Introduzione alla chemiometria e disegno sperimentale

Modulo 3: Il Software R (grafici di base)

Docente: Dr. Sabina Licen (slicen@units.it)

Visualizzare velocemente l'aspetto di matrici e dataframe

str(nomeoggetto) restituisce la struttura di un oggetto

```
> str(Ogg1)
'data.frame':   28 obs. of  4 variables:
 $ giorno: int  1 2 3 4 5 6 7 8 9 10 ...
 $ T2017 : num  7.3 8.5 11.4 9.2 9.4 10.6 9.4 7 5 6 ...
 $ T2018 : num  11.3 7.4 5.7 6.8 6.2 6.1 5.5 8.3 8.2 6.5 ...
 $ T2019 : num  NA NA NA NA 9.6 9.9 8.5 11.1 NA NA ...
> 
```

Visualizzare velocemente l'aspetto di matrici e dataframe

str(nomeoggetto) restituisce la struttura di un oggetto

head(nomeoggetto) restituisce le prime sei righe dell'oggetto

```
> str(Ogg1)
'data.frame':   28 obs. of  4 variables:
 $ giorno: int  1 2 3 4 5 6 7 8 9 10 ...
 $ T2017 : num  7.3 8.5 11.4 9.2 9.4 10.6 9.4 7 5 6 ...
 $ T2018 : num  11.3 7.4 5.7 6.8 6.2 6.1 5.5 8.3 8.2 6.5 ...
 $ T2019 : num  NA NA NA NA 9.6 9.9 8.5 11.1 NA NA ...
> 
>
> head(Ogg1)
  T2017 T2018 T2019
1   7.3  11.3    NA
2   8.5   7.4    NA
3  11.4   5.7    NA
4   9.2   6.8    NA
5   9.4   6.2   9.6
6  10.6   6.1   9.9
```

Visualizzare velocemente l'aspetto di matrici e dataframe

str(nomeoggetto) restituisce la struttura di un oggetto

head(nomeoggetto) restituisce le prime sei righe dell'oggetto

tail(nomeoggetto) restituisce le ultime sei righe dell'oggetto

```
> str(Ogg1)
'data.frame':   28 obs. of  4 variables:
 $ giorno: int  1 2 3 4 5 6 7 8 9 10 ...
 $ T2017 : num  7.3 8.5 11.4 9.2 9.4 10.6 9.4 7 5 6 ...
 $ T2018 : num  11.3 7.4 5.7 6.8 6.2 6.1 5.5 8.3 8.2 6.5 ...
 $ T2019 : num  NA NA NA NA 9.6 9.9 8.5 11.1 NA NA ...
> 
```

```
> head(Ogg1)
  T2017 T2018 T2019
1   7.3  11.3    NA
2   8.5   7.4    NA
3  11.4   5.7    NA
4   9.2   6.8    NA
5   9.4   6.2   9.6
6  10.6   6.1   9.9
```

```
> tail(Ogg1)
   T2017 T2018 T2019
23   8.6   4.7   5.2
24   9.7   4.6   6.1
25   9.2  -1.5  10.0
26   7.9  -3.5  10.4
27   8.6  -3.0  10.2
28  12.6  -2.2  10.0
```

Visualizzare velocemente l'aspetto di matrici e dataframe

str(nomeoggetto) restituisce la struttura di un oggetto

head(nomeoggetto) restituisce le prime sei righe dell'oggetto

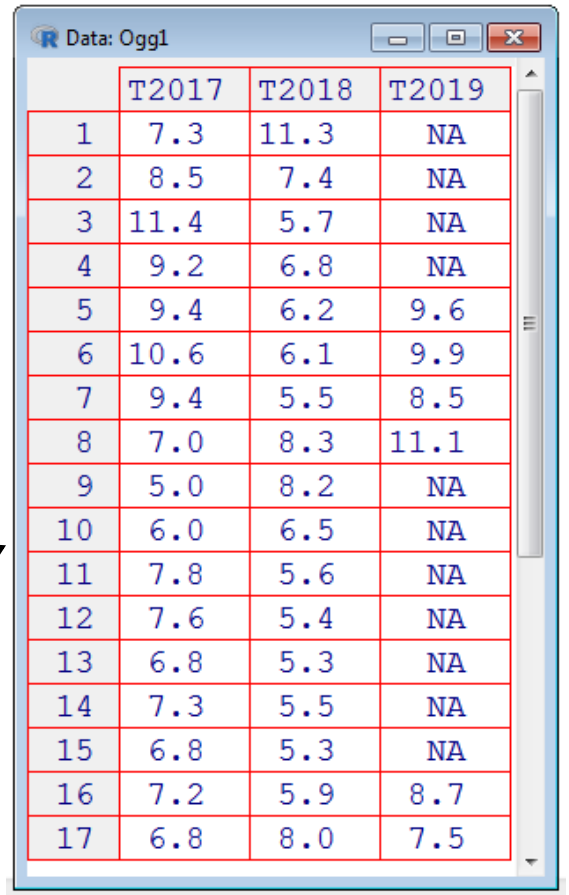
tail(nomeoggetto) restituisce le ultime sei righe dell'oggetto

View(nomeoggetto) apre una finestra che visualizza **tutto** l'oggetto

```
> str(Ogg1)
'data.frame':   28 obs. of  4 variables:
 $ giorno: int  1 2 3 4 5 6 7 8 9 10 ...
 $ T2017 : num  7.3 8.5 11.4 9.2 9.4 10.6 9.4 7 5 6 ...
 $ T2018 : num  11.3 7.4 5.7 6.8 6.2 6.1 5.5 8.3 8.2 6.5 ...
 $ T2019 : num  NA NA NA NA 9.6 9.9 8.5 11.1 NA NA ...
> 
```

```
> 
> head(Ogg1)
  T2017 T2018 T2019
1   7.3  11.3    NA
2   8.5   7.4    NA
3  11.4   5.7    NA
4   9.2   6.8    NA
5   9.4   6.2   9.6
6  10.6   6.1   9.9
```

```
> tail(Ogg1)
   T2017 T2018 T2019
23   8.6   4.7   5.2
24   9.7   4.6   6.1
25   9.2  -1.5  10.0
26   7.9  -3.5  10.4
27   8.6  -3.0  10.2
28  12.6  -2.2  10.0
> 
> View(Ogg1)
> 
```



	T2017	T2018	T2019
1	7.3	11.3	NA
2	8.5	7.4	NA
3	11.4	5.7	NA
4	9.2	6.8	NA
5	9.4	6.2	9.6
6	10.6	6.1	9.9
7	9.4	5.5	8.5
8	7.0	8.3	11.1
9	5.0	8.2	NA
10	6.0	6.5	NA
11	7.8	5.6	NA
12	7.6	5.4	NA
13	6.8	5.3	NA
14	7.3	5.5	NA
15	6.8	5.3	NA
16	7.2	5.9	8.7
17	6.8	8.0	7.5

La funzione summary()

summary(nomeoggetto) restituisce dati statistici "di base" per ogni colonna dell'oggetto

```
> summary(originale2)
                 date            PM03              PM05              PM07              PM1
 07/07/2015 09:52:00:    2   Min.   :   505   Min.   :    87   Min.   :   41   Min.   :   13
 10/07/2015 11:27:00:    2   1st Qu.: 23100   1st Qu.:  1782   1st Qu.:  570   1st Qu.:  319
 14/06/2015 16:50:00:    2   Median : 49590   Median :  3067   Median :  901   Median :  511
 18/06/2015 01:12:00:    2   Mean   : 61767   Mean   :  4172   Mean   : 1483   Mean   :  911
 19/06/2015 12:41:00:    2   3rd Qu.: 88919   3rd Qu.:  4874   3rd Qu.: 1668   3rd Qu.: 1023
 20/06/2015 23:26:00:    2   Max.   :885568   Max.   :225385   Max.   :74506   Max.   :46388
 (Other)            :84813   NA's   :7        NA's   :7        NA's   :7       NA's   :7
      PM2               PM3               PM5              PM10
 Min.   :    0.0   Min.   :    0.0   Min.   :   0.00   Min.   : 0.000
 1st Qu.:  131.0   1st Qu.:   34.0   1st Qu.:   7.00   1st Qu.: 0.000
 Median :  214.0   Median :   58.0   Median :  13.00   Median : 1.000
 Mean   :  411.4   Mean   :  115.6   Mean   :  26.89   Mean   : 1.604
 3rd Qu.:  451.0   3rd Qu.:  120.0   3rd Qu.:  29.00   3rd Qu.: 2.000
 Max.   :27345.0   Max.   :10766.0   Max.   :4058.00   Max.   :651.000
 NA's   :7         NA's   :7         NA's   :7         NA's   :7
```

Ordinare vettori, matrici e dataframe

sort(nomevettore, decreasing = FALSE)

VETTORI

Ordinare vettori, matrici e dataframe

sort(nomevettore, decreasing = FALSE)

VETTORI

---

order()

```
> Df
   V1 V2   V3
1   a  1  5.6
2   b  5  8.2
3   c  4  7.4
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
7   c  4  1.3
8   d  9 11.7
9   a  5 25.1
10  b  3  7.5
```

MATRICI & DATAFRAME

```
> Df2<-Df[order(Df$V3),]
> Df2
   V1 V2   V3
7   c  4  1.3
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
1   a  1  5.6
3   c  4  7.4
10  b  3  7.5
2   b  5  8.2
8   d  9 11.7
9   a  5 25.1
```

Ordinare vettori, matrici e dataframe

sort(nomevettore, decreasing = FALSE)

VETTORI

---

order()

```
> Df
   V1 V2   V3
1   a  1  5.6
2   b  5  8.2
3   c  4  7.4
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
7   c  4  1.3
8   d  9 11.7
9   a  5 25.1
10  b  3  7.5
```

