Laborarator 5 – Probabilități și Statistică Matematică

ELEMENTE DE GRAFICA IN R

Graficele reprezintă, în general, un instrument folositor pentru vizualizarea rezultatelor unei analize dar și un mod prin care se poate verifica dacă am obținut ce ni s-a cerut. Este important să folosim opțiunile corecte atunci când trasăm un grafic, de multe ori valorile predefinite putând conduce la figuri eronate. În tabelul de mai jos avem listate principalele instrucțiuni grafice

Tabelul 8. Functii grafice uzuale in R

Funcția	**Scurtă descriere**
`plot`	O funcție generică folosită pentru plotarea unui obiect (puncte, curbe, etc.)
`barplot`	Trasează un grafic de bare orizontale sau verticale (folosit pentru variabile discrete)
`hist`	Desenează histograme cu frecvanțe sau probabilități
`boxplot`	Desenează una sau mai multe boxplot-uri în paralel
`pie`	Desenează o diagramă de tip plăcintă

Metodele grafice din R nu se limitează la cele din tabelul de mai sus. De exemplu, pachetul `ggplot2`[1] este foarte răspândit la ora actuală deoarece pune la dispoziție o serie de instrumente grafice (folosind o gramatică de grafice) cu ajutorul cărora se pot produce figuri de calitate deosebită (ce se folosesc în special pentru publicații științifice).

În continuare vom prezenta o parte din funcțiile cel mai des folosite pentru trasarea graficelor în R împreună cu proprietățile de bază ale acestora.

5.1 Funcția `plot`

Cea mai folosită funcție (high-level) de plotare în R este funcția `plot()`. Aceasta permite trasarea unei diagrame de împrăștiere (*scatterplot*) între doi vectori `x` și `y`, unde vectorul `x` indică valorile pe axa abscisei iar `y` pe axa ordonatelor.

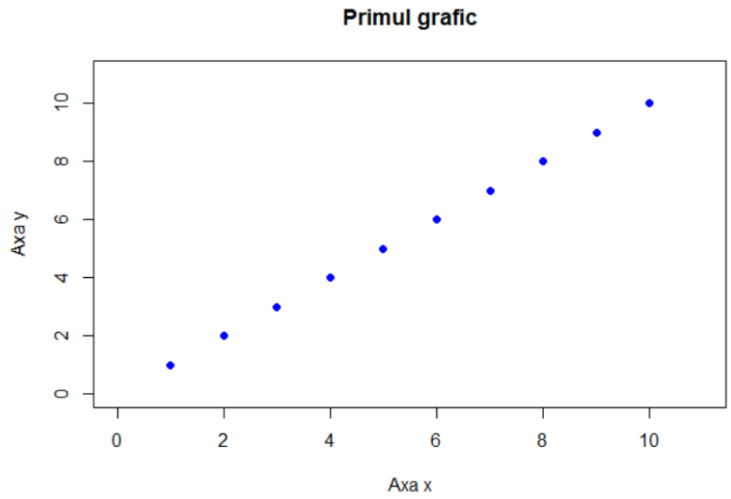
Argumentele principale ale funcției `plot()` se găsesc în tabelul următor.

Tabelul 9. Argumentele principale ale functiei plot()

Argument	Descriere
`x, y`	Vectori de aceeași lungime care specifică valorile coordonatelor de pe x și de pe y
`type`	Tipul de grafic: `"l"` reprezintă linii, `"p"` reprezintă puncte, `"b"` reprezintă și linii și puncte, `"n"`înseamnă că nu trasează nimic
`main`, `xlab`, `ylab`	String-uri folosite pentru titlul graficului și etichetarea axelor x și y
`xlim`, `ylim`	Limitele pe axa x și y. De exemplu, `xlim = c(0, 100)` va seta valoarea minimă și maximă de pe axa x la 0 și respectiv 100.
`pch`	Un număr întreg care denotă tipul de simbol folosit pentru trasarea punctelor (vezi `?points`), sau un șir de caractere care specifică simbolurile ca text. De exemplu, `pch = 21` va crea un cerc colorat cu 2 culori, iar `pch = "P"` va trasa caracterul `"P"` pentru fiecare punct.
`col`	Culoarea principală a simbolurilor desenate. De exemplu `col = "blue"` va trasa simbolurile în culoarea albastră.
`lty`	Tipul de linie folosit. De exemplu `lty = 1` reprezintă o linie solidă pe când `lty = 2` reprezintă o linie întreruptă.
`lwd`	Grosimea liniei folosite, valoare prestabilită este `lwd = 1`.
`cex`	Un vector numeric folosit pentru a specifica mărimea simbolurilor trasate. Valoarea prestabilită este 1. De exemplu `cex = 4` va face punctele foarte mari pe când `cex = .5` le va face mai mici.

```
plot(x = 1:10,                          # x-coordonate
     y = 1:10,                          # y-coordinate
     type = "p",                        # puncte (nu linii)
     main = "Primul grafic",
     xlab = "Axa x",
     ylab = "Axa y",
     xlim = c(0, 11),                   # Valorile min si max pe
axa x
     ylim = c(0, 11),                   # Valorile min si max pe
axa y
```

```
     col = "blue",                    # Culoarea punctelor
     pch = 16,                        # Tipul simbolului
     cex = 1)                          # Marimea simbolului
```



În afară de vectorii `x` și `y` toate celelalte argumente sunt opționale, dacă nu sunt specificate atunci R-ul folosește valorile prestabilite. De exemplu dacă nu specificăm limitele `xlim` și `ylim`, R-ul calculează aceste limite astfel încât toate punctele să fie încadrați în interiorul graficului.

