INTRODUCCIÓ AL R

R és un programari per realitzar manipulació de dades, càlculs i gràfics. Es tracta d'un entorn utilitzat bàsicament per fer tractament estadístic de dades. R és en realitat una implementació del llenguatge S.

És programari lliure. El podeu baixar desde http://www.r-project.org/ (només cal buscar R al google i us sortirà com a primera opció aquesta pàgina.). Un cop us l'heu baixat ja us el podeu instal.lar. En aquesta pàgina també hi trobareu diversos manuals, incloent-ne un de bàsic com *Introduction to R.*

Un cop instal.lat us trobareu amb una interfície gràfica bastant estàndard. Observeu només un parell de particulars:

1. **File**. Hi ha l'opció **Change dir.** que ens portaraà a la carpeta on hi tindrem els fitxers. **Source R code** serveix per executar fitxers, usualment amb extensió R, amb codi R.

2. **Package.** Els packages són col.leccions de funcions, dades i/o documentació. N'hi ha de molt especialitzats i de molt generals i es tracta de contribucions de la comunitat d'usuaris de R. **Load package** carrega un package que prèviament s'hagi instal.lat, bé des d'un fitxer .zip o des d'un node CRAN.

Aquesta primera sessió està dedicada a explicar-vos el funcionament bàsic del R. Per tant, si no el teniu instal.lat a l'ordinador on esteu treballant, el primer que heu de fer és instal.lar-lo i obrir-lo. Després es tracta que aneu provant tot el que s'explica.

El mateix R té ajudes incorporades. Si teniu dubte sobre algun concepte poseu

```
> help(concepte)
```

i s'obrirà una finestra d'ajuda.

NOMBRES I VECTORS

Tenim diferents tipus de dades. Les més simples són els nombres reals, peró també es poden utilitzar nombres complexes, valors lògics (és a dir TRUE o FALSE) o cadenes de caràcters, etc..

El primer pas és entendre l'operador d'assignació $<-$. Suposem que volem assignar a x el valor 3. Farem

```
> x<-3
```

També hi ha l'operador concatenació. Si volem crear el vector $(1, 4, -21)$ només cal fer

```
> y<-c(1,4,-21)
```

Ens permet a més operar diversos vectors. Així podem fer

```
> z<-c(x,10,10,y,x)
```

que ens donarà el vector

```
> z
[1]   3  10  10   1   4 -21   3
```

Podem fer les operacions aritmètiques més habituals: $+$, $-$, $*$, $/$ i per fer potències $**$. Una de les gràcies del R és que si fem operacions arimètiques entre vectors, aquestes operacions es fan component a component i si, els vectors tenen longituts diferents, el vector més curt es repeteix tantes vegades com faci falta. Vegeu els exemples següents i ho entendreu.

Si fem

```
> x<-c(1,2,3,4,5)
> x*x
```

obtenim

```
[1]  1  4  9 16 25
```

és a dir, el resultat serà el vector (1,4,9,16,25).

Si operem entre dos vectors de longitut diferent obtenim el següent:

```
> x<-c(1,5)
> y<-c(0,10,20,30,40,50)
> z<-y/x
> z
[1]  0  2 20  6 40 10
```

és a dir, el resultat és $(0, 2, 20, 6, 40, 10)$.

Una altre eina imporant és saber utilitzar les funcions que estan ja implementades al R. L'estructura és sempre la mateixa: el nom seguit de parèntesi i entre els parèntesi els arguments de la funció separats per comes. Si escriviu només el nom de la funció, us donarà la descripció de la funció.

De funcions n'hi ha moltes, entre elles: sumar `sum()`, el màxim `max()`, el mínim `min()`, l'arrel quadrada `sqrt()`, el valor absolut `abs()`, l'exponencial `exp()`, el factorial `factorial()`, el sinus `sin()`, el cosinus `cos()`, el logaritme `log()`, la tangent `tan()` o el càlcul de números combinatoris `choose(n,k)`.

Per exemple:

```
> x<-c(1,9)
> y<-sqrt(x)
> y
[1] 1 3
```

El resultat és $(1, 3)$.