MATRICI & DATAFRAME

- L'ordine dell'elenco è importante!!!
- Il segno meno significa decrescente

```
> Df2<-Df[order(Df$V3),]
> Df2
   V1 V2   V3
7   c  4  1.3
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
1   a  1  5.6
3   c  4  7.4
10  b  3  7.5
2   b  5  8.2
8   d  9 11.7
9   a  5 25.1
```

```
> Df3<-Df[order(-Df$V2,Df$V1),]
> Df3
   V1 V2   V3
8   d  9 11.7
5   a  8  3.6
4   d  7  3.5
9   a  5 25.1
2   b  5  8.2
3   c  4  7.4
7   c  4  1.3
6   b  3  4.2
10  b  3  7.5
1   a  1  5.6
```

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

```
> Vector<-c(1:10)
> Vector
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector)
[1] "integer"
```

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

```
> Vector<-c(1:10)
> Vector
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector)
[1] "integer"
>
> Vector2<-as.character(Vector)
> Vector2
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
> class(Vector2)
[1] "character"
>
```

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

```
> Vector<-c(1:10)
> Vector
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector)
[1] "integer"
>
> Vector2<-as.character(Vector)
> Vector2
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
> class(Vector2)
[1] "character"
>
> Vector3<-as.factor(Vector)
> Vector3
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 2 3 4 5 6 7 8 9 10
> class(Vector3)
[1] "factor"
```

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

```
> Vector<-c(1:10)
> Vector
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector)
[1] "integer"
>
> Vector2<-as.character(Vector)
> Vector2
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
> class(Vector2)
[1] "character"
>
> Vector3<-as.factor(Vector)
> Vector3
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 2 3 4 5 6 7 8 9 10
> class(Vector3)
[1] "factor"
>
> Vector4<-as.factor(Vector2)
> Vector4
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 10 2 3 4 5 6 7 8 9
> class(Vector4)
[1] "factor"
```

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

```
> Vector<-c(1:10)
> Vector
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector)
[1] "integer"
>
> Vector2<-as.character(Vector)
> Vector2
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
> class(Vector2)
[1] "character"
>
> Vector3<-as.factor(Vector)
> Vector3
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 2 3 4 5 6 7 8 9 10
> class(Vector3)
[1] "factor"
>
> Vector4<-as.factor(Vector2)
> Vector4
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 10 2 3 4 5 6 7 8 9
> class(Vector4)
[1] "factor"
```

osservare la differenza dell'ordine dei livelli tra un vettore numerico e un vettore *character* trasformati in *factor!*

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

```
> Vector<-c(1:10)
> Vector
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector)
[1] "integer"
>
> Vector2<-as.character(Vector)
> Vector2
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
> class(Vector2)
[1] "character"
>
> Vector3<-as.factor(Vector)
> Vector3
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 2 3 4 5 6 7 8 9 10
> class(Vector3)
[1] "factor"
>
> Vector4<-as.factor(Vector2)
> Vector4
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 10 2 3 4 5 6 7 8 9
> class(Vector4)
[1] "factor"
>
>
> Vector5<-as.numeric(as.character(Vector4))
> Vector5
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector5)
[1] "numeric"
>
```

osservare la differenza dell'ordine dei livelli tra un vettore numerico e un vettore *character* trasformati in *factor!*

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

```
> Vector<-c(1:10)
> Vector
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector)
[1] "integer"
>
> Vector2<-as.character(Vector)
> Vector2
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
> class(Vector2)
[1] "character"
>
> Vector3<-as.factor(Vector)
> Vector3
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 2 3 4 5 6 7 8 9 10
> class(Vector3)
[1] "factor"
>
> Vector4<-as.factor(Vector2)
> Vector4
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 10 2 3 4 5 6 7 8 9
> class(Vector4)
[1] "factor"
>
>
> Vector5<-as.numeric(as.character(Vector4))
> Vector5
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector5)
[1] "numeric"
>
```

osservare la differenza dell'ordine dei livelli tra un vettore numerico e un vettore *character* trasformati in *factor!*

Si possono usare anche per cambiare la classe di una colonna di un dataframe, es.

```
> Df$V1
 [1] "a" "b" "c" "d" "a" "b" "c" "d" "a" "b"
> Df$V1<-as.factor(Df$V1)
> Df[,1]
 [1] a b c d a b c d a b
Levels: a b c d
> Df$V1<-as.character(Df$V1)
> Df[,1]
 [1] "a" "b" "c" "d" "a" "b" "c" "d" "a" "b"
```

La funzione as.*()

as.*(nomeoggetto) consente di cambiare la classe di un oggetto in un'altra classe

(se l'oggetto è formato da elementi che consentono di assegnarlo a più classi)

```
> Vector<-c(1:10)
> Vector
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector)
[1] "integer"
>
> Vector2<-as.character(Vector)
> Vector2
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
> class(Vector2)
[1] "character"
>
> Vector3<-as.factor(Vector)
> Vector3
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 2 3 4 5 6 7 8 9 10
> class(Vector3)
[1] "factor"
>
> Vector4<-as.factor(Vector2)
> Vector4
 [1] 1  2  3  4  5  6  7  8  9  10
Levels: 1 10 2 3 4 5 6 7 8 9
> class(Vector4)
[1] "factor"
>
>
> Vector5<-as.numeric(as.character(Vector4))
> Vector5
 [1]  1  2  3  4  5  6  7  8  9 10
> class(Vector5)
[1] "numeric"
>
```

Si può anche convertire un dataframe in matrice o viceversa con le funzioni

as.matrix

as.data.frame

osservare la differenza dell'ordine dei livelli tra un vettore numerico e un vettore *character* trasformati in *factor!*

Si possono usare anche per cambiare la classe di una colonna di un dataframe, es.

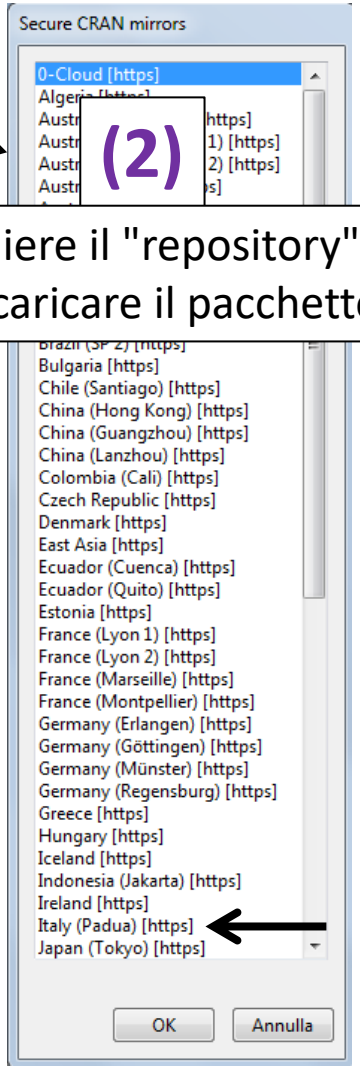
```
> Df$V1
 [1] "a" "b" "c" "d" "a" "b" "c" "d" "a" "b"
> Df$V1<-as.factor(Df$V1)
> Df[,1]
 [1] a b c d a b c d a b
Levels: a b c d
> Df$V1<-as.character(Df$V1)
> Df[,1]
 [1] "a" "b" "c" "d" "a" "b" "c" "d" "a" "b"
```

Installare e caricare pacchetti

Per installare un pacchetto bisogna procedere come segue:

(1) `install.packages("nomepacchetto")`

(2) Scegliere il "repository" da cui scaricare il pacchetto

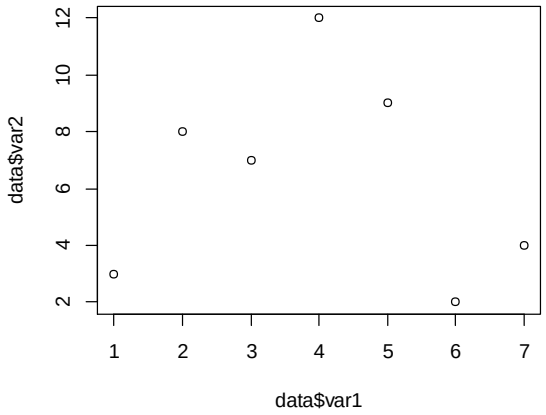


(3) Attendere...

(4) Ogni volta che si apre R bisogna caricare il pacchetto:

`library(nomepacchetto)`

Grafici a punti

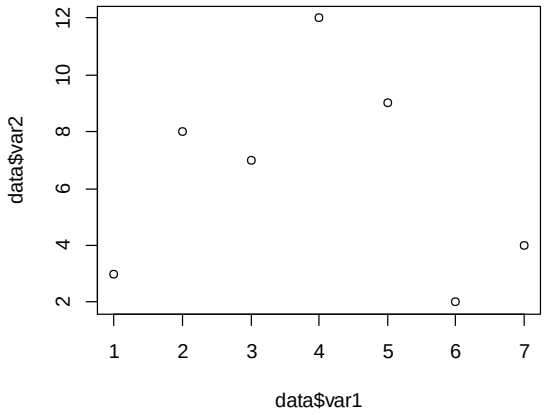


Configurazione "minima" del comando:

plot(nomeoggettoA, nomeoggettoB)

X **Y**

Grafici a punti



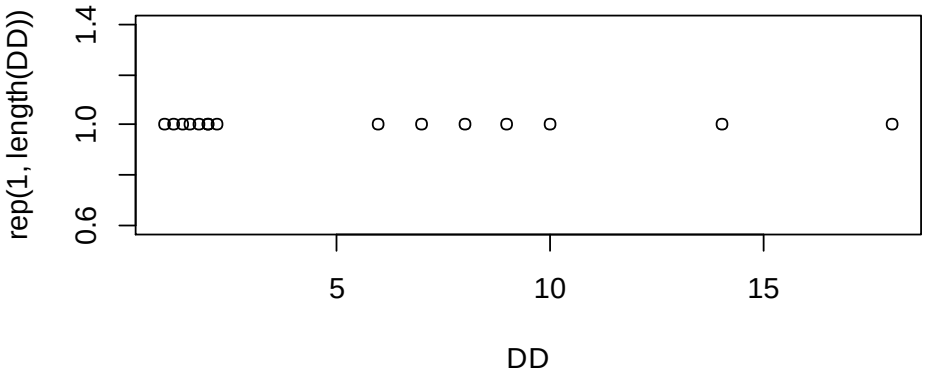
Configurazione "minima" del comando:

plot(nomeoggettoA, nomeoggettoB)