▶ Trasați următoarele diagrame de împrăștiere:

```
x <- seq(0, 1, 0.1); plot(x, x - x * x + 2)plot(x, x - x * x +
2, type = "l"); plot(x, x - x * x + 2, type = "b", pch = 19)
```

Tipul de simbol pe care vrem să-l folosim atunci când trasăm un grafic este specificat prin argumentul `pch`. Figura de mai jos ne prezintă tipurile simboluri pe care atunci când atribuim argumentului `pch` o valoare întreagă.

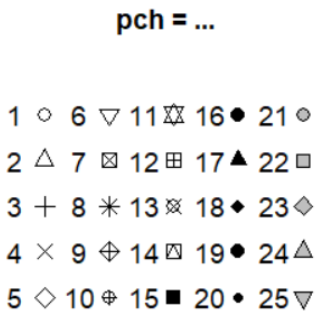
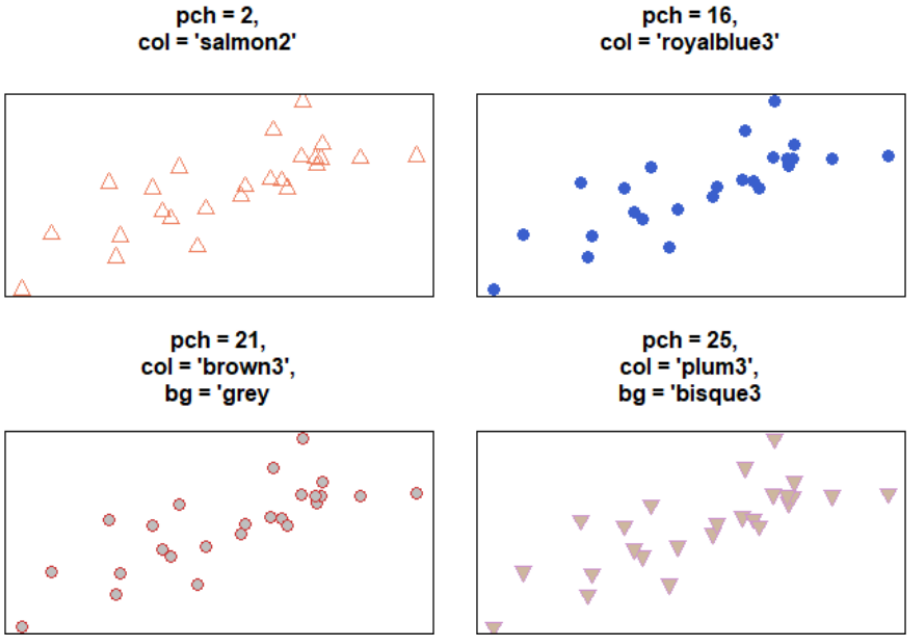


Figura 4. Tipurile de simboluri asociate parametrului pch.

Următorul exemplu ilustrează câteva tipuri de simboluri:



- Considerăm următoarea funcție $g:\mathrm{R}\to\mathrm{R}$,

$$g(x)=\begin{cases} sin^2 x log(x), & x>0 \\ sin^2(x)x, & x\le 0 \end{cases}$$

a. Definiți funcția folosind comenzile `if-else` și `Vectorize` iar apoi folosind comanda `ifelse`.

b. Trasați graficul curbei pe intervalul $[-\pi,\pi]$.

5.2 Culori

Majoritatea funcțiilor de desenare au un argument în care pot specifica culoarea elementelor pe care vrem să le trasăm, de cele mai multe ori acesta este `col`. Cel mai ușor mod de a introduce o culoare este prin specificarea numelui acesteia, de exemplu `col = 'red'` este culoarea roșie. Figura 1 prezintă 144 de culori alese la întâmplare din totalul de 657 câte există în R.

Figura 5. 144 de culori din totalul de 657 din R

Pentru a vedea toate culorile din R putem rula comanda `colors()`.

5.3 Funcția `hist`

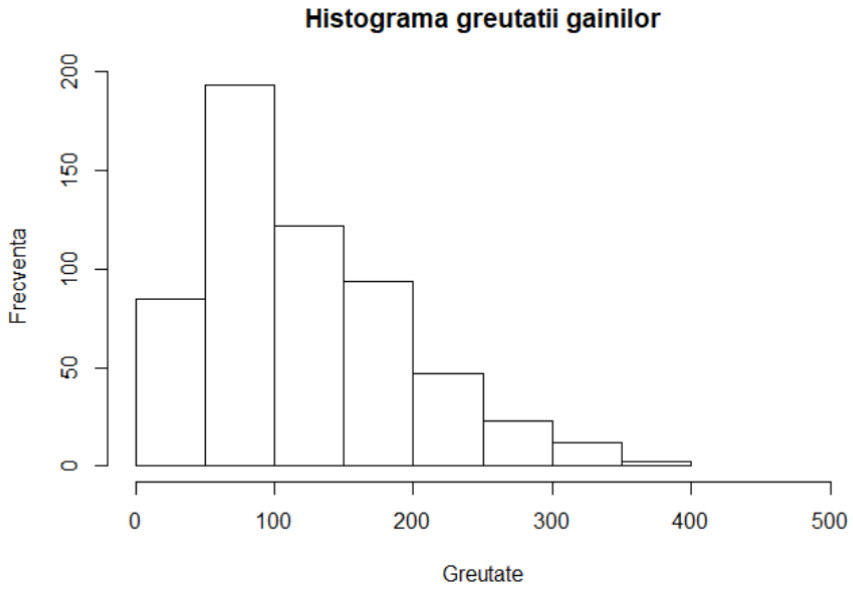
Histograma[2] reprezintă metoda grafică, cea mai comună, de reprezentare a repartiției unui vector numeric. Pentru a crea o histogramă în R folosim funcția `hist()` în care argumentul principal este un vector numeric. Tabelul de mai jos prezintă argumentele principale ale funcției `hist`.

