello del 95%. Osserviamo che nessuno degli intervalli di confidenza contiene il valore 0.

Il summary del modello contiene inoltre informazioni globali, quali il valore della log-verosimiglianza, o i valori di indici quali AIC, BIC e QIC

Possiamo anche confrontare i valori osservati con quelli interpolati secondo il modello; questi possono essere estratti, in modo analogo a quanto fatto con le funzioni `lm` o `glm`, con l'opzione `fitted.vallues`. Estraiamo i primi 10 e gli ultimi 10 valori.

```
A <- cbind(df$Terapia, M3Italy$fitted.values)
colnames(A) <- c("Osservati", "Interpolati")
A <- data.frame(A)
head(A)
```

Osservati	Interpolati
26	260.32774
35	29.00266
36	38.68364
56	39.94722
64	61.11467
105	69.69253

```
tail(A)
```

	Osservati	Interpolati
971	234	242.7163
972	227	234.9091
973	229	228.0710
974	226	230.0014
975	232	227.0598
976	227	232.8862

Si osseerva che all'inizio delle rilevazioni i valori osservati e quelli interpolati sono piuttosto diversi tra loro, segno che il modello non è in grado di spiegare in modo corretto il conteggio dei pazienti ricoverati in terapia intensiva nei primi giorni dell'epidemia. Viceversa, le rilevazioni relative agli ultimi 10 giorni sono molto simili ai valori interpolati e non si notano differenze rilevanti.

19.3

Calcoliamo i conteggi attesi per i prossimi cinque giorni. Definiamo quindi la componente temporale del modello, in modo analogo a quanto fatto in precedenza, ma specificando un numero di istanti temporali, successivi all'ultima rilevazione, pari a 5. Analogamente alla funzione `glm`, i valori previsti si ottengono con la funzione `predict`. Questa richiede in input il modello stimato, la nuova componente temporale per le covariate (`newxreg`), ed il numero di istanti temporali per cui si vuole effettuare la previsione (`n.ahead`). Si noti inoltre che l'opzione `method = "bootstrap"` permette di ottenere le stime degli estremi dell'intervallo predittivo per ogni previsione, calcolate in base al metodo del percentile al livello di confidenza del 95%.

```
go <- 5
TT <- length(df$Data)
newRegressors <- data.frame(linearTrend = ((TT+1):(TT+go)),
                            quadTrend = ((TT+1):(TT+go))^2/100,
                            linlogTrend = log((TT+1):(TT+go)))

P3Italy <- predict(M3Italy,
                   newxreg = newRegressors,
                   n.ahead = go,
                   method = "bootstrap")
```

Stampiamo i risultati affiancando ciascuna previsione puntuale al rispettivo intervallo.

```
pred <- cbind(P3Italy$pred, P3Italy$interval,
              P3Italy$interval[, 2] - P3Italy$interval[, 1])
colnames(pred) <- c("Stime puntuali", "PIinf", "PIsup", "Ampiezza")
rownames(pred) <- format(df$Data[TT] + (1:go), "%d %b")
pred <- data.frame(pred)
pred
```

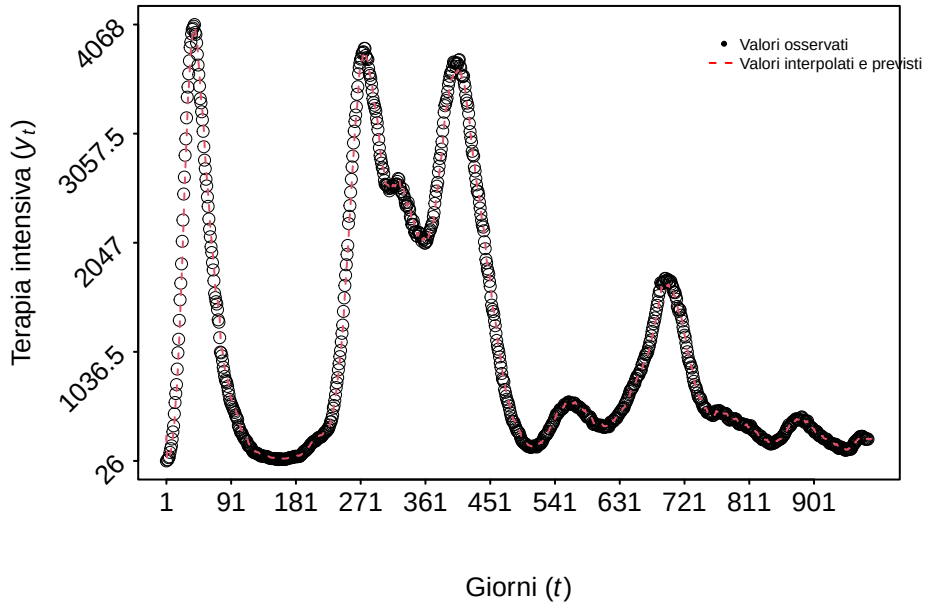
	Stime.puntuali	PIinf	PIsup	Ampiezza
27 ott	227.9977	198	257	59
28 ott	228.9513	191	266	75
29 ott	229.8617	185	275	90
30 ott	230.7298	176	282	106
31 ott	231.5565	174	294	120

Osserviamo, per i prossimi 5 giorni, un andamento decrescente per il numero di pazienti ricoverati in terapia intensiva (seguendo il calo che si è registrato negli utlimi 5 giorni). Per quanto riguada gli intervalli predittivi, si nota che le ampiezze sono elevate e che tendono ad aumentare per i giorni distanti dell'ultima rilevazione (ovviamente più si va avanti nel tempo, più la previsione risulta imprecisa).

Le seguente figura mostra i conteggi osservati dei pazienti ricoverati in in terapia intensiva per tutto il periodo considerato insieme ai corrispondenti valori interpolati con il modello stimato a cui vengono aggiunti i valori previsti per il periodo temporale riportato sopra.

```
ytick <- seq(m1, m2, length.out = 5)
xtick <- seq(1, n, by = 90)

plot(df$Terapia,
     ylab = expression("Terapia intensiva ("*italic(y[t])*")"),
     xlab = expression("Giorni ("*italic(t)*")"),
     yaxt = "n",
     xaxt = "n",
     xlim = c(1, n+5),
     ylim = c(m1-10, m2+10),
     lwd = 0.5,
     lty = 1,
     col ="black" )

axis(side=2, at=ytick, labels = FALSE,
     cex.lab = 0.5, padj = 2,tck=-0.009)
text(par("usr")[1],
     ytick,
     labels = ytick, srt = 45,
     pos = 2, xpd = TRUE, cex.lab = 0.1)
axis(side=1, at=xtick,
     labels = FALSE, cex=0.1,tck=-0.009)
text(x=xtick,
     par("usr")[3],
     labels = xtick,
     cex.lab = 0.5,
     pos = 1, xpd = TRUE)

lines(c(M3Italy$fitted.values, P3Italy$pred),
      lwd = 1, col = 2, lty = 2)
```

```
legend(x = 750, y = m2,
       pch = c(20, NA),
       lty = c(NA, 2),
       legend = c("Valori osservati", "Valori interpolati e previsti"),
       col = c("black", "red"),
       bty = "n",
       x.intersp = 0.1,
       cex = 0.6,  pt.cex = .5,
       xpd = TRUE,
       text.width = 0.0001)
```



Si osserva che la linea tratteggiata corrispondente ai valori interpolati ricalca quasi perfettamente la curva formata dalle osservazioni rilevate. Si conferma quindi che il modello spiega in modo appropriato i dati considerati.

SOLUZIONE ESERCIZIO 20

Si rimanda alle soluzioni dell'esercizio precedente ed alle dispense delle applicazioni.

SOLUZIONE ESERCIZIO 21

21.1

Nel seguito si caricano i dati e si prepara la formulazione del modello in modo del tutto analogo a quanto fatto nell'esercizio 19, al quale si rimanda per i commenti su questa parte.

```
repository <- "https://raw.githubusercontent.com/pcm-dpc/COVID-19/master/"
overall.dataset <- "dati-andamento-nazionale/dpc-covid19-ita-andamento-nazionale.csv"
overall.filename <- paste(repository, overall.dataset,sep = "")
Italy <- read.csv(overall.filename)
df <- Italy[, c(1, 4)]
df$data <- as.Date(as.POSIXct(df$data, format = "%Y-%m-%dT %H:%M:%S"))
df <- subset(df, data <= as.Date('2022-10-26'))
colnames(df) <- c("Data", "Terapia")
rownames(df) <- 1:nrow(df)
```

Stimiamo il modello utilizzando nuovamente il comando `tsglm`, seguendo la stessa sintassi introdotta nell'Esercizio 19, specificando però `distr = "nbinom"` per utilizzare la distribuzione Binomiale Negativa.

```
options(scipen = 100)
require(tscount)

Regressors <- cbind(linearTrend = seq(along = df$Terapia),
                    quadTrend = seq(along = df$Terapia)^2/100,
                    linlogTrend = log(seq(along = df$Terapia)))

M4Italy <- tsglm(ts = df$Terapia,
                 link = "log",
                 model = list(past_obs = 1),
                 xreg = Regressors,
                 distr = "nbinom")
summary(M4Italy)
```

```
#>
#> Call:
#> tsglm(ts = df$Terapia, model = list(past_obs = 1), xreg = Regressors,
#>     link = "log", distr = "nbinom")
#>
#> Coefficients:
#>                 Estimate   Std.Error    CI(lower)   CI(upper)
#> (Intercept)   0.14320423  0.03258229   0.07934412   0.2070643
#> beta_1        0.97425368  0.00211713   0.97010419   0.9784032
#> linearTrend  -0.00018265  0.00007188  -0.00032352  -0.0000418
#> quadTrend     0.00000422  0.00000525  -0.00000608   0.0000145
#> linlogTrend   0.01957353  0.00798593   0.00392139   0.0352257
```

```
#> sigmasq       0.00104318          NA           NA          NA
#> Standard errors and confidence intervals (level =  95 %) obtained
#> by normal approximation.
#>
#> Link function: log
#> Distribution family: nbinom (with overdispersion coefficient 'sigmasq')
#> Number of coefficients: 6
#> Log-likelihood: -4866.576
#> AIC: 9745.152
#> BIC: 9774.453
#> QIC: 9950.531
```

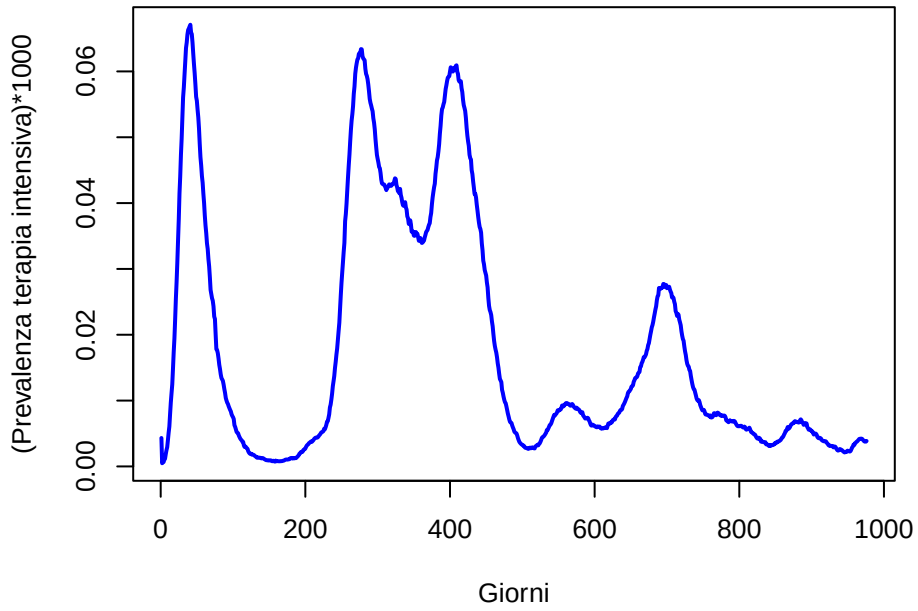
I valori dei coefficienti stimati sono uguali a quelli ottenuti stimando il modello precedente (con distribuzione condizionata di Poisson). Si rimanda quindi all'Esercizio 19 per i commenti. In questo caso si aggiunge la stima del parametro σ^2 di overdispersion, pari a 0.0009. Si osserva che il grado di overdispersion sembra essere molto limitato, dal momento che il valore di questo parametro è estremamente basso, prossimo a zero.

21.2

Riportiamo graficamente l'andamento temporale della prevalenza prevista dal modello per i pazienti in terapia intensiva rispetto alla numerosità complessiva della popolazione. Il valore della popolazione italiana, al 1 gennaio 2020, è pari a 60317000 unità (dati ISTAT). La prevalenza si calcola quindi dividendo la serie dei conteggi per questo valore e moltiplicando per 1000.

```
pop_Ita <- 60317000
prev <- M4Italy$fitted.values/pop_Ita
prev <- prev*1000
```

```
plot(prev, type = "l",
     xlab = "Giorni",
     lwd = 2,
     col = "blue", ylab = "(Prevalenza terapia intensiva)*1000")
```



SOLUZIONE ESERCIZIO 22

Si rimanda alle soluzioni dell'esercizio precedente ed alle dispense delle applicazioni.

Soluzioni esercizi "Bootstrap"

SOLUZIONI ESERCIZIO 23

23.1

Si caricano i dati contenuti nel file `cranio1.RData` e se ne calcolano le statistiche descrittive.

```
load("cranio1.Rdata")
skimr::skim_without_charts(cranio1)
```

Table 37: Data summary

Name	cranio1
Number of rows	98
Number of columns	1

Table 37: Data summary

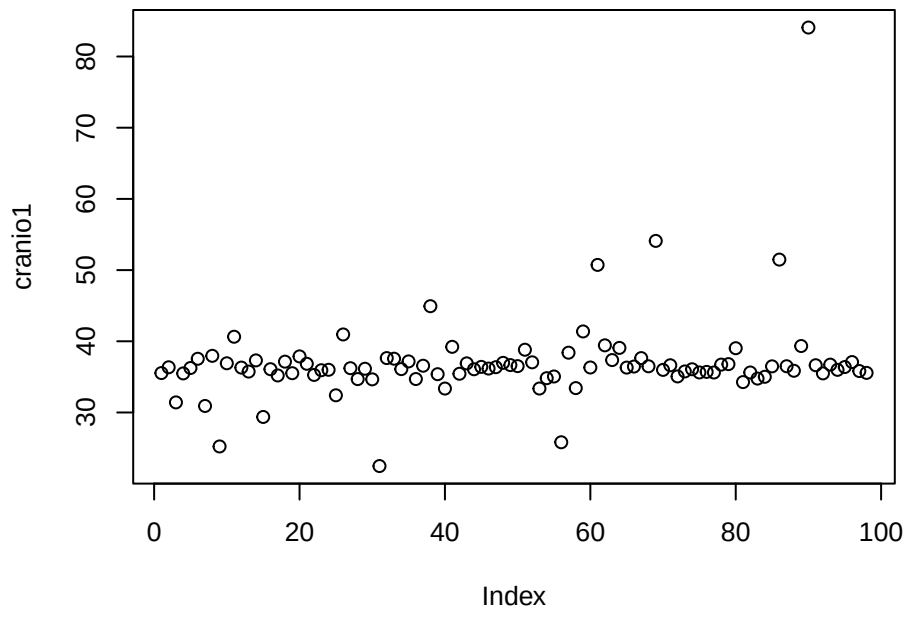
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

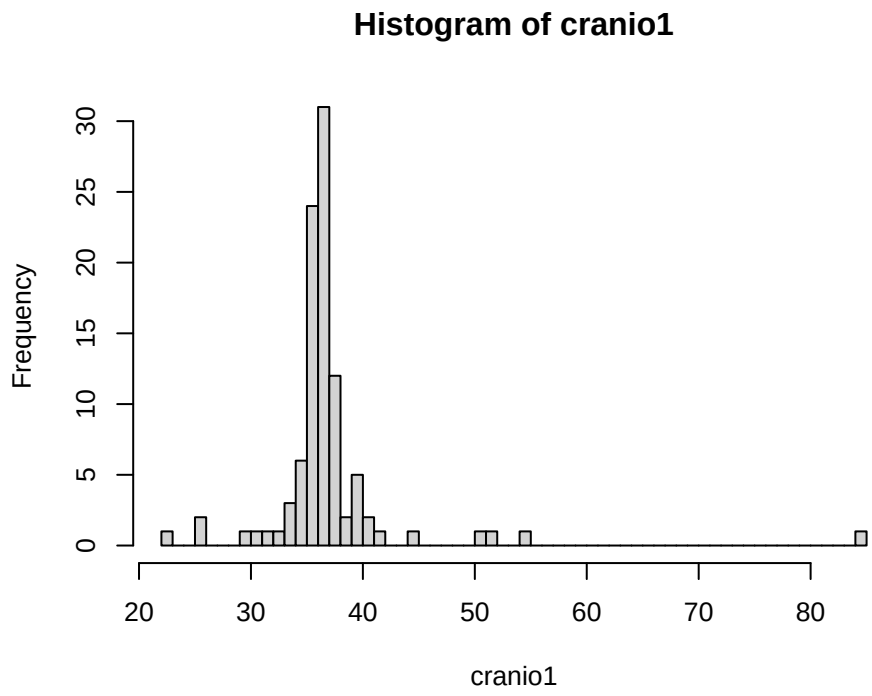
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	36.94	6.26	22.48	35.5	36.31	37.13	84.07

Il dataset contiene 98 osservazioni per un'unica variabile (la misura della circonferenza del cranio). Il campo di variazione va da un minimo di 23 cm ad un massimo di 84 cm. La circonferenza media è di 37 cm, valore molto vicino a quello della mediana. Il 75% dei nenonati osservati ha misura della circonferenza inferiore a 37cm. Si osserva quindi che il valore massimo (84 cm) è molto superiore alle rimanenti osservazioni. La variabilità media intorno alla media è di circa 6 cm, valore piuttosto limitato se confrontato con la media. Per valutare meglio i dati osservati, ed in particolare il valore del massimo rispetto al resto delle osservazioni, li rappresentiamo graficamente.

```
plot(cranio1, type = "p")
```



```
hist(cranio1, breaks = 60)
```

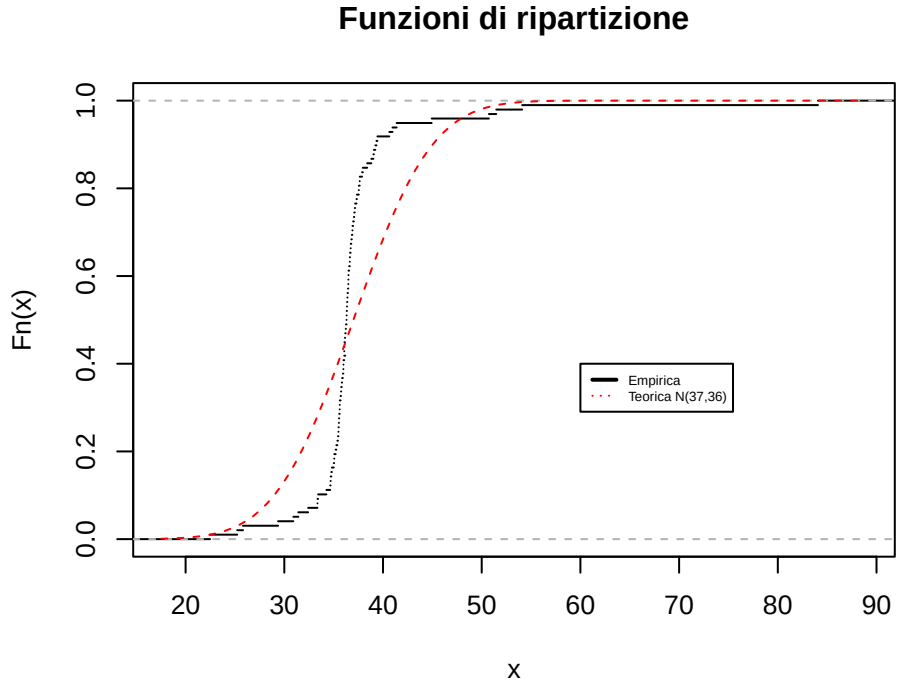


Si osserva che il valore massimo di 84 cm è molto isolato rispetto al resto delle osservazioni, nessuna delle quasi supera i 60 cm di circonferenza. Più in generale si nota che la maggiore parte delle osservazioni è concentrata tra i 30 e i 40 cm, e solo poche si trovano al di fuori di questa regione.

Si ipotizza una distribuzione Normale univariata per la V.C. sottostante la misura della circonferenza del cranio; disegnamo quindi la funzione di ripartizione empirica dei dati insieme quella teorica di una V.C. di Gauss, fissando i valori dei parametri in base alle statistiche campionarie, ovvero $X \sim N(37, 36)$.

```
ind <- which.max(cranio1)
plot(ecdf(cranio1),
     do.points=FALSE,
     main ='Funzioni di ripartizione')
curve(pnorm(x, 37, sd(cranio1)),
      lty = 'dashed',
      col = 'red',
      add = TRUE)

legend(60, 0.4,
       c("Empirica", "Teorica N(37,36)"),
       col = c("black", "red"),
       lty = c(1, 3),
```

```
       lwd = c(2, 1),
       cex = 0.5)
```



Si nota che la curva empirica non si sovrappone in modo perfetto a quella teorica. Si evidenzia che la distrbuzione dei dati osservati presenta code meno accentuate rispetto ad una V.C. Normale.

23.2

La stima dell'errore standard (misura di accuratezza) per la circoferenza media si ottiene con il bootstrap non parametrico utilizzano la funzione `bootstrap` dell'omonimo pacchetto. Questa richiede in input il nome del dataset, il numero di replicazioni, e la funzione che si vuole calcolare su ciascina delle replicazioni. In questo caso si considerano 2000 replicazioni del dataset originale.

```
require(bootstrap)
set.seed(153)
b.boot <- bootstrap::bootstrap(cranio1, 2000, mean)
summary(b.boot$thetastar)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   35.32   36.52   36.92   36.96   37.35   39.60
```

L'oggetto `b.boot$thetastar` contiene i risultati della media calcolata su ciascuna delle 2000 replicazioni bootstrap. Visualizzandone le statistiche descrittive, si osserva che i valori delle medie sono

tutti piuttosto vicini tra di loro. La media (è una media di medie) è pari a 39.96, approssimativamente uguale a quella calcolata sul dataset originale. Il campo di variazione va da un minimo di 35.32 ad un massimo di 39.60.

```
sd(b.boot$thetastar)
```

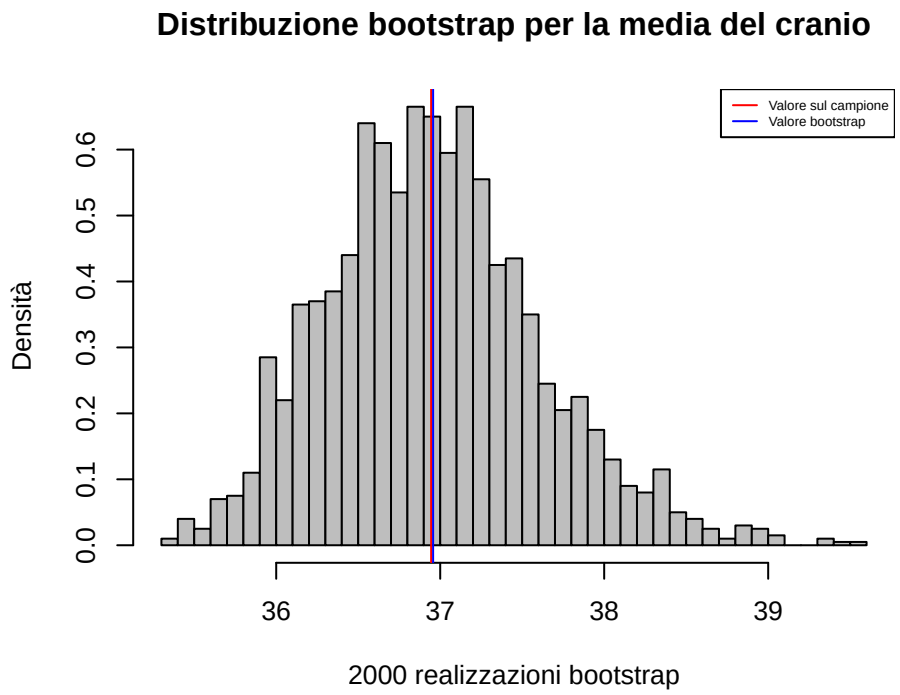
```
#> [1] 0.6463134
```

La misura di accuratezza si calcola attraverso la deviazione standard delle 2000 medie ottenute attraverso le replicazioni bootstrap; tale valore è pari a 0.64 e risulta piuttosto limitato.

23.3

Si disegna la distribuzione bootstrap per la media attraverso l'istogramma delle 2000 medie calcolate sulle replicazioni bootstrap del dataset originale.

```
temp <- range(b.boot$thetastar)
hist(b.boot$thetasta,
      breaks = 50,
      freq = FALSE,
      main = "Distribuzione bootstrap per la media del cranio",
      xlab = "2000 realizzazioni bootstrap",
      # xlim = temp,
      # ylim = c(0,0.7),
      col = "gray",
      ylab = "Densità")
abline(v =  mean(cranio1), col = "red")
abline(v = mean(b.boot$thetastar), col = "blue")
legend("topright",
       c("Valore sul campione", "Valore bootstrap"),
       col = c("red", "blue"),
       lty = c(1, 1),
       cex = 0.5)
```

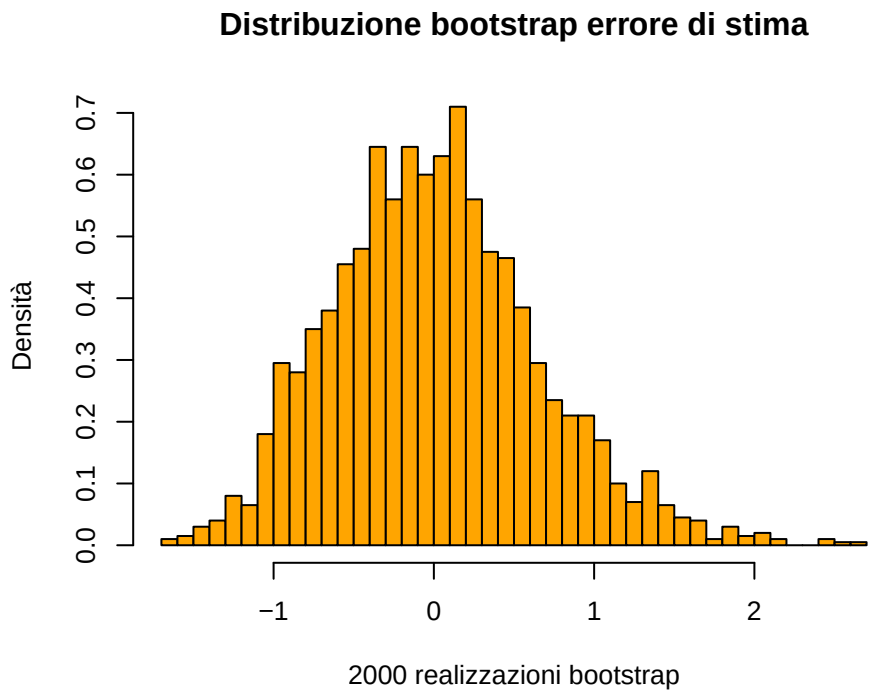


La distribuzione bootstrap per le medie delle misura di coroconferenza del cranio si presenta abbastanza simmetrica. La media calcolata sul campione originale e la medie delle 2000 medie sulle replicazioni bootstrap sono coincidenti (risultano sovrapposte nel grafico) e si trovano nel picco centrale dell'istogramma.