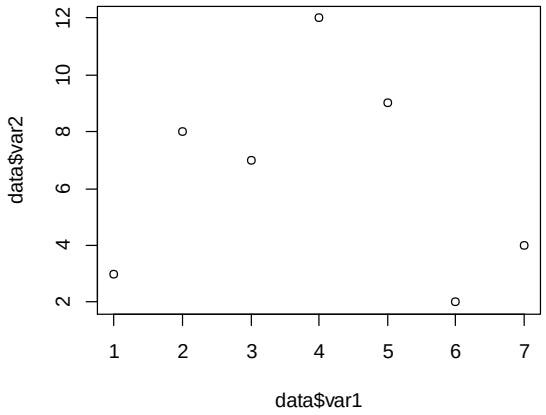
X **Y**

Per visualizzare una distribuzione univariata "orizzontale" o "verticale":

plot(nomeoggetto, rep(1,length(nomeoggetto)))



Grafici a punti



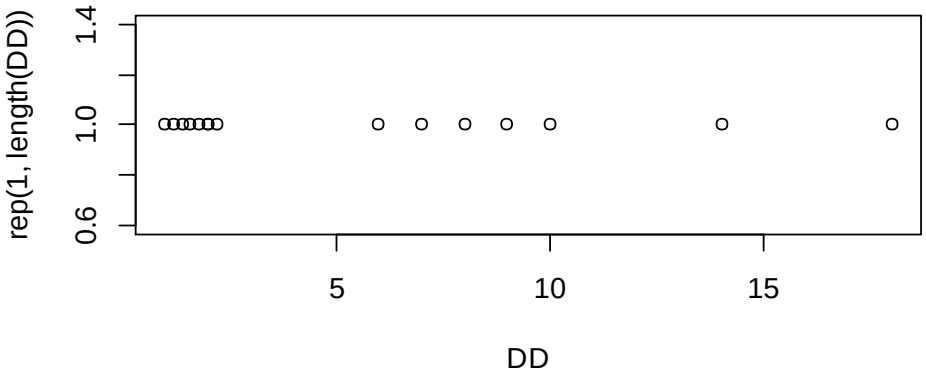
Configurazione "minima" del comando:

plot(nomeoggettoA, nomeoggettoB)

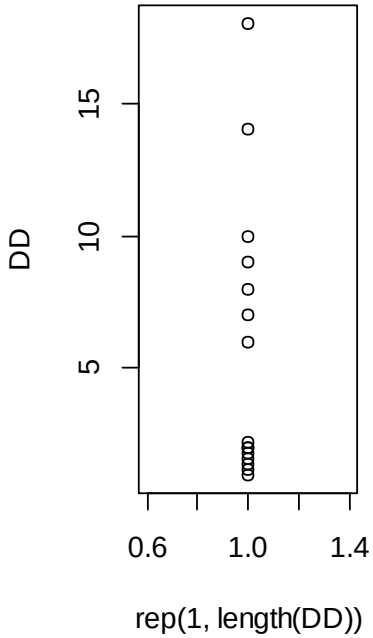
- nomeoggettoA → X
- nomeoggettoB → Y

Per visualizzare una distribuzione univariata "orizzontale" o "verticale":

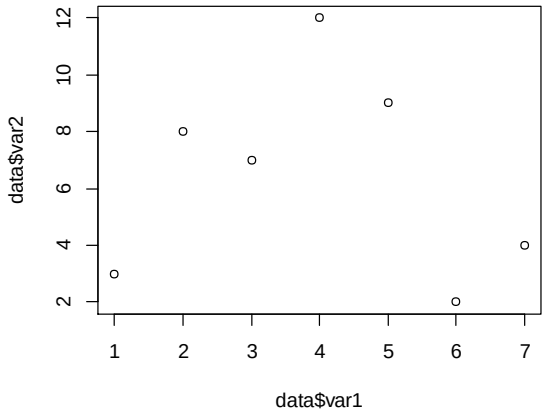
plot(nomeoggetto, rep(1,length(nomeoggetto)))



plot(rep(1,length(nomeoggetto)), nomeoggetto)



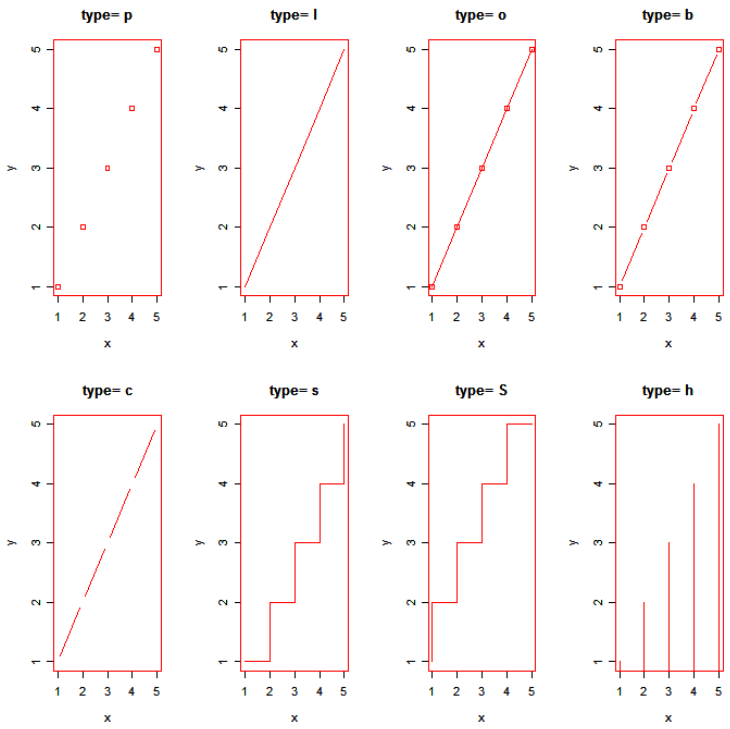
Grafici a punti



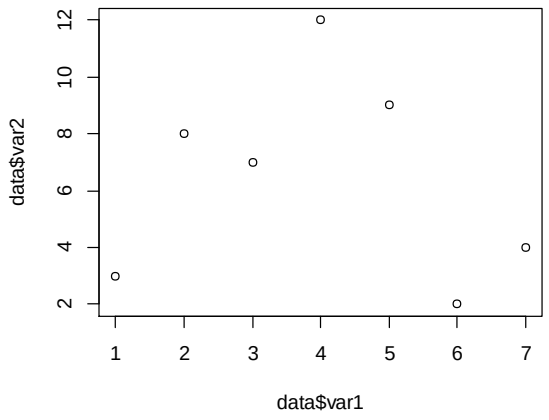
plot(nomeoggetto, nomeoggetto, type="p")

X **Y**

p = punti;
l = linee;
o = linee+punti;
b = linee interrotte in corrispondenza dei punti
c = linee interrotte in corrispondenza dei punti (i punti non si vedono)
s = linee che collegano i punti a scalino
S = linee che collegano i punti a scalino (altro tipo)
h = linee verticali da asse x a punto, tipo istogramma
n = si crea il grafico ma senza i punti



Argomenti opzionali per i Grafici a punti



plot(nomeoggetto, nomeoggetto, type="p",)

- **X**
- **Y**

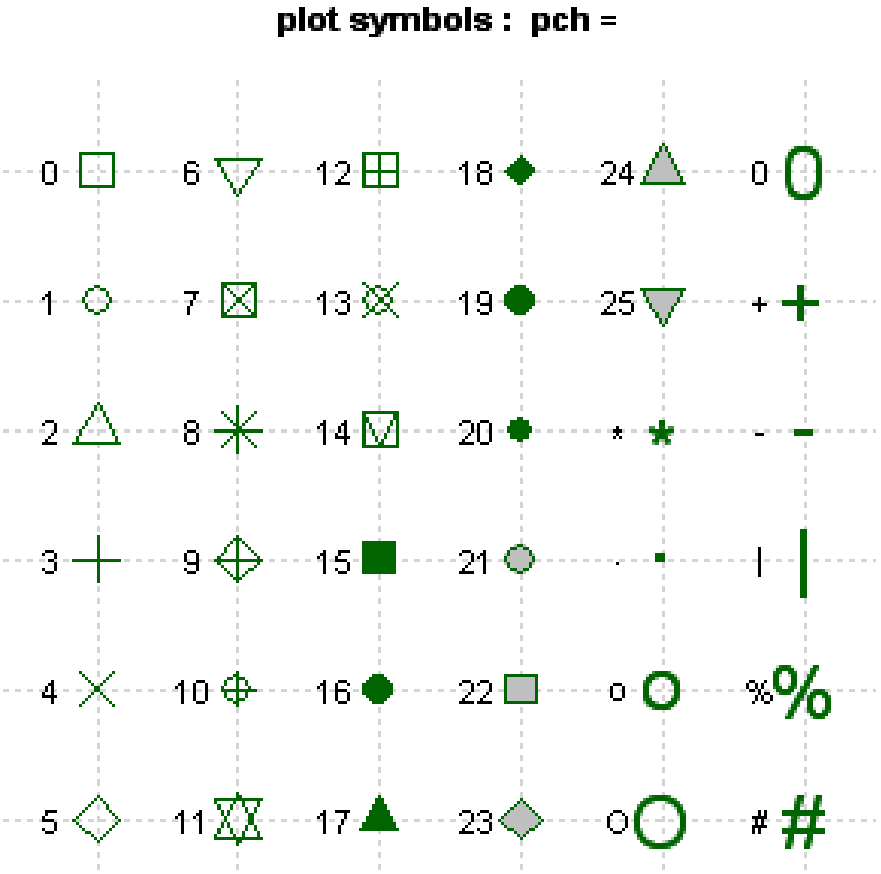
dopo la virgola si possono aggiungere altri argomenti (tutti separati da una virgola)