Hi ha també funcions que no són numèriques. Si fem `ls()` ens dóna la llista d'objectes presents en memòria.

Per acabar aquesta primera part vegem com crear successions regulars de nombres . Hi ha dues maneres. Podem fer

```
> 1:6
[1] 1 2 3 4 5 6
```

i el resultat és el vector $(1, 2, 3, 4, 5, 6)$ o podem fer

```
> seq(-1,1,by=0.5)
[1] -1.0 -0.5  0.0  0.5  1.0
```

i el resultat és $(-1, -0.5, 0, 0.5, 1)$.

MATRIUS

Una matriu és un vector de dades al qual s'assigna uns estructura de matriu.

Per exemple, considerem el vector

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

Si el volem convertir en una matriu 3×4, ho fem donant-li aquesta estructura:

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

de manera que ara u serà

$$\begin{pmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{pmatrix}$$

Ho veurem com

```
> u
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10

[2,]    2    5    8   11

[3,]    3    6    9   12
```

De la mateixa manera ho podem convertir en una matriu 4×3 fent `dim(u) <- c(3,4)`, en una vector fila 1×12 fent `dim(u) <- c(1,12)` o en un vector columna fent `dim(u) <- c(12,1)`.

En resum, si volem crear la matriu

$$\begin{pmatrix} 1 & 8 & -3 \\ 0 & 0 & 11 \\ 0 & -1 & -1 \\ 0 & 5 & -5 \\ 3 & 6 & 9 \end{pmatrix}$$

i anomenar-la x hem de fer les comandes

```
> x<-c(1,0,0,0,3,8,0,-1,5,6,-3,11,-1,-5,9)
> dim(x)<-c(5,3)
```

Hi ha dues altres maneres equivalents de fer-ho

```
> y<-c(1,0,0,0,3,8,0,-1,5,6,-3,11,-1,-5,9)
> x<-matrix(y,nrow=5,ncol=3)
```

o

```
> y<-c(1,0,0,0,3,8,0,-1,5,6,-3,11,-1,-5,9)
> x<-array(y,dim=c(5,3))
```

Continuem amb la matriu x. Ara tenim la posibilitat de considerar només el vector fila format per la tercera fila de la matriu escrivint

```
> x[3,]
[1]  0 -1 -1
```

o el vector columna format per la primera columna de la matriu

```
> x[,1]
[1] 1 0 0 0 3
```

Quines operacions podem fer entre matrius?

Si tenim dues matrius compatibles pel producte les podem multiplicar fent

```
> a%*%b
```

Per exemple, imaginem que volem fer el producte de les matrius

$$\begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$

i

$$\begin{pmatrix} 1 & 2 & 0 \\ 0 & 1 & 10 \end{pmatrix}$$

ho podem fer de la manera següent

```
> a1<-c(2,1,1,3)
> a<-matrix(a1,nrow=2,ncol=2)
> b1<-c(1,0,2,1,0,10)
> b<-matrix(b1,nrow=2,ncol=3)
> a%*%b
     [,1] [,2] [,3]
[1,]    2    5   10
[2,]    1    5   30
```

Si dues matrius són de la mateixes dimensions podem també fer la suma (resta, producte, etc..) component a component. Proveu de fer

```
> a+a
     [,1] [,2]
[1,]    4    2
[2,]    2    6
```

o

```
> a*a
     [,1] [,2]
[1,]    4    1
[2,]    1    9
```

Veureu que $a+a$ ens dóna la suma de dues matrius pero $a*a$ no dóna el producte de matrius.

Es poden fer operacions més estranyes amb les matrius. Podeu provar de fer

```
> outer(a,b,"*")
```

o

```
> outer(a,b,"+")
```

i mireu d'entendre què fan.

FUNCIONS

Ja hem vist com utilitzar les funcions implementades al R. Tenim també la possibilitat de definir-escriure noves funcions.

La sintaxi d'una nova funció en R és

```
> nom<-function(<arguments>){codi}
```

Després la podem guardar en un fitxer *nom.R*.