23.4

Rappresentiamo graficamente la distribuzione dell'errore di stima, calcolato come differenza tra la media calcolata sul campione originale e la media calcolata su ciascun campione bootstrap.

```
dif <- b.boot$thetastar - mean(cranio1)
hist(dif,
     breaks = 50,
     freq = FALSE,
     main = "Distribuzione bootstrap errore di stima",
     xlab = "2000 realizzazioni bootstrap",
     col  = "orange",
     ylab = "Densità")
```



La distribuzione dell'errore di stima è approssimativamente simmetrica, centrata intorno a 0. Il campo di variazione è piuttosto limitato, andando da un minimo di -2 ad un massimo poco superiore a 2. Inoltre la maggior parte delle osservazioni si trovano nella zona centrale, segno di un errore di stima particolarmente limitato.

SOLUZIONI ESERCIZIO 24

24.1

Valore del peso medio: 91.3 g.

```
x <- c(80, 103, 91)
mean(x)
```

```
#> [1] 91.33333
```

24.2

Numero di possibili campioni bootstrap: $\binom{5}{3} = 10$.

```
n <- length(x)
choose(2*n-1, n)
```

```
#> [1] 10
```

24.3

Possibili campioni Bootstrap: (80, 80, 80), (80, 80, 91), (80, 80, 103), (80, 91, 91), (80, 103, 103), (80, 91, 103), (91, 91, 91), (91, 91, 103), (91, 103, 103), (103, 103, 103).

Li raccogliamo nel dataframe `df`; ogni colonna corrisponde ad una diversa replicazione.

```
df <- data.frame(c(80, 80, 80), c(80, 80, 91), c(80, 80, 103), c(80, 91, 91), c(80, 103, 103),
colnames(df) <- as.character(1:10)
```

24.4

Medie (nello stesso ordine dei campioni scritti al punto precedente): 80.0, 83.7, 87.7, 87.3, 95.3, 91.3, 91.0, 95.0, 99.0, 103.0.

```
medie_bootstrap <- round(apply(df, 2, mean), 1)
```

24.5

Medie delle medie ottenute dai campioni bootstrap: 91.3 Il valore coincide con quello determinato sul campione originale al punto 1.

```
mean(medie_bootstrap)
```

```
#> [1] 91.33
```

24.6

Valori minimi:

```
apply(df, 2, min)
```

```
#>   1   2   3   4   5   6   7   8   9  10
#>  80  80  80  80  80  80  91  91  91 103
```

Valori massimi:

```
apply(df, 2, max)
```

```
#>   1   2   3   4   5   6   7   8   9  10
#>  80  91 103  91 103 103  91 103 103 103
```

SOLUZIONI ESERCIZIO 25

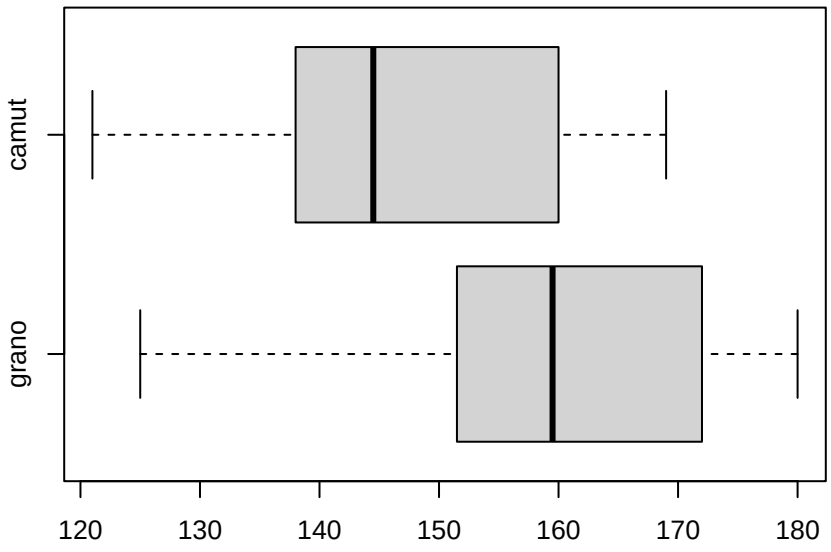
25.1

Si raccolgono i dati in due vettori e si effettuano le analisi delle statistiche descrittive e delle rappresentazioni grafiche.

```
grano <- c(176, 125, 152, 180, 159, 168, 160, 151)
camut <- c(164, 121, 137, 169, 144, 145, 156, 139)
dat <- data.frame(grano, camut)
summary(dat)
```

```
#>      grano           camut
#>  Min.   :125.0   Min.   :121.0
#>  1st Qu.:151.8   1st Qu.:138.5
#>  Median :159.5   Median :144.5
#>  Mean   :158.9   Mean   :146.9
#>  3rd Qu.:170.0   3rd Qu.:158.0
#>  Max.   :180.0   Max.   :169.0
```

```
boxplot(dat, horizontal = T)
```



I commenti dei risultati sono lasciati allo studente. Si noti che le osservazioni sono poche ed il campione potrebbe non essere abbastanza rappresentivo della popolazione.

25.2

Le medie delle due serie di dati sono rispettivamente 159 g (per il grano) e 147 (per il camut). La differenza è di 12 g.

Nel seguito, ad ogni iterazione del ciclo `for`, vengono selezionate in modo casuale (e con ripetizioni) n righe del dataset originale; si calcolano le medie delle due colonne del nuovo dataset e se ne calcola la differenza. Il tutto viene ripetuto 2000 volte (i.e. si ottengono 2000 ripetizioni bootstrap) e i risultati vengono raccolti nel vettore `DDB`.

```
n <- dim(dat)[1]
B <- 1000
DDB <- rep(0, B)
set.seed(123)
for(i in 1:B){
  ind <- sample(1:n, size = n, replace = TRUE)
  Y <- dat[ind,"grano"]
  Z <- dat[ind,"camut"]
  CC <- mean(Y)-mean(Z)
  DDB[i] <- CC
}
summary(DDB)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>    4.00   10.50   11.88   11.86   13.25   19.62
```

Considerando 1000 ripetizioni bootstrap la differenza tra le medie relative a grano e camut variano da un minimo di 4 ad un massimo di quasi 20. Il valore medio delle differenza è invece pari a 12, approssimativamente uguale alla mediana. I due indici coincidono con il valore della differenza tra le medie calcolate sul campione originale.

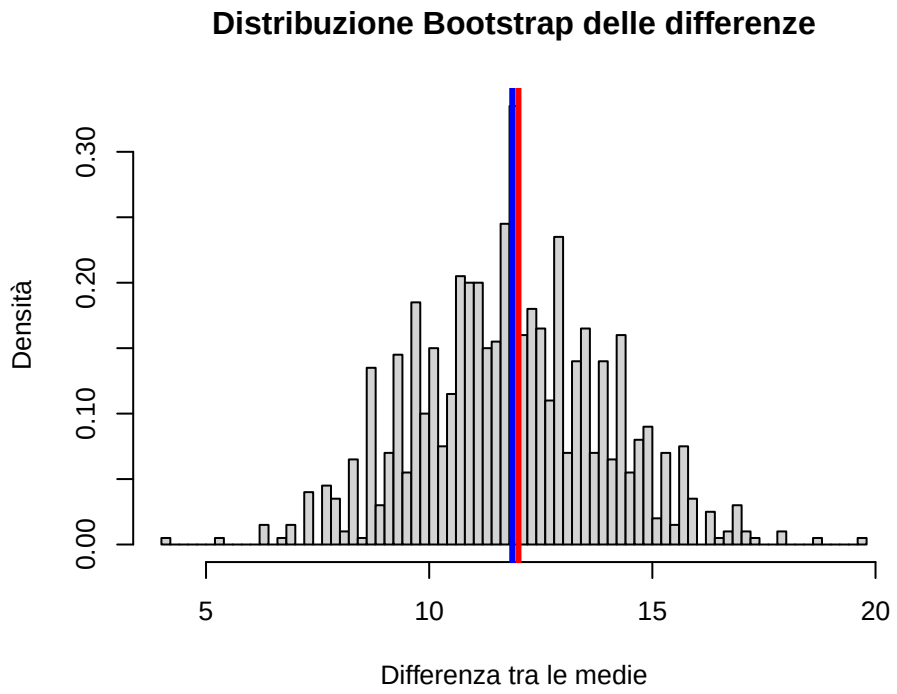
Si provi ad implementare la stessa richiesta utilizzando la funzione `bootstrap::bootstrap`; si confrontino i risultati. (Hint: si costruisca una funzione che riceve in input le righe corrispondenti alla replica bootstrap e calcoli la differenza delle medie. Si applichi questa funzione al comando `bootstrap`)

NOTA: nello svolgimento di questo esercizio ogni ricampionamento viene effettuato congiuntamente per le due rilevazioni: si estrae (casualmente e con reimmissione) un indice $i \in \{1, \dots, n\}$ e si considera l'i-esimo elemento sia del vettore grano sia del vettore camut. In generale si potrebbe anche procedere indipendentemente: estraggo due indici $i, j \in \{1, \dots, n\}$ e considero l'i-esimo elemento del vettore grano ed il j-esimo del vettore camut. La definizione del metodo più opportuno dipende dalle singole applicazioni. Nel nostro caso i due dataset originali sembrano da considerare congiuntamente: è sufficiente plottarli nel piano cartesiano (`plot(grano, camut)`), o più semplicemente calcolarne l'indice di correlazione per rilevare che sono fortemente (e positivamente) linearmente associate. In altre applicazioni potrebbe essere più opportuno procedere invece considerando i due dataset indipendnetemente.

25.3

Si rappresenta graficamente la distribuzione bootstrap in forma di istogramma. Si aggiungono poi le linee corrispondenti alla differenza calcolata sul dataset originale e la media delle differenze calcolate sui dataset bootstrap.

```
diff <- mean(grano)-mean(camut)
B_diff <- mean(DDB)

hist(DDB,
     main = "Distribuzione Bootstrap delle differenze",
     breaks = 60,
     freq = FALSE,
     ylab = "Densità",
     xlab = "Differenza tra le medie")
abline(v = c(diff, B_diff),
       col = c("red", "blue"),
       lwd = c(3, 3))
legend(15, 0.7,
       c("difOR", "difB"),
       col = c("red", "blue"),
       lty = c(1, 1),
       cex = 0.6)
```



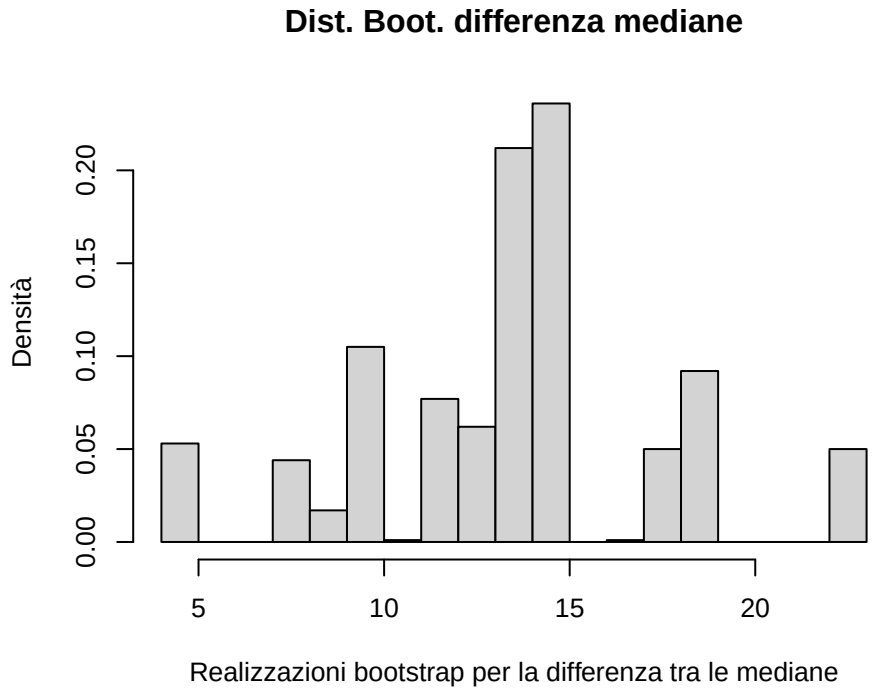
Il campo di variazione, come già descritto in precedenza, varia all'incirca tra 5 e 20. La distribuzione è approssimativamente simmetrica attorno al picco centrale, posto intorno al valore 12, e coincidente

sia con la differenza tra le medie calcolate sui dati originali, sia con la medie delle differenze bootstrap.

25.4

Si ripete quanto implementato in precedenza sostituendo la differenza delle mediane alla differenza delle medie.

```
DDB <- rep(0,B)
set.seed(123)
for(i in 1:B){
  ind <-sample(1:n, size = n, replace = TRUE)
  Y <- dat[ind,"grano"]
  Z <- dat[ind,"camut"]
  CC<- median(Y)-median(Z)
  DDB[i] <- CC
}

hist(DDB,
     main = "Dist. Boot. differenza mediane",
     breaks = 15,
     freq = FALSE,
     ylab = "Densità",
     xlab = "Realizzazioni bootstrap per la differenza tra le mediane")
```



In questo caso si nota che le differenze tra le mediane calcolate sui campioni bootstrap tendono ad assumere sempre un ristretto numero di valori. Per esempio si osserva un elevato numero di differenze con valore tra 13 e 15 e quasi nulla tra 15 e 17. Questo è spiegabile con le poche osservazioni disponibili originariamente: il bootstrap tende a ripetere le osservazioni e di conseguenza anche i valori del risultato finale. In questo caso con il bootstrap non è possibile stimare alcuni quantili della distribuzione.

SOLUZIONI ESERCIZIO 26

26.1

Si caricano i dati in un vettore e se ne forniscono le statistiche descrittive e le principali rappresentazioni grafiche. I commenti sono lasciati allo studente.

```
x <- c(23, 16, 21, 24, 34, 28, 28, 28, 24, 30, 28, 24, 26, 18,
       23, 23, 36, 37, 49, 50, 51, 56, 46, 41, 54, 30, 40, 31)
summary(x)
```

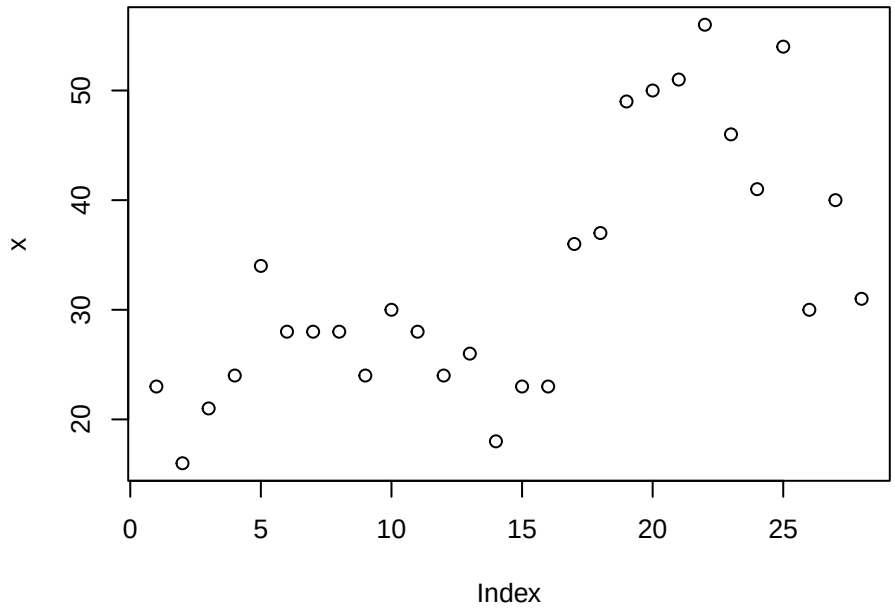
```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   16.00   24.00   29.00   32.82   40.25   56.00
```

```
sd(x)
```

```
#> [1] 11.41491
```

Media e mediana, che saranno oggetto di analisi con la tecnica bootstrap nei prossimi punti dell'esercizio, valgono rispettivamente 33 e 29.

```
plot(x)
```



Si osserva che le prime rilevazioni presentano tutte valori bassi (inferiori a 40) per il numero di casi di tumore raro. Le ultime rilevazioni assumono invece valori molto più elevati.

26.2

Per applicare la procedura bootstrap (con il comando `bootstrap::bootstrap`), definiamo dapprima la funzione necessaria per il calcolo di media e mediana sui campioni bootstrap.

La funzione `bootstrap::bootstrap` implementata esegue la procedura bootstrap campionando in modo casuale e con reinserimento gli indici delle righe del dataset originario. Il numero di diverse replicazioni è fissato a 3000.

```
n <- length(x)
B <- 3000
set.seed(2410)
mean_B <- bootstrap::bootstrap(1:n, B, function(i) mean(x[i]))
set.seed(2410)
median_B <- bootstrap::bootstrap(1:n, B, function(i) median(x[i]))
```

Si noti che settando lo stesso valore per il seme prim di entrambe le chiamate della funzione `bootstrap::bootstrap` si utilizzano gli stessi campioni sia per il calcolo con le medie sia per quello con le mediane. Analizziamo ora i risultati relativi ai due indici calcolandone l'errore standard.

```
sd(mean_B$thetastar)
```

```
#> [1] 2.161022
```

```
sd(median_B$thetastar)
```

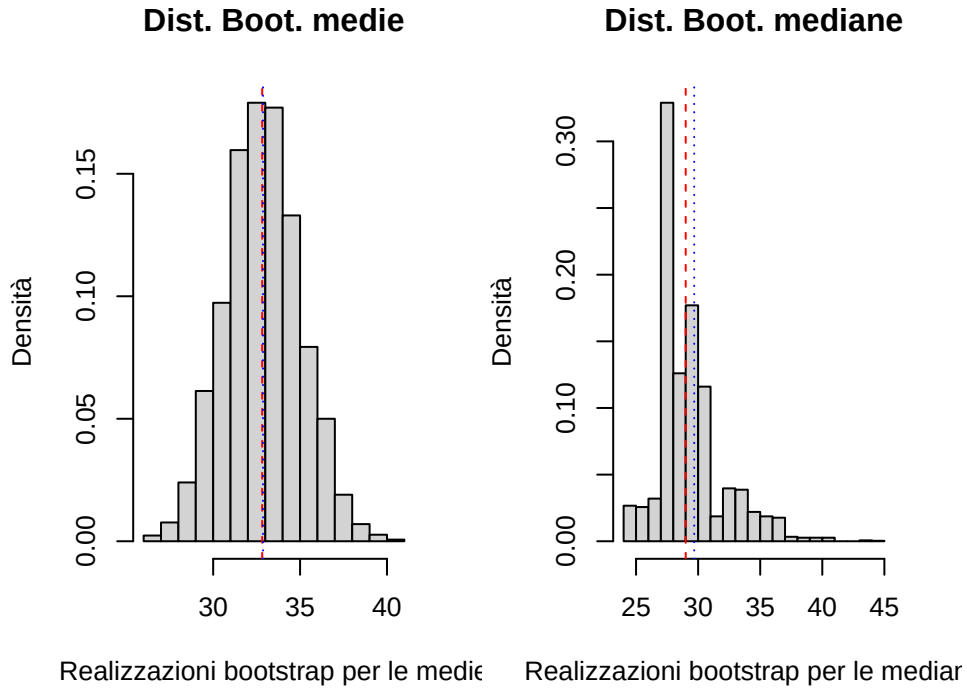
```
#> [1] 2.650219
```

Si osserva che entrambi i valori dello standard error sono piuttosto contenuti; l'accuratezza per entrmbi gli stimatori (media e mediana campionarie), stimata attraverso il metodo bootstrap sulla base di 3000 replicazioni del campione originario, è quindi elevata. In particolare che il valore dello standard error per la media risulta essere leggermente più basso rispetto a quello per la mediana; è noto che il metodo bootstrap non funziona sempre in modo ottimale per i quartili.

26.3

Riportiamo in un'unica finestra grafica gli istogrammi delle due distribuzioni bootstrap.

```
par(mfrow = c(1, 2))
hist(mean_B$thetastar,
     main = "Dist. Boot. medie",
     breaks = 15,
     freq = FALSE,
     ylab = "Densità",
     xlab = "Realizzazioni bootstrap per le medie")
abline(v = mean(x), col = "red", lty = 2)
abline(v = mean(mean_B$thetastar), col = "blue", lty = 3)

hist(median_B$thetastar,
     main = "Dist. Boot. mediane",
     breaks = 15,
     freq = FALSE,
     ylab = "Densità",
     xlab = "Realizzazioni bootstrap per le mediane")
abline(v = median(x), col = "red", lty = 2)
abline(v = mean(median_B$thetastar), col = "blue", lty = 3)
```



I due grafici appaiono significativamente diversi tra loro. Quello relativo alla media risulta quasi perfettamente simmetrico; i valori della media sul dataset orioginae e della media sui 3000 valori bootstrap coincidono tra loro e si posizionano in corrispondenza del picco centrale. Il ricampionamento bootstrap non è invece stato altrettanto efficacie per quanto riguarda la mediama: la forma dell'istogramma non si presenta simmetrica, mostrando un unico picco molto elevato in corrispondenza di un valore compreso tra 27 e 28 (non al centro della distribuzione). Questo picco non coincide nè con la mediana calcolata sul dataset originale, nè con la media delle mediane calcolate sui 3000 campioni bootstrap. Inoltre questi due valori sono a loro volta non coincidenti. Il range di questa disytribuzione bootstrap è più ampio rispetto a quello ottenuto nel caso della media.

Per quanto riguarda la distorsione dei due stimatori, calcoliamo la differenza tra il valore dello stimatore ottenuto sul dataset originale e la media dei valori bootstrap.

```
mean(mean_B$thetastar) - mean(x)
```

```
#> [1] 0.04861905
```

```
mean(median_B$thetastar) - median(x)
```

```
#> [1] 0.6853333
```

La distorsione ottenuta per lo stimatore della media è molto bassa (pari a 0.05), segno che lo stimatore (media campionaria) non risulta distorta. Al contrario, il valore della distorsione per la mediana è di 0.7, indice di una leggera distorsione per lo stimatore corrispondente.

26.4

Si rimanda alle soluzioni dei punti e degli esercizi precedenti.

SOLUZIONI ESERCIZIO 27

27.1

Si carica il dataset e se ne analizzano le principali statistiche decsrittive.

```
load("warfarin.Rdata")
skimr::skim_without_charts(warfarin[, -1])
```

Table 39: Data summary

Name	warfarin[, -1]
Number of rows	104
Number of columns	2
_____	
Column type frequency:	
numeric	2
_____	
Group variables	None

Variable type: numeric

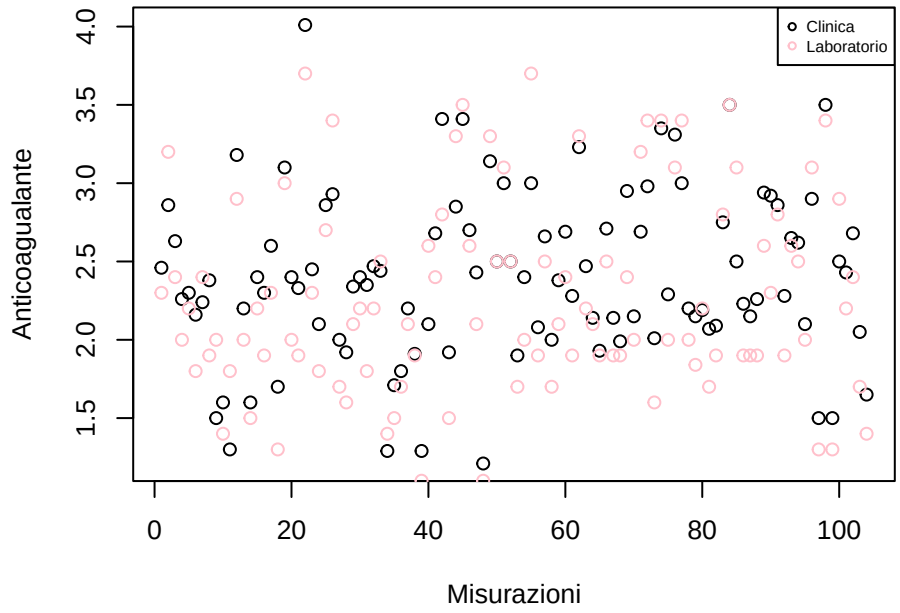
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
cread	0	1	2.40	0.54	1.21	2.1	2.38	2.7	4.01
lread	0	1	2.28	0.63	1.10	1.9	2.15	2.6	3.70

Sono presenti 104 individui, per i quali sono rilevate le misurazioni di due variabili. Le due distribuzioni hanno campo di variazione molto simile, assumendo solo valori positivi, e con un massimo intorno a 4. La media delle misurazioni dell'anticoagulante rilevate in clinica risulta superiore a quella dei valori rilevati in laboratorio. Viceversa queste misurazioni risultano maggiormente variabili, avendo un valore della deviazione standard più elevato rispetto a quelle rilevate in clinica.