Argumentele funcției `hist()`

Argument	Descriere
`x`	Vector de valori
`breaks`	Cum să calculăm mărimea bin-urilor (vezi `?hist`)
`freq`	Opțiune pentru trasarea histogramei de frecvență și de probabilități, `freq = TRUE` arată frecvențele, `freq = FALSE` arată probabilitățile.
`col`, `border`	Culoarea interioară a bin-urilor (`col`) și culoarea conturului lor (`border`)

Putem crea o histogramă folosind setul de date `ChickWeight` (`?ChickWeight`)

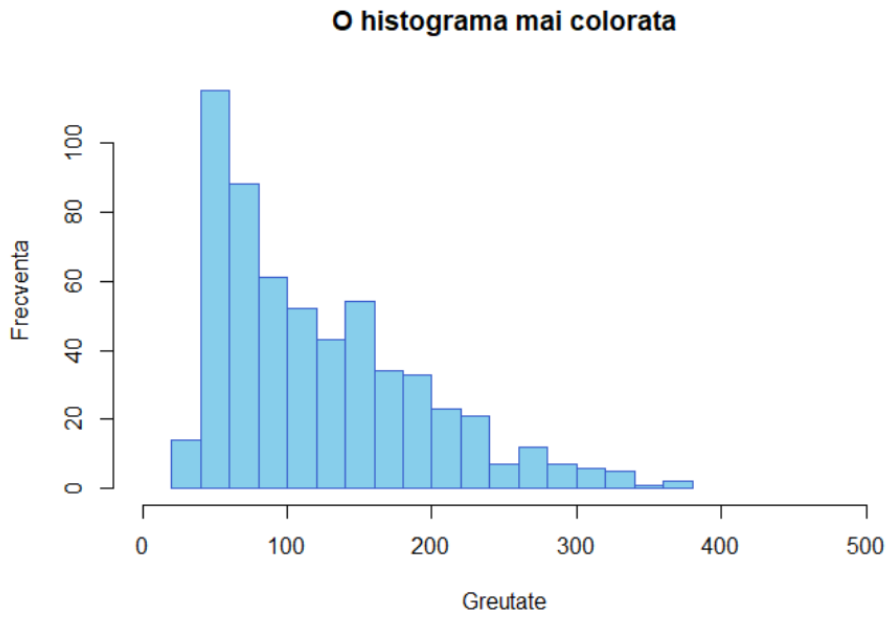
```
hist(x = ChickWeight$weight,
     main = "Histograma greutatii gainilor",
     xlab = "Greutate",
     ylab = "Frecventa",
     xlim = c(0, 500))
```



Putem modifica histograma de mai sus, schimbând numărul de bin-uri și culoarea acestora:

```
hist(x = ChickWeight$weight,
```

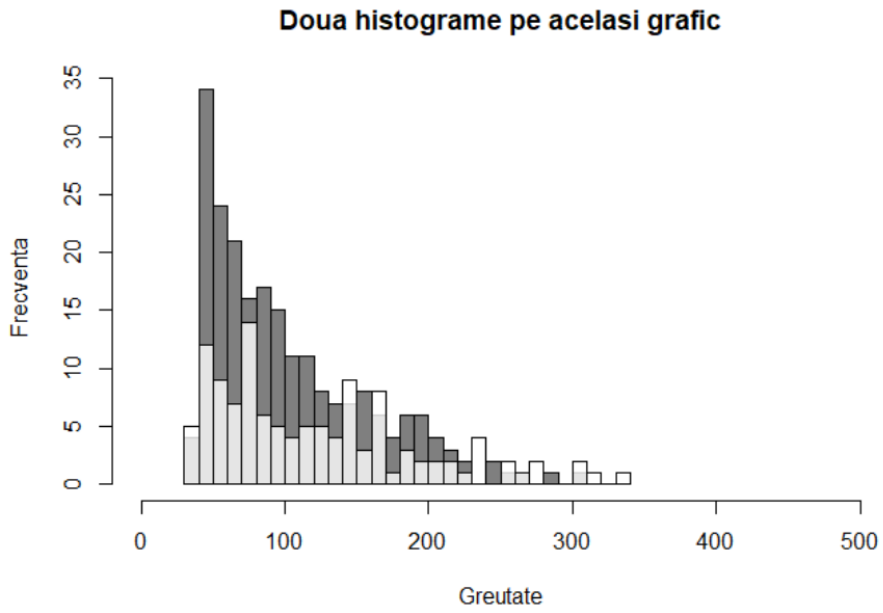
```
     main = "O histograma mai colorata",
     xlab = "Greutate",
     ylab = "Frecventa",
     breaks = 20, # 20 Bins
     xlim = c(0, 500),
     col = "skyblue", # Culoarea de umplere
     border = "royalblue3") # Culoarea conturului
```



Dacă vrem să ilustrăm două histograme pe aceeași figură, pentru a evidenția repartiția după două clase, putem folosi argumentul `add = TRUE` la cel de-al doilea plot:

```
hist(x = ChickWeight$weight[ChickWeight$Diet == 1],
     main = "Doua histograme pe acelasi grafic",
     xlab = "Greutate",
     ylab = "Frecventa",
     breaks = 20,
     xlim = c(0, 500),
```

```
      col = gray(0, .5))

hist(x = ChickWeight$weight[ChickWeight$Diet == 2],
      breaks = 30,
      add = TRUE, # Adauga graficul la cel de dinainte
      col = gray(1, .8))
```