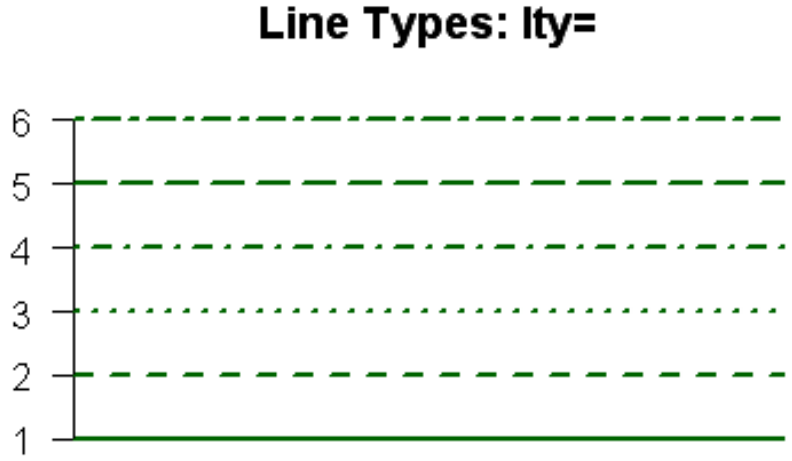
xlab ="Asse X"	→ titolo asse X (*label*)
ylab ="Asse Y"	→ titolo asse Y (*label*)
main ="Titolo"	→ titolo del grafico
cex.lab =0.75	→ grandezza titoli assi
cex.axis =0.65	→ grandezza etichette assi
cex.main =1.5	→ grandezza titolo grafico
col.lab ="green"	→ colore titoli assi
col.axis ="purple"	→ colore etichette assi
col.main ="red"	→ colore titolo grafico

xlim =c(0,8)	→ limiti asse X
ylim =c(0,12)	→ limiti asse Y
pch =16	→ tipo di punto
col ="red"	→ colore del punto (o della linea)
cex = 0.75	→ grandezza punto
bg ="blue"	→ riempimento del punto (solo per tipi da 21 a 25)
lty = 2	→ tipo di linea (*line type*)
lwd = 1.2	→ spessore della linea (*line width*)

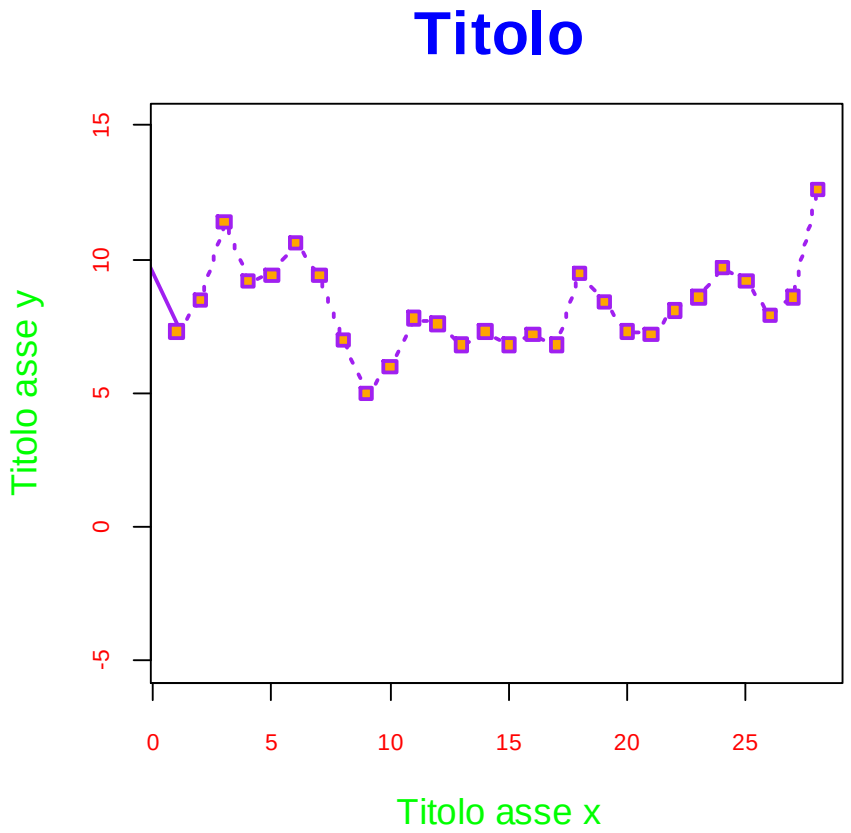
Inoltre ponendo l'argomento **log=** a **"x"**, **"y"** o **"xy"** uno o entrambi gli assi vengono disegnati in scala logaritmica (funziona anche in altri tipi di grafici, es. boxplot)

Tipi di punti e di linee





Migliorare l'estetica dei Grafici a punti con l'uso degli altri argomenti



plot(nomeoggetto, nomeoggetto, type="o", ylim=c(-5,15), xlim=c(1,28), xlab="Titolo asse x", ylab="Titolo asse y", main="Titolo", lwd=2, lty=3, pch=22, col="purple", bg="orange", col.axis="red", col.main="blue", col.lab="green", cex.main=2, cex.lab=1.2, cex.axis=0.75)

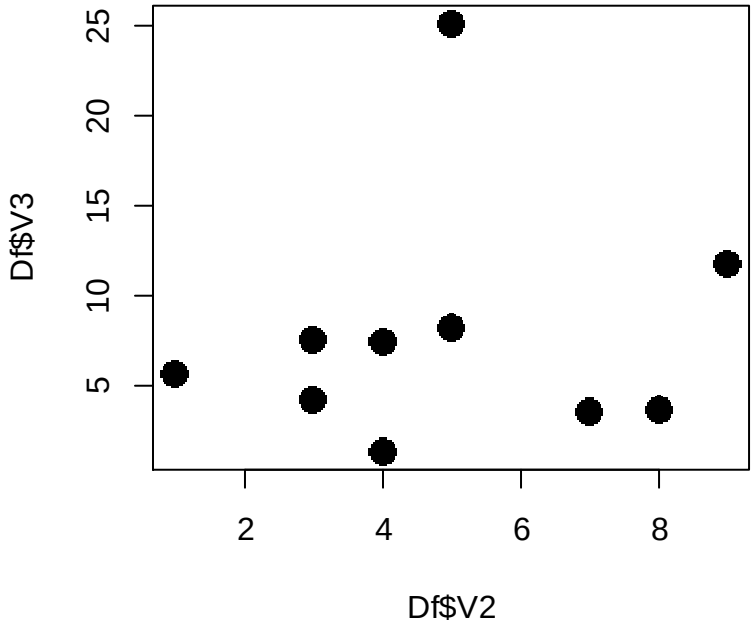
L'ordine degli argomenti dopo "type" è INDIFFERENTE!!!

Nello script si può andare a capo nella scrittura, basta poi selezionare tutto il comando per eseguirlo

Generare un vettore di colori per categoria da utilizzare per l'argomento col= nei grafici

```
> Df
   V1 V2   V3
1   a  1  5.6
2   b  5  8.2
3   c  4  7.4
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
7   c  4  1.3
8   d  9 11.7
9   a  5 25.1
10  b  3  7.5
```

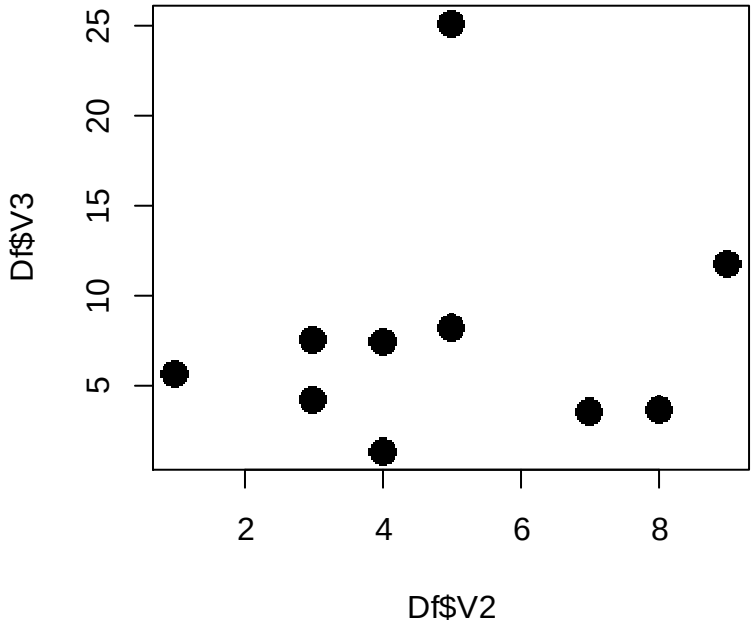
```
> plot(Df$V2,Df$V3)
```



Generare un vettore di colori per categoria da utilizzare per l'argomento col= nei grafici

```
> Df
   V1 V2   V3
1   a  1  5.6
2   b  5  8.2
3   c  4  7.4
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
7   c  4  1.3
8   d  9 11.7
9   a  5 25.1
10  b  3  7.5
```

```
> plot(Df$V2,Df$V3)
```



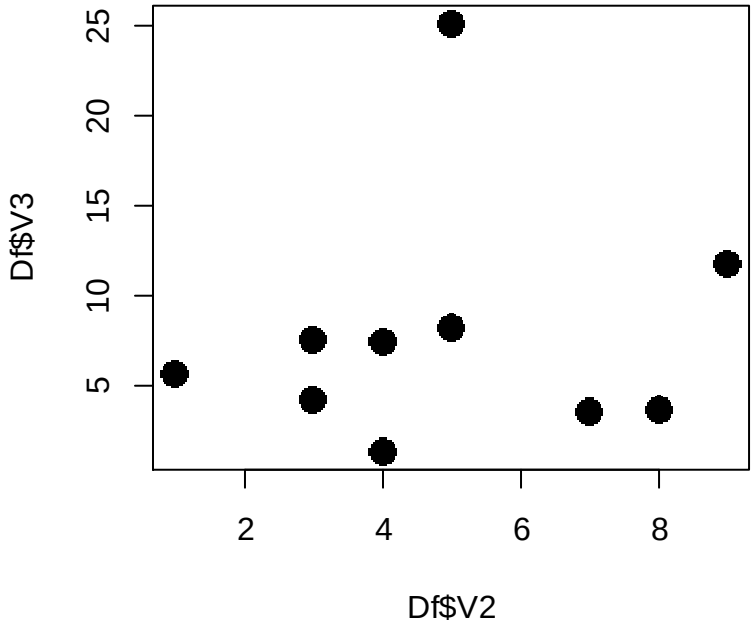
(1)

```
> Color<-as.factor(Df$V1)
> levels(Color)
[1] "a" "b" "c" "d"
```