Per exemple, imagineu que volem crear una funció que donats dos vectors calculi el producte dels màxim del primer pel mínim del segon. Podriem escriure:

```
> prodmaxmin<-function(x,y){
+ m1<-max(x)
+ m2<-min(y)
+ m1*m2}
```

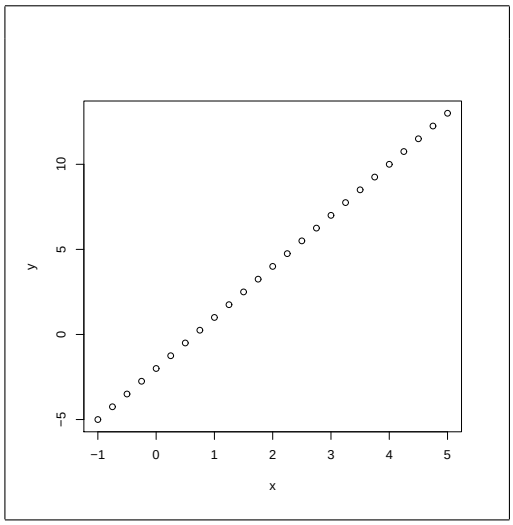
DIBUIXAR

El R té també una part gràfica. Veurem aquí com dibuixar punts i funcions.

La manera de dibuixar funcions és dibuixant uns quants punts. Evidentment si agafem més punts, millor serà el dibuix

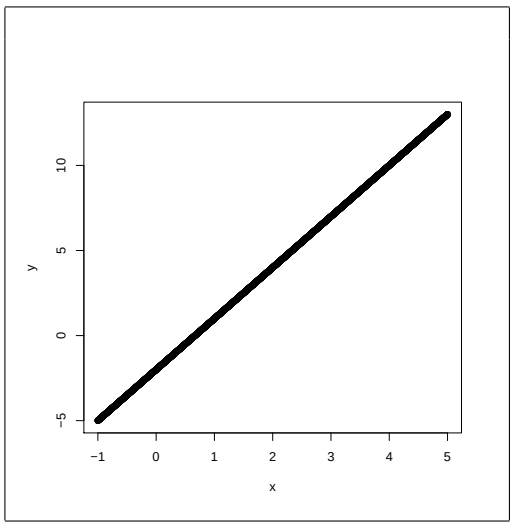
Dibuixarem primer la recta $y = 3*x-2$ a l'interval $[-1, 5]$. hi ha tres etapes:
1) Escollim els punts que dibuixarem
2) Calculem la funció
3) Dibuixem els punts

```
> x<-seq(-1,5,by=0.25)
> y<-3*x-2
> plot(x,y,type="p")
```



Si volem podem unir els punts canviant la darrera comanda per

```
> plot(x,y,type="l")
```

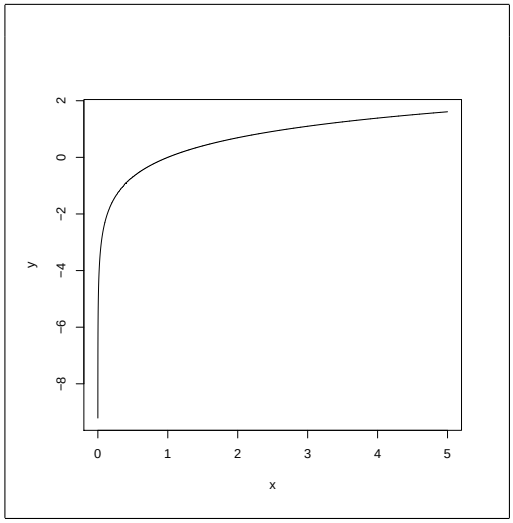


També ho podem millorar utilitzant més punts

```
> x<-seq(-1,5,by=0.001)
> y<-3*x-2
> plot(x,y,type="p")
```

Dibuixem ara la funció logaritme

```
> x<-seq(0,5,by=0.0001)
> y<-log(x)
> plot(x,y,type="l")
```



LLEGIR FITXERS EXTERNS

Per llegir un fitxer de dades extern de nom *fitxer.txt* que es trobi a la carpeta *usuari* podem utilitzar la comanda

```
> x <- read.table("d:/usuari/fitxer.txt")
```

Ha de ser un fitxer ASCII contenint dades numèriques, ordenades en files (individus) i columnes (variables).