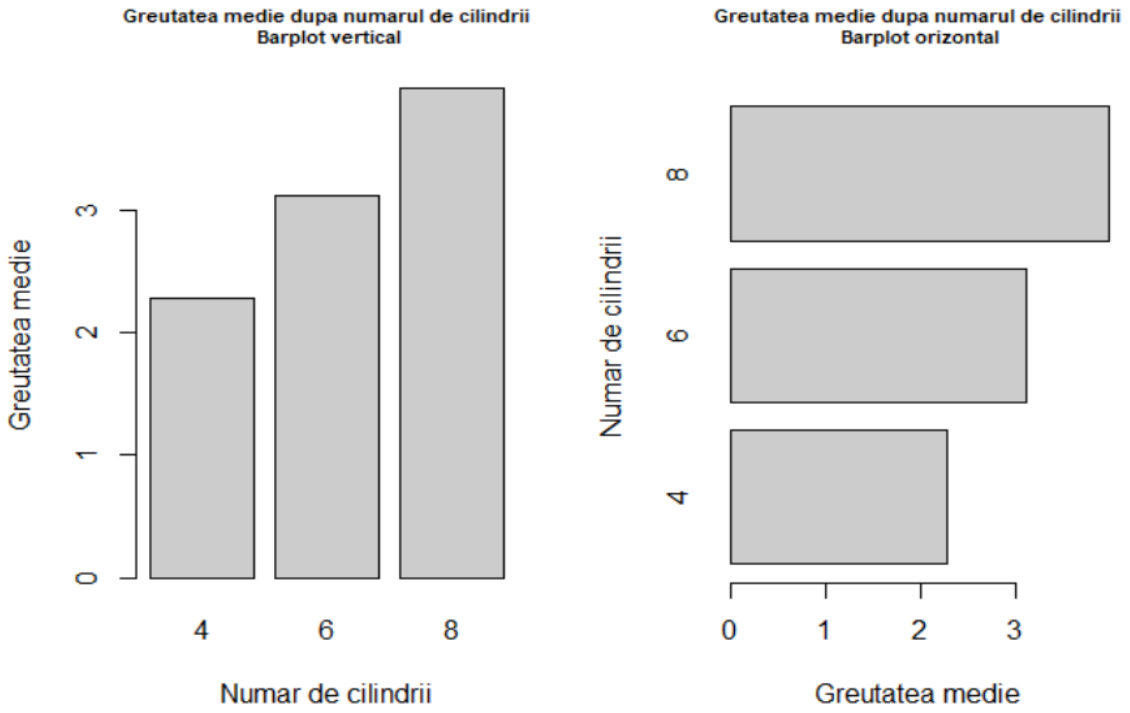
5.4 Funcția `barplot`

Funcția `barplot` este folosită în special atunci când avem de-a face cu o variabilă discretă. Argumentul principal al funcției este `height`, un vector numeric care va genera înălțimea fiecărei bare. Pentru a adăuga nume sub fiecare bară putem folosi argumentul `names.arg`.

De exemplu, folosind setul de date `mtcars` putem să afișăm greutatea medie a mașinilor în funcție de numărul de cilindrii:

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

```
weight_cars = aggregate(wt ~ cyl,
                        data = mtcars,
                        FUN = mean)

barplot(height = weight_cars$wt,
        names.arg = weight_cars$cyl,
        xlab = "Numar de cilindrii",
        ylab = "Greutatea medie",
        main = "Greutatea medie dupa numarul de cilindrii\n Barp
lot vertical",
        col = "grey80",
        cex.main = 0.7)

barplot(height = weight_cars$wt,
        names.arg = weight_cars$cyl,
        horiz = TRUE,
        ylab = "Numar de cilindrii",
        xlab = "Greutatea medie",
        main = "Greutatea medie dupa numarul de cilindrii\n Barp
lot orizontal",
        col = "grey80",
        cex.main = 0.7)
```

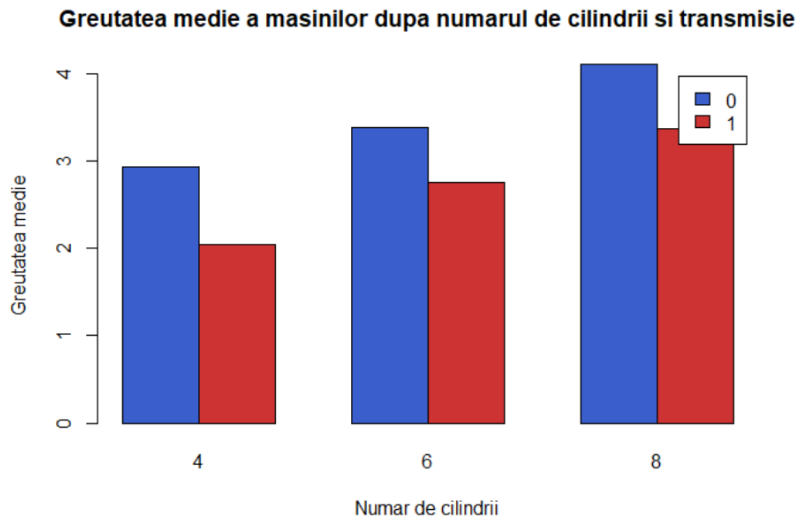


- Folosind setul de date `ChickWeight` afișați, cu ajutorul funcției `barplot`, greutatea medie a găinilor în raport cu numărul de zile de la naștere.