Rappresentiamo ora le osservazioni nel piano cartesiano (grafico a dispersione), utilizzando colori diversi per le misurazioni in clinica e quelle in laboratorio.

```
plot(warfarin[, 2],
     ylab="Anticoagualante",
     main = "",
     xlab = "Misurazioni")
points(warfarin[, 3],
```

```
       col="pink")
legend("topright",
       col = c("black", "pink"),
       c("Clinica", "Laboratorio"),
       pch = c(1, 1),
       cex = 0.6)
```

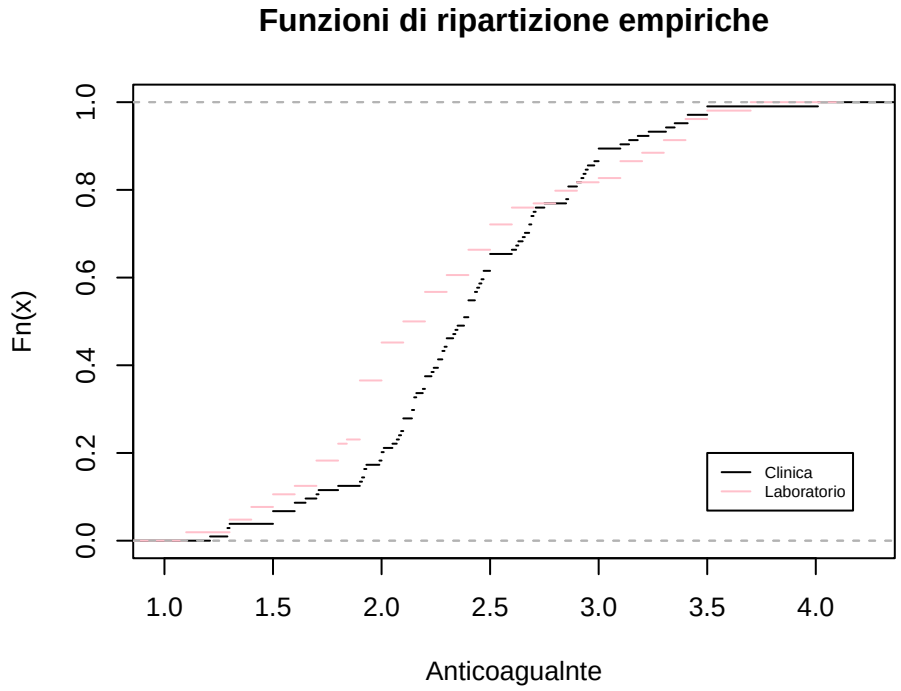


I valori ottenuti attraverso entrambe le tipologie di misurazione si dispongono in modo uniforme nel piano, senza formare strutture di punti particolari (pattern) o prevalenza di uno dei due metodi di misurazione in determinate regioni del piano. I valori ottenuti in clinica sembrano essere superiori rispetto a quelli ottenuti in laboratorio, ma la differenza è estremamente limitata (come già visto con le statistiche descrittive).

Rappresentiamo anche le due funzioni di ripartizione empiriche in uno stesso grafico in modo da poterle confrontare.

```
plot(ecdf(warfarin[, 2]),
     col = "black",
     xlab = "Anticoagualnte",
     do.points = FALSE,
     main = "Funzioni di ripartizione empiriche")
plot(ecdf(warfarin[, 3]),
     col = "pink",
     do.points = FALSE,
     add = TRUE)
```

```
legend(3.5, 0.2,
       col = c("black", "pink"),
       c("Clinica", "Laboratorio"),
       lty = c(1, 1),
       cex = 0.6)
```



A parità di ascissa (ovvero di misurazione di anticoagualante), e fino al valore $x = 2.7$, la funzione di ripartizione per i valori ottenuti in clinica è inferiore rispetto a quella dei valori ottenuti in laboratorio. In altre parole, per valori $x \leq 2.7$ di anticoagulante, la probabilità $\mathbb{P}(X_{Clinica} \leq x)$ è sempre minore della probabilità $\mathbb{P}(X_{Laboratorio} \leq x)$. La situazione si inverte per valori di x superiori a 2.7. L'inclinazione di entrambe le curve si attenua per valori elevati dell'anticoagulante.

27.2

Si applica lo stimatore di plug-in per il rapporto tra le medie aritmetiche, dove si indica con C le misurazioni in clinica e con L quelle in laboratorio:

$$\theta_{plug-in} = \frac{\bar{C}}{\bar{L}}.$$

```
th <- mean(warfarin$cread)/mean(warfarin$lread); th
```

```
#> [1] 1.054828
```

Si nota che la stima $\hat{\theta}_{plug-in}$ sul campione è di 1.05, ovvero di poco superiore a 1.

Occorre considerare che lo stimatore è **distorto** in quanto è il rapporto tra due medie aritmetiche.

27.3

Per ottenere l'errore standard della stima calcolata al punto precedente si utilizza la funzione `bootstrap::bootstrap`; si implementa quindi dapprima la funzione che definisce lo stimatore plug-in di interesse per un generico campione bootstrap. Tale funzione richiede in input l'argomento numerico che definisce la posizione delle unità `ind` che si usano per il campione bootstrap di volta in volta e restituisce la stima plug-in calcolata su quel campione.

```
theta1 <- function(ind) {
  Y <- warfarin[ind, 2]
  Z <- warfarin[ind, 3]
  mean(Y)/mean(Z)
}
```

La funzione `bootstrap::bootstrap` esegue la procedura bootstrap campionando in modo casuale e con reinserimento gli indici delle righe del dataset originario. Il numero di diverse replicazioni è fissato a 2000.

```
n <- dim(warfarin)[1]
B <- 2000
set.seed(18243)
bootw <- bootstrap::bootstrap(1:n, B, theta1)
head(bootw$thetastar)
```

```
#> [1] 1.056795 1.056041 1.034042 1.058653 1.040985 1.049019
```

L'oggetto in output, `bootw$thetastar`, restituisce il vettore che contiene i 2000 valori della stima plug-in relizzati su ogni campione estratto. Visualizzando i primi 10 valori si nota che sono tutti leggermente superiori a 1. Attraverso la funzione summary ne analizziamo le prinicipali statistiche descrittive.

```
summary(bootw$thetastar)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   1.006   1.046   1.055   1.055   1.064   1.096
```

Il valore minimo è pari a 1.006, confermando che nessuna delle stime bootstrap assume valore inferiore a 1. Il valore massimo è invece di 1.096, denotando così un range molto limitato. Media e mediano coincidono tra loro e approssimativamente con il valore del rapporto tra le medie calcolato sul dataset originale.

La deviazione standard delle replicazioni bootstrap consente invece di valutare la variabilità dovuta al campionamento casuale (errore standard).

```
sd(bootw$thetastar)
```

```
#> [1] 0.01262892
```

Infine si calcola la differenza riscontrata tra la stima del parametro calcolata sui dati originali ed il valore medio ottenuto attraverso il bootstrap.

```
Tm1 <- mean(bootw$thetastar)
Tm1 - th
```

```
#> [1] 0.0002483279
```

Tale valore risulta piuttosto piccolo; la distorsione è pertanto trascurabile e non occorre correggere la stima.

SOLUZIONI ESERCIZIO 28

28.1

Si rimanda alle soluzioni del punto 28.3.

28.2

Si rimanda alle soluzioni del punto 28.3.

28.3

Si considerano i dati relativi all'Esercizio 27 circa le misure di anticoagulante. Si riporta l'intervallo di confidenza a livello del 90% per il rapporto tra le medie ottenuto con il metodo del percentile e con il metodo Bias Corrected Accelerated Bootstrap (BCAB). Si noti che le quantità già introdotte nell'esercizio 27 non vengono qui ricalcolate.

L'intervallo di confidenza bootstrap calcolato con il *metodo del percentile* si basa sulla distribuzione empirica delle replicazioni bootstrap. Si utilizzano come limiti dell'intervallo di confidenza i quantili appropriati della distribuzione delle replicazioni bootstrap.

In particolare fissando il livello di copertura pari a 0.90 si determinano gli estremi considerando il quinto e il novantacinquesimo percentile. Il comando `quantile` consente di calcolare i quantili specificati.

```
Q <- quantile(bootw$thetastar, c(0.05,0.95)); Q
```

```
#>       5%      95%
#> 1.034852 1.075985
```

Al livello di confidenza del 90% si nota che l'intervallo contiene valori sempre superiori a 1; possiamo quindi concludere che i valori rilevati in clinica sono significativamente superiori a quelli rilevati nel laboratorio.

Per il calcolo dell'intervallo di confidenza (a livello di confidenza del 90%) con il *metodo BCAB* si utilizza invece la funzione `bcanon` del pacchetto `bootstrap`. Questa richiede gli stessi argomenti della funzione bootstrap (intervallo di valori su cui effettuare il ricampionamento, numero di copie bootstrap, funzione da applicare) oltre al valore del livello di confidenza. Ricordiamo che questo intervallo di confidenza corregge per distorsione e utilizza una costante di accellerazione.

```
require(bootstrap)
set.seed(1023)
n <- dim(warfarin)[1]
CIBca <- bootstrap::bcanon(1:n, nboot = 2000, theta1, alpha = c(0.05,0.95))
CIBca$confpoints
```

```
#>      alpha bca point
#> [1,]  0.05  1.032924
#> [2,]  0.95  1.076306
```

E' possibile ottenere l'intervallo così calcolato specificando il campo `$confpoints` dell'oggetto definito. Notiamo che al livello di confidenza del 90% gli estremi dell'intervallo di confidenza differiscono da quelli ottenuti con il metodo del percentile. In particolare l'ampiezza dell'intervallo risulta leggermente maggiore. L'estremo inferiore risulta comunque minore di 1, confermando la conclusione ottenuta in precedenza che i valori rilevati in clinica sono significativamente superiori a quelli rilevati in laboratorio.

```
CIBca$acc
```

```
#> [1] -0.01751753
```

```
CIBca$z0
```

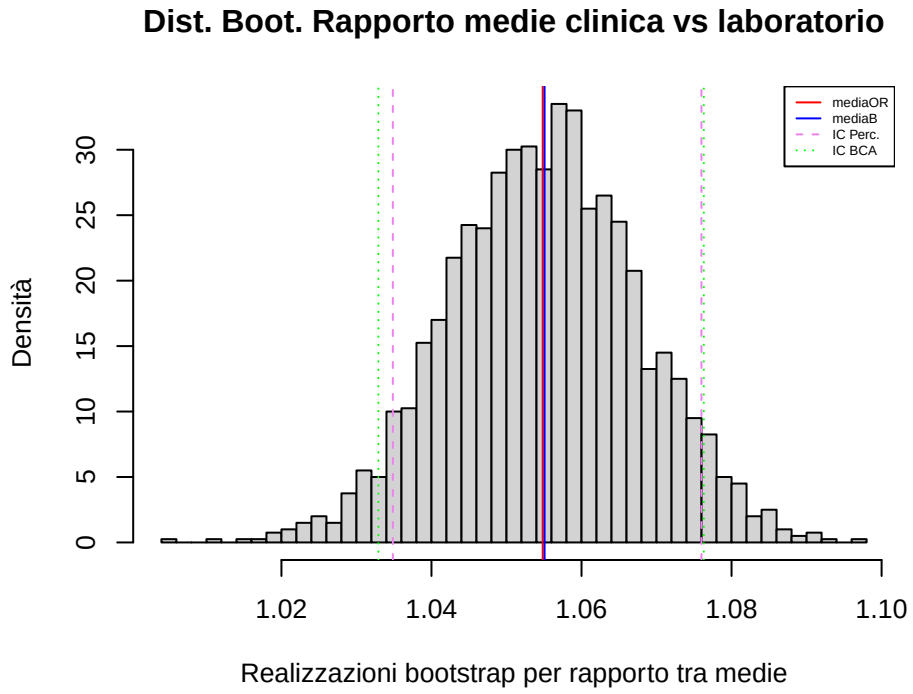
```
#> [1] 0.03133798
```

Possiamo anche ottenere i valori per le costanti di accelerazione e di correzione per la distorsione, pari rispettivamente a -0.02 e 0.03, entrambe diverse da zero (seppur di poco).

Infine aggiungiamo gli estremi degli intervalli di confidenza ottenuti con i due metodi alla rappresentazione grafica della distribuzione bootstrap.

```
hist(bootw$thetastar,
     main = "Dist. Boot. Rapporto medie clinica vs laboratorio",
     breaks = 60,
     freq = FALSE,
     ylab = "Densità",
```

```
     xlab = "Realizzazioni bootstrap per rapporto tra medie")

abline(v = c(th, Tm1,
             Q[1], Q[2],
             CIBca[["confpoints"]][3], CIBca[["confpoints"]][4]),
       col = c("red", "blue", "violet","violet", "green", "green"),
       lty = c(1, 1, 2, 2, 3, 3))

legend("topright",
       c("mediaOR", "mediaB", "IC Perc.", "IC BCA"),
       col = c("red", "blue", "violet", "green"),
       lty = c(1, 1, 2, 3),
       cex = 0.5)
```



La rappresentazione grafica della distribuzione bootstrap mostra che i valori ottenuti si distribuiscono in modo approssimativamente simmetrico. La media calcolata sul campione originale e la media delle stime bootstrap coincidono e si dispongono approssimativamente in corrispondenza del picco centrale e più alto dell'istogramma. Le linee degli intervalli di confidenza sono molto vicine tra loro, anche se l'ampiezza di quello ottenuto con metodo bootstrap risulta leggermente maggiore (come già notato in precedenza).

SOLUZIONI ESERCIZIO 29

29.1

Si analizzano i dati riferiti alla presenza di metastasi relativi a 139 donne con tumore al seno rispetto al trattamento immunoterapico. Si costruisce il dataframe corrispondente, denotando con X la variabile trattamento immunoterapico e con Y la variabile metastasi. Il rischio relativo si calcola come rapporto tra la frequenza delle donne con metastasi e trattamento immunoterapico e la frequenza delle donne con metastasi ma senza trattamento immunoterapico.

```
dataR <- data.frame(X = rep(c("TRUE", "FALSE"), c(59, 80)),
                    Y = rep(c(TRUE, FALSE, TRUE, FALSE), c(10, 59-10, 50, 80-50)))
table(dataR)
```

```
#>         Y
#> X        FALSE TRUE
#>   FALSE    30   50
#>   TRUE     49   10
```

```
CC <- proportions(table(dataR), 1); CC
```

```
#>         Y
#> X            FALSE      TRUE
#>   FALSE 0.3750000 0.6250000
#>   TRUE  0.8305085 0.1694915
```

```
RR <- CC[4]/CC[3]; RR
```

```
#> [1] 0.2711864
```

La stima di massima verosimiglianza del rischio relativo di sviluppare metastasi è 0.27, significativamente distante da 1. Se ne deduce che c'è un'associazione tra la presenza di metastasi e il trattamento immunoterapico: la probabilità di avere metastasi è circa quattro volte inferiore per le donne che hanno eseguito un trattamento immunoterapico rispetto alle donne che non l'hanno eseguito.

29.2

Utilizziamo il metodo bootstrap per determinare una misura di accuratezza per il risultato ottenutp al punto precedente. Il dataset da cui ricampionare è costituito da due colonne, le variabili presenza/assenza di metastasi e trattamento immunoterapico. Ogni riga del dataset contiene le informazioni relative ad una donna: occorre quindi ricampionare *congiuntamente* le due colonne. Si procede quindi ricampionando l'indice di riga (da 1 ad n) e successivamente ricavando la riga corrispondente nel dataframe. Dopo aver determinato un campione bootstrap si applica la funzione `theta1` che permette di calcolare il rischio relativo in modo analogo a quanto fatto nel punto precedente.

```r
n <- dim(dataR)[1]

theta1 <- function(i) {
    CC <- prop.table(table(dataR[i, ]), 1)
    CC[4]/CC[3]
}

set.seed(2753)
boot_RR <- bootstrap::bootstrap(x = 1:n, nboot = 2000, theta = theta1)

summary(boot_RR$thetastar)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#> 0.06623 0.21818 0.27119 0.27561 0.32670 0.57845
```

Commentiamo brevemente i risultati attraverso le principali statistiche descrittive (funzione `summary`). Il campo di variazione delle 2000 replicazioni bootstrap del valore del rischio relativo è piuttosto limitato, variando da un minimo di 0.07 ad un massimo di 0.58. Si tratta quindi di valori sempre inferiori ad 1, segno che la probabilità di avere metastasi si riduce sensibilmente per le donne che si sono sottoposte al trattamento immunoterapico. In media (valor medio pari a 0.28) questa riduzione è pari a circa 3 volte e mezza (1/0.28). Il valore della media è approssimativamente uguale a quello della mediana e a quello del rischio relativo calcolato sul dataset originale.

Infine, per ottenere una misura di accuratezza per il risultato determinato al punto precedente, calcoliamo la deviazione standard dei valori del rischio relativo sulle 2000 replicazioni bootstrap.

```r
sd(boot_RR$thetastar)
```

```
#> [1] 0.08048537
```

Il risultato, pari a 0.08, sottolinea che l'errore standard è molto contenuto, segno di un'elevata accuratezza del valore del rischio relativo calcolato sul dataset originale.

29.3

Prima di rappresentare graficamente la distribuzione bootstrap, calcoliamo l'intervallo di confidenza con il metodo del percentile; si tratta semplicemente di determinare gli opportuni quantili della distribuzione bootstrap.

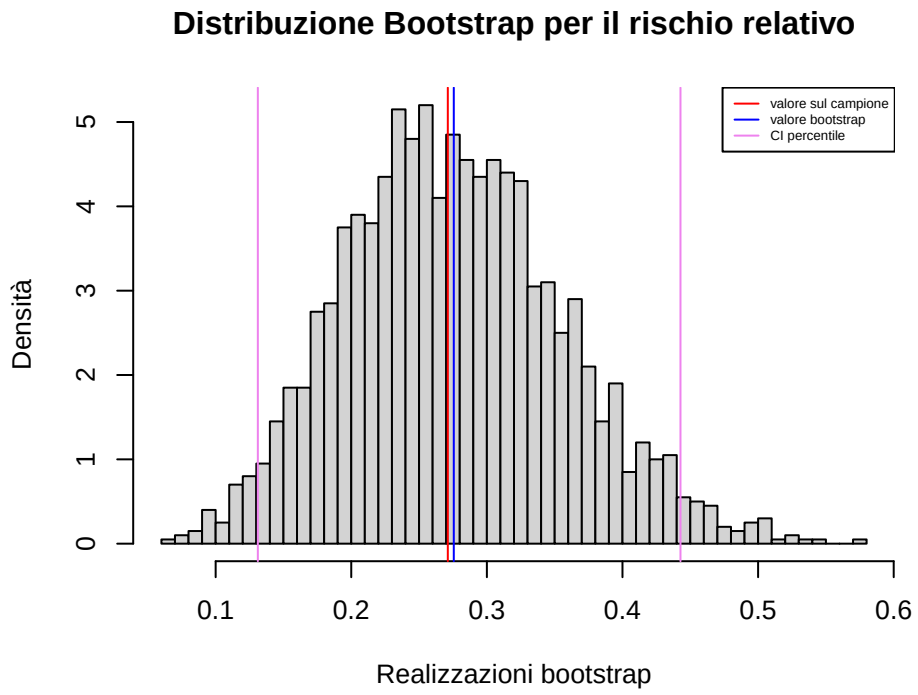
```r
Q <- quantile(boot_RR$thetastar, c(0.025, 0.975)); Q
```

```
#>      2.5%     97.5%
#> 0.1311163 0.4427633
```

Gli estremi dell'intervallo di confidenza bootrstap con livello di significatività pari a 0.95 sono 0.13 e 0.44: nel 95% dei casi (ovvero, considerando il 95% delle repliche bootstrap) il valore del rischio relativo si trova all'interno di questo intervallo.

Rappresentiamo ora la distribuzione bootrstap in forma di istogramma, aggiungendo gli estremi dell'intervallo di confidenza determinato, oltre al valore del rischio relativo calcolato sul dataset originale, e al valore medio sulle replicazioni bootstrap.

```
hist(boot_RR$thetastar,
     main = "Distribuzione Bootstrap per il rischio relativo",
     breaks = 60,
     freq = FALSE,
     ylab = "Densità",
     xlab = "Realizzazioni bootstrap")
abline(v = c(RR, mean(boot_RR$thetastar), Q[1], Q[2]),
       col = c("red", "blue", "violet", "violet"))
legend("topright",
       c("valore sul campione", "valore bootstrap", "CI percentile"),
       col = c("red", "blue", "violet"),
       lty= c(1, 1, 1),
       cex = 0.5)
```



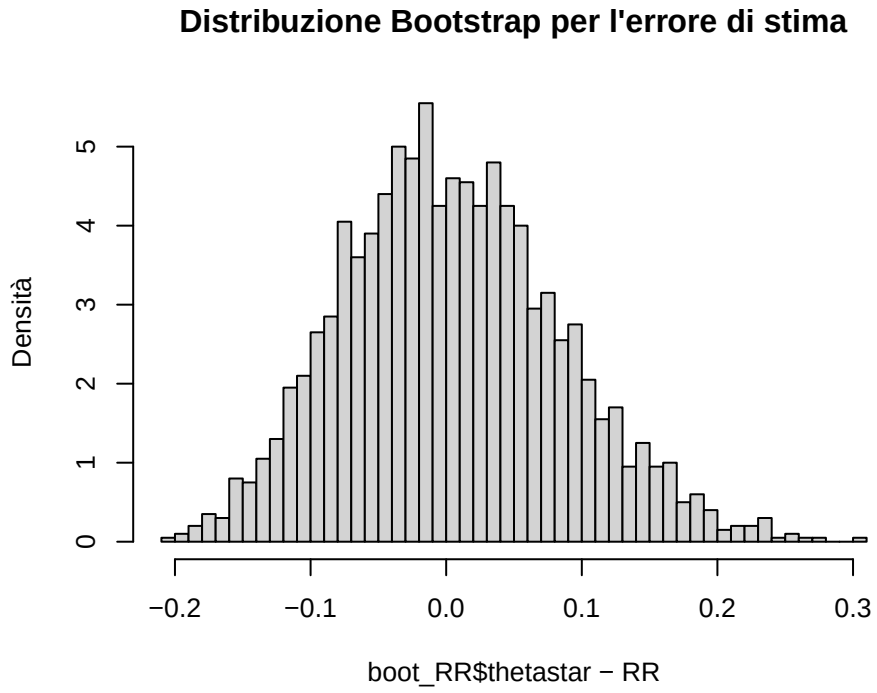
La forma dell'istogramma è approssimativamente simmetrica, pur presentando una coda destra leggermente più lunga rispetto a quella sinistra. Gli estremi dell'intervallo di confidenza comprendono la maggior parte delle osservazioni (il 95%), e i valori del rischio relativo e della media sulle replicazioni bootstrap si dispongono approssimativamente al centro dell'intervallo. Si osserva inoltre che

questi due valori si posizionano nella parte centrale dell'istogramma, pur rimanendo leggermente discostati dal picco più elevato della distribuzione bootstrap.

29.4

Per ottenere la distribuzione dell'errore di stima è sufficiente calcolare la differenza tra il valore del rischio relativo calcolato sul campione originale e la media dei valori bootstrap.

```
hist(boot_RR$thetastar - RR,
     main = "Distribuzione Bootstrap per l'errore di stima",
     breaks = 60,
     freq = FALSE,
     ylab = "Densità")
```



Anche in questo caso la forma dell'istogramma è approssimativamente simmetrica (ricalca ovviamente la forma dell'istogramma precedente). Si osserva che il campo di variazione è piuttosto limitato, avendo un minimo pari a -0.2 ed un massimo pari a 0.3. La distribuzione è centrata in 0 (dove si trova anche il picco più alto), segno che in media l'errore di stima è nullo.

29.5

Consideriamo ora lo stimatore della differenza tra le probabilità.

```
DD <- CC[4]-CC[3]; DD
```

```
#> [1] -0.4555085
```

La stima di massima verosimiglianza della differenza delle probabilità è -0.47. Trattandosi di un valore inferiore a 0 si conferma la conclusione tratta precedentemente che c'è un'associazione tra la presenza di metastasi e il trattamento immunoterapico. E' infatti noto che $RR = 1$ e $DD = 0$ sono due condizioni equivalenti (necessarie e sufficienti) per avere indipendenza statistica tra i due fenomeni.