Generare un vettore di colori per categoria da utilizzare per l'argomento col= nei grafici

```
> Df
   V1 V2   V3
1   a  1  5.6
2   b  5  8.2
3   c  4  7.4
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
7   c  4  1.3
8   d  9 11.7
9   a  5 25.1
10  b  3  7.5
```

```
> plot(Df$V2,Df$V3)
```



(1)

```
> Color<-as.factor(Df$V1)
> levels(Color)
[1] "a" "b" "c" "d"
```

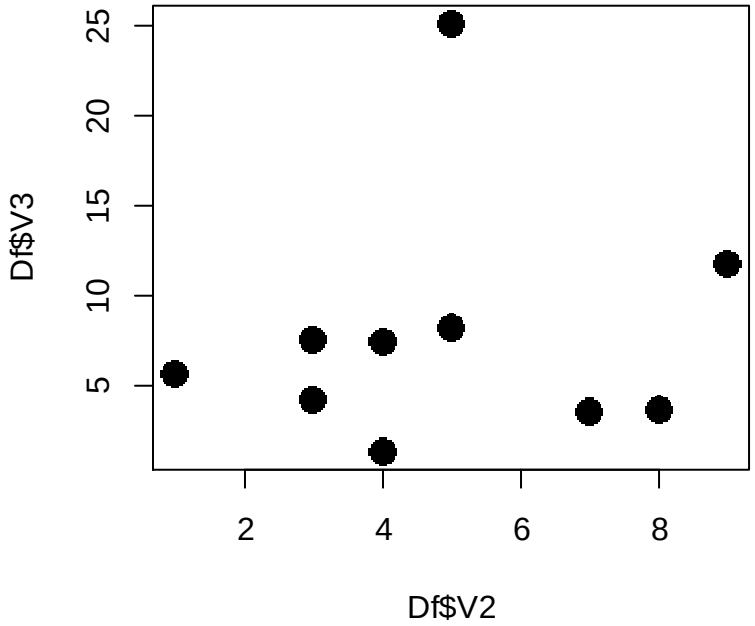
(2)

```
> levels(Color)<-c("red","blue","green","purple")
> Color
 [1] red    blue   green  purple red    blue   green  purple red    blue
Levels: red blue green purple
```

Generare un vettore di colori per categoria da utilizzare per l'argomento col= nei grafici

```
> Df
   V1 V2   V3
1   a  1  5.6
2   b  5  8.2
3   c  4  7.4
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
7   c  4  1.3
8   d  9 11.7
9   a  5 25.1
10  b  3  7.5
```

```
> plot(Df$V2,Df$V3)
```



(1)
```
> Color<-as.factor(Df$V1)
> levels(Color)
[1] "a" "b" "c" "d"
```

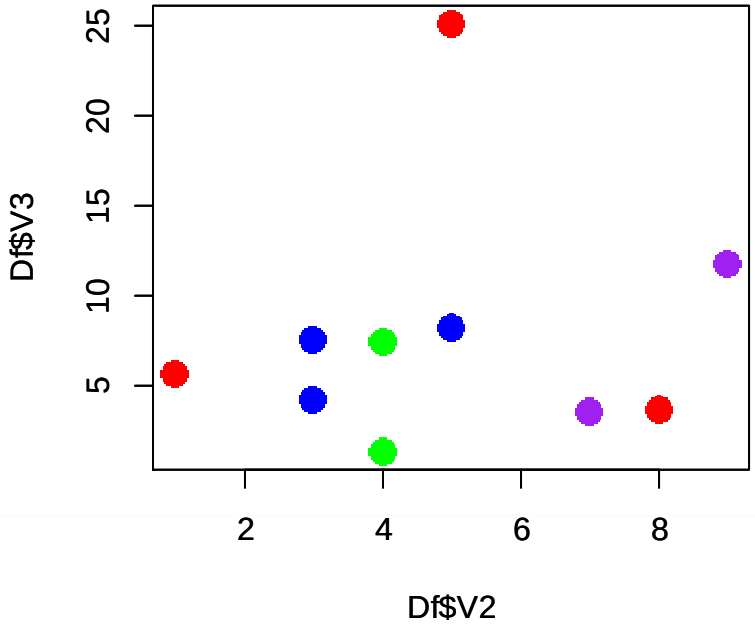
(2)
```
> levels(Color)<-c("red","blue","green","purple")
> Color
 [1] red    blue   green  purple red    blue   green  purple red    blue
Levels: red blue green purple
```

(3)
```
> Color<-as.character(Color)
> Color
 [1] "red"    "blue"   "green"  "purple" "red"    "blue"   "green"  "purple" "red"    "blue"
```

Generare un vettore di colori per categoria da utilizzare per l'argomento col= nei grafici

```
> Df
   V1 V2   V3
1   a  1  5.6
2   b  5  8.2
3   c  4  7.4
4   d  7  3.5
5   a  8  3.6
6   b  3  4.2
7   c  4  1.3
8   d  9 11.7
9   a  5 25.1
10  b  3  7.5
```

```
> plot(Df$V2,Df$V3)
```



(1)
```
> Color<-as.factor(Df$V1)
> levels(Color)
[1] "a" "b" "c" "d"
```

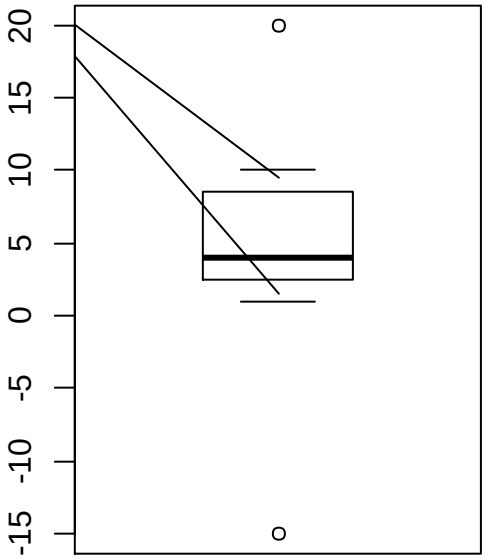
(2)
```
> levels(Color)<-c("red","blue","green","purple")
> Color
 [1] red    blue   green  purple red    blue   green  purple red    blue
Levels: red blue green purple
```

(3)
```
> Color<-as.character(Color)
> Color
 [1] "red"    "blue"   "green"  "purple" "red"    "blue"   "green"  "purple" "red"    "blue"
>
```

(4)
```
> plot(Df$V2,Df$V3,col=Color,pch=16,cex=2)
```

Grafici a scatola (boxplot)

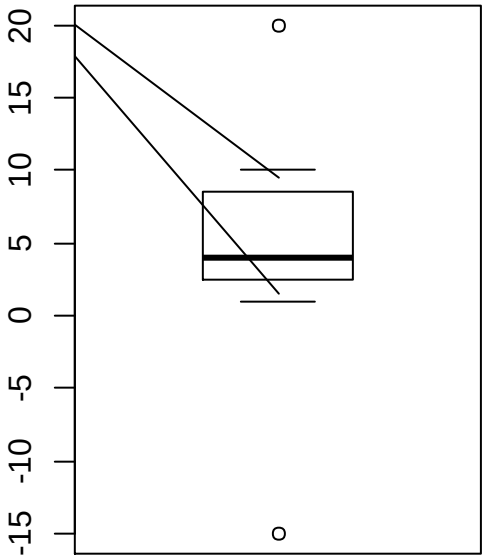
Configurazione "minima" del comando:



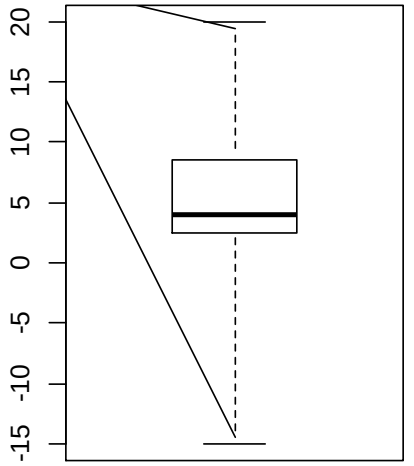
boxplot(nomeoggetto)

Grafici a scatola (boxplot)

Configurazione "minima" del comando:



boxplot(nomeoggetto)



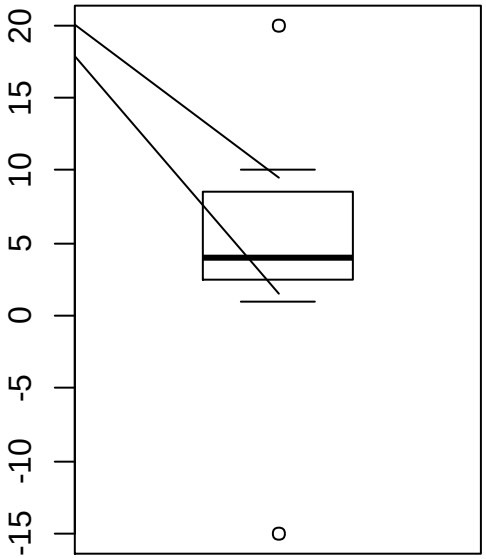
boxplot(nomeoggetto, range=0)

↓

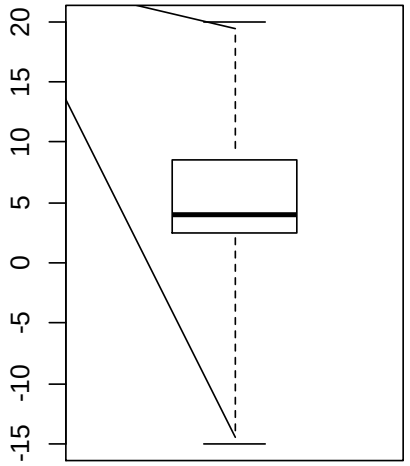
i "baffi" sono il min e il max

Grafici a scatola (boxplot)