De asemenea putem crea un barplot clusterizat în funcție de mai multe grupuri de date. De exemplu să presupunem că vrem să vedem dacă există diferențe între greutatea medie a mașinilor (din setul de date `mtcars`) care au transmisie manuală sau automată și numărul de cilindrii.

```
# calculam greutatea medie dupa numarul de cilindrii si transmis
ie
carWeight_cyl_am = aggregate(mtcars$wt, by = list(mtcars$cyl, mt
cars$am), FUN = mean)


# transformam rezultatul sub forma de matrice
carWeight_cyl_am = as.matrix(carWeight_cyl_am)
carWeight_cyl_am
     Group.1 Group.2        x
[1,]       4       0 2.935000
```

```
[2,]        6        0 3.388750
[3,]        8        0 4.104083
[4,]        4        1 2.042250
[5,]        6        1 2.755000
[6,]        8        1 3.370000

# aducem la forma necesara pentru barplot
carWeight = matrix(carWeight_cyl_am[,3], nrow = 3)
colnames(carWeight) = unique(carWeight_cyl_am[,2])
rownames(carWeight) = unique(carWeight_cyl_am[, 1])

carWeight = t(carWeight)

barplot(carWeight,
        beside = TRUE,
        legend.text = TRUE,
        col = c("royalblue3", "brown3"),
        main = "Greutatea medie a masinilor dupa numarul de cili
ndrii si transmisie",
        xlab = "Numar de cilindrii",
        ylab = "Greutatea medie")
```



5.5 Funcția `boxplot`

Pentru a vedea cât de bine sunt repartizate datele în setul de date putem folosi funcția `boxplot` (box and whisker plot - cutie cu mustăți). Această funcție prezintă într-o manieră compactă modul în care este repartizată o variabilă. Această metodă grafică prezintă principalii indicatori de poziție ai variabilei studiate: cuartilele de ordin 1 și 3 ($Q1$, $Q3$) care delimitează cutia ($IQR=Q3-Q1$) și cuartila de ordin 2 sau mediana (linia din interiorul cutiei). Mustățile sunt calculate în modul următor: mustața superioară este determinată de valoarea celei mai mari observații care este mai mică sau egală cu $Q3+1.5IQR$ iar mustața inferioară este valoarea celei mai mici observații mai mari sau egale cu $Q1-1.5IQR$. Valorile din afara cutiei cu mustăți se numesc valori aberante. Principalele argumente ale funcției `boxplot` se regăsesc în tabelul următor, pentru mai multe detalii apelați `?boxplot`.

Tabelul 10. Principalele argumente ale functiei boxplot.

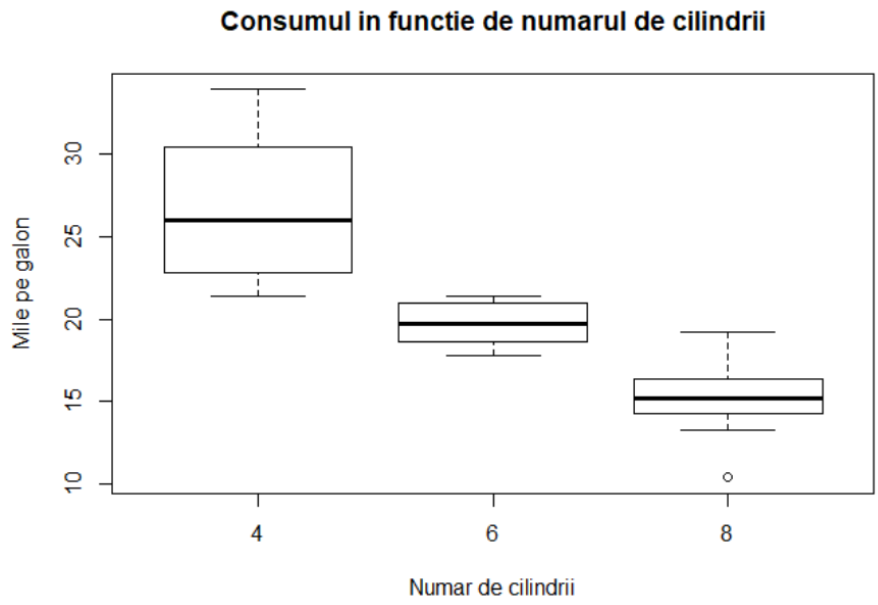
Argument	Descriere
`formula`	O formulă de tip `y ~ grp`, unde `y` este variabila investigată iar `grp` este variabila care descrie grupurile după care vrem să trasăm grafiul
`data`	Un data frame (sau listă) în care variabilele din formulă sunt definite
`subset`	Un vector care specifică o submulțime a observațiilor
`x`	Un vector care specifică valorile ce urmează să fie trasate

Tabelul 10. Principalele argumente ale functiei boxplot.

Argument	Descriere
`horizontal`	O valoare logică care indică dacă trasăm boxplot-urile vertical (`FALSE`) sau orizontal (`TRUE`)
`add`	O valoare logică prin care se permite adăugarea graficului la unul deja existent

Următorul exemplu ne prezintă relația dintre consum (`mpg`) și numărul de cilindrii (`cyl`) în cazul mașinilor din setul de date `mtcars`.

