Introducció al programa R

1 L'entorn estadístic R

R és un entorn de tractament estadístic de dades, construït al voltant d'un llenguatge de programació, dotat d'una interfície gràfica amb capacitat interactiva.
El llenguatge de programació és orientat a objectes i interpretat, anàleg a d'altres llenguatges dels anomenats *de scripting*, com per exemple, el Python.
La finestra principal del programa permet entrar ordres, que s'escriuen a partir de l'indicador `>`.
L'interfície gràfica s'integra en la dels sistemes operatius, com Windows o Linux, i té un menú amb tasques bàsiques. Apart de les comunes a tots els programes, en comentarem algunes d'específiques més en detall.
El software R és lliure, la instal·lació es realitza a través del CRAN (Comprehensive R Achive Network), podeu trobar tota la informació sobre R a

http://www.r-project.org.

Un manual introductori sobre R que us pot ser útil el trobareu a

http://www.math.csi.cuny.edu/Statistics/R/simpleR/.

2 Variables

A R les dades es classifiquen en:

- Numèriques: `2.3`, `pi`, `4.6e17`.
- Complexos: `3 + 2.3i`.
- Caràcters: `"Barcelona"`.
- Lògiques: `T (TRUE)` o `F (FALSE)`.
- Missing, és a dir, una dada que falta a algun lloc o que no entén: `NA` (Not Available).

3 Operadors d'assignació

En R és possible fer servir el signe `=` com a operador d'assignació,

```
a = 3
```

però és més propi el `<-`

```
a <- 3
```

que també serveix cap a la dreta:

```
2 -> x
```

De fet, el signe `=` es fa servir, com veurem més endavant, per donar als paràmetres opcionals de les funcions valors diferents dels que tenen per defecte.

4 Vectors

La comanda `c()` permet formar vectors a partir d'escalars o altres vectors:

```
x <- c(1,2,3,4)
y <- c(4,3,2,1)
u <- c(x,4,x)
```

En el cas de cadenes de caràcters, el resultat de `c( )` és un vector de cadenes de caràcters, però no és la mateixa cosa que la cadena de caràcters més llarga obtinguda enganxant les cadenes operands. Per aquest propòsit hi ha un operador específic, `paste()`:

```
paste("A", 1:6, sep = "")
paste("Today is", date())
```

La comanda `rep(n,k)` repeteix `k` vegades el valor `n`. Per exemple,

```
rep(5,3)
```

La comanda `seq()` crea successions. Alguns exemples:

```
seq(-pi, pi, by=.5)
seq(-pi, pi, length=10)
seq(1, by=.05, length=10)
```

Per obtenir successions d'enters de `k` a `n` podem escriure `k:n`. Per exemple,

```
1:7
```

5 Operacions aritmètiques

Les usuals són `+ - * /`. Per calcular potències podem posar `**` o bé `^`.

Exercici 1. Agafeu els vectors `x` i `u` definits anteriorment i calculeu

```
z <- x+u
1/x
```

Què passa si són vectors de diferent longitud? Per exemple, proveu d'executar

```
x/u
```

6 Funcions

En `R` les funcions s'escriuen amb un nom, seguit de parèntesis, entre els quals es posen els arguments. Cada funció té 0 o més arguments obligatoris, i potser alguns d'opcionals.
Hi ha funcions primitives i d'altres, escrites en el propi llenguatge i que es troben en fitxers amb extensió `.r`. Qualsevol successió de comandes que s'escriu de manera interactiva pot gravar-se en un tal fitxer, per poder-la repetir en una altra ocasió.

Atenció! Quan s'invoca una funció cal posar-hi els parèntesis, fins i tot si la funció no té cap argument. El nom sense parèntesis produeix que `R` volqui a la pantalla la definició de la funció. Per exemple,

```
ls()
```

és una funció sense arguments, que produeix la llista d'objectes presents en memòria. En canvi, podeu veure què passa si entrem només

```
ls
```

Funcions i comandes més usuals

Per fer el màxim, el mínim, l'arrel quadrada, el valor absolut, el logaritme o l'exponencial, les funcions que necessitem són:

```
max(), min(), sqrt(), abs(), log(), exp().
```