29.6

In modo analogo a quanto fatto per lo stimatore del rischio relativo, si ottengono 2000 replicazioni bootstrap del dataset e s calcola la differenza delle probabilità su ciascuna di queste.

```
n <- dim(dataR)[1]

theta2 <- function(i) {
  CC <- prop.table(table(dataR[i, ]), 1)
  DD <- CC[4] - CC[3]
}

set.seed(2753)
boot_DD <- bootstrap::bootstrap(x = 1:n, nboot = 2000, theta = theta2)

summary(boot_DD$thetastar)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#> -0.6617 -0.5017 -0.4523 -0.4525 -0.4036 -0.2270
```

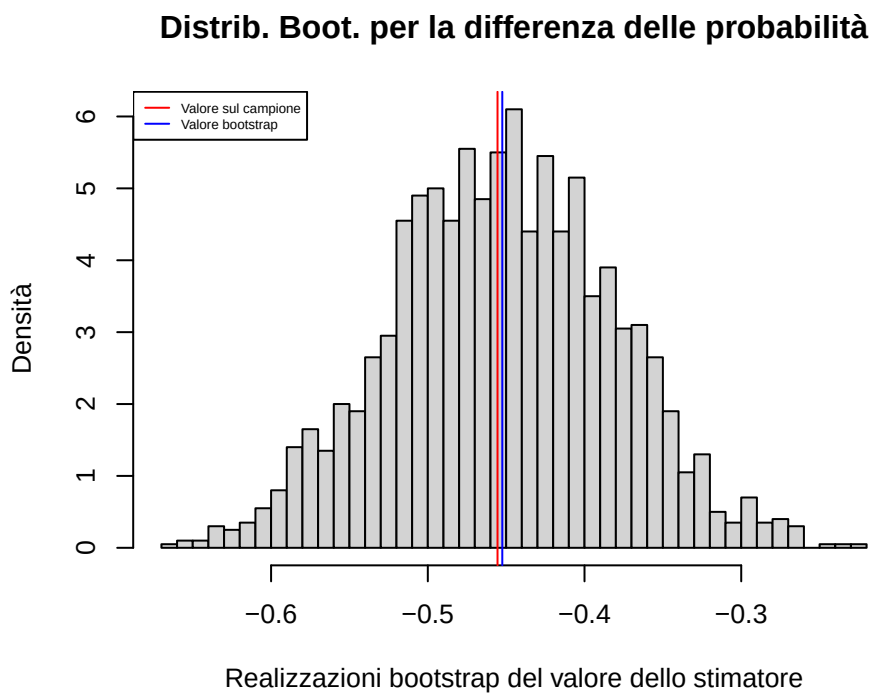
```
sd(boot_DD$thetastar)
```

```
#> [1] 0.0709495
```

I valori ottenuti per ciascuno dei 2000 campioni bootstrap presentano un campio di variazione interamente contenuto nell'insieme dei numeri reali minori di zero; il valore del massimo è pari a -0.23 (il minimo è invece -0.66), confermandil fatto che, per ogni replicazione bootstrap, le donne che non hanno eseguito il trattamento immunoterapico hanno una probabilità più elevata di sviluppare metastasi. In media la differenza tra le probabilità è pari a -0.45, valore coincidente sia con la mediana sia con la differenza ricavata sul dataset originale.

```
hist(boot_DD$thetastar,
     main = "Distrib. Boot. per la differenza delle probabilità",
     breaks = 60,
```

```
     freq = FALSE,
     ylab = "Densità",
     xlab = "Realizzazioni bootstrap del valore dello stimatore")
abline(v = c(DD, mean(boot_DD$thetastar)),
       col = c("red", "blue"))
legend("topleft",
       c("Valore sul campione", "Valore bootstrap"),
       col = c("red", "blue"),
       lty= c(1, 1),
       cex = 0.5)
```



La rappresentazione grafica della distribuzione bootstrap conferma le conclulsioni tratte finora; osserviamo inoltre che presenta una forma approssimativamente simmetrica attorno ad un picco centrale che coincide sia con la media dei valori bootstrap, sia con il risultato ottenuto sul campione originale.

29.7

Calcliamo infine gli intervalli di confidenza (al 95%) per il valore della differenza delle probabilità utilizzando sia il metodo del percentile sia il metodo BCA.

```
quantile(boot_DD$thetastar, c(0.025, 0.975))
```

```
#>       2.5%      97.5%
#> -0.5898717 -0.3172210
```

```
boot_DD_CI <- bootstrap::bcanon(x = 1:n,
                                nboot = 2000,
                                theta = theta2,
                                alpha = c(0.025, 0.975))
boot_DD_CI$confpoints
```

```
#>      alpha  bca point
#> [1,] 0.025 -0.5904762
#> [2,] 0.975 -0.3035191
```

L'intervallo di confidenza ottenuto con il metodo del percentile è $[-0.59, -0.32]$; include solo valori negativi, ad ulteriore conferma che vie è associazione tra l'insorgenza di metastasi e la (non) esecuzione del trattamento immunoterapico. Si interpreta considerando che nel 95% dei casi (95% dei campioni bootstrap considerati) il valore della differenza tra le probabilità cade all'interno dell'intervallo. Il metodo BCA consente di correggere per la distorsione. L'intervallo risultante è molto simile al precedente, ma presenta un'ampiezza leggermente più ampia. Osserviamo che i due parametri di quest'ultimo intervallo, accelerazione e correzione per la distorsione, sono entrambi diversi da 0.

```
boot_DD_CI$acc
```

```
#> [1] 0.01549495
```

```
boot_DD_CI$z0
```

```
#> [1] -0.001253314
```

SOLUZIONI ESERCIZIO 30

Si rimanda alle soluzioni dell'esercizio 29 e alle dispense delle applicazioni.

Soluzioni esercizi "Realizzazioni dalla distribuzione Normale Bivariata"

SOLUZIONI ESERCIZIO 31

.1

Per ottenere realizzazioni pseudo-casuali da una Variabile Casuale Normale Bivariata $(X_1, X_2) \sim N(\mu_1, \mu_2, \sigma_1^2, \sigma_2^2, \sigma_{12})$ si utilizza la funzione `rmvnorm` del pacchetto `mvtnorm`. Questo comando richiede come argomenti il numero di valori da generare (`n`), il vettore delle medie (`mean`) e la matrice di varianza covarianza (`sigma`).

```
require(mvtnorm)

medie <- c(0, 0)
varcov <- matrix(c(3, 0, 0, 3), ncol = 2, byrow = TRUE)

set.seed(14263)
x <- mvtnorm::rmvnorm(n = 3000, mean = medie, sigma = varcov)
```

.2

L'oggetto restituito in outpput dalla funzione `rmvnorm` è una matrice di due colonne e 3000 righe: le due colonne rappresentano le due componenti (X_1 e X_2) della variabile bivariata, mentre ciascuna riga corrisponde ad una diversa osservazione.

Calcoliamo i valori campionari delle medie e della matrice di varianza covarianza con i corrispondenti teorici.

```
apply(x, 2, mean)
```

```
#> [1] -0.03755349 -0.07563272
```

```
cov(x)
```

```
#>              [,1]         [,2]
#> [1,]  2.984626483 -0.004400598
#> [2,] -0.004400598  3.068788617
```

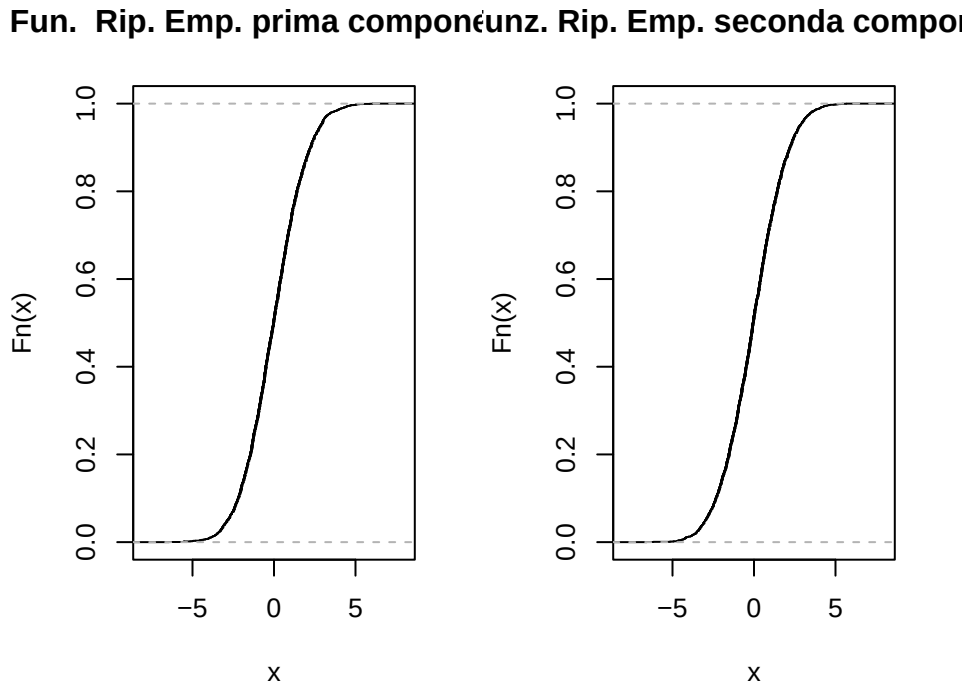
Si osserva, per tutti gli indici considerati, una buona aderenza dei valori empirici ottenuti a quelli della distribuzione teorica.

Notiamo inoltre che, in quetso caso, la covarianza tra le due componenti è pari a 0, ovvero le due componenti sono incorrelate. Ne consegue che sono anche indipendenti: la prima componente (prima colonna) ha distribuzione Normale Univariata a media 0 e varianza 3, ovvero $X_1 \sim N(0, 3)$, mentre la seconda componente (seconda colonna) ha a sua volta distribuzione Normale Univariata a media 0 e varianza 3, ovvero $X_2 \sim N(0, 3)$, indipendente dalla prima. E' importante sottolineare che questo risultato non vale in generale se le due componenti sono correlate, ovvero se $\sigma_{12} \neq 0$.

In conclusione, rappresentiamo le due funzioni di ripartizioni empiriche in una stessa finestra grafica.

```
par(mfrow=c(1, 2))
plot(ecdf(x[, 1]),
     do.points = FALSE,
     xlim = c(-8, 8),
     main = 'Fun.  Rip. Emp. prima componente')

plot(ecdf(x[, 2]),
```

```
     do.points = FALSE,
     xlim = c(-8, 8),
     main = 'Funz. Rip. Emp. seconda componente')
```

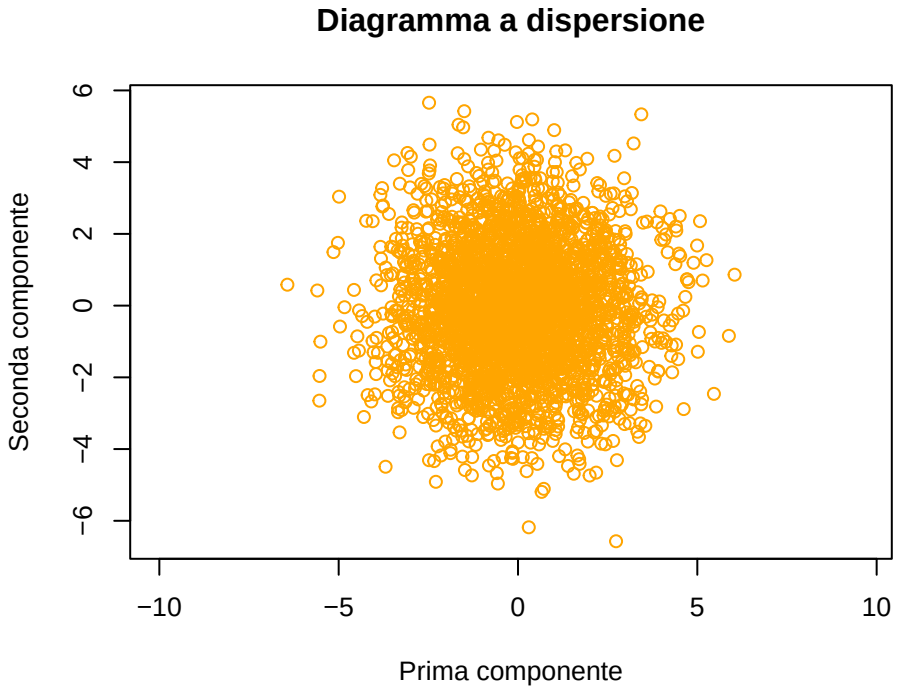


A conferma del fatto che le due componenti possono essere considerate come generate dalla stessa distribuzione Gaussiana univariata ($N(0, 3)$), le due curve sono sostanzialmente indistinguibili l'una dall'altra; si può inoltre osservare che seguono in modo praticamente perfetto la curva della funzione di ripartizione teorica per una V.C. corrispondente.

.3

Visualizziamo il grafico a dispersione semplicemente plottando nel piano i vettori relativi alle due componenti della distribuzione bivariata (asse x: prima componente, asse y: seconda componente).

```
plot(x[, 1], x[, 2],
     col = "orange",
     asp = 1,
     main = "Diagramma a dispersione",
     xlab = "Prima componente",
     ylab = "Seconda componente")
```



L'opzione `asp = 1` permette di utilizzare la stessa unità di misura per asse x ed asse y. Questo consente di visualizzare in modo preciso la forma della nuvola di punti, evidenziando in particolare modo le differenze tra le due componenti e le relazioni che intercorrono tra loro. In questo caso si osserva che la nuvola di punti assume una forma perfettamente circolare, conseguenza del fatto che le varianze σ_1^2 e σ_2^2 sono uguali e la covarianza σ_{12} è pari a 0. Il campo di variazione per le due componenti è approssimativamente il medesimo, variando in entrambi i casi da un minimo di circa -6 ad un massimo di circa 6. Infine osserviamo che i punti si distribuiscono maggiormente nella zona centrale, corrispondente al punto dato dalle medie delle due componenti, con una densità che man mano cala all'allontanarsi dal centro.

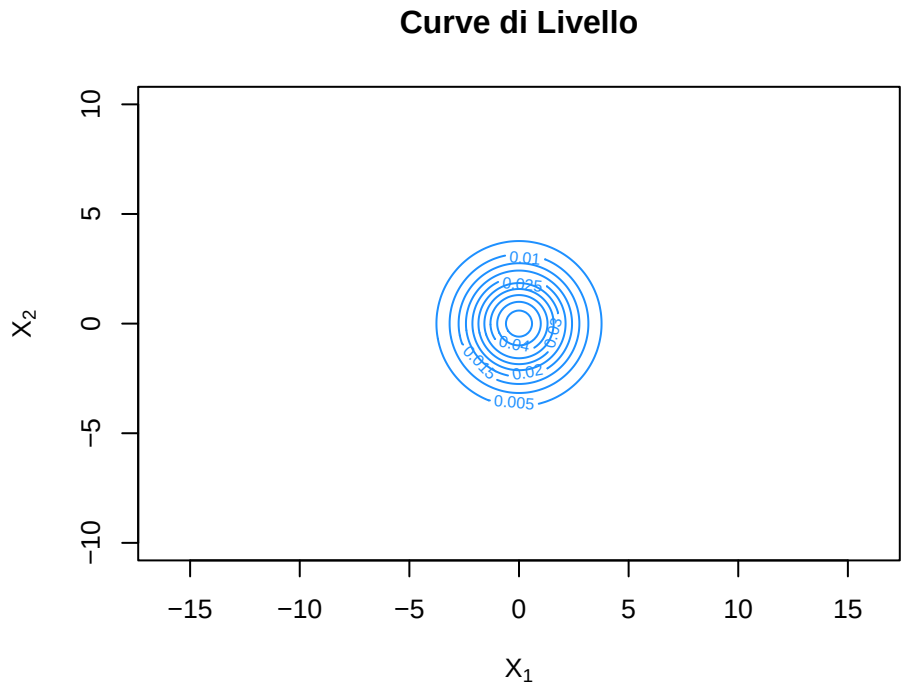
.4

La rappresentazione grafica della densità di una V.C. Normale Bivariata si ottiene attraverso una raffigurazione tridimensionale (spazio a tre dimensioni). Una rappresentazione in 2D può essere ricavata intersecando la superficie tridimensionale con dei piani posti a quote diverse e paralleli tra loro. Il risultato sono le cosiddette curve di livello. Oprativamente, in `R`, i passi da seguire sono i seguenti:

- si definisce una griglia (2D) di punti: in questo caso selezioniamo i vettori `z1` e `z2` come una successione di punti equispaziati da -10 a 10. Il comando `expand.grid` consente di ottenere la griglia costruendo tutte le possibili coppie di punti `(z1[i], z2[j])`;
- si calcola il valore della densità in ognuno dei punti della griglia: la funzione `dmvnorm` (in modo analogo alla funzione `dnorm` nel caso univariato) richiede in input la griglia di punti, il vettore delle medie e la matrice di varianza-covarianza e restituisce la densità nei punti della griglia in forma di vettore. E' successivamente necessario trasformarlo in un oggetto matriciale;

- infine si utilizza il comando `contour` per plottare i valori della densità appena calcolati.

```
z1 <- z2 <- seq(-10, 10, by = 0.1)
griglia <- expand.grid(z1, z2)

dens <- matrix(dmvnorm(griglia,
                       mean = medie,
                       sigma = varcov),
                       ncol = length(z1))

contour(z1, z2, dens,
        asp = 1,
        # nlevels = 3,
        # levels = c(0.01, 0.02, 0.04, 0.07),
        main = "Curve di Livello", xlab = expression(X[1]), ylab = expression(X[2]),
        col = "dodgerblue")
```



Di default la funzione contour stabilisce automaticamente le curve di livello da rappresentare; è comunque possibile specificare il numero di curve che si vuole ottenere o anche i valori specifici delle quote. Come già osservato nel grafico a dispersione, notiamo che le curve di livello hanno forma perfettamente circolare (lunghezza degli assi uguale e di conseguenza nessuna orientazione specifica). Il centro di ciascuna circoferenza si trova in corrispondenza del punto (0,0), in corrispondenza delle medie delle due componenti.

SOLUZIONI ESERCIZIO 32

Nel seguito si rimanda alle soluzioni dell'Esercizio 31. Si riporta solo il codice e i commenti significativamente diversi rispetto all'esercizio precedente.

.1

Realizzazioni dalla distribuzione normale bivariata con covarianza positiva e stessa variabilità elevata delle due componenti.

```
require(mvtnorm)

medie <- c(2, 2)
varcov <- matrix(c(100, 4, 4, 100), ncol = 2)

set.seed(14263)
x <- rmvnorm(n = 3000, mean = medie, sigma = varcov)
```

.2

Analisi statistiche descrittive (in questo caso la covarianza è diversa da 0, quindi le due componenti non possono essere considerate indipendenti).

```
mean(x)
```

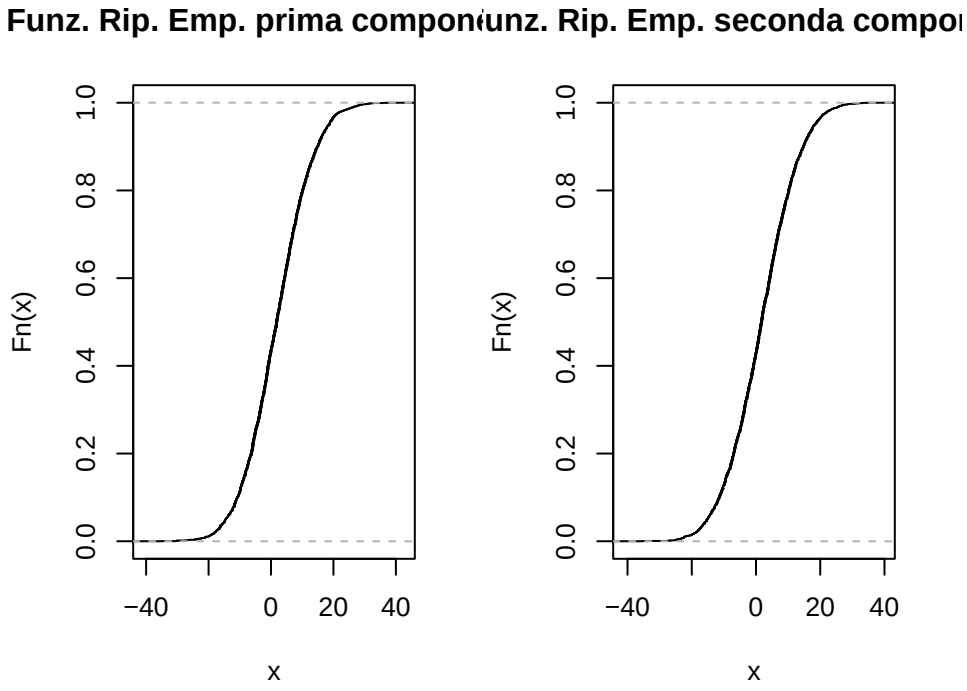
```
#> [1] 1.666789
```

```
cov(x)
```

```
#>           [,1]       [,2]
#> [1,] 99.482805   3.888923
#> [2,]  3.888923 102.285964
```

Grafico delle funzioni di ripartizione empirica

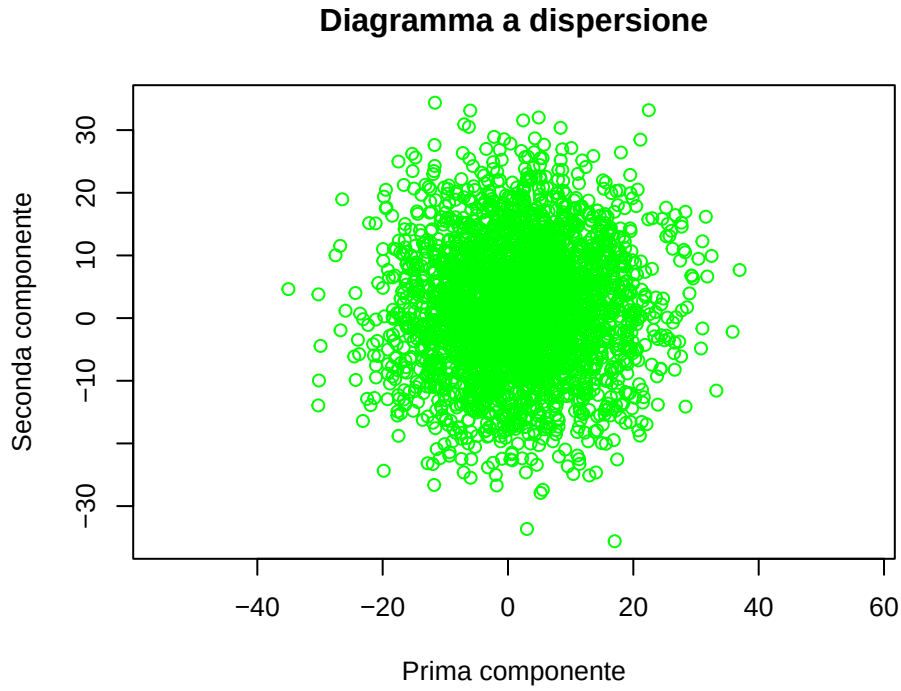
```
par(mfrow = c(1, 2))
plot(ecdf(x[, 1]),
     do.points = FALSE,
     main = 'Funz. Rip. Emp. prima componente')

plot(ecdf(x[, 2]),
     do.points = FALSE,
     main = 'Funz. Rip. Emp. seconda componente')
```



.3

Grafico a dispersione delle due componenti.

```
plot(x[, 1], x[, 2],
     col = "green",
     asp = 1,
     main = "Diagramma a dispersione",
     xlab = "Prima componente",
     ylab = "Seconda componente")
```

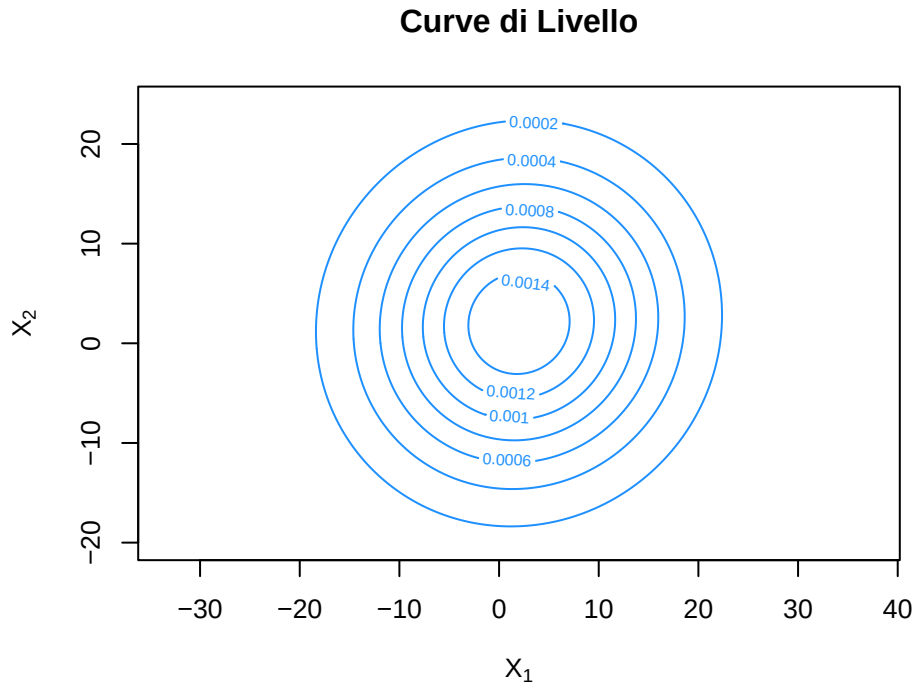


Nonostante dal grafico sia impossibile rilevarlo con precisione, è noto che la nuvola di punti è in questo caso centrata in (2, 2), corrispondente alle medie delle due componenti. E' anche difficile rilevare l'orietazione della nuvola di punti, che sappiamo essere verso l'alto (l'asse maggiore dell'ellisse si dispone lungo una retta con coefficiente angolare positivo). La causa è il valore delle varianze che è sensibilmente superiore a quello della covarianza. Si osserva infatti che il campo di variazione delle due componenti è molto più elevata rispetto a quanto notato nell'esercizio precedente. Ribadiamo infine che, nonostante la forma della nuvola di punti sembra essere una circonferenza, è in questo caso un ellisse, visto il valore della covarianza diverso da 0.

##.4 La procedura è identica a quella introdotta nell'Esercizio 31; osserviamo solo che i vettori `z1` e `z2` devono essere definiti in modo diverso: in primo luogo, nell'esercizio precedente le medie erano pari a 0, quindi i due vettori erano simmetrici rispettoi a 0 (andavano da -5 a 5); ora le medie sono uguali a 2, quindi `z1` e `z2` dovranno essere simmetrici rispetto a 2. In secondo luogo, la variabilità era molto limitata nell'esercizio precedente (varianze pari a 3); in questo caso abbiamo invece una variabilità molto maggiore (varianze pari a 100): è quindi necessario che gli intervalli rappresentati da `z1` e `z2` siano molto più ampi.

```
z1 <- z2 <- seq(-20, 24, by = 0.1)
griglia <- expand.grid(z1, z2)

dens <- matrix(dmvnorm(griglia,
                       mean = medie,
                       sigma = varcov),
                       ncol = length(z1))

contour(z1, z2, dens,
```

```
        asp = 1,
        # nlevels = 3,
        # levels = c(0.01, 0.02, 0.04, 0.07),
        main = "Curve di Livello", xlab = expression(X[1]), ylab = expression(X[2]),
        col = "dodgerblue")
```



L'interpretazione ed i commenti relativi al grafico ricalcano per lo più quelli dell'esercizio precedente; ci limitiamo ad osservare che i valori della quota dei piani paralleli sono significativamente inferiori rispetto a quelli mostrati nell'esercizio precedente. Questo testimonia una forma tridimensionale della distribuzione motlo più bassa, schiacciata ed estesa nel piano.