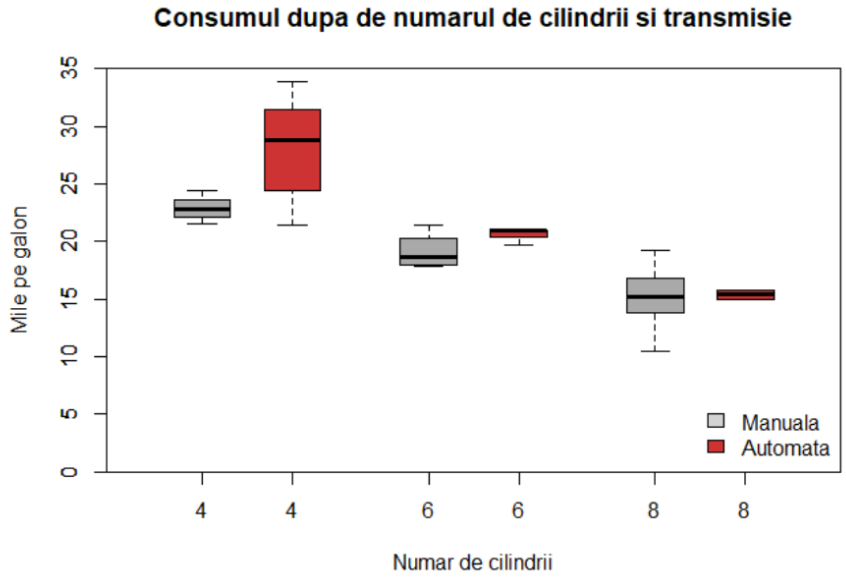
```
boxplot(mpg ~ cyl,
        data = mtcars,
        xlab = "Numar de cilindrii",
        ylab = "Mile pe galon",
        main = "Consumul in functie de numarul de cilindrii")
```



Putem să vedem această relație și în raport cu tipul de transmisie.

```
boxplot(mpg ~ cyl,
        data = mtcars,
        subset = am == 0,
        boxwex = 0.25,
```

```
        at = 1:3 - 0.2,
        col = "darkgrey",
        xlab = "Numar de cilindrii",
        ylab = "Mile pe galon",
        main = "Consumul dupa de numarul de cilindrii si transmi
sie",
        xlim = c(0.5, 3.5),
        ylim = c(0, 35),
        yaxs = "i")

boxplot(mpg ~ cyl,
        data = mtcars,
        subset = am == 1,
        add = TRUE,
        boxwex = 0.25,
        at = 1:3 + 0.2,
        col = "brown3")

legend("bottomright" ,c("Manuala", "Automata"),
       fill = c("lightgray", "brown3"), bty = "n")
```



5.6 Funcții pentru adăugarea unor elemente la un grafic

Funcțiile (low-level) din această secțiune sunt folosite pentru a adăuga elemente, de tipul linii, puncte, text la un grafic deja existent.

Tabelul 11. Functii low-level uzuale

Funcția	**rezultatul**
`points(x, y)`	Adaugă puncte la un grafic.
`abline(),segments()`	Adaugă linii sau segmente la un grafic existent.
`arrows()`	Adaugă săgeți.
`curve()`	Adaugă o curbă care reprezintă graficul unei funcții.
`rect(),polygon()`	Adaugă un dreptunghi sau un poligon oarecare.
`text(),mtext()`	Adaugă text la o figură.

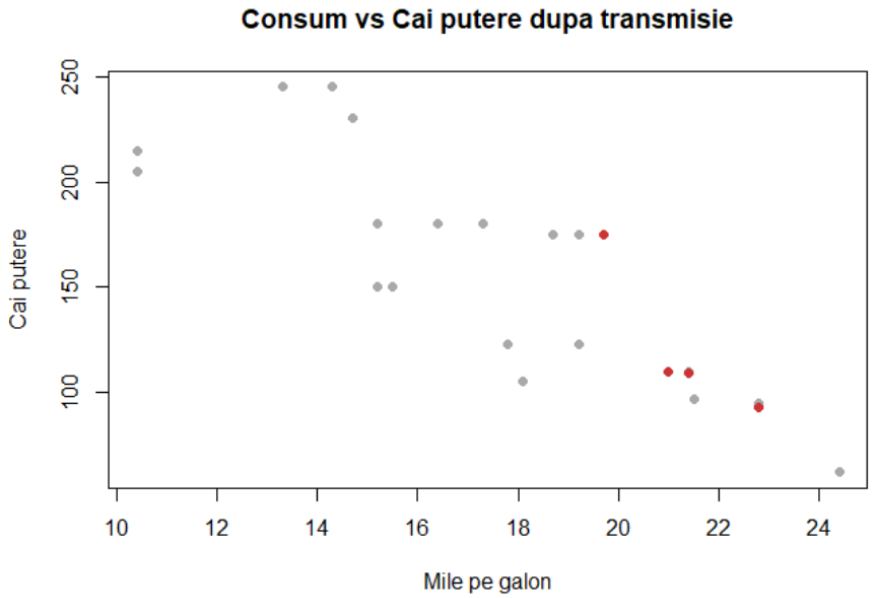
Tabelul 11. Functii low-level uzuale

Funcția	rezultatul
`legend()`	Adaugă legenda.
`axis()`	Adaugă o axă.

Pentru a adăuga noi puncte la un grafic deja existent putem folosi funcție `points()`. Pentru a vedea toate argumentele acestei funcții apelați `?points`.