Tenim també les funcions trigonomètriques:

```
sin(), cos(), tan() ...
```

Escriptura de noves funcions

La sintaxi d'una nova funció en R és

```
nom <- function(variables) {codi}
```

Per exemple, suposem que donades les probabilitats dels diferents punts (ordenats) d'una variable discreta volem una funció que ens retorni el valor de la Funció de Distribució en aquests punts, és a dir, la probabilitat acumulada:

```
Fdd <- function(x){
  if (sum(x)!=1 | sum(x<0)>0)
    {"Error, no és un vector de probabilitats"}
  else{
    n <- length(x)
    fdd <- rep(0,n)
    fdd[1] <- x[1]
    for (i in 2:n)
      fdd[i] <- x[i] + fdd[i-1]
    fdd
  }
}
```

Proveu ara de fer:

```
Fdd( c(2/3,1/6,1/6) )
Fdd( c(2,3,4,5) )
```

7 Gràfiques

La funció genèrica que fa gràfiques és `plot()`. Per exemple, podem obtenir la gràfica de la funció $f(x) = x^2 \sin(\frac{1}{x})$ amb aquestes instruccions:

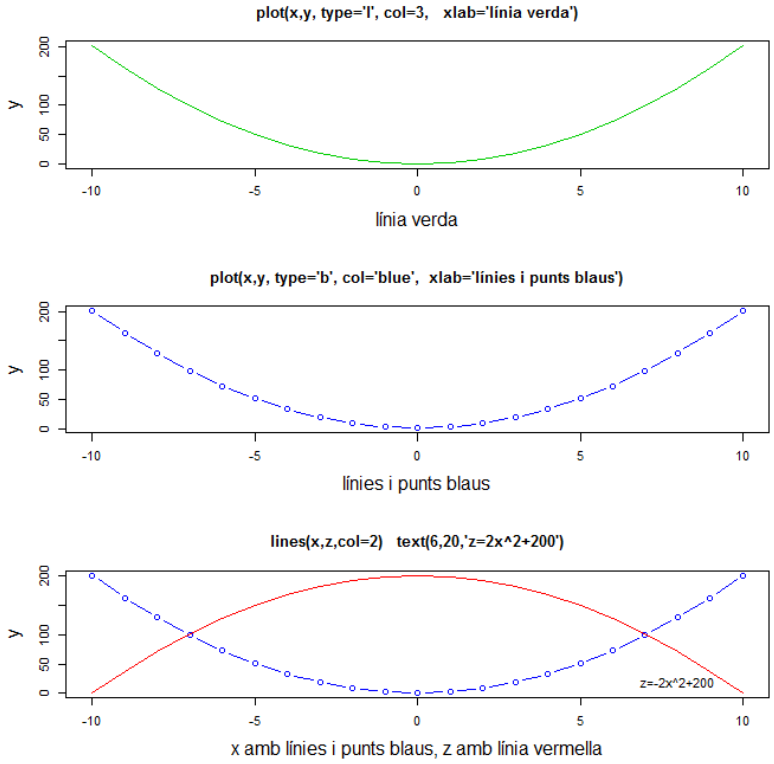
```
x <- seq(-0.5, 0.5, by=0.005)
y <- x^2*sin(1/x)
plot(x, y, type= "l")
```

La comanda `plot` té moltes opcions referents a tota mena d'aspectes del gràfic. (Vegeu el help per la sintaxi.) Per exemple, podem fer línies en lloc de punts, podem posar colors, títols, noms als eixos... A més amb les comandes `lines, points, legend, text` podem afegir línies, punts, títols o textos a un gràfic.

Podem veure més d'una gràfica alhora utilitzant la funció `par`. Per exemple, si volem posar dues gràfiques de costat escrivim `par(mfrow=c(1,2))`, si volem dues gràfiques una sobre l'altre podem fer `par(mfrow=c(2,1))`.

En el següent exemple juntem tres gràfiques amb la comanda `par(mfrow=c(3,1))`:

```
x <- -10:10
y <- 2*x^2 + 1
z <- -2*x^2 + 200
```



Exercici 2. Feu la gràfica de la funció $f(x) = \frac{1}{x}e^{2x^2}$ posant títols al gràfic i a cada un dels eixos.

8 Matrius i arrays de més dimensions

Una matriu o, en general, un array de dimensió més gran és un vector de valors, amb informació addicional: un vector `dim` d'enters positius, amb producte igual a la longitud del vector de valors.