Soluzioni esercizi "Applicazione dell'algoritmo EM"

SOLUZIONI ESERCIZIO 33

.1

Si considera una tabella di contingenza costituita da 2 righe e 3 colonne e contenente un valore mancante (`NA`). Si utilizza l'algoritmo di Expectation-Maximization (EM) per stimare il dato mancante sulla base di un modello lineare $y_{ij} = \mu + \alpha_i + \beta_j + e_{ij}$, dove μ è l'intercetta, α_i è il parametro di riga, β_j è il parametro di colonna e $e_{ij} \sim N(0, \sigma^2)$ è un termine di errore.

L'algoritmo EM è costituito da due passi (si vedano le dispense di applicazioni per maggiori dettagli):

- passo **E** (expectation): assegnando un valore iniziale al valore mancante, si calcolano i valori dei parametri del modello;
- passo **M** (maximization): si utilizzano i parametri del modello per stimare ed aggiornare il valore del dato mancante. La funzione `em1` definita di seguito implementa i due passi descritti; restituisce in output il valore di tutte le quantità stimate.

```
em1 <- function(y21, y){
  ystar <- y
  ystar[2, 1] <- y21

  # Expectation Step
  mu.hat <- mean(ystar)
  alpha.hat <- apply(ystar, MAR = 1, mean) - mean(ystar)
  beta.hat  <- apply(ystar, MAR = 2, mean) - mean(ystar)

  # Maximization Step
  y21 <- mu.hat + alpha.hat[2] + beta.hat[1]

  return(c(mu = mu.hat,
           alpha = alpha.hat,
           beta = beta.hat,
           y21 = y21))
}

y <- matrix(c(16, 7, 7, NA, 64, 5), nrow = 2, ncol = 3, byrow = TRUE); y
```

```
#>      [,1] [,2] [,3]
#> [1,]   16    7    7
#> [2,]   NA   64    5
```

```
em1(19.8, y)
```

```
#>     mu alpha1 alpha2  beta1  beta2  beta3    y21
#>   19.8   -9.8    9.8   -1.9   15.7  -13.8   27.7
```

Eseguendo la funzione fissando come valore iniziale per il dato mancante il valore di 19.8 (pari alla media dei dati presenti), si ottengono i valori mostrati di sopra. In particolare il valore della frequenza mancante viene stimato pari a 27.7, aggiornando così il valore assegnato inizialmente di 19.8.

L'algoritmo EM prevede di iterare la procedura iterando questi due passi fino a convergenza. Si itera quindi l'esecuzione della funzione `em1` attraverso un ciclo `while` fino a quando la stima dei parametri rimane "stabile", ovvero fino a quando la differenza tra il valore dei parametri a due iterazioni *successive* dell'algoritmo è inferiore ad una certa soglia ε (10^{-8} nell'esempio di seguito).

```r
em.step <- function(y, epsilon = 1e-8) {
  trace <- NULL
  convergenza <- FALSE
  trace <- t(em1(y21 = mean(y, na.rm = TRUE), y = y))
  y21id <- grep("y21", colnames(trace))
  i <- 0
  while(!convergenza) {
    i <- i + 1
    trace <- rbind(trace, em1(y21 = trace[i, "y21"], y = y))
    convergenza <- (dist(trace[i:(i+1), -y21id]) < epsilon)
  }
  return(trace)
}

set.seed(183)
est <- em.step(y)
```

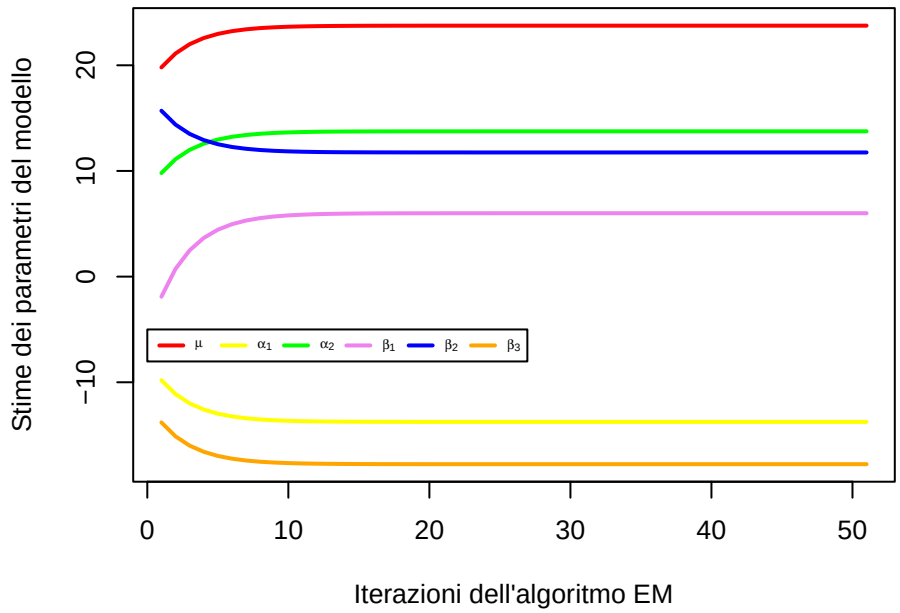
L'output generato è una matrice di 51 righe e 7 colonne. Ogni colonna rappresenta un diverso parametro stimato dal modello. L'ultima colonna in particolare è il valore ottenuto per il dato mancante. Ciascuna riga contiene invece le stime ottenute ad un'iterazione dell'algoritmo. Per giungere a convergenza l'algoritmo EM richiede quindi 51 iterazioni. In particolare nell'ultima riga troviamo i valori dei parametri a convergenza (quindi "definitivi"). Il valore del dato mancante stimato attraverso l'algoritmo EM è quindi pari a 43.5 (deve essere approssimato, trattandosi di una frequenza).

.2

Dal momento che la lettura della matrice in output non è agevole, possiamo rappresentare i valori ottenuti attraverso un grafico (trace plot). Si utilizza la funzione `matplot` che consente di rappresentare ciascuna delle serie di dati contenute nella matrice (escludiamo il valore stimato per la frequenza mancante).

```r
names1 <- expression(mu, alpha[1], alpha[2], beta[1], beta[2], beta[3])
pal1 <- c("red", "yellow", "green", "violet", "blue", "orange")
matplot(est[, -7],
        type = "l",
        col = pal1,
        lwd = 2,
        lty = 1,
        xlab = "Iterazioni dell'algoritmo EM",
        ylab = "Stime dei parametri del modello")

legend(x = 0,
       y = -5,
       legend = names1,
       lwd = 2 ,
```

```
       col = pal1,
       lty = 1,
       horiz = TRUE,
       cex = 0.5)
```



Osserviamo che i parametri μ, α_2 e β_3 hanno un andamento cresecnte all'aumentare delle iterazioni dell'algoritmo, al contrario dei restanti parametri. Si nota inoltre che l'algoritmo EM, aggiornando i valori delle stime da un'iterazione e la seguente, comporta un aggiustamento sensibile durante i primi passi (prima del decimo), mentre gli aggiustamenti sono molto più limitati per i passi successivi.

.3

Di seguito si modifica la funzione `em.step` in modo tale che il criterio di stop dell'algoritmo EM, non sia la convergenza dei parametri stimati, ma il raggiungimento di un numero massimo di iterazioni (NB. Nella pratica le due condizioni vengono spesso affiancate: si richiede che la convergenza venga raggiunta, ma se dopo un numero massimo di iterazioni, l'algoritmo EM sta ancora "girando", allora lo si blocca ugualmente. Tale numero massimo di iterazioni è solitamente piuttosto elevato. E' assolutamente sconsigliato utilizzare come condizione di stop dell'algoritmo solo l'aver raggiunto un massimo numero i di iterazioni.). Non serve alcuna modifica alla funzione `em1`. Nella funzione `em.step` si modifica solo la definizione dell'oggetto `convergenza` (rinominata `maxit_reached`).

```
em.step <- function(y, maxit = 20) {
  trace <- NULL
  maxit_reached <- FALSE
  trace <- t(em1(y21 = mean(y, na.rm = TRUE), y = y))
  y21id <- grep("y21", colnames(trace))
  i <- 0
  while(!maxit_reached) {
    i <- i + 1
    trace <- rbind(trace, em1(y21 = trace[i, "y21"], y = y))
    maxit_reached <- (i >= (maxit-1))
  }
  return(trace)
}

est_mod <- em.step(y, maxit = 20)
```

Si noti che in questo caso abbiamo interrotto l'esecuzione dell'algoritmo prima della convergenza, pertanto non vi è garanzia che le stime dei parametri siano effettivamente attendibili (in questo caso non si riscontrano differenze significative, ma le stime ottenute con il secondo metodo risultano comunque meno precise).

SOLUZIONI ESERCIZIO 34

Si rimanda alle soluzioni dell'esercizio precedente.

SOLUZIONI ESERCIZIO 35

Si rimanda alle soluzioni dell'esercizio precedente.

Soluzioni esercizi "Densità miscuglio"

SOLUZIONI ESERCIZIO 36

.1

Consideriamo la funzione `funcmxn` implementata a lezione per rappresentare graficamente la funzione di densità di un miscuglio a due componenti Gaussiane.

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

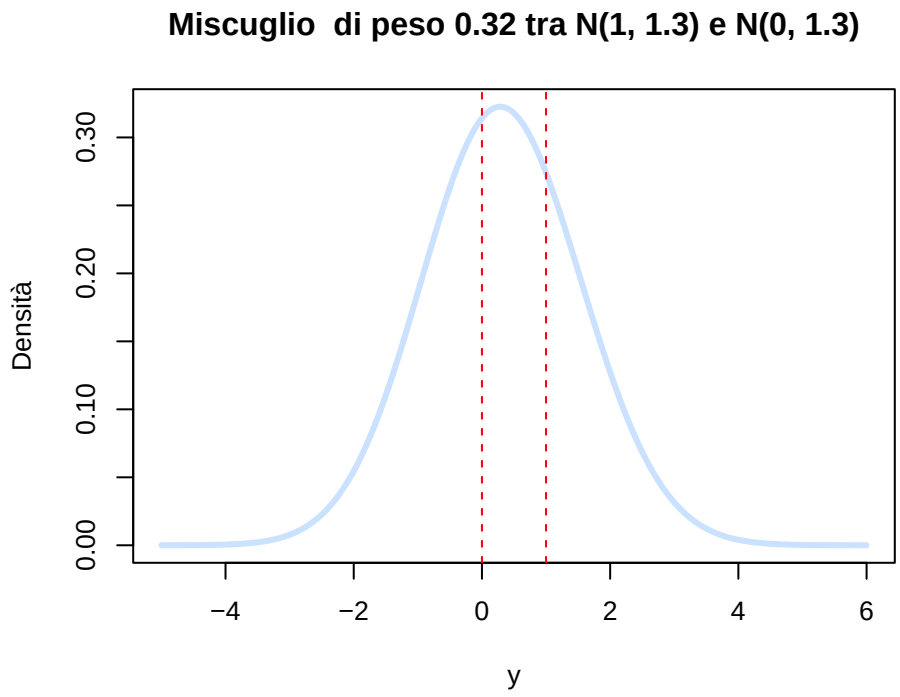
Definiamo i vettori per le medie e le deviazioni standard e il valore del peso della prima componente; la funzione `funcmxn` determina il valore della densità nel punto (`x`) che le viene passato in input.

```
mu <- c(1, 0)
sd <- c(sqrt(1.3), sqrt(1.3))
p <- 0.32

funcmxn(0.5, p, mu, sd)
```

```
#> [1] 0.3178186
```

Per esempio, nel punto $x = 0.5$ la funzione di densità del miscuglio Gaussiano considerato ha valore pari a 0.32. Una rappresentazione grafica dei valori della desnità consenste di valutare meglio il miscuglio considerato. Per ottenere il grafico è necessario determinare la densità in un intervallo di punti. Consideriamo quindi una successione di numeri equispaziati da -5 a 10 (in generale gli estremi devono essere valutati caso per caso ed eventualmente aggiustati dopo una visione preliminare del grafico). La successione di punti viene poi passata in argomento alla funzione `funcmxn`. Aggiungiamo le righe tratteggiate in corrispondenza delle medie delle due componenti.

```
y <- seq(-5, 6, by = 0.01)
pr <- funcmxn(x = y, p = p, mu = mu, sd = sd)

plot(y, pr,
     xlab = "y", ylab = "Densità",
     lwd = 3,
     col = "lightsteelblue1",
     type = "l",
     main = "Miscuglio  di peso 0.32 tra N(1, 1.3) e N(0, 1.3)")

abline(v = c(0, 1), lty = c(2, 2), col = c("red", "red"))
```



In questo caso il grafico della densià non permettere di distinguere le due componenti; si osserva infatti un unico massimo, con un picco centrale apparentemente simmetrico rispetto ad un punto compreso tra 0 e 1 (le medie delle due componenti). I valori delle medie delle due componenti risultano infatti toppo vicini tra loro (in relazione ad una varianza superiore ad 1), per consintire di distinguere i due picchi. Sarebbe necessario aumentare la distanza tra le due medie o diminuire il valore della vaianza.

.2

Definiamo la funzione `rnormxn` che genera delle pseudo realizzazioni da una funzione miscuglio con due componenti Gaussiane. Analogamente alla funzione precedente, richiede in input i valori del peso della prima componente (`w`) e i vettori delle medie e delle varianze delle due componenti. Ogni osservazione viene estratta con probabilità pari a `w` dalla prima componente e con probabilità pari a `1-w` dalla seconda componente.

```
rnormxn <- function(n, w, mu, sd){
  x = numeric(n)
  for(i in 1:n){
    dm = runif(1, 0, 1)
    if(dm < w)
      x[i] = rnorm(1, mu[1], sd[1])
    else
      x[i] = rnorm(1, mu[2], sd[2])
    }
```

```
  x
}

val <- rnormxn(1000, w = p, mu = mu, sd = sd)

summary(val)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#> -3.6071 -0.4704  0.2855  0.3358  1.1425  4.1535
```

```
var(val)
```

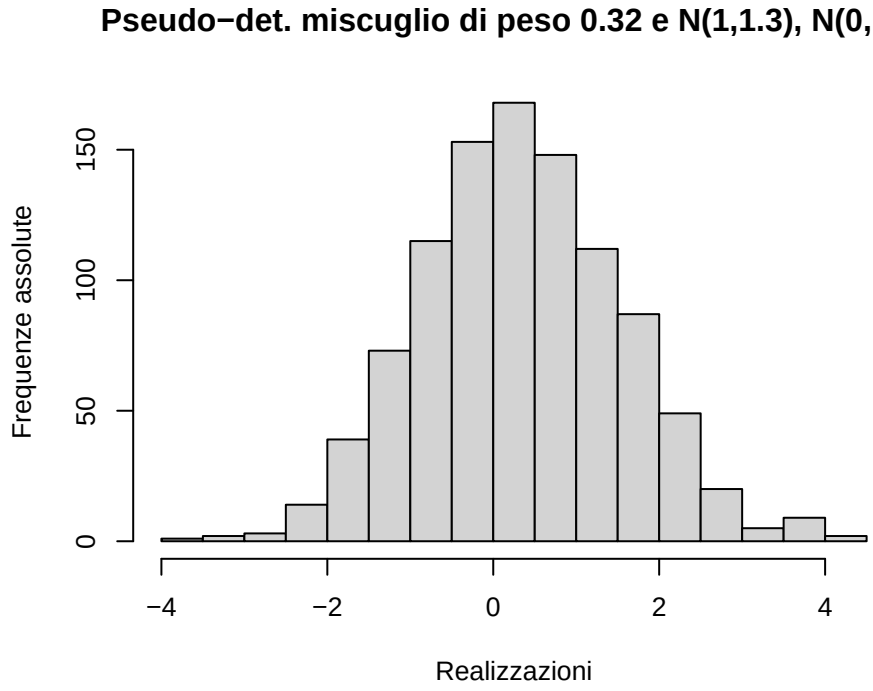
```
#> [1] 1.441945
```

Calcoliamo le statistiche descrittive delle 1000 pseudo-realizzazioni ottenute dalla funzione miscuglio. Il campo di variazione dei dati osservati va da un minimo di -3.3 ad un massimo di 4.3. Media e mediana sono approssimativamente coincidenti: metà delle osservazioni assumono un valore inferiore a 0.35. In geenrale sembra essere confermata la simmetria già osservata dal grafico della densità del miscuglio. Il valore della varianza è 1.44, di poco superiore al valore (comune) delle due componenti Gaussiane del miscuglio.

.3

L'istogramma delle realizzazioni ottenute al punto precedente conferma sostanzialmente le osservazione del punto precedente: il campo di variazione è piuttosto limitato, il centro (unico) della distribuzione è di poco superiore a 0, e la forma dell'istogramma è approssimativamente simmetrica. Non si nota la presenza di un doppio massimo corrispondente alle due componenti del miscuglio.

```
hist(val,
     breaks = 25,
     ylab = "Frequenze assolute",
     xlab = "Realizzazioni",
     main = "Pseudo-det. miscuglio di peso 0.32 e N(1,1.3), N(0,1.3)")
```



SOLUZIONI ESERCIZIO 37

Consideriamo ora una densità miscuglio con due componenti Gaussiane $X_1 \sim N(0,1)$ e $X_2 \sim N(1.5, 0.1)$. Il peso della prima componente è 0.75. Osserviamo che le medie delle due componenti non si discostano in modo significativo, mentre la prima componente (che presenta un peso maggiore) ha una variabilità molto superiore rispetto alla seconda. La prima componente sarebbe quindi molto più dispersa (e quindi più "bassa"), ma presente un peso superiore.

.1

Utilizziamo ancora la funzione `funcmxn` per determinare i valori della densità del miscuglio (si rimanda all'esercizio precedente o alle dispense per i dettagli).

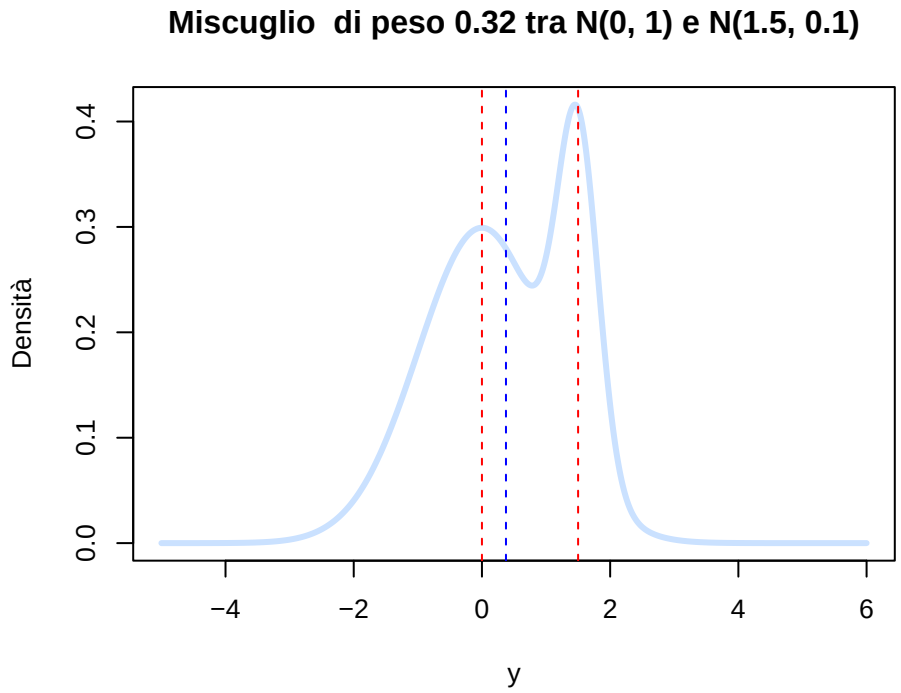
```
mu <- c(0, 1.5)
sd <- c(sqrt(1), sqrt(0.1))
p <- 0.75

y <- seq(-5, 6, by = 0.01)
pr <- funcmxn(y, p, mu, sd)

summary(pr)
```

```
#>      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
#> 0.0000000 0.0000592 0.0068202 0.0908265 0.2043185 0.4160466
```

.2

Rappresentiamo graficamente i valori della densità determinati al putno precedente.

```
plot(y, pr,
     xlab = "y", ylab = "Densità",
     lwd = 3,
     col = "lightsteelblue1",
     type = "l",
     main = "Miscuglio  di peso 0.32 tra N(0, 1) e N(1.5, 0.1)")

media_miscuglio <- sum(c(0.75, 0.25) * c(0, 1.5))
abline(v = c(mu, media_miscuglio), lty = c(2, 2, 2), col = c("red", "red", "blue"))
```



In questo caso la funzioen densità del miscuglio presenta due picchi (due massimi), rendendo le due componenti chiaramente distinguibili. Come anticipato, la prima componente presenta un valore della varianza molto più alto rispetto alla seconda, presentando quindi un massimo molto più basso, ma beneficia di un valore del peso più elevato. La seconda componente ha una variabilità molto bassa attorno alla propria media. Osserviamo che il valore medio del miscuglio, pari a circa 0.09, si posiziona molto vicino a quello della prima componente.

Soluzioni esercizi “Modelli miscuglio univariati e multivariati”

SOLUZIONE ESERCIZIO 38

.1

Si descrivono i dati analizzandone le principali statistiche descrittive.

```
load("emoglobina.Rdata")
require(skimr)
skimr::skim_without_charts(emoglobina)
```

Table 41: Data summary

Name	emoglobina
Number of rows	70
Number of columns	1
_____	
Column type frequency:	
numeric	1
_____	
Group variables	None

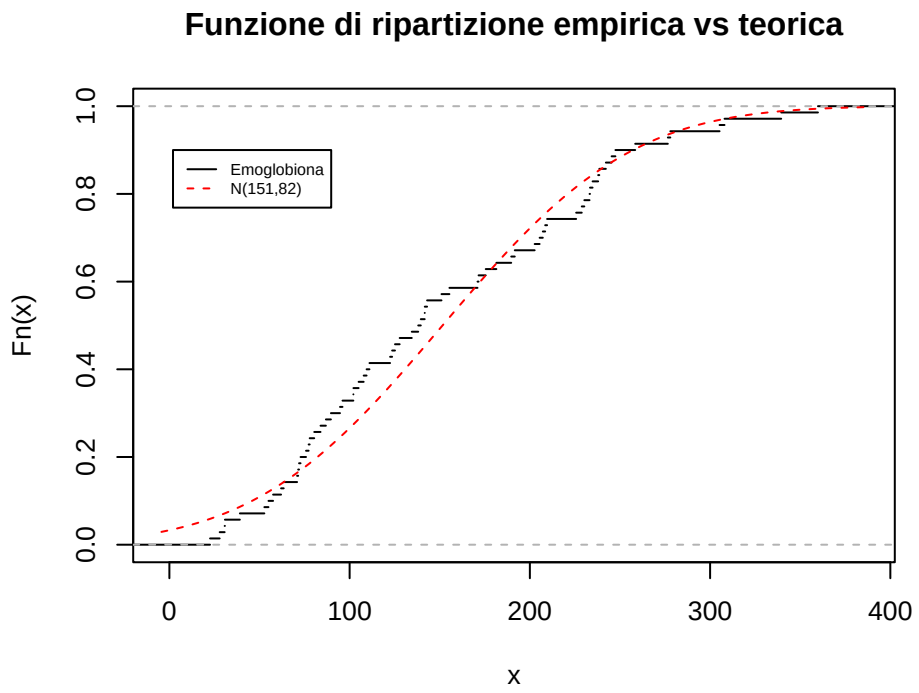
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	151.55	82.48	22.55	81.24	138.77	221.66	359.84

Il dataset contiene 70 righe (70 osservazioni) ed una sola colonna, la variabile che rileva il componente del plasma osservato. Mediamente le osservazioni presentano un valore dell'emoglobina pari a 152. Il campo di variazione risulta piuttosto ampio, con un minimo approssimativamente pari a 23 ed un massimo di circa 360. La variabilità si conferma piuttosto elevata osservando il valore della deviazione standard: in media le osservazione hanno uno scostamento medio dalla media pari a 82.5. Calcolando approssimativamente il valore del coefficiente di variazione questo si traduce in una variabilità pari a circa il 50% della media. La mediana è leggermente distante rispetto alla media, indizio di una possibile asimmetria della distribuzione.

Forniamo inoltre una rappresentazione grafica dei dati tramite la funzione di ripartizione empirica. Confrontiamo questa curva rispetto al modello teorico della distribuzione normale con media e deviazione standard uguali a quelle campionarie. E' chiaro che questa risulterà probabilmente un'approssimazione imprecisa, dal momento che i dati provengono da un miscuglio Gaussiano e non da una variabile di Gauss.

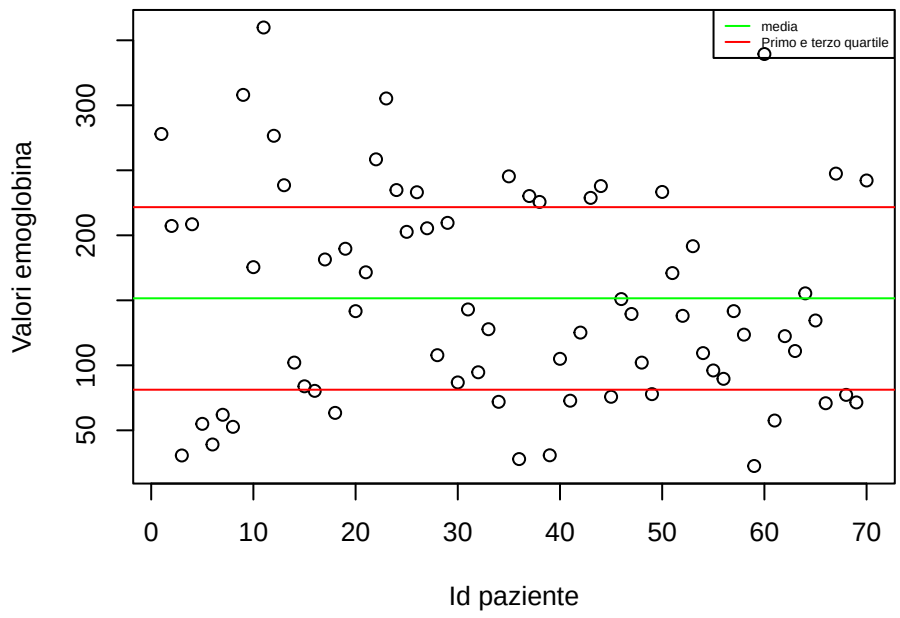
```r
plot(ecdf(emoglobina),
     do.points = FALSE,
     main = "Funzione di ripartizione empirica vs teorica",
     col = "black")

curve(pnorm(x, mean = mean(emoglobina), sd = sd(emoglobina)),
      add = TRUE,
      lty = 2,
      col = "red")

legend(2, 0.9,
       col = c("black", "red"),
       c("Emoglobiona", "N(151,82)"),
       lty = c(1, 2),
       cex = 0.6)
```



Osserviamo che le due curve, empirica e teorica, presentano degli sconstamenti abbastanza significativi; in particolare evidenziamo che la distribuzione dei dati osservati ha una coda sinistra più leggera (meno osservazioni) dell'atteso, mentre la coda destra risltra coerente con l'ipotesi di normalità. Infine si osserva che la regione centrale della distribuzione empirica presenta più osservazioni rispetto all'atteso (sempre secondo l'ipotesi di normalità).