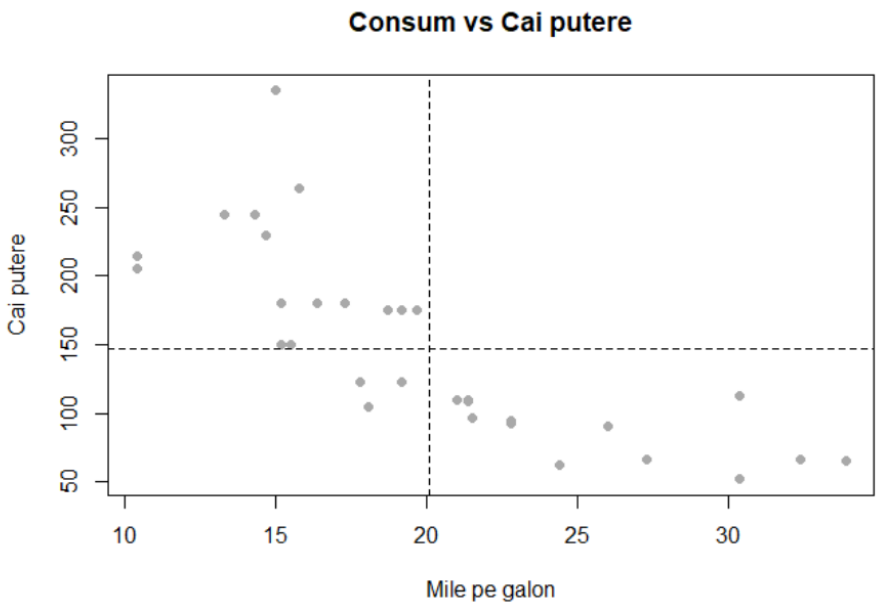
Să considerăm următorul exemplu în care trasăm diagrama de împrăștiere după consum (`mpg`) și putere (`hp`) pentru mașinile din setul de date `mtcars` în raport cu tipul de transmisie.

```
plot(x = mtcars$mpg[mtcars$am == 0],
     y = mtcars$hp[mtcars$am == 0],
     xlab = "Mile pe galon",
     ylab = "Cai putere",
     main = "Consum vs Cai putere dupa transmisie",
     pch = 16,
     col = "darkgrey")

points(x = mtcars$mpg[mtcars$am == 1],
      y = mtcars$hp[mtcars$am == 1],
      pch = 16,
      col = "brown3")
```



Dacă vrem să adăugăm linii drepte la un grafic putem folosi comanda `abline()` sau `segments()`. De exemplu în figura de mai sus vrem să adăugăm o linie verticală și una orizontală care să marcheze media variabilelor de pe axa x și y.

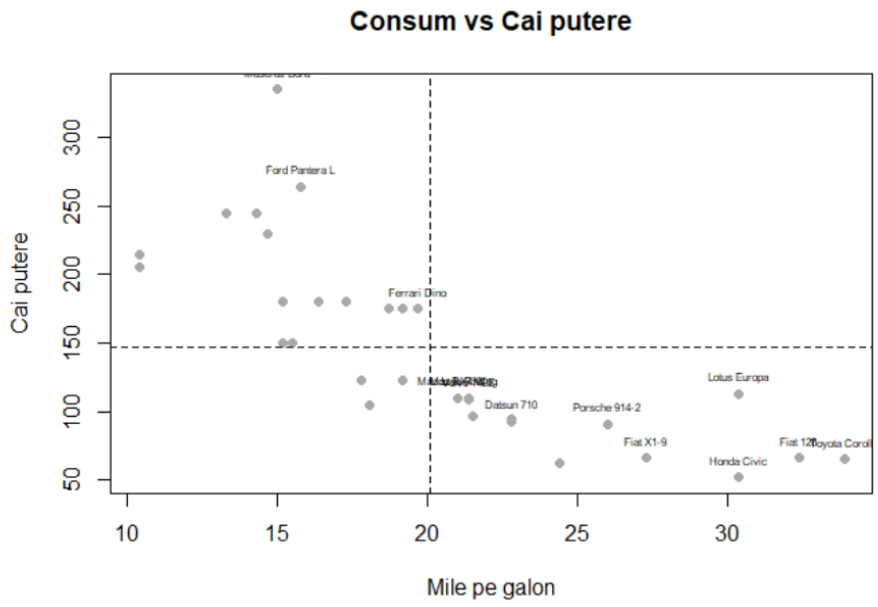
```
plot(x = mtcars$mpg,
     y = mtcars$hp,
     xlab = "Mile pe galon",
     ylab = "Cai putere",
     main = "Consum vs Cai putere",
     pch = 16,
     col = "darkgrey")

abline(h = mean(mtcars$hp), lty = 2)
abline(v = mean(mtcars$mpg), lty = 2)
```



Pentru a adăuga numele mașinilor cu transmisie automată în fiecare punct putem folosi comanda `text()`. Argumentele principale ale acestei funcții sunt `x`, `y` care descriu coordonatele etichetelor și `labels` care reprezintă etichetele.

```
plot(x = mtcars$mpg,
     y = mtcars$hp,
     xlab = "Mile pe galon",
     ylab = "Cai putere",
     main = "Consum vs Cai putere",
     pch = 16,
     col = "darkgrey")

abline(h = mean(mtcars$hp), lty = 2)
abline(v = mean(mtcars$mpg), lty = 2)

text(x = mtcars$mpg[mtcars$am == 1],
     y = mtcars$hp[mtcars$am == 1],
```

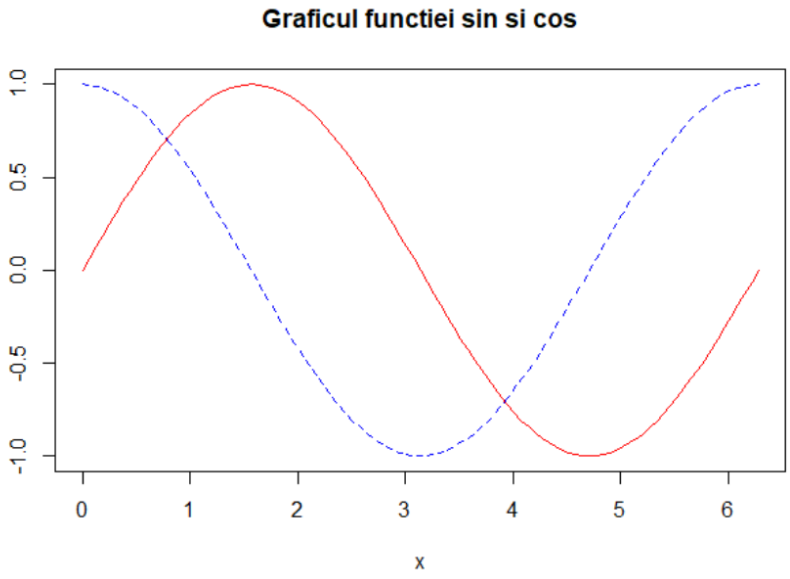
```
      labels = rownames(mtcars[mtcars$am == 1, ]),
      pos = 3,
      cex = 0.5)
```