Generem un vector, sense més estructura

```
u <- 1:12
```

L'operador `dim(u)` retorna `NULL`. Li podem assignar a `u` una estructura de matriu 3×4 amb la comanda

```
dim(u) <- c(3,4)
```

També, es pot crear directament una matriu (2-dimensional) amb

```
u <- matrix(1:12, nrow=3, ncol=4)
```

o, en cas de més de dues dimensions, amb

```
u <- array(1:12, dim=c(2,3,2))
```

Observem que el primer argument pot ser de longitud inferior a la deduïda del vector `dim`, igual al producte dels seus elements. En aquest cas s'anirà repetint fins a omplir el total d'elements del vector de valors:

```
u <- array(c(1,2), dim=c(3,4))
```

Seccions d'una matriu

La fila 1 d'una matriu: `u[1,]`.
La columna 1 d'una matriu: `u[,1]`
La segona secció 2×2 d'un array de dimensió: `c(2,2,3)` : `u[,,2]`
Una matriu, sense la fila 3:

```
A <- matrix(1:20,nrow=4)
A[-3,]
```

Una matriu, sense les columnes 1 a 3:

```
A[,-(1:3)]
```

Operacions amb matrius

- La transposta d'una matriu `A` s'obté amb la comanda `t(A)`.
- Dones dues matrius `A` i `A`, el producte i la divisió element a element s'obté amb `A*B` i `A/B`.
- El producte de matrius s'obté amb

  ```
  A %*% B
  ```

- Per calcular la inversa s'utilitza la instrucció `solve`:

  ```
  A = matrix(c(1,2,3,4), 2)
  solve(A)
  ```

9 Llistes

Estructures de dades formades per llistes de components de caràcter heterogeni.
Aquests components poden tenir noms. Per exemple,

```
x <- list("valor"=1, "retol"="exemple", "vector"=c(1,2,3))
```

construeix un objecte `x` amb tres components que tenen, respectivament, els noms `valor`, `retol` i `vector`. Podem adreçar-nos als components d'una llista pel número (amb doble parèntesi quadrat):

```
x[[1]]
x[[2]]
x[[3]]
```

o bé, pel nom (si en tenen) utilitzant el separador `$`:

```
x$valor
x$retol
x$vector
```

Encara que en general no necessitarem construir aquests objectes, sí que els farem servir implícitament. Per exemple, els `data.frame` són els objectes bàsics on emmagatzemar dades estadístiques. Una "matriu de dades" $n \times p$ amb informació de p variables observades sobre n individus anirà bé tenir-la com un `data.frame` amb p components, les p variables, on cada component és un vector de longitud n.

Exercicis

1. Trobeu els resultats de les següents operacions amb 6 decimals:

   a) $\ln(5) + 3^5 - \sqrt{3\pi}\sin\left(\frac{2\pi}{3}\right) - \sqrt[3]{e}$

   b) $\dfrac{\sqrt{3}+5\pi}{7-\sqrt[5]{2}}$

   c) $2^{-\frac{7}{13}}\left(\dfrac{11}{9}\right)^{-\frac{8}{7}}$

   d) $\displaystyle\sum_{k=763}^{825}\frac{1}{k}$

   e) $\displaystyle\sum_{k=4}^{9}\frac{3^k k!}{k^k}$

2. Completeu els llocs buits a la comanda `rep(seq(_,_,_),_)` de `R`, per tal que generi les següents successions (primer una i després l'altra):

   `20, 15, 15, 10, 10, 10, 5, 5, 5, 5`

   i

   `20, 20, 20, 20, 15, 15, 15, 10, 10, 5`.

3. Considereu les següents matrius:

$$A = \begin{pmatrix} 7 & 10 \\ 8 & 5 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 5 & 0 \\ 4 & 0 & 11 \end{pmatrix} \qquad C = \begin{pmatrix} 7 & 12 \\ 0 & 9 \\ 3 & 3 \end{pmatrix} \qquad D = \begin{pmatrix} 13 & 2 \\ 7 & 3 \end{pmatrix}.$$

   a) Calculeu la suma $A + D$.

   b) Calculeu els productes de matrius: AB, CA i AD.

4. Escriviu les instruccions per definir les següents funcions:

   a) $f(n) = \displaystyle\sum_{k=1}^{n}\frac{1}{e^k}$

   b) $g(j,n) = \displaystyle\sum_{k=j}^{n}\binom{n}{k}\frac{1}{k!}$

   i calculeu $f(10)$, $f(100)$, $g(4,10)$ i $g(6,23)$.