Infine si fornisce la rappresentazione grafica dei valori nel piano cartesiano (grafico a dispersione rispetto all'id dell'unità), confrontando con valore medio e primo e terzo quartile della distribuzione.

```r
plot(emoglobina,
     xlab = "Id paziente",
     ylab = "Valori emoglobina")

q <- quantile(emoglobina, c(0.25,0.75))
m <- mean(emoglobina)

abline(h = c(m, q[1], q[2]),
       col = c("green", "red", "red"))

legend("topright",
       c("media", "Primo e terzo quartile"),
       col = c("green", "red"),
       lty = c(1, 1), cex = 0.5)
```



I punti si distribuiscono nel piano in modo casuale (non si evidenzia la presenza di sistematicità o strutture particolari). Evidenziamo una leggera asimmetria della loro disposizione rispetto al valore medio: nella parte bassa del grafico si evidenzia una dispersione dei punti inferiore alla zona superiore: i punti si distanziano meno dalla linea del terzo quartile.

.2

Si utilizza la funzione `mclust::mclustBIC` per stimare dei modelli miscuglio per un numero di componenti crescente (da 1 a 9). Di default la funzione considera sia i modelli che suppongono

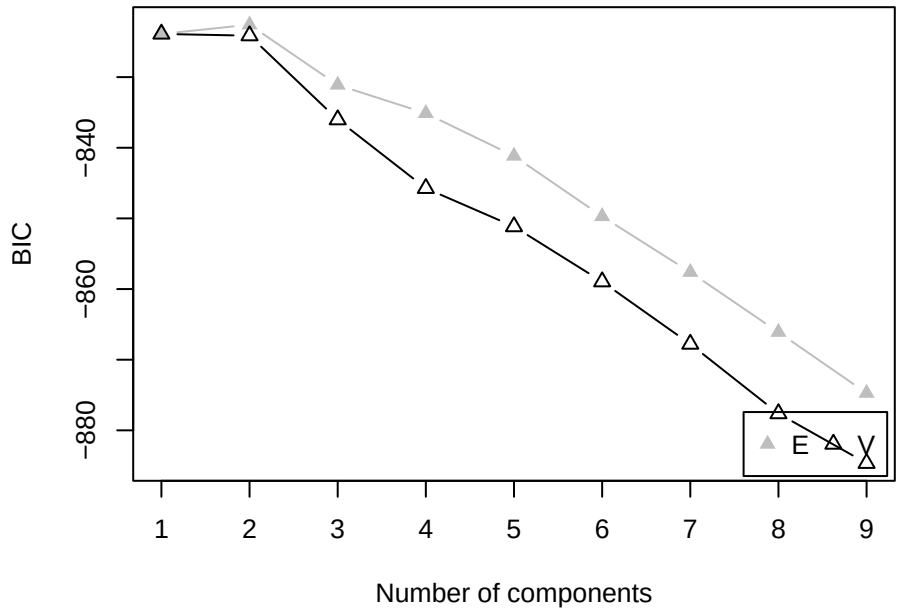
uguale varianza tra le diverse componenti (modello più parsimonioso, specificato con la lettera "E") sia i modelli con varianza specifica per ciascuna componente (specificato con la lettera "V"). Tutti i modelli stimati sono confrontati con il criterio BIC al fine di selezionare quelli che hanno un adattamento migliore.

```
library(mclust)
mcc <- mclust::mclustBIC(emoglobina); mcc
```

```
#> Bayesian Information Criterion (BIC):
#>           E         V
#> 1 -823.8906 -823.8906
#> 2 -822.5878 -824.1333
#> 3 -831.1105 -836.0230
#> 4 -835.1245 -845.7390
#> 5 -841.1645 -851.1584
#> 6 -849.7154 -858.9562
#> 7 -857.6157 -867.7858
#> 8 -866.1007 -877.6227
#> 9 -874.7309 -884.6570
#>
#> Top 3 models based on the BIC criterion:
#>       E,2       E,1       V,1
#> -822.5878 -823.8906 -823.8906
```

L'output della funzione utilizzata mostra una matrice con i valori dell'indice BIC per relativo a tutti i modelli stimati; le righe corrispondono al numero delle componenti utilizzate, le colonne alle due specificazioni per la varianza. In aggiunta l'output presenta la lista dei tre modelli migliori dal punto di vista del BIC; in questo caso la scelta ottimale è selezionare 2 gruppi ed assumere che la varianza sia comune tra tutte le componenti del modello. Si osserva che il secondo e il terzo migliore modello assumono una sola componente; in quetso caso il modello presuppone che non ci sia eterogeneità tra le osservazioni.

```
plot(mcc)
```



E' anche possibile rappresentare graficamente i risultati ottenuti per il BIC dei diversi modelli, semplicemente utilizzando la funzione `plot`. Il grafico mostra le due serie di valori (una per i modelli con varianza specifica per ciascuna componente, uno per i modelli con varianza comune) ed il relativo andamento rispetto al numero di componenti. Il valore migliore per la definizione dell'indice BIC implementata nel pacchetto `mclust` è quello maggiore; il risultato conferma ovviamente quello già osservato: il modello ottimale utilizzando l'indice BIC per la selezione è quello con due componenti e varianza comune.

.3

Stimiamo quindi il modello selezionato e ne commentiamo i parametri. La funzione utilizzata è `mclust::Mclust`, che richiede in input i dati, il numero di componenti e la specificazione per la varianza del modello.

```
mod1 <- Mclust(emoglobina,
               G = 2,
               modelNames = "E")
summary(mod1)
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust E (univariate, equal variance) model with 2 components:
```

```
#>
#>  log-likelihood  n df       BIC       ICL
#>       -402.7969 70  4 -822.5878 -832.2396
#>
#> Clustering table:
#>  1  2
#> 43 27
```

Il summary del modello mostra solo informazioni relative al valore della log-verosimiglianza, indici BIC ed ICL, numero di osservazioni e gradi di libertà, oltre al numero di osservazioni classificate in ciascuno dei due gruppi. Osserviamo che il primo gruppo risulta più numeroso del secondo, con 43 soggetti contro 27. Per ottenere maggiori informazioni (sui parametri del modello), è necessario specificare l'opzione `parameters=TRUE`.

```
summary(mod1, parameters=TRUE)
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust E (univariate, equal variance) model with 2 components:
#>
#>  log-likelihood  n df       BIC       ICL
#>       -402.7969 70  4 -822.5878 -832.2396
#>
#> Clustering table:
#>  1  2
#> 43 27
#>
#> Mixing probabilities:
#>         1         2
#> 0.6132646 0.3867354
#>
#> Means:
#>         1         2
#>  97.43502 237.36860
#>
#> Variances:
#>        1        2
#> 2060.818 2060.818
```

I parametri del modello sono (i) i pesi delle due diversi componenti, (ii) le medie e (iii) le varianze relativi alle due componenti del miscuglio.

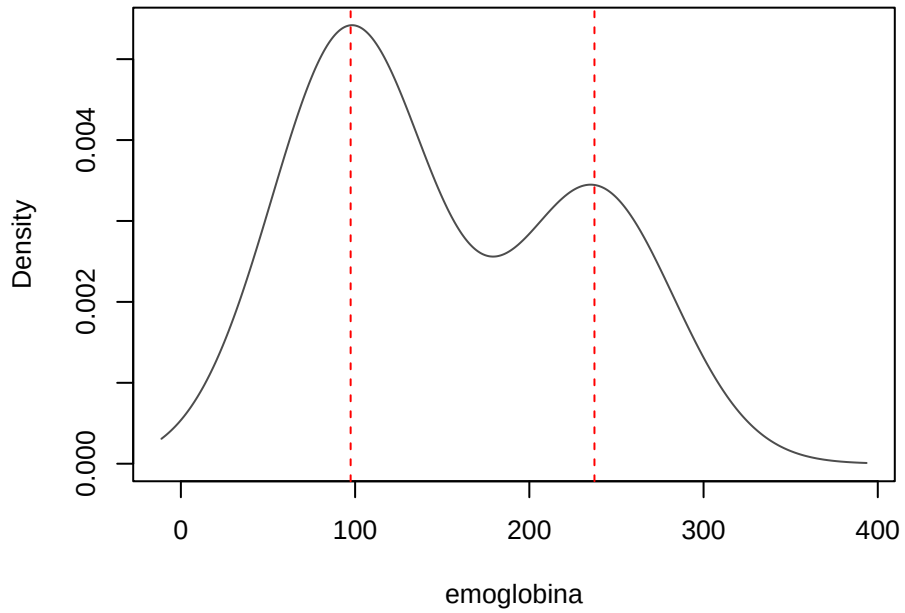
- Le due componenti hanno peso pari a 0.61 e 0.39 rispettivamente; trattandosi di un modello che assume varianza comune per entrambe le convergenze, la componente con peso maggiore è quella con il maggior numero di osservazioni allocate. Si noti che questo non è necessariamente vero nel caso di un modello in cui le varianze sono specifiche per ciascuna componente.

- Il valore medio per la componente del plasma per i soggetti nel primo gruppo è pari a 97, molto inferiore rispetto al valore corrispondente per le osservazioni allocate nel secondo gruppo (pari a 237). Possiamo quindi caratterizzare le due sottopopolazioni in base al valore della componente del plasma: valori mediamente alti per la seconda sottopopolazione, mediamente bassi per la prima.
- I valori delle varianze per le due componenti sono uguali, avendo selezionato il modello più parsimonioso. La deviazione standard corrispondente vale circa 45; questo valore rappresenta la variabilità media attorno al valor medio specifico per ciascuna componente.

.4

Rappresentiamo il grafico della densità del miscuglio specificando l'opzione `what='density'` nella funzione `plot`.

```
plot(mod1, what = 'density')
abline(v = mod1$parameters$mean, col = rep("red", 2), lty = c(2, 2))
```



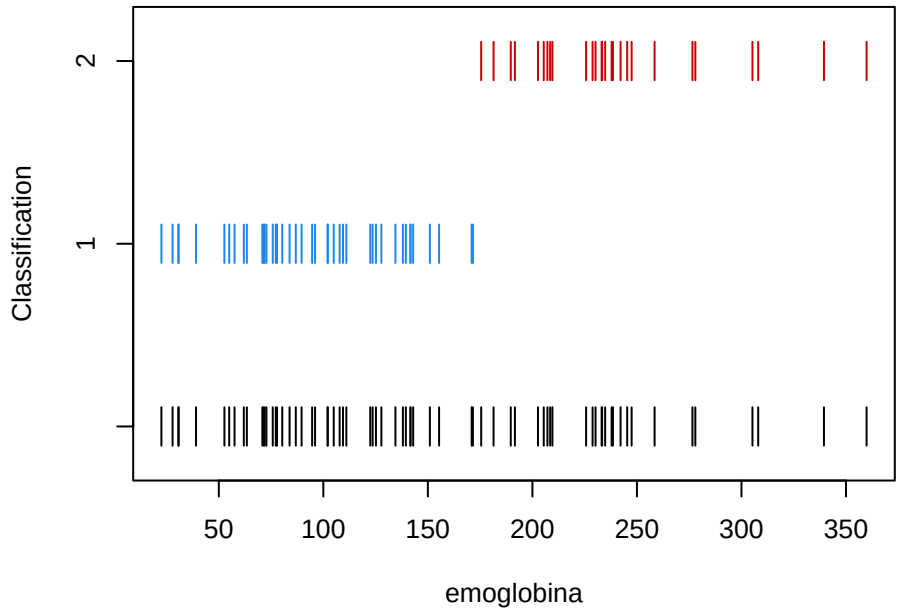
La curva della densità del miscuglio mostra in modo molto chiaro le due componenti, con due massimi, posti in corrispondenza delle medie delle due componenti (come evidenziato della righe verticali posti in corrispondenza delle medie delle due componenti). La seconda componente presenta una media più elevata. Si osserva che il picco della prima componente risulta più elevato rispetto all'altro a causa del maggior peso di quella componente (se i pesi fossero stati uguali, avendo anche uguale varianza, il massimo delle due componenti sarebbe risultato alla stessa quota).

Si noti che se non si specifica il tipo di grafico, vengono stampati tutti i grafici (BIC, classificazione, incertezza di classificazione e densità).

.5

Rappresentiamo anche il grafico che mostra la classificazione delle unità nei due gruppi. Anche in questo caso, è sufficiente specificare l'opzione `what='classification'` per ottenere il grafico.

```
plot(mod1, what='classification')
```



Il grafico rappresenta la classe di allocazione per ciascuna unità, sulla base della regola della massima probabilità a posteriori. Osserviamo che il valore della componente del plasma che discrimina l'appartenenza ad una classe o all'altra è intorno a 170. Si conferma la caratterizzazione della prima classe come quella con valori della componente del plasma più bassi. Si osserva inoltre che entrambe le classi hanno un numero di persone ragionevole (abbiamo già osservato che ci sono 43 osservazioni nella prima classe e 27 nella seconda).

SOLUZIONE ESERCIZIO 39

.1

Si descrivono i dati analizzandone le principali statistiche descrittive.

```
load("./datln.Rdata")
skimr::skim_without_charts(data1)
```

Table 43: Data summary

Name	data1
Number of rows	32
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

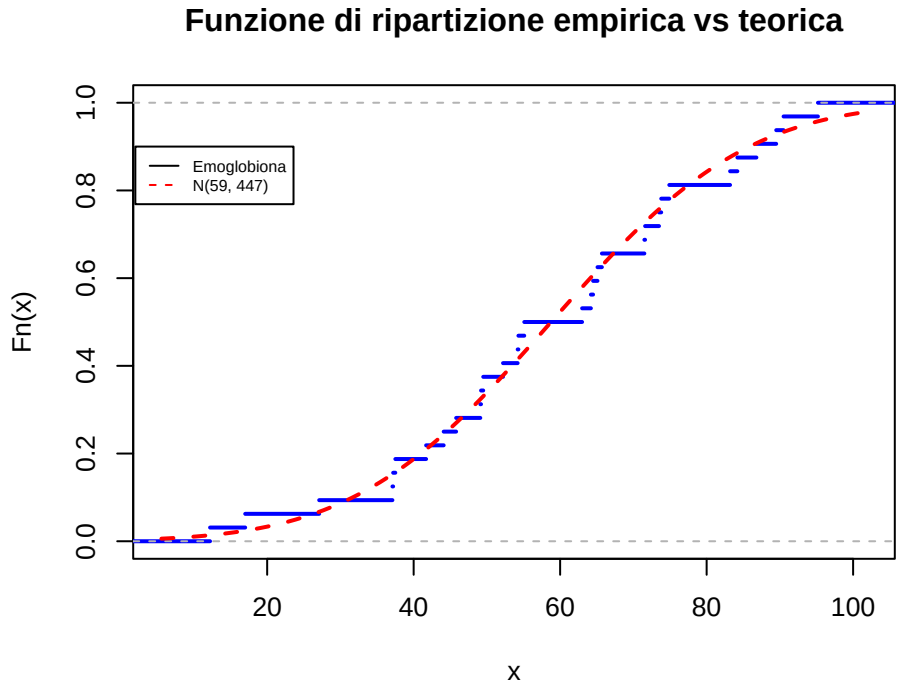
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	58.77	21.14	12.2	45.37	59.05	73.58	95.2

Il dataset contiene solo 32 osservazioni per un'unica variabile, la lunghezza delle ferite. La variabilità delle osservazioni è piuttosto limitata, presentando un valore della deviazione standard pari a circa 21; anche il campo di variazione non è particolarmente esteso, andando d aun minimo di 12 mm ad un massimo di 95 mm. La lunghezza media delle ferite è pari a quasi 6 cm, valore approssimativamente coincidente con la mediana: metà dei soggetti esaminati ha riportato ferite di lunghezza superiore ai 6cm. La lunghezza di un quarto delle ferite non supera i 4 cm e mezzo (primo quartile).

Rappresentiamo ora graficamente la funzione di ripartizione empirica delle osservazioni confrontandola con quella teorica relativa ad una V.C. Gaussiana univariata.

```
plot(ecdf(data1),
     do.points = FALSE,
     main = "Funzione di ripartizione empirica vs teorica",
     col = "blue",
     lwd = 2)

curve(pnorm(x, mean = mean(data1), sd = sd(data1)),
      add = TRUE,
      lty = 2,
      col = "red",
      lwd = 2)

legend(2, 0.9,
       col = c("black", "red"),
       c("Emoglobiona", "N(59, 447)"),
       lty = c(1, 2),
       cex = 0.6)
```



Osseraviamo che la curva empirica si sovrappone in modo molto accurato a quella teorica. Il risultato può risultare condizionato dalle poche osservazioni disponibili, ma sembra suggerire che una distribuzione normale univariata sia particolarmente adatta per interpretare i dati osservati. Possiamo ipotizzare che il modello miscuglio non sia la scelta più adatta in questo caso.

.2

Utilizziamo il criterio di informazione Bayesiano (BIC) per valutare il modello ottimale, variando sia il numero di componenti, sia l'assunzione sulla varianza delle diverse componenti (ocmune a tutte o specifica per ciascuna).

```
mclust::mclustBIC(data1)
```

```
#> Bayesian Information Criterion (BIC):
#>           E         V
#> 1 -291.9992 -291.9992
#> 2 -298.3368 -299.5150
#> 3 -303.3167 -310.9852
#> 4 -306.4282 -315.5288
#> 5 -309.9439 -310.3042
#> 6 -316.6261 -306.8127
#> 7 -323.5604 -315.4417
#> 8 -330.4907 -330.3440
#> 9 -337.4117 -321.3681
```

```
#>
#> Top 3 models based on the BIC criterion:
#>       E,1       V,1       E,2
#> -291.9992 -291.9992 -298.3368
```

Come ipotizzato in precedenza, si osserva che il modello ottimale è quello con una sola componente (in questo caso non vi è differenza tra modello con varianza comune e varianza specifica: essendo una sola la componente c'è una sola varianza). Segue, in ordine di preferenza il modello omoscedastico con due componenti.

.3

Riportiamo i commentiamo i valori dei parametri del modello selezionato.

```
est <- mclust::Mclust(data1, G = 1)
summary(est, parameters = TRUE)
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust X (univariate normal) model with 1 component:
#>
#>  log-likelihood  n df       BIC       ICL
#>       -142.5339 32  2 -291.9992 -291.9992
#>
#> Clustering table:
#>  1
#> 32
#>
#> Mixing probabilities:
#> 1
#> 1
#>
#> Means:
#> [1] 58.76563
#>
#> Variances:
#> [1] 432.8929
```

I commenti sonoper lo più una ripetizione di quanto mostrato al punto 1. Per quanto riguarda la tabella di classificazione, tutte le osservazioni sono ovviamente allcoate nell'unico gruppo; il peso dell'unica componente è pari a 1.

.4 e .5

Tutte le osservazioni sono allocate all'unico gruppo esistente.

SOLUZIONE ESERCIZIO 40

.1

Si descrivono i dati analizzandone le principali statistiche descrittive.

```
load("Dentice.Rdata")
summary(SS)
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   2.900   5.000   5.850   6.225   7.425  12.800
```

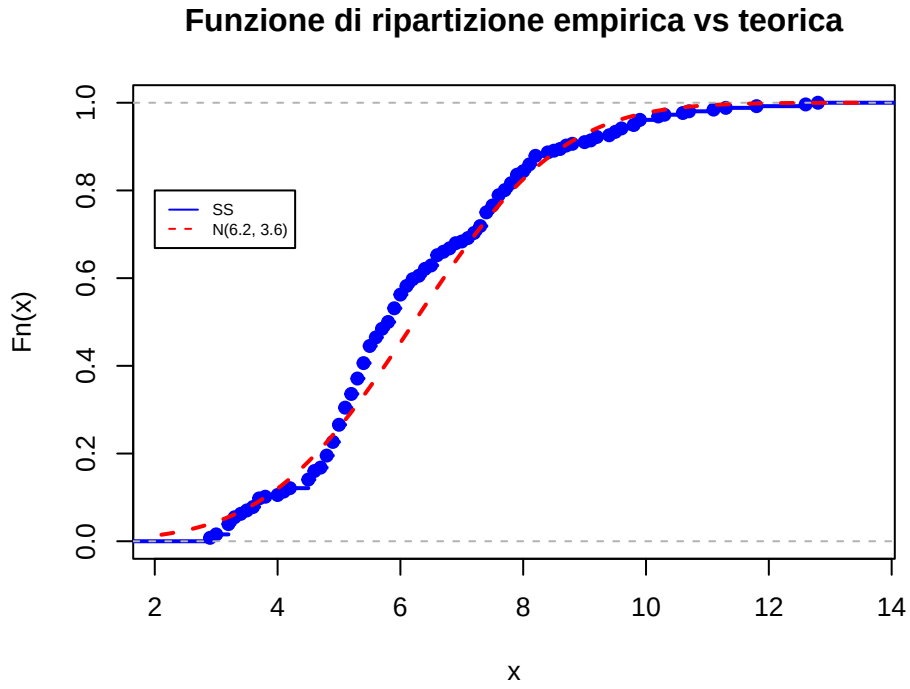
La lunghezza media dei 256 dentici considerati è pari a 6.2; il valore della mediana è di poco più basso. Lo scarto interquartile, misurato come differenza tra terzo e primo quartile è di circa 2.5, valore molto basso se confrontato con la distanza tra massimo e minimo (circa 10): ne consegue che metà dei dentici (la parte centrale della distribuzione) hanno una lunghezza compresa in un intervalo di valori molto contenuto, tra 5 e 7.5, primo e terzo quartile. Le code della distribuzione presentano invece valori molto dispersi. Il valore della deviazione standard, calcolato qui di seguito, è piuttosto limitato: le lunghezze considerate hanno uno scostamento medio dalla media di circa 1.9.

```
sd(SS)
```

```
#> [1] 1.8949
```

Rappresentiamo graficamente i valori osservati calcolando la funzione di ripartizione empirica e confrontandola con quella teorica sotto l'ipotesi di distribuzione normale con media e varianza pari ai valori campionari.

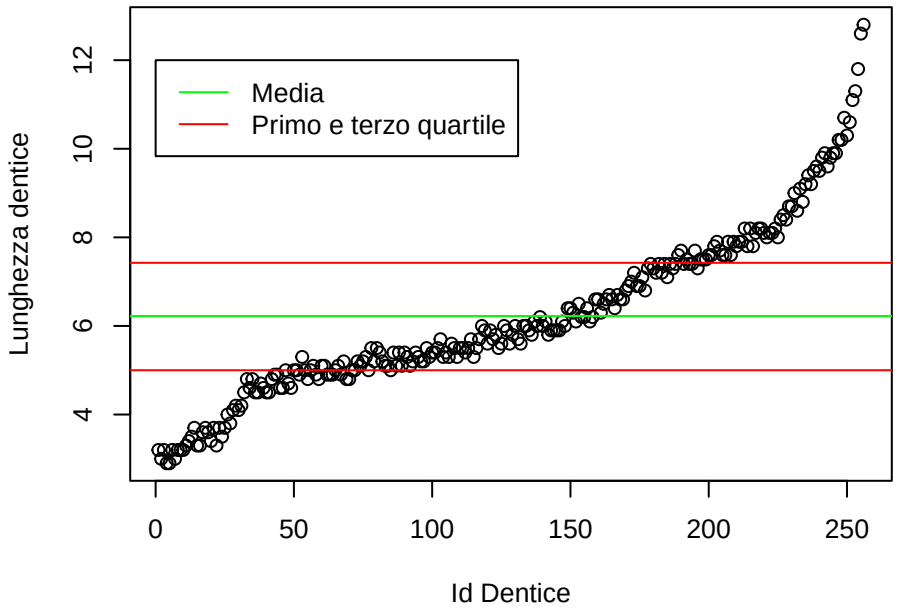
```
plot(ecdf(SS),
     main = "Funzione di ripartizione empirica vs teorica",
     col = "blue",
     lwd = 2)

curve(pnorm(x, mean = mean(SS), sd = sd(SS)),
      lty = 2, lwd = 2, col = "red",
      add = TRUE)

legend(2, 0.8,
       col = c("blue", "red"),
       c("SS", "N(6.2, 3.6)"),
       lty = c(1, 2),
       cex = 0.6)
```



La prima parte della distribuzione mostra degli scostamenti significativi tra la funzione empirica e quella teorica. Possiamo ipotizzare quindi che le osservazioni corrispondenti a questa zona faranno riferimento ad una componente del modello miscuglio con distribuzione piuttosto diversa rispetto a quella assunta qui. Al contrario oltre ad un certo valore per la lunghezza (approssimativamente 8), le due curve sembrano coincidere quasi perfettamente.

Infine rappresentiamo i dat nel diagramma di dispersione, aggiungendo le linee orizzontali per media e primo e terzo quartile della distribuzione.

```
plot(SS,
     xlab = "Id Dentice",
     ylab = "Lunghezza dentice")
q <- quantile(SS, c(0.25, 0.75))

abline(h = c(mean(SS), q[1], q[2]),
       col = c("green", "red", "red"))

legend(0, 12,
       c("Media", "Primo e terzo quartile"),
       col = c("green", "red"),
       lty = c(1,1))
```



In questo i dati seguono una forma particolarmente definita; questo non ha alcun reale significato, dal momento che stimao plottando i valori delle lunghezze rispetto all'id, e non rispetto ad un'altra variabile. Osserviamo comunque che la coda destra della distribuzione (valori elevati) sembra essere più lunga rispetto a quella sinistra, allontanadosi maggiormente dal "proprio" quartile (il quartile più vicino); anche la dispersione aumenta.

.2

Dato che non è richiesto di specificare sul modello il vincolo di uguale variabilità, si stima sia il modello in cui le due componenti hanno uguale varianza (più parsimonioso) sia il modello in cui le due componenti hanno diversa variabilità.