Funcția `curve()` permite trasarea/adăugarea unei linii care descrie o funcție. Printre argumentele funcției regăsim `expr` care reprezintă expresia funcției care depinde de `x` (se pot folosi și funcții customizate), `from, to` care reprezintă intervalul de valori pentru `x` și `add` care permite adăugarea unei curbe la un grafic existent.

```
curve(expr = sin(x),
      from = 0,
      to = 2*pi,
      ylab = "",
      main = "Graficul functiei sin si cos",
      col = "red")

curve(expr = cos(x),
      from = 0,
      to = 2*pi,
```

```
      add = TRUE,
      col = "blue",
      lty = 2)
```



Atunci când vrem să adăugăm o legendă la un grafic folosim funcția `legend()`.

Argumentele acestei funcții se regăsesc în tabelul de mai jos.

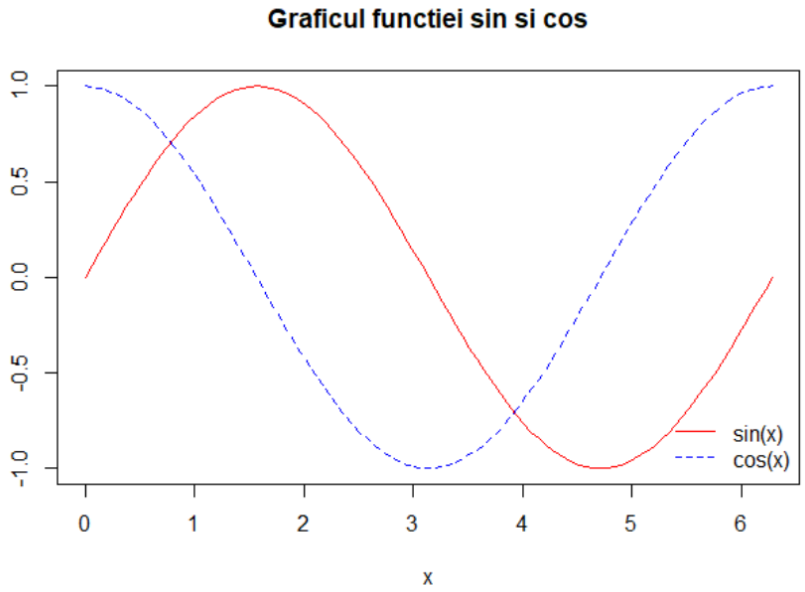
Tabelul 12. Argumentele functiei legend()

Argument	**Rezultat**
`x, y`	Coordonatele legendei - de exemplu, `x = 0, y = 0` va pune legenda la coordonatele (0, 0). Alternativ, putem indica poziția unde vrem legenda (i.e. `"topright"`, `"topleft"`).
`legend`	Un vector de caractere care precizează textul care vrem să apară în legendă.
`pch, lty, lwd, col, pt.bg, ...`	Argumente grafice adiționale (pentru detalii apelați `?legend`).

Ca exemplu să considerăm graficele de funcții de mai sus la care vrem să specificăm care grafic corespunde funcției sin și care funcției cos:

```
curve(expr = sin(x),
      from = 0,
```

```
      to = 2*pi,
      ylab = "",
      main = "Graficul functiei sin si cos",
      col = "red")

curve(expr = cos(x),
      from = 0,
      to = 2*pi,
      add = TRUE,
      col = "blue",
      lty = 2)

legend("bottomright",
       legend = c("sin(x)", "cos(x)"),
       col = c("red", "blue"),
       lty = c(1, 2),
       bty = "n")
```



5.7 Salvarea figurilor

Odată ce am creat un grafic putem să-l salvăm într-un fișier extern. Pentru aceasta folosim funcțiile `pdf()`, `png()` sau `jpeg()`. Aceste funcții vor salva figura ca fișier de tip .pdf, .png sau .jpeg.

Tabelul 13. Argumente pentru functiile pdf, jpeg si png

Argument	Rezultat
`file`	Directorul și numele fișierului sub formă de șir de caractere. De exemplu, pentru a salva un grafic pe desktop scriem `file = "/Users/.../Desktop/plot.pdf"` pentru un fișier pdf.
`width, height`	Dimensiunea graficului final în inchi.
`dev.off()`	Acesta nu este un argument al funcțiilor `pdf()` și `jpeg()`. Trebuie executat acest cod după ce trasarea graficului a fost efectuată pentru a finaliza crearea imaginii.

Pentru a salva o imagine avem de parcurs următorii pași:

1. Execută funcțiile `pdf()` sau `jpeg()` cu argumentele `file, width, height`.
2. Execută codul care generează figura (e.g. `plot(x = 1:10, y = 1:10)`)
3. Completează scrierea fișierului prin execuția comenzii `dev.off()`. Această comandă spune R-ului că am finalizat crearea fișierului.

```
# Pasul 1
pdf(file = "/Users/.../Desktop/MyPlot.pdf",   # directorul cu fi
sierul
    width = 4, # latimea in inchi
    height = 4) # inaltimea in inchi
# Pasul 2
plot(x = 1:10,
     y = 1:10)
abline(v = 0)
text(x = 0, y = 1, labels = "Ceva text aleator")
# Pasul 3
dev.off()
```