Configurazione "minima" del comando:

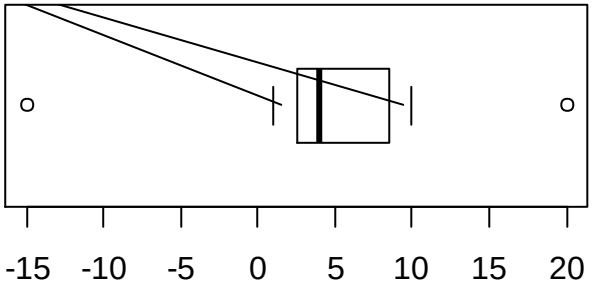


boxplot(nomeoggetto)

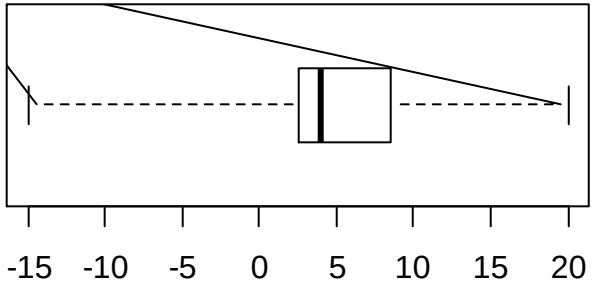


boxplot(nomeoggetto, range=0)

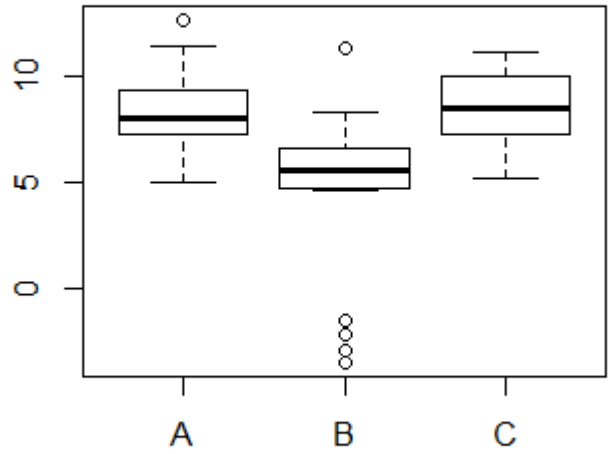
i "baffi" sono il min e il max



aggiungendo alla funzione
l'argomento **horizontal = TRUE**



Grafici a scatola (boxplot)



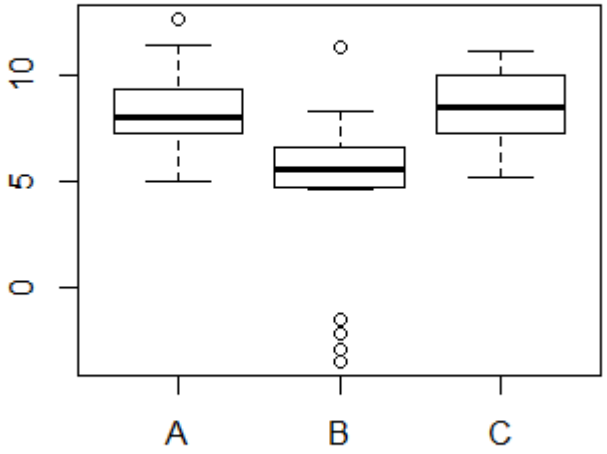
boxplot(nomeoggetto)

↓

è una matrice o un dataframe o un elenco di vettori separati da una virgola

boxwex = 0.9	→ determina la larghezza del box (per estetica grafico)
varwidth = FALSE	→ se TRUE la larghezza del box è proporzionale alla radice quadrata delle osservazioni da cui è stato costruito
notch = FALSE	→ se TRUE un "intaglio" (notch) è disegnato sul box (vedi help)
log =...	→ indica se disegnare un asse ("x" o "y") o entrambi ("xy") devono essere disegnati in scala logaritmica
whisklty =1	→ se = 0 i "baffi" non vengono disegnati
staplelty =1	→ se = 0 la riga alla fine dei "baffi" non viene disegnata
outline =TRUE	→ se FALSE gli outlier non vengono disegnati

Argomenti opzionali per i Grafici a scatola



boxplot(nomeoggetto,)

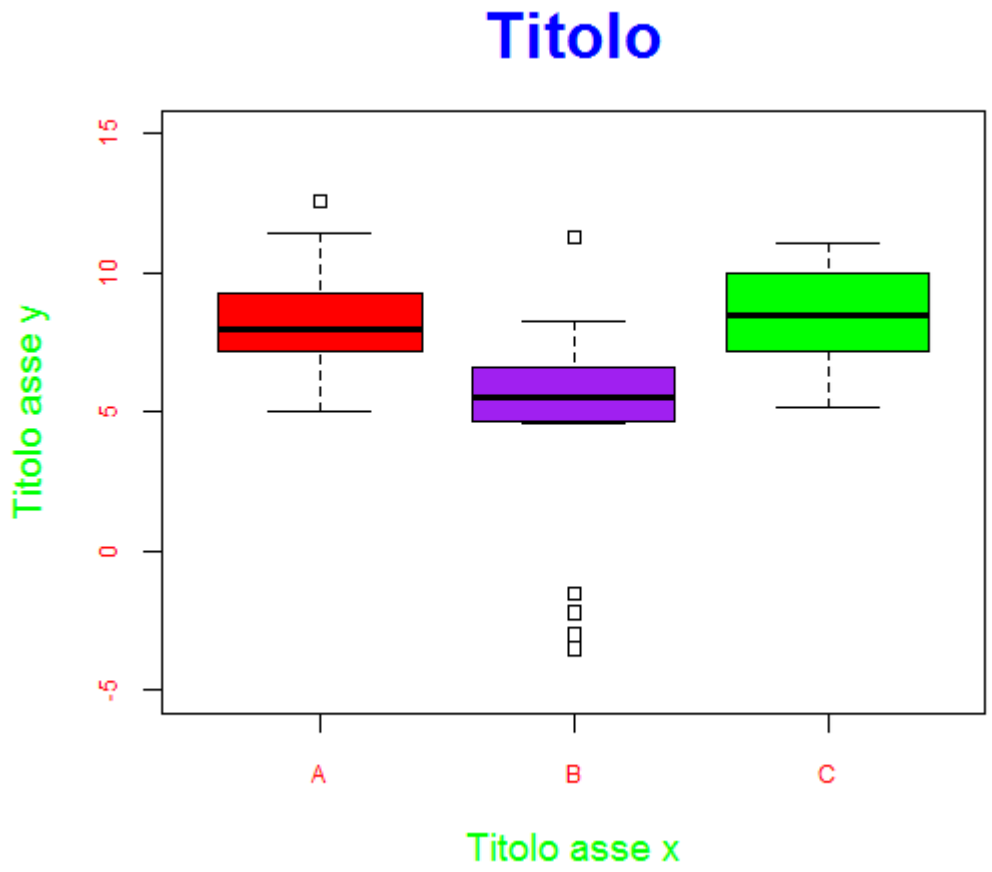
dopo la virgola si possono aggiungere altri argomenti (tutti separati da una virgola)

xlab ="Asse X" → titolo asse X (*label*)
ylab ="Asse Y" → titolo asse Y (*label*)
main ="Titolo" → titolo del grafico
cex.lab =0.75 → grandezza titoli assi
cex.axis =0.65 → grandezza etichette assi
cex.main =1.5 → grandezza titolo grafico
col.lab ="green" → colore titoli assi
col.axis ="purple" → colore etichette assi
col.main ="red" → colore titolo grafico

ylim =c(0,12) → limiti asse Y
pch =16 → tipo di punto *outlier*
col ="red" → colore box

può essere anche un vettore che indica un colore diverso per ogni box, es. c("red","blue","green")

Migliorare l'estetica dei Grafici a scatola con l'uso degli altri argomenti

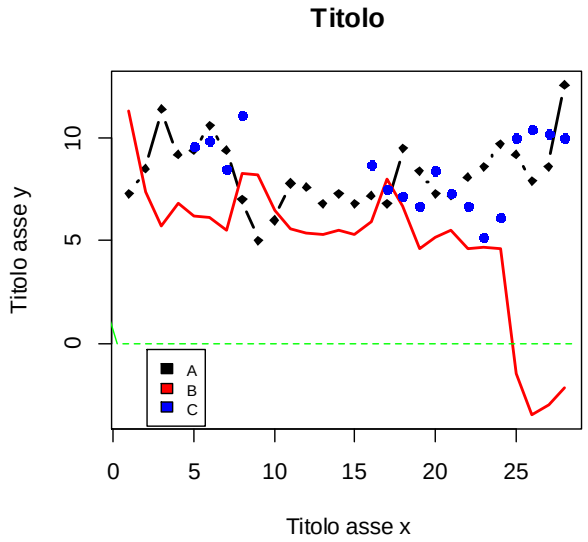


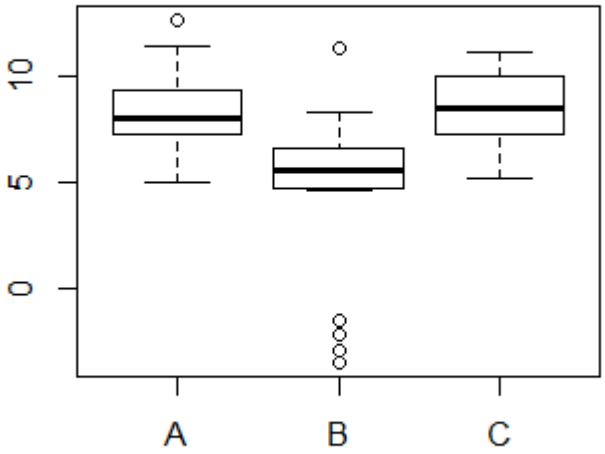

```
boxplot(nomeoggetto, ylim=c(-5,15),  xlab="Titolo asse x", ylab="Titolo asse y", main="Titolo",
pch=22, col=c("red","purple","green"), col.axis="red", col.main="blue", col.lab="green",
cex.main=2, cex.lab=1.2, cex.axis=0.75)
```