Modello con uguale varianza delle componenti

```
require("mclust")
mod_E <- Mclust(SS, G = 2, modelNames = "E")
summary(mod_E)
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust E (univariate, equal variance) model with 2 components:
#>
#>  log-likelihood   n df      BIC       ICL
```

```
#>       -515.2898 256  4 -1052.76 -1094.472
#>
#> Clustering table:
#>   1   2
#> 225  31
```

Modello con varianza specifica per le componenti

```
mod_V <- Mclust(SS, G = 2, modelNames = "V")
summary(mod_V)
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust V (univariate, unequal variance) model with 2 components:
#>
#>  log-likelihood   n df       BIC       ICL
#>       -513.3384 256  5 -1054.403 -1175.986
#>
#> Clustering table:
#>   1   2
#> 159  97
```

Commentiamo ora i risultati ottenuti per i due modelli stimati. I valori della log-verosimiglianza per i due modelli sono piuttosto simili; quello per il modello con varianza comune tra le due componenti presenta un valore leggermente più basso (-515 contro -513). Considerando il valore dell'indice BIC (e quindi andando a penalizzare la log-verosimiglianza per il numero di parametri del modello), la situazione si inverte; si seleziona quindi come modello ottimale quello con BIC = -1052.760 ovvero quello con uguale varianza. Il numero di parametri del modello è infatti diverso tra i due: quello con uguale varianza ha solo 4 parametri liberi, quello con varianze specifiche ne ha 5 (bisogna aggiuneger la varianza per la seconda componente). Osserviamo infine che la classificazione delle unità nele due componente presenta differenze significative tra i due risultati.

.3

Interpretiamo ora i risultati relativi ai parametri di ciascuno dei due modelli.

```
summary(mod_E, parameters=TRUE)
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust E (univariate, equal variance) model with 2 components:
```

```
#>
#>  log-likelihood   n df      BIC       ICL
#>       -515.2898 256  4 -1052.76 -1094.472
#>
#> Clustering table:
#>   1   2
#> 225  31
#>
#> Mixing probabilities:
#>         1         2
#> 0.8471224 0.1528776
#>
#> Means:
#>        1        2
#> 5.684987 9.214754
#>
#> Variances:
#>        1        2
#> 1.963072 1.963072
```

```
summary(mod_V, parameters=TRUE)
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust V (univariate, unequal variance) model with 2 components:
#>
#>  log-likelihood   n df       BIC       ICL
#>       -513.3384 256  5 -1054.403 -1175.986
#>
#> Clustering table:
#>   1   2
#> 159  97
#>
#> Mixing probabilities:
#>       1       2
#> 0.51532 0.48468
#>
#> Means:
#>        1        2
#> 5.151611 7.365439
#>
#> Variances:
#>        1        2
#> 1.098866 3.685410
```

Osserviamo innanzitutto i diversi pesi che vengono assegnati alle due componenti dai due modelli:

considerando quello con uguale variabilità, il peso della prima componente è notevolmente più alto rispetto a quello della seconda componente. Al contrario, assumendo il modello con varianze specifiche, la differenza è molto meno marcata e le due componenti hanno approssimativamente lo stesso peso nel miscuglio.

Per quanto riguarda le medie, si osserva che il valore relativo alla prima componente non varia in modo sensibile a seconda del modello: risulta uguale a 5.6 per il modello con uguale varianza e a 5.1 per quello con varianza specifica. Al contrario la media relativa alla seconda componente è sensibilmente più alto (circa 9) se consideriamo il modello con varianza comune alle due componenti. In ogni caso possiamo caratterizzare le due componenti in base alla lunghezza dei dentici: i più lunghi nella seconda, i più corti nella prima.

La varianza (unica) del primo modello stimato vale circa 1.96; calcolandone la radice, questo risultato si interpreta affermando che, per entrambe le componenti, la variabilità media intorno alla rispettiva media è di circa 1.4. Nel caso del secondo modello, la varianza della prima componente assume un valore molto inferiore rispetto a quella della seconda.

.4

Per il confronto tra i soggetti appartenenti alle due diverse componenti consideriamo unicamente il modello con uguale varianza (quello suggerito dal BIC). Gli stessi passi si potrebbero ripetere sull'altro modello. Estraiamo per prima cosa gli indici dei soggetti appartenenti alle due diverse componenti.

```
ind1 <- which(mod_E$classification == 1)
ind2 <- which(mod_E$classification == 2)

summary(SS[ind1])
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   2.900   4.900   5.500   5.726   6.700   8.200
```

```
summary(SS[ind2])
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   8.400   9.050   9.600   9.842  10.250  12.800
```

Oltre alle già osservate differenze per medie e varianze, possiamo notare come il valore del massimo per la prima componente sia inferiore al valore del minimo per la seconda. In entrambi i casi i valori di media e mediana coincidono approssimativamente: bisogna ricordare che entrambe le componenti assumono distribuzione Gaussiana univariata, pertanto è naturale individuarne le caratteristiche principali.

SOLUZIONI ESERCIZIO 41

Si rimanda alle soluzioni degli esercizi precedenti.

SOLUZIONI ESERCIZIO 42

.1

Si descrivono i dati analizzandone le principali statistiche descrittive.

```
load("fluidip.Rdata")
skimr::skim_without_charts(flu)
```

Table 45: Data summary

Name	flu
Number of rows	875
Number of columns	1
_____	
Column type frequency:	
numeric	1
_____	
Group variables	None

Variable type: numeric

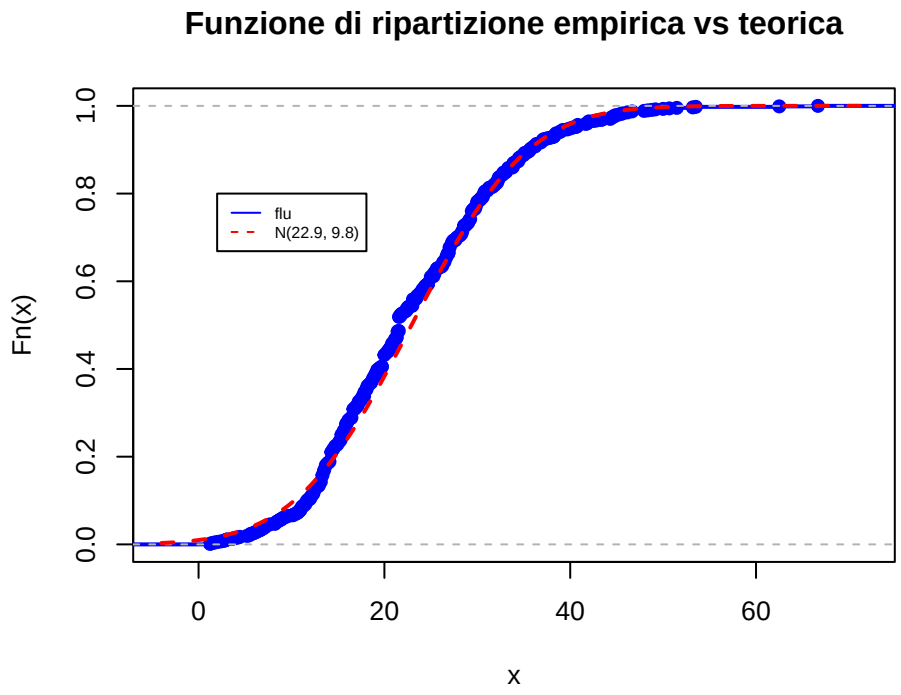
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
data	0	1	22.92	9.83	1.23	15.5	21.57	29.41	66.67

Il valore medio della quantità di fluidi somministrati ai pazienti è di circa 23 mL/kg; tale valore coincide approssimativamente con la mediana: la metà dei pazienti assume più di 22 mL/kg di fluidi in seguito all'operazione. I valori di primo e terzo quartile sono molto vicini tra loro; lo scarto interquartile vale approssimativamente 15, a fronte di un campo di variazione per l'intera distribuzione di circa 65. Ne consegue che metà delle osservazioni si distribuisce molto densamente nella parte centrale della distribuzione; le due code risultano invece molto allungate e con valori delle osservazioni abbastanza dispersi. In particolare la coda destra è particolarmente pesante, arrivando ad un valore massimo di 67 mL/kg di fluidi assunti.

Rappresentiamo ora i dati osservati attraverso il grafico della loro funzione di ripartizione empirca e confrontandola con quella teorica; si assume per questa una distribuzione Gaussiana con media e varianza date dai valori campionari.

```
plot(ecdf(flu),
     main = "Funzione di ripartizione empirica vs teorica",
     col = "blue",
     lwd = 2)

curve(pnorm(x, mean = mean(flu), sd = sd(flu)),
      lty = 2, lwd = 2, col = "red",
```

```
      add = TRUE)

legend(2, 0.8,
       col = c("blue", "red"),
       c("flu", "N(22.9, 9.8)"),
       lty = c(1, 2),
       cex = 0.6)
```



La funzione di ripartizione empirica sembra aderire in modo piuttosto accurato a quella teorica assunta; si evidenziano dei leggeri scostamenti solo nella parte iniziale della distribuzione. Apparentemente potremmo quindi ipotizzare che una variabile di Gauss così definita possa essere adatta ad interpretare i dati osservati. In realtà, effettuando il test di Kolmogorov-Smirnov per la verifica della normalità, si nota un valore del p-value piuttosto basso (ordine di 10^{-4}), il che porterebbe a rigettare l'ipotesi nulla che i dati abbiano distribuzione normale con media e varianza definite sopra.

```
ks.test(flu, "pnorm", mean = mean(flu), sd = sd(flu))
```

```
#>
#>  Asymptotic one-sample Kolmogorov-Smirnov test
#>
#> data:  flu
#> D = 0.07338, p-value = 0.0001617
#> alternative hypothesis: two-sided
```

.2

Stimiamo il modello miscuglio a due componenti Gaussiane e in cui si suppone varianza specifica per ogni componente.

```
mod_flu<- Mclust(flu, G = 2, modelNames = "V")
summary(mod_flu, parameters = TRUE)
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust V (univariate, unequal variance) model with 2 components:
#>
#>  log-likelihood   n df       BIC       ICL
#>       -3221.907 875  5 -6477.685 -6963.479
#>
#> Clustering table:
#>   1   2
#> 465 410
#>
#> Mixing probabilities:
#>        1        2
#> 0.446132 0.553868
#>
#> Means:
#>        1        2
#> 16.74911 27.88629
#>
#> Variances:
#>        1        2
#> 36.48648 89.53369
```

Il modello stimato utilizza un totale di 5 parametri liberi (uno dei pesi, le due medie e le due varianze); il valore della log-likelihood ottenuto è pari a -3221.9. Per quanto riguarda la classificazione delle osservazioni nelle due componenti del miscuglio, nella prima sono allocate 465 unità, nella seconda 410. Le due sottopopolazioni risultano quindi abbastanza bilanciate.

Passando all'analisi dei coefficienti, si osserva che la seconda componente ha un peso maggiore rispetto alla prima: la probabilità di essere allocati nella seconda componente è pari a 0.55, mentre per la prima è 0.45 (nonostante la seconda abbia effettivamente un minor numero di soggetti allocati).

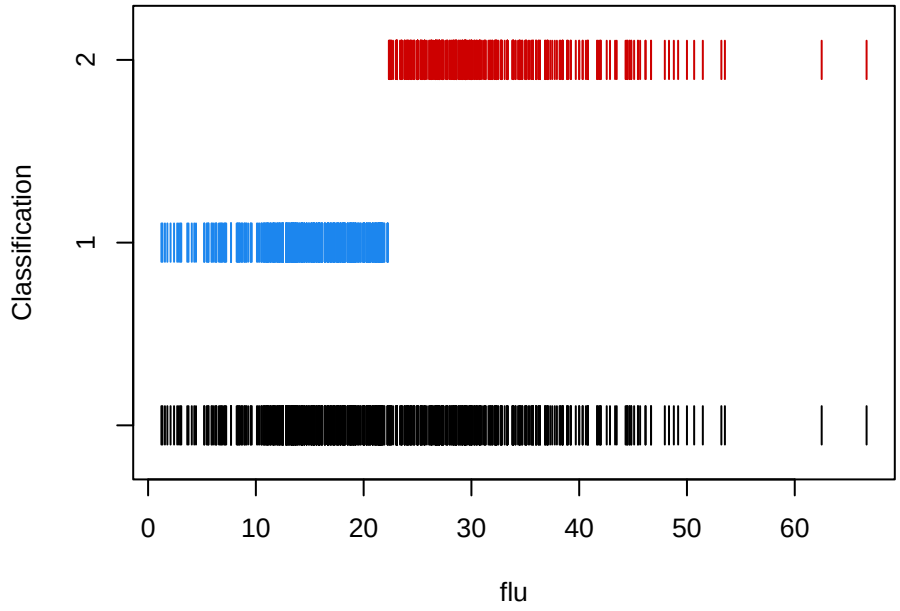
Analizzando le medie delle osservazioni delle due sottopopolazioni possiamo caratterizzare le componenti: la prima contiene infatti i pazienti che hanno assunto livelli mediamente bassi di fludi (la media di queste unità è 16.8 mL/kg); nella seconda, viceversa, sono allocati i soggetti che hanno assunto livelli alti di fluidi (media pari a 27.9 mL/kg). Anche i valori delle varianze sono sensibilmente diversi tra le due componenti. Le osservazioni della seconda sottopopolazione risultano

infatti molto più disperse rispetto a quelle della prima: hanno una variabilità media intorno alla propria media di circa 9 (per quelle della prima componente è invece pari a circa 6).

.3

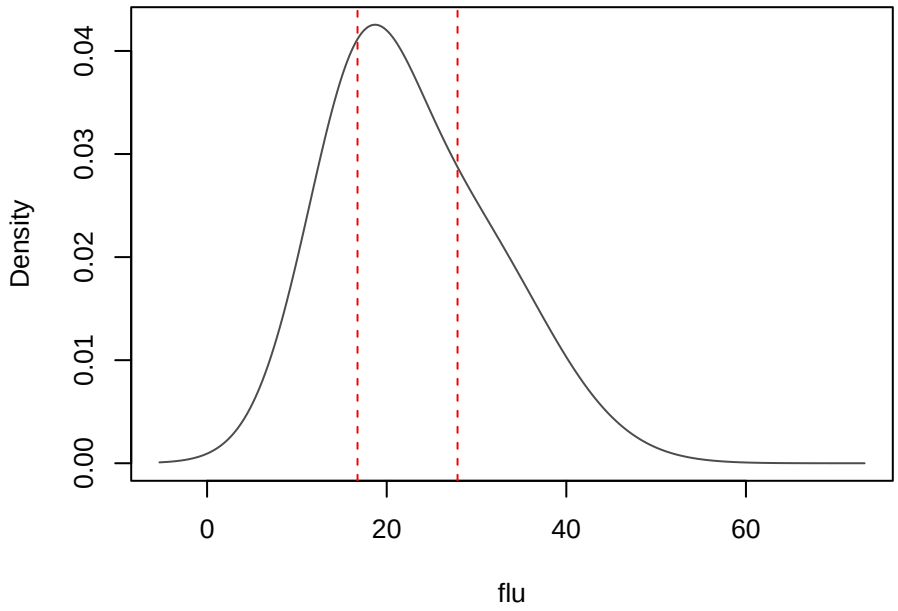
Riportiamo i grafici principali per l'interpretazione del modello stimato.

```
plot(mod_flu, what = "classification")
```



Il grafico relativo alla classificazione delle unità viene realizzato sulla base della regola della massima probabilità a posteriori; mostra che entrambe le componenti contengono un numero ragionevole di unità. La dose di liquidi somministrati che discrimina l'appartenenza ad una sottopopolazione o all'altra è intorno a 25 mL/kg; questo consente di confermare la caratterizzazione della prima sottopopolazione come quella con un basso valore di fluidi assunti dal paziente (viceversa per la seconda componente). Si evidenzia inoltre la presenza, nel secondo gruppo, di un ristretto numero di osservazioni che si discostano in modo significativo dal resto della distribuzione: si tratta di pazienti che hanno assunto una quantità particolarmente elevata di9 fluidi.

```
plot(mod_flu, what = "density")
abline(v = mod_flu$parameters$mean, col = rep("red", 2), lty = rep(2, 2))
```



Il grafico della densità mostra la curva della funzione di densità del miscuglio. Questa presenta un unico punto di massimo posizionato in prossimità della media della prima componente. Tale situazione è casua della variabilità della seconda componente che è molto più elevata rispetto a quella della prima, a fronte di due pesi molto simili tra le due. Si rileva comunque una certa asimmetria verso destra (coda più lunga), segno della presenza della seconda sottopopolazione.

.4

Calcoliamo infine le probabilità a posteriori, ovvero le probabilità con cui ciascuna unità viene allocata alla prima o alla seconda componente. Stampiamo in particolare i valori relativi alle prime 6 unità del dataset.

```
head(round(mod_flu$z, 2))
```

```
#>      [,1] [,2]
#> [1,] 0.54 0.46
#> [2,] 0.04 0.96
#> [3,] 0.54 0.46
#> [4,] 0.01 0.99
#> [5,] 0.81 0.19
#> [6,] 0.00 1.00
```

```
tail(round(mod_flu$z, 2))
```

```
#>        [,1] [,2]
#> [870,] 0.00 1.00
#> [871,] 0.80 0.20
#> [872,] 0.73 0.27
#> [873,] 0.79 0.21
#> [874,] 0.04 0.96
#> [875,] 0.01 0.99
```

Si nota per esempio che le unità 1 e 3 presentano maggior incertezza di classificazione, venendo allocate alla prima componente sulla base del metodo della massima probabilità a posteriori, ma presentando comunque una probabilità per la seconda componente piuttosto elevata. Viceversa per le rimanenti osservazioni esaminate, lo'allocazione non presenta incertezza.

SOLUZIONI ESERCIZIO 43

.1

Si descrivono i dati analizzandone le principali statistiche descrittive.

```
load("gel.Rdata")
data <- as.data.frame(data)
skimr::skim_without_charts(data)
```

Table 47: Data summary

Name	data
Number of rows	122
Number of columns	2
_____	
Column type frequency:	
numeric	2
_____	
Group variables	None

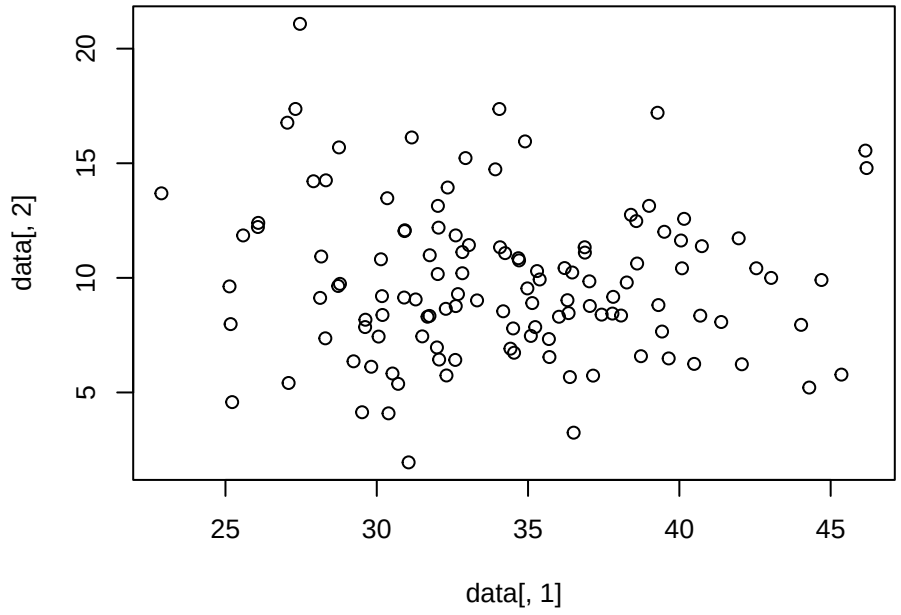
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
A	0	1	34.19	5.04	22.88	30.57	34.06	37.7	46.19
B	0	1	9.81	3.31	1.95	7.69	9.42	11.7	21.08

Il dataset raccoglie i dati relativi al livello di piastrine (variabile `A`) e di globuli bianchi (variabile `B`) nel sangue di 122 pazienti. I dati relativi al livello di piastrine presentano un campo di variazione

abbastanza ampio, con un massimo pari a 46 ed un minimo di 23. Leggermente più limitato è invece il campo di variazione per il livello di globuli bianchi (va da 2 a 21). In entrambi i casi i valori di media e mediana sono approssimativamente coincidenti. La variabilità è contenuta per entrambi i caratteri.

```
plot(data[, 1], data[, 2])
```



Il grafico a dispersione delle osservazioni mostra che i punti corrispondenti sono disposti nel piano cartesiano in modo casuale; non si evidenzia infatti la presenza di pattern particolari tra i dati. Si segnala la presenza di alcuni punti che risultano isolati rispetto alla distribuzione; ci sono ad esempio due punti con un valore molto elevato per il livello di piastrine (e mediamente alto anche per i globuli bianchi), o un unico punto con un livello di piastrine molto basso.

.2

La scelta delle componenti del miscuglio multivariato si basa sulla funzione `mclust::mclustBIC`. Nel caso multivariato le possibili specificazioni del modello sono molteplici. Nel seguito ci concentriamo sull'assunzione di componenti sferiche (ovvero assumendo che non ci sia correlazione tra le variabili A e B). La codifica per questa situazione è `EII` (nel caso in cui la variabilità sia la stessa tra tutte le componenti) o `VII` (nel caso in cui ciascuna componente abbia una specifica variabilità). Esistono due analoghe codifiche, `EEE` e `VEE`, nel caso di modelli non sferici (ovvero in cui si assume la presenza di correlazione tra e variabili).

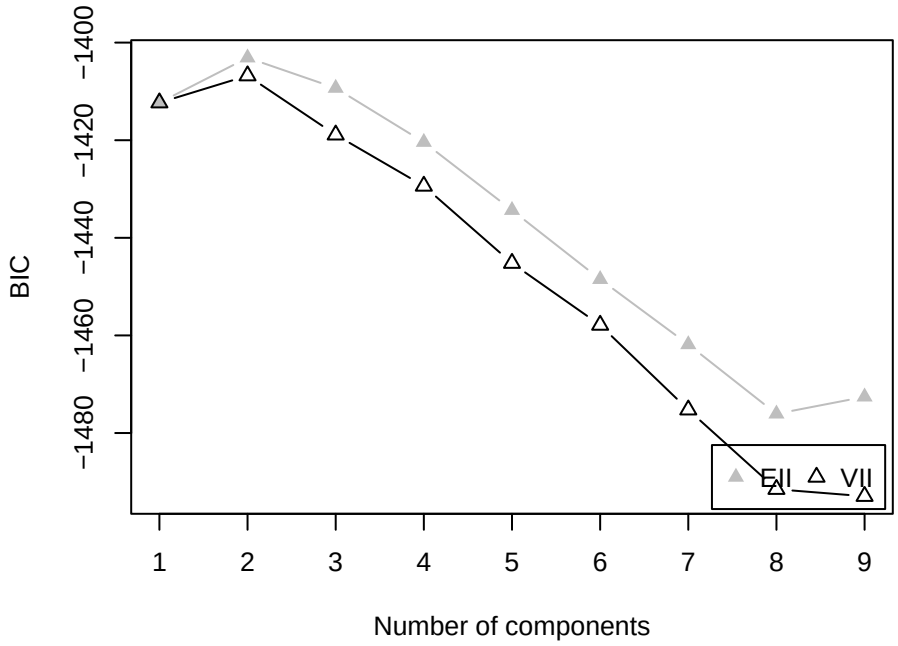
```
require(mclust)
mcc <- mclust::mclustBIC(data, modelNames = c("EII", "VII")); mcc
```

```
#> Bayesian Information Criterion (BIC):
#>         EII       VII
#> 1 -1412.321 -1412.321
#> 2 -1403.100 -1406.778
#> 3 -1409.323 -1418.899
#> 4 -1420.404 -1429.384
#> 5 -1434.344 -1445.197
#> 6 -1448.501 -1457.882
#> 7 -1461.857 -1475.242
#> 8 -1476.073 -1491.518
#> 9 -1472.578 -1492.956
#>
#> Top 3 models based on the BIC criterion:
#>     EII,2     VII,2     EII,3
#> -1403.100 -1406.778 -1409.323
```

Analogamente al caso univariato, l'output mostra i risultati in forma di matrice, con il numero di componenti sulle righe e la specificazione del modello sulle colonne. Considerando i tre migliori modelli dal punto di vista dell'indice BIC, si rileva che la scelta migliore è selezionare due componenti ed assumere variabilità comune tra entrambe. Questo risultato è preferibile (valore del BIC leggermente più alto) rispetto al modello con 2 componenti e variabilità specifica.