Altri elementi dei grafici

Grafici a punti e/o linee

Grafici a scatola (boxplot)





Aggiunta di elementi ai grafici

- **punti e linee**
- **testo**
- **legenda**

Aggiungere elementi ad un grafico già disegnato

(1) Punti e linee

Funzione usata ad es. per aggiungere serie ad un Grafico XY

points(nomeoggetto, nomeoggetto, type="p", argomenti)

- 1° nomeoggetto → **X**
- 2° nomeoggetto → **Y**
- type="p", argomenti → come per la funzione **plot**

Argomenti:

pch =16	→ tipo di punto
col ="red"	→ colore del punto (o della linea)
cex = 0.75	→ grandezza punto
bg ="blue"	→ riempimento del punto (solo per tipi da 21 a 25)
lty = 2	→ tipo di linea
lwd = 1.2	→ spessore della linea (*line width*)

SEGUE

(2) Linee orizzontali e verticali

abline(h=numero, argomenti)

↓

h → per linee orizzontali (*horizontal*)
v → per linee verticali (*vertical*)

oppure per disegnare una retta y=mx + q:

abline(a=q, b=m, argomenti)

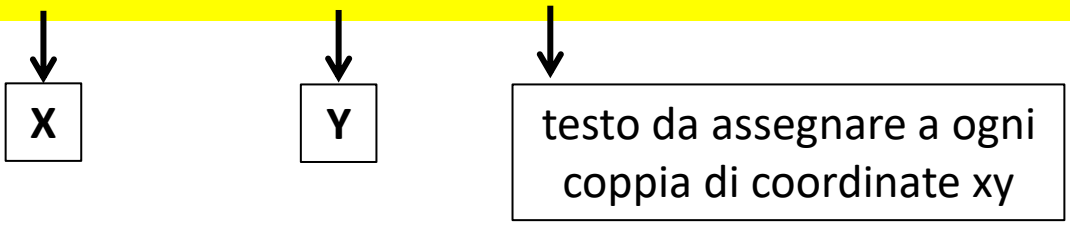
Argomenti:

col ="red"	→ colore del punto (o della linea)
lty = 2	→ tipo di linea
lwd = 1.2	→ spessore della linea (*line width*)

SEGUE

(3) Testo

text(nomeoggetto, nomeoggetto, label= vettoretesto, argomenti)



Argomenti:

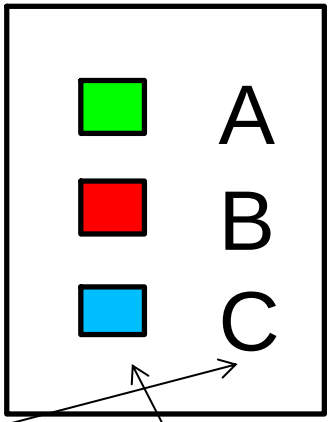
col ="red" → colore del testo
cex = 0.75 → grandezza del testo

SEGUE

(4) Legenda

legend(posizionenelgrafico, legend=vettorenomi, fill= vettorecolori)

- posizionenelgrafico:
  - "bottomright"
  - "bottom"
  - "bottomleft"
  - "left"
  - "topleft"
  - "top"
  - "topright"
  - "right"
  - "center"
- vettorenomi: nomi delle serie



legend("left", **legend**=c("A","B","C"), **fill**=c("green","red","deepskyblue"))

(5) Assi aggiuntivi

axis(side, argomenti)

1= in basso
2 = a sinistra
3 = in alto
4 = a destra

Diverse specifiche per indicare in che posizioni si trovano le divisioni (*tickmarks*) dell'asse ed eventuale testo/numeri per ognuna di esse

(6) Titoli aggiuntivi assi

mtext("testo", side, argomenti)

Diverse specifiche, ad es. per indicare su che "linea" porre il testo

Queste funzioni sono utili quando si vogliono **personalizzare gli assi**, per cui nelle funzioni che disegnano il grafico vengono eliminate

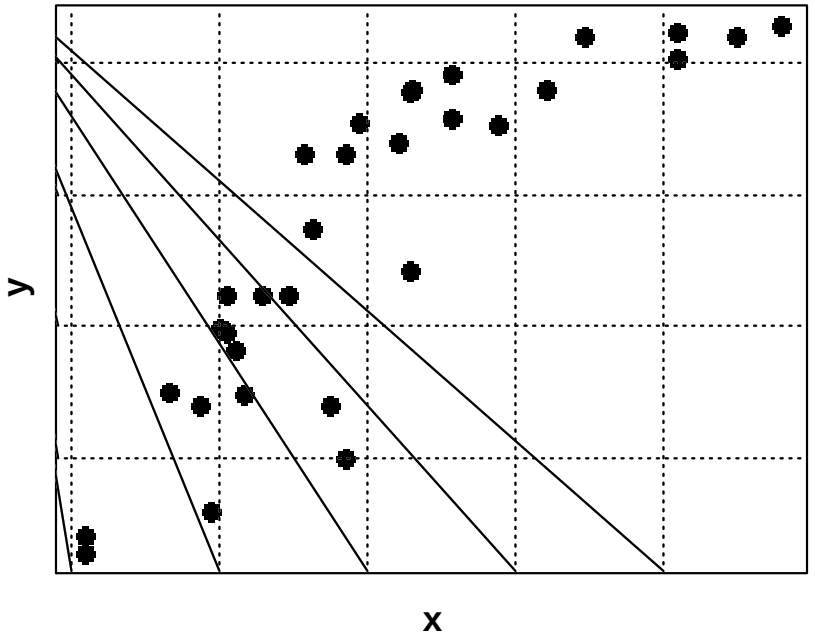
(i) la generazione automatica degli assi (con xaxt= "n" e yaxt= "n") e

(ii) le etichette degli assi (con xlab="" e ylab="")

(iii) si aggiungono novi assi ed etichette con **axis()** e **mtext()**

(7) Griglia

grid(argomenti) → Diverse specifiche, ad es. colore, tipo di linea, "passo" della griglia, etc..

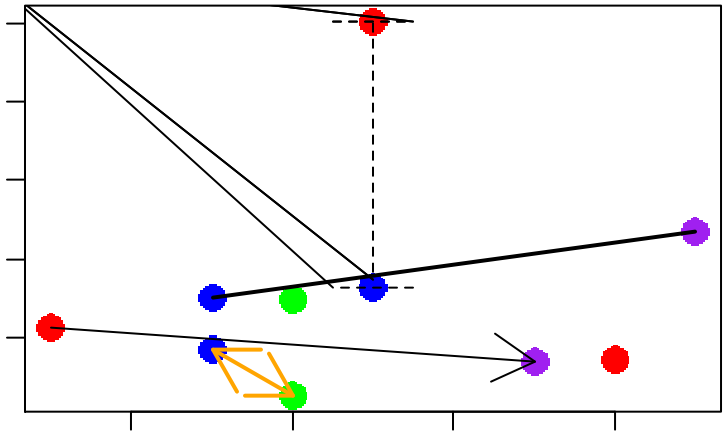


(8) Frecce

arrows(x_0,y_0,x_1,y_1,argomenti) → Permette di aggiungere delle frecce al grafico

coordinate punto iniziale

coordinate punto finale



Argomenti:

code=...	→	tipo di freccia (1-la freccia sul punto iniziale; 2-la freccia sul punto finale; 3-la freccia su entrambi i punti)
length= ...	→	lunghezza dei lembi della freccia (unità di misura = pollici); se =0 non vengono disegnate frecce, solo la linea
angle= ...	→	angolo tra la linea i lembi della freccia
col = ...	→	colore
lwd = ...	→	grossezza della linea
lty= ...	→	tipo di linea

La funzione locator

locator(n) → **numero di punti da individuare nel grafico**

Questa funzione serve ad individuare punti con un click del mouse all'interno dell'area di un grafico già disegnato. Restituisce le coordinate dei punti cliccati.

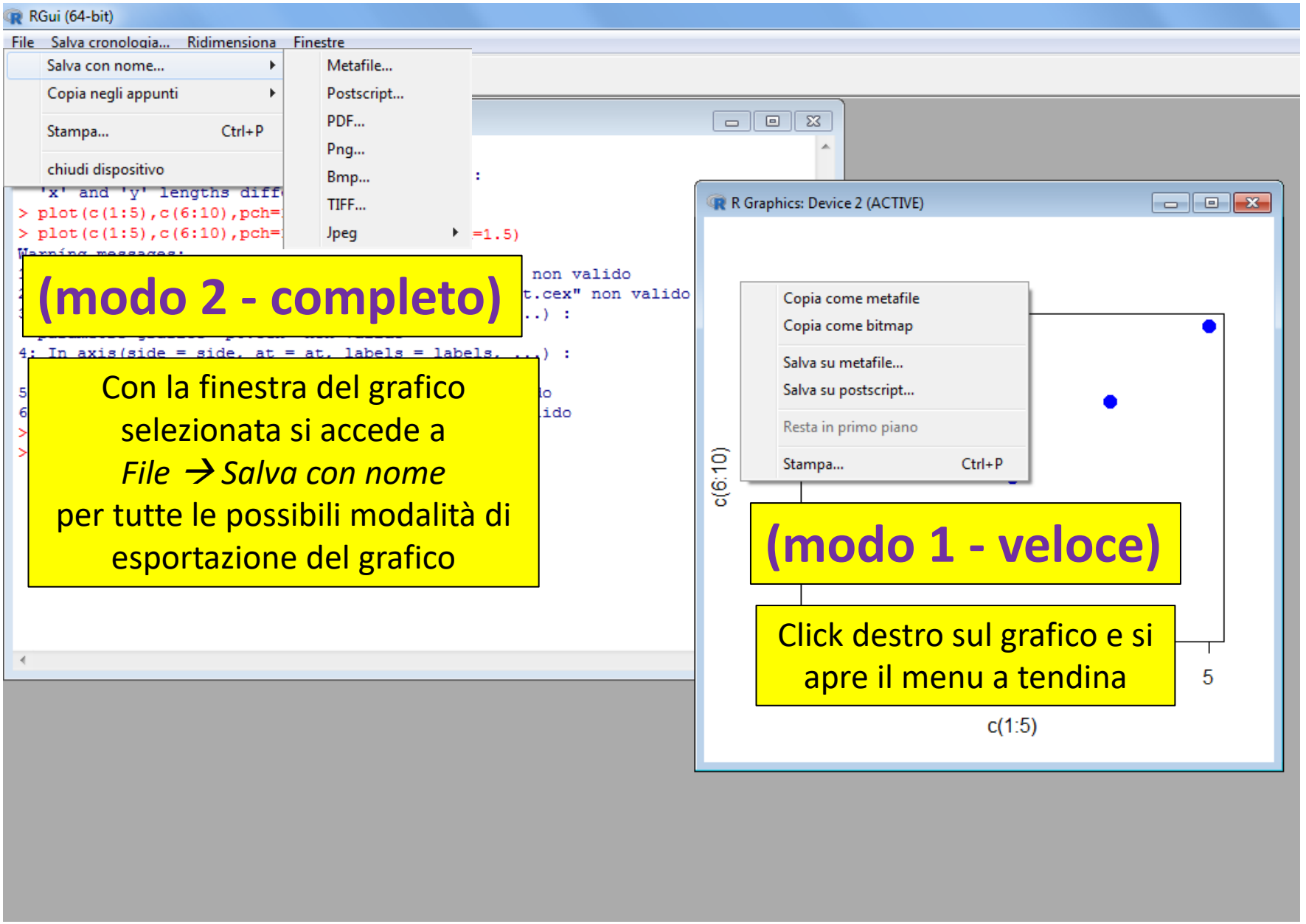
Può essere utilizzata all'interno di altre funzioni come **points**, **text** e **legend** al posto dell'inserimento di coordinate xy.