Gli stessi risultati si ottengono dal grafico corrispondente, che mostra i due simboli a quota maggiore in corrispondenza di un numero di componenti pari a 2.

```
plot(mcc)
```



.3

Stimiamo ora i parametri del modello selezionato al punto precedente.

```
mc <- Mclust(data, G = 2, modelNames = "EII")
summary(mc, parameters = TRUE )
```

```
#> ----------------------------------------------------
#> Gaussian finite mixture model fitted by EM algorithm
#> ----------------------------------------------------
#>
#> Mclust EII (spherical, equal volume) model with 2 components:
#>
#>  log-likelihood   n df     BIC       ICL
#>       -687.1381 122  6 -1403.1 -1441.014
#>
#> Clustering table:
#>  1  2
#> 78 44
#>
#> Mixing probabilities:
#>         1         2
#> 0.6283537 0.3716463
#>
```

```
#> Means:
#>        [,1]      [,2]
#> A 31.399671 38.916329
#> B  9.933265  9.614923
#>
#> Variances:
#> [,,1]
#>          A        B
#> A 11.40698  0.00000
#> B  0.00000 11.40698
#> [,,2]
#>          A        B
#> A 11.40698  0.00000
#> B  0.00000 11.40698
```

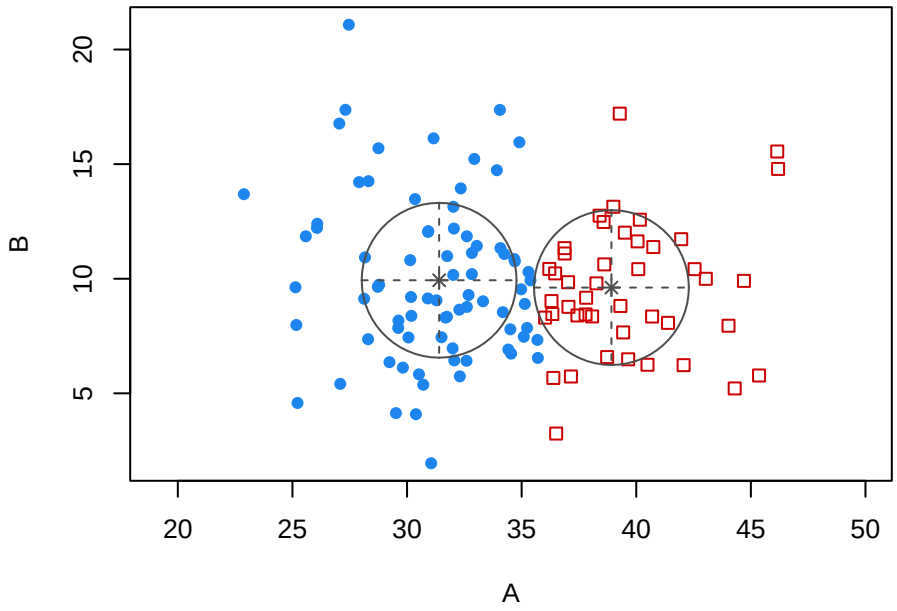
Osserviamo dapprima che il numero di parametri liberi del modelli è pari a 6: si tratta di uno dei due pesi, dei 4 valori delle medie e dell'unica varianza. Il valore corrispondente per la funzione di log-verosimiglianza è pari a -687. Sono inoltre riportati gli indici BIC (con cui è stata fatta la selezione del modello) e ICL, basato sull'entropia. Per quanto riguarda la classificazione delle unità nei due gruppi, la prima sottopopolazione risulta significativamente più numerosa, contenendo 78 osservazioni delle 122 totali.

L'analisi dei parametri del modello mostra che la prima componente ha un peso molto superiore rispetto alla seconda: la probabilità che un'unità selezionata in modo casuale appartenga alla prima classe è pari a 0.63. Le medie consentono di caratterizzare le due sottopopolazioni dal punto di vista del contesto applicativo; si osserva infatti che la prima componente presenta una media per il livello di piastrine (pari a 31.4) significativamente inferiore rispetto all'analogo parametro per le unità del secondo gruppo (38.9). La seconda sottopopolazione risulta essere quindi caratterizzata da quei pazienti con un livello di piastrine medio-alto. Viceversa, per quanto riguarda il livello di globuli bianchi, la differenza delle medie tra le due componenti è molto più limitata: i soggetti allocati alla prima componente presentano un valore leggermente più alto, ma la differenza è minima (9.9 contro 9.6). La variabile relativa al livello delle piastrine sembra essere quindi meno rilevante per discriminare l'appartenenza delle osservazioni a uno dei due gruppi. Infine, si rileva che la matrice di varianza covarianza è (come richiesto) uguale per entrambe le componenti: in entrambi i gruppi, la variabilità media rispetto alla media è pari a 3.4 per entrambe le variabili.

.4

Come nel caso univariato, il grafico di classificazione delle osservazioni si ottiene specificando l'opzione `what = "classification"` nel comando `plot`.

```
plot(mc, what = "classification", asp = 1)
```

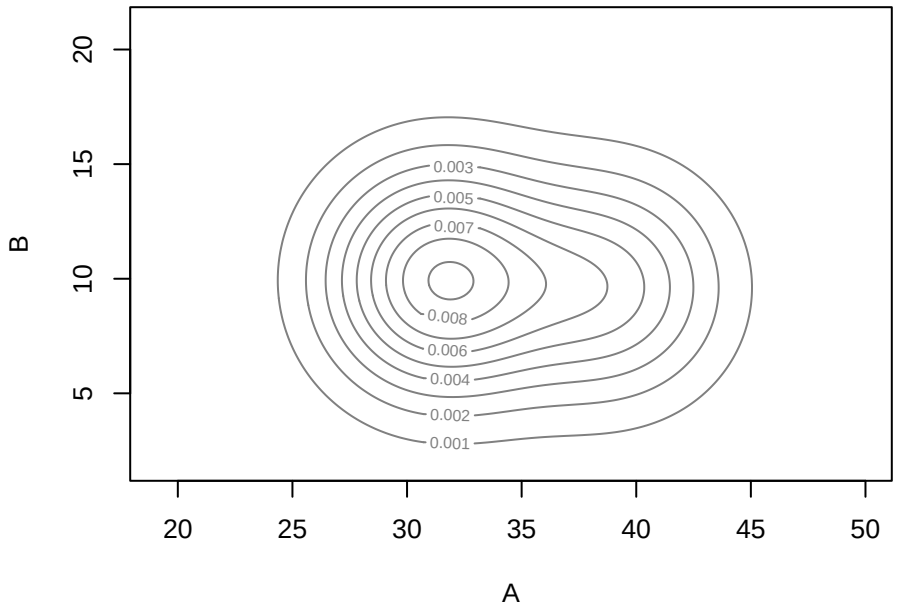


Il grafico mostra in questo caso la disposizione delle osservazioni nel piano cartesiano rispetto alle due variabili (grafico a dispersione). I punti corrispondenti alle unità vengono raffigurati con forme e colori diversi per sottolineare l'appartenenza ad una sottopopolazione o all'altra. Si osserva nuovamente che la variabile che misura il livello di globuli bianchi (asse y) non è rilevante al fine della classificazione delle unità: il passaggio da una componente all'altra non dipende dalla quota dei punti, ma solo dalla loro posizione lungo l'asse y (l'iperpiano che divide le due regioni è una retta quasi verticale). Osserviamo inoltre la presenza delle circonferenze relative alle due componenti (attraverso l'opzione `asp=1` è possibile fissare la stessa scala per asse x e asse y). Si evidenzia che le due circonferenze (modello sferico) sono uguali, oltre che come forma, anche come volume (area in questo caso), segno che la variabilità delle due componenti è stata vincolata ad assumere lo stesso valore.

.5

In modo analogo è possibile ottenere il grafico della densità del miscuglio.

```
plot(mc, what = "density", asp = 1)
```



La forma della densità viene rappresentata per mezzo delle curve di livello (proiezioni delle intersezioni della superficie 3D con piani paralleli al piano xy e posti a quote diverse). Si osserva che la figura è leggermente allungata nella direzione orizzontale, con un prolungamento più marcato verso destra. Questo è l'effetto di una prima componente con un peso maggiore (posizionata a sinistra) e di una meno pesante che produce il prolungamento. Non si rileva quindi la presenza di due picchi diversi.

.6

Per ottenere gli intervalli di confidenza bootstrap per i parametri del modello si utilizza la funzione `mclust::MclustBootstrap`. L'unico argomento è il modello stimato; il summary richiede poi l'opzione `what = "ci"` per ottenere gli intervalli di confidenza (eventualmente esiste un'analoga opzione `what = "se"` per ricavare gli standard error).

```
mc_boot <- mclust::MclustBootstrap(mc)
summary(mc_boot, what = "ci")
```

```
#> ----------------------------------------------------
#> Resampling confidence intervals
#> ----------------------------------------------------
#> Model                     = EII
#> Num. of mixture components = 2
#> Replications              = 999
#> Type                      = nonparametric bootstrap
```

```
#> Confidence level           = 0.95
#>
#> Mixing probabilities:
#>                1         2
#> 2.5%  0.4394546 0.2006242
#> 97.5% 0.7993758 0.5605454
#>
#> Means:
#> [,,1]
#>                A         B
#> 2.5%  29.99300  9.063553
#> 97.5% 32.65313 10.940278
#> [,,2]
#>                A         B
#> 2.5%  37.19494  8.781025
#> 97.5% 41.19874 10.692207
#>
#> Variances:
#> [,,1]
#>                A         B
#> 2.5%   8.825493  8.825493
#> 97.5% 13.606830 13.606830
#> [,,2]
#>                A         B
#> 2.5%   8.825493  8.825493
#> 97.5% 13.606830 13.606830
```

Osserviamo per esempio che l'intervallo di confidenza (al 95%) per il peso della prima componente è piuttosto ampio, avendo 0.4 e 0.8 come estremi inferiore e superiore rispettivamente. E' bene sottolineare come questo intervalo contenga quindi anche valori inferiori a 0.5, che genererebbero un miscuglio in cui la componente più pesante non sarebbe più la prima ma la seconda. I commenti per l'intervallo relativo al peso della seconda cmomponente sono speculari. Per quanto riguarda le medie gli intervalli di confidenza sono invece generalmente abbastanza ristretti; per esempio la l'intervallo per la media del livello di piastrine tra i soggetti del primo gruppo ha 30 e 32.7 come estremi: in altre parole nel 95% dei campioni 999 bootstrap il valore calcolato per questo parametro ricade all'interno di tale intervallo.

SOLUZIONI ESERCIZIO 44

Si rimanda alle soluzioni dell'esercizio precedente.

Soluzioni esercizi “Modelli a classi latenti”

SOLUZIONI ESERCIZIO 45

.1

Descriviamo brevemente i dati osservati riportando le frequenze osservate circa i due test che valutano la mobilità del ginocchio.

```
load("Mobility.Rdata")
n <- dim(Y)[1]
apply(Y, 2, table)/n
```

```
#>             Y         Y1
#> 0 0.13678696 0.13154831
#> 1 0.52619325 0.53492433
#> 2 0.17636787 0.20081490
#> 3 0.12630966 0.10069849
#> 4 0.03434226 0.03201397
```

Osserviamo che ciascuna variabile è categoriale e presenta cinque possibili categorie di risposta diverse (da 0, mobilità bassa, a 4, mobilità elevata). I risultati dei due test seguono uno stesso andamento e le differenze sono minime. In particolare osserviamo che la frequenza più alta è, per entrambi, in corrispondenza dei soggetti con mobilità medio-bassa; la frequenza di questa tipologia di mobilità all'interno del dataset è superiore al 50%. La frequenza minore è invece relativa ai soggetti con mobilità del ginocchio elevata, che risultano essere poco più del 3% dei casi totali per entrambi i test. Le restanti tre categorie mostrano frequenze piuttosto simili, sempre comprese tra il 0.1 e 0.2.

.2

Per poter applicare la funzione di stima del modello a classi latenti, è necessario modificare il formato dei dati, in modo da determinarne i pattern di frequenza; questo passaggio comporta una notevole riduzione della dimensionalità dei dati. L'operazione viene eseguita in modo automatico dal comando `MultiLCIRT::aggr_data`. La funzione restituisce in output le configurazioni di risposta uniche (in modo tale che se due o più soggetti presentano la stessa risposta per ogni variabile, la configurazione corrispondente viene mostrata una sola volta) e le corrispondenti frequenze assolute.

```
require(MultiLCIRT)
Yout <- aggr_data(Y)
S <- Yout$data_dis
yv <- Yout$freq
cbind(S, yv)
```

```
#>       Y Y1  yv
```

```
#>  [1,] 1  1 591
#>  [2,] 1  3  60
#>  [3,] 1  2 127
#>  [4,] 2  3  36
#>  [5,] 0  1 120
#>  [6,] 2  2 114
#>  [7,] 0  3  11
#>  [8,] 0  0  79
#>  [9,] 3  2  67
#> [10,] 2  0  22
#> [11,] 2  1 125
#> [12,] 3  4  15
#> [13,] 2  4   6
#> [14,] 3  3  49
#> [15,] 1  0 109
#> [16,] 0  4   4
#> [17,] 3  1  71
#> [18,] 4  2  16
#> [19,] 0  2  21
#> [20,] 1  4  17
#> [21,] 4  3  17
#> [22,] 4  4  13
#> [23,] 4  1  12
#> [24,] 3  0  15
#> [25,] 4  0   1
```

I risultati mostrano che le configurazioni uniche di risposta sono pari a 25; la più frequente, con 591 soggetti che riportano questa situazione, è quella corrispondente a un livello di mobilità medio-basso per entrmbi i test. La frequenza minore (solo 4 unità) si osserva invece per i soggetti che mostrano mobilità elevata con il primo test e bassa con il secondo. Si tratta chiaramente di una situazione limite, che giustifica il numero molto ridotto di soggetti che la segnalano. In generale prevalgono infatti le configurazioni con risposte analoghe o simili per entrambi i test.

.3

La stima del modello a classi latenti viene effettuata con la funzione `MultiLCIRT::est_multi_poly` specificando il dataset con le configurazioni uniche (`S`), le relative frequenze (`yv`) e il numero di componenti (`k=2`). E' inoltre opportuno utilizzare l'opzione `output=TRUE` per ricavare un maggior numero di informazioni dall'output del modello. Il livello di tolleranza necessario per valutare la convergenza dell'algoritmo EM è fissato a `tol = 10^-10`.

```
modlc2 <- est_multi_poly(S, yv, k=2, output = TRUE)
```

```
#> *-----------------------------------------------------------------------------*
#> Link of type =                  0
#> Discrimination index =          0
```

```
#> Constraints on the difficulty = 0
#> Type of initialization =        0
#> *-------------------------------------------------------------------------*
```

```
length(modlc2$lkv)
```

```
#> [1] 150
```

```
modlc2$np
```

```
#> [1] 17
```

Il numero di passi può essere visto considerando che nell'oggetto `lkv`, dove sono salvati i valori che assume la funzione di log-verosimiglianza ad ogni step dell'algoritmo EM. In questo caso l'algoritmo richiede 150 passi per raggiungere la convergenza.

Il numero di parametri è invece contenuto nell'oggetto `np`; si tratta di uno dei due pesi e delle 16 probabilità condizionate di risposta.

```
summary(modlc2)
```

```
#>
#> Call:
#> est_multi_poly(S = S, yv = yv, k = 2, output = TRUE)
#>
#> Log-likelihood:
#> [1] -4267.8
#>
#> AIC:
#> [1] 8569.6
#>
#> BIC:
#> [1] 8662.23
#>
#> Class weights:
#> [1] 0.6131 0.3869
#>
#> Conditional response probabilities:
#> , , class = 1
#>
#>          item
#> category      1      2
#>        0 0.1953 0.2067
#>        1 0.6533 0.7325
#>        2 0.1025 0.0461
#>        3 0.0490 0.0135
```

```
#>         4 0.0000 0.0012
#>
#> , , class = 2
#>
#>          item
#> category      1      2
#>         0 0.0441 0.0125
#>         1 0.3248 0.2218
#>         2 0.2935 0.4460
#>         3 0.2488 0.2389
#>         4 0.0888 0.0809
```

Stampiamo il summary del modello in modo da poterne valutare e commentare i parametri. Il valore della funzione di log-verosimiglianza a convergenza è pari a -4268; i valori corrispondenti degli indici BIC e AIC sono rispettivamente 8662 e 8569.

Per quanto riguarda i parametri, consideriamo prima il peso di ciascuna delle due classi latenti. La prima classe risulta quella con peso maggiore: la probabilità che un'osservazione selezionata in modo casuale sia allocata qui è pari a 0.61.

Le probabilità condizionate di risposta rappresentano invece i valori della probabilità di avere, condizionatamente alla classe latente di appartenenza, una determinata categoria di risposta per le diverse variabili. Per esempio, osservando i valori relativi alla prima classe latente, si osserva che il 65% dei soggetti ha mobilità medio-bassa secondo il primo test, mentre lo stesso livello di mobilità è raggiunto dal 73% dei pazienti considerando il secondo test. Anche la probabilità di avere soggetti con una bassa mobilità è piuttosto elevata (circa 0.2 per entrambi i test). le rimanenti categorie di risposta sono molto meno frequenti. In conclusione si può caratterizzare la prima classe latente come quella in cui sono allocati i sogegtti che hanno mediamente una mobilità bassa o medio-bassa rispetto ad entrambi i test.

La seconda classe latente contiene invece soggetti in cui la mobilità è leggermente più alta: si evidenziano probabilità piputtosto elevate per i livelli medio-basso, medio e medio-alto, mentre la probabilità di avere soggetti con mobilità bassa cala drasticamente rispetto alla prima classe latente e non supera il 4%. Inoltre si evidenziano probabilità non nulle (per quanto basse) per i soggetti con mobilità elevata. La seconda classe latente caratterizza quindi quei soggetti con mobilità media, includendo anche i pochi che mostrano una mobilità medio-alta o alta. Entrambe le variabili sembrano essere rilevanti per l'allocazione delle unità nelle classi.

.4

Le probabilità a posteriori vengono fornite per ciascuna configurazione di risposte (non per ciascun soggetto). Sono contenute nell'oggetto `Pp` dell'output del modello.

```
round(modlc2$Pp, 2)
```

```
#>       [,1] [,2]
#>  [1,] 0.91 0.09
#>  [2,] 0.15 0.85
```

```
#>  [3,] 0.25 0.75
#>  [4,] 0.03 0.97
#>  [5,] 0.96 0.04
#>  [6,] 0.05 0.95
#>  [7,] 0.28 0.72
#>  [8,] 0.99 0.01
#>  [9,] 0.03 0.97
#> [10,] 0.90 0.10
#> [11,] 0.65 0.35
#> [12,] 0.00 1.00
#> [13,] 0.01 0.99
#> [14,] 0.02 0.98
#> [15,] 0.98 0.02
#> [16,] 0.09 0.91
#> [17,] 0.51 0.49
#> [18,] 0.00 1.00
#> [19,] 0.42 0.58
#> [20,] 0.05 0.95
#> [21,] 0.00 1.00
#> [22,] 0.00 1.00
#> [23,] 0.00 1.00
#> [24,] 0.84 0.16
#> [25,] 0.00 1.00
```

Confrontando i valori delle probabilità con la matrice delle configurazioni uniche di risposta, si osserva per esempio che i pazienti con pattern 4 (ovvero che punteggio ai due test 2 e 3 indicanti mobilità medio bassa e mobilità media) vengono assegnato alla seconda classe latente con probabilità molto elevata (0.97). Viceversa la prima classe contiene con alta probabilità i soggetti che presentano livello di mobilità medio-basso rispetto ad entrambi i test. Si osservano anche configurazioni di risposta con incertezza di classificazione. Per esempio la configurazione 17, relativa ai soggetti che presentano mobilità medio-alta rispetto al primo test ma medio-bassa rispetto al secondo, hanno probabilità di classificazione nella prima classe pari a 0.51 e nella seconda pari a 0.49. Per la regola della massima probabilità a posteriori risultano comunque allocate nella prima classe.

SOLUZIONI ESERCIZIO 46

Si rimanda alle soluzioni dell'esercizio precedente.

SOLUZIONI ESERCIZIO 47

Questo esercizio contiene due covariate (parte non svolta durante il corso). Per la risoluzione senza l'inserimento di covariate (considerando solo le prime due colonne del dataset) si rimanda alle soluzioni dell'esercizio 45.

SOLUZIONI ESERCIZIO C

.1

Descriviamo brevemente i dati osservati riportando le frequenze osservate circa i due test che valutano la mobilità del ginocchio.

```
load("mate.Rdata")
n <- dim(mate)[1]
apply(mate, 2, table)/n
```

```
#>       Item1     Item2     Item3     Item4
#> 0 0.2642384 0.2569536 0.3834437 0.1324503
#> 1 0.7357616 0.7430464 0.6165563 0.8675497
```

Osserviamo che, per ciascuno dei quattro quesiti, la maggior parte degli studenti ha risposto correttamente. La domanda con al frequenza maggiore di risposte errate è stata la terza (moltiplicazione di due interi negativi!!), alla quale quasi il 40% degli studenti non ha saputo fornire la risposta corretta. Le altre domande presentano frequenze di errori più bassi, fino al quesito 4, a cui solo il 13% degli studenti non ha risposto correttamente.

.2

Un metodo alternativo di rappresentazione di questa tipologia di dati consiste nel ricavare le configurazioni di risposta (ovvero tutte le possibili combinazioni di risposte ai diversi quesiti) ed associare la frequenza assoluta di ciascuna configurazione. Questo consente di ridurre la dimensione del dataset e permette di condurre analisi descrittive leggermente diverse rispetto a quelle di cui al punto precedente.

```
require(MultiLCIRT)
mate_mod <- aggr_data(mate)
S <- mate_mod$data_dis
yv <- mate_mod$freq
cbind(S, yv)
```

```
#>       Item1 Item2 Item3 Item4  yv
#>  [1,]     1     1     0     1 245
#>  [2,]     0     1     1     1 120
#>  [3,]     1     1     1     1 598
#>  [4,]     0     0     0     1  72
#>  [5,]     1     1     0     0  24
#>  [6,]     1     0     1     1  85
#>  [7,]     0     1     0     1  61
#>  [8,]     0     0     1     1  43
#>  [9,]     1     1     1     0  36
#> [10,]     1     0     0     1  86
```

```
#> [11,]     0     1     0     0  22
#> [12,]     0     0     0     0  44
#> [13,]     0     0     1     0  21
#> [14,]     1     0     1     0  12
#> [15,]     1     0     0     0  25
#> [16,]     0     1     1     0  16
```

In questo caso le diverse configurazioni di risposta sono solo 16 (a fronte di 1510 soggetti). Affiancandole alle corrispondenti frequenze assolute, si nota che la configurazione più frequente è quella secondo cui gli studenti rispondono correttamente a tutti i quesiti; ben 598 soggetti ricadono in questa situazione. Viceversa, solo 44 studenti sbagliano la risposta di tutte e quattro le domande. La situazione meno frequente è però quelle in cui si risponde correttamente alle domande 1 e 3, ma non alle domande 2 e 4. In generale, è piuttosto raro il caso di due domande corrette e due sbagliate.

.3

Stimiamo ora un modello a classi latenti assumendo la presenza di 2 classi.

```
est <- MultiLCIRT::est_multi_poly(S = S, yv = yv, k = 2, output = TRUE)
```

```
#> *-------------------------------------------------------------------------*
#> Link of type =                  0
#> Discrimination index =          0
#> Constraints on the difficulty = 0
#> Type of initialization =        0
#> *-------------------------------------------------------------------------*
```

```
summary(est)
```

```
#>
#> Call:
#> MultiLCIRT::est_multi_poly(S = S, yv = yv, k = 2, output = TRUE)
#>
#> Log-likelihood:
#> [1] -3171
#>
#> AIC:
#> [1] 6360
#>
#> BIC:
#> [1] 6407.87
#>
#> Class weights:
#> [1] 0.2579 0.7421
#>
```

```
#> Conditional response probabilities:
#> , , class = 1
#>
#>          item
#> category      1      2      3      4
#>        0 0.5993 0.6985 0.6955 0.3694
#>        1 0.4007 0.3015 0.3045 0.6306
#>
#> , , class = 2
#>
#>          item
#> category      1      2     3      4
#>        0 0.1478 0.1035 0.275 0.0501
#>        1 0.8522 0.8965 0.725 0.9499
```

Il peso associato alla seconda componente risulta molto maggiore rispetto a quello della prima classe: considerando un'osservazione selezionata in modo casuale, la probabilità che appartenga alla seconda classe latente è di circa 0.75.

Le probabilità condizionate di risposta consentono poi di caratterizzare le due classi (finora sappiamo solo che una delle due contiene un numero maggiore di osservazioni). Osserviamo che, considerando le osservazioni classificate nella seconda classe (condizionatamente alla seconda classe), la probabilità di rispondere correttamente è molto elevata per ciascun quesito. Gli studenti allocati in questa classe forniscono la risposta corretta alla domanda 4 con probabilita pari quasi a 1; il quesito più complicato si conferma essere il terzo, con una probabilità di fornire la risposta corretta pari a 0.73. Considerando la prima classe latente, osserviamo invece che le probabilità di rispondere in modo sbagliato sono più alte rispetto alle probabilità di rispondere correttamente; l'unica eccezione è il quesito 4, per cui, anche tra gli studenti di questa classe latente, sono maggioritari coloro che hanno risposto correttamente.

L'interpretazione delle due classi latenti è quindi piuttosto semplice: la prima contiene gli studenti che hanno risposto per lo più in modo sbagliato ai quesiti proposti (non necessariamente a tutti i quesiti, ma globalmente il risultato non è stato positivo). Viceversa la seconda classe latente raccoglie gli studenti che sono stati in gradi di rispondere complessivamente in modo corretto all'insieme dei quesiti.

.4

Estraiamo ora dall'output del modello le probabilità a posteriori di assegnazione degli studenti alle classi latenti. Si ricorda che, nel caso del modello a classi latenti non otteniamo questo risultato per ogni unità del dataset, ma per ogni configurazione di risposte. Dobbiamo quindi confrontare questo risultato con la matrice delle configurazioni di risposta e le relative frequenze.

```
round(est$Pp, 2)
```

```
#>       [,1] [,2]
#>  [1,] 0.08 0.92
#>  [2,] 0.12 0.88
```

```
#>  [3,] 0.02 0.98
#>  [4,] 0.94 0.06
#>  [5,] 0.51 0.49
#>  [6,] 0.24 0.76
#>  [7,] 0.44 0.56
#>  [8,] 0.73 0.27
#>  [9,] 0.15 0.85
#> [10,] 0.65 0.35
#> [11,] 0.90 0.10
#> [12,] 0.99 0.01
#> [13,] 0.97 0.03
#> [14,] 0.77 0.23
#> [15,] 0.95 0.05
#> [16,] 0.59 0.41
```

Osserviamo per esempio che gli studenti che hanno risposto correttamente a tutti i quesiti (configurazione 3) viene allocata con probabilità molto alta (98%) alla seconda classe latente, confermandone l'interpretazione del punto precedente. Viceversa gli studenti con configurazione di risposta 12 (fornte risposte sbagliate a tutti i quesiti) vengono allocati con probabilità 0.99 alla prima classe latente. Vi sono poi configurazioni con un'incertezza di allocazione maggiore. Consideriamo per esempio gli studenti che hanno risposto correttamente solo alle domande 2 e 4 (configurazione 7): questi vengono allocati con probabilità 0.44 alla prima classe e con probabilità 0.56 alla seconda (secondo la regola di classificazione della massima probabilità a posteriori, *tutti* loro saranno comunque allocati alla seconda classe). Infine è interessante notare come altre configurazioni (per esempio la numero 8), pur mantenendo lo stesso numero di risposte corrette e risposte sbagliate (2 e 2) vengono allocate con incertezza molto minore alla classe latente 1.