points(locator(n), type="p", argomenti)

text(locator(n), label= vettoretesto, argomenti)

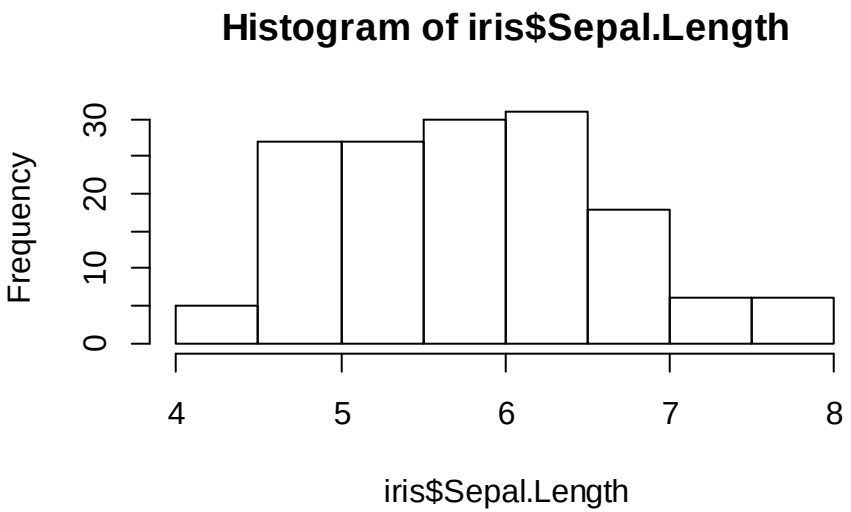
legend(locator(1), legend=vettorenomi, fill= vettorecolori)

Salvare/Esportare un grafico da R

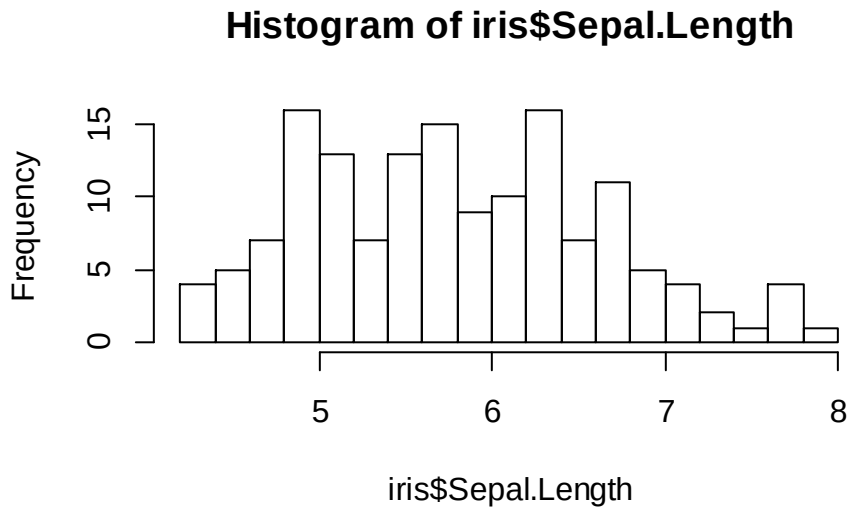


Istogrammi

Configurazione "minima" del comando:



hist(nomeoggetto)



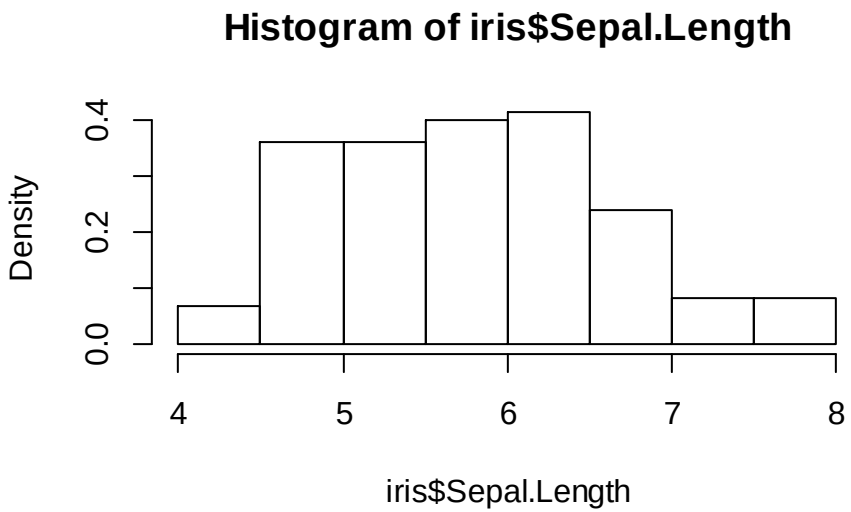
hist(nomeoggetto, breaks=20)

↓

per cambiare il numero di intervalli (*bins*)

SEGUE

Istogrammi (2)



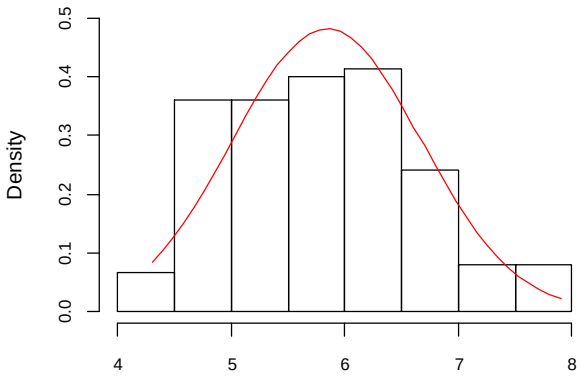
hist(nomeoggetto, freq=FALSE)

↓

per visualizzare la densità di probabilità sull'asse y

Per sovrapporre al grafico una funzione gaussiana di densità di probabilità eseguire:

```
xfit<-seq(min(nomeoggetto),max(nomeoggetto),length=100)
yfit<-dnorm(xfit,mean=mean(nomeoggetto),sd=sd(nomeoggetto))
hist(nomeoggetto, freq=FALSE)
lines(xfit, yfit, col="red")
```



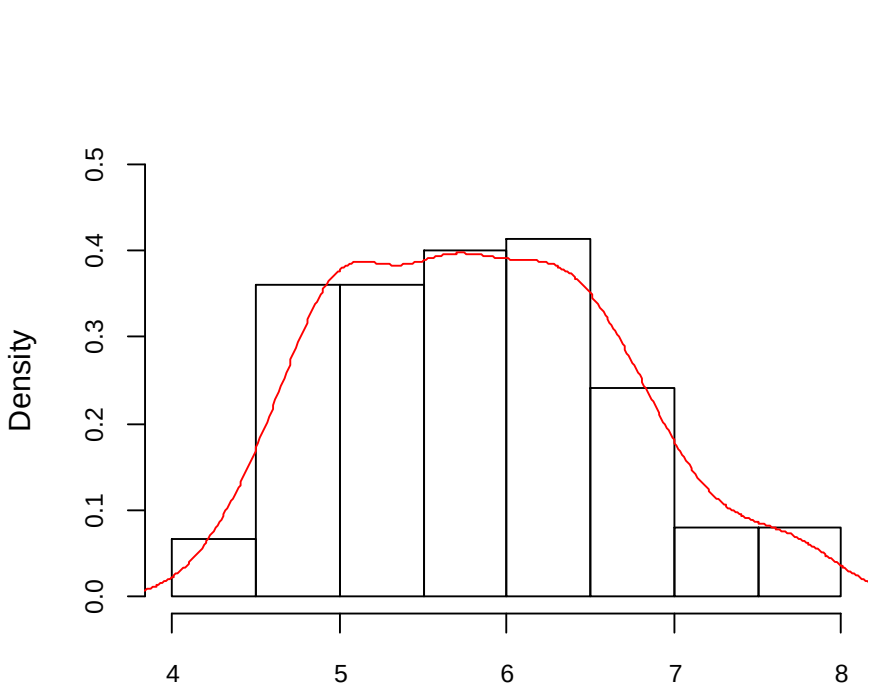
SEGUE

Istogrammi (3)

Per sovrapporre al grafico una funzione kernel di densità di probabilità eseguire:

```
d <- density(nomeoggetto)
hist(nomeoggetto, freq=FALSE)
lines(d, col="red")
```

Nella funzione **density**
si possono scegliere diversi valori di ampiezza di banda e diversi tipi di funzione per le "gobbe" con gli argomenti
bw = e **kernel =**



cambio bw

cambio kernel