Si el fitxer té una primera línia que conté els noms de les variables, aleshores la instrucció és

```
> x <- read.table("d:/usuari/fitxer.txt", header = TRUE)
```

Hi ha moltes altres opcions, segons les diverses possibilitats de formats de fitxers, per exemple amb valors separats per comes, de formats numèrics, especificant el caràcter separador de decimals.

El resultat és un objecte del tipus data.frame.

Podem recuperar els components pel número(amb doble parèntesi quadrat) `x[[1]]` o bé, pel nom, si en si tenen, amb el separador $, `x$nom`.

Per exemple, considereu el fitxer *exemplefitxer.txt*. Agafeu-lo del dossier, copieu-lo al vostre disc dur i després llegiu-lo des del R amb el nom de x. Comproveu què fan les comandes

```
> x
> x[[1]]
> x$y1
```

EXERCICIS

1. Digueu què fa

```
> y<-seq(2,10,by=2)
> z<-y**2
> sum(z/factorial(y))
```

2. Calculeu

$$\sum_{k=2}^{100} \frac{1.1^k + sin(3)}{k \log(k)}$$

3. Siguin A i B les matrius

$$\begin{pmatrix} 1 & 1 \\ 1 & 3 \end{pmatrix}$$

i

$$\begin{pmatrix} 2 & 2 \\ 3 & 4 \end{pmatrix}$$

Calculeu $A + B$ i $A * B * B * A$

4. Dibuixeu el punts $(-1, 1), (1, 1), (-1 - 1), (1, -1)$.

5. Dibuixeu la funció $\sin(x)/x$ a $(0, 1)$.

6. Agafeu el fitxer *vida.txt* del dossier (al *vidadoc.txt* el trobareu documentat). Llegiu-lo amb el R i feu un gràfic de punts de les variables *longitut* i *edat*.

SOLUCIONS

1. Aquestes comandes calculen

$$\frac{2^2}{2!} + \frac{4^2}{4!} + \frac{6^2}{6!} + \frac{8^2}{8!} + \frac{10^2}{10!}$$

2. Fem les comandes

```
> k<-2:100
> x<-((1.1)^k+sin(3))/(k*log(k))
> sum(x)
[1] 387.9854
```

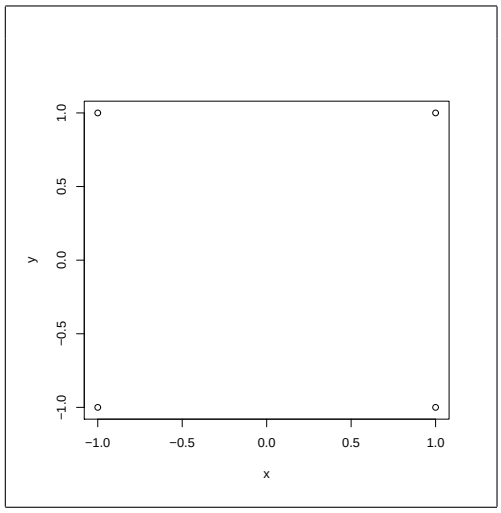
3. hem de fer

```
> a1<-c(1,1,1,3)
> a<-matrix(a1,nrow=2,ncol=2)
> b1<-c(2,3,2,4)
> b<-matrix(b1,nrow=2,ncol=2)
> a+b
     [,1] [,2]
[1,]    3    3
[2,]    4    7
> a%*%b%*%b%*%a
     [,1] [,2]
[1,]   62  130
[2,]  142  298
```

4. Les comandes són

```
> x<-c(-1,-1,1,1)
> y<-c(-1,1,-1,1)
> plot(x,y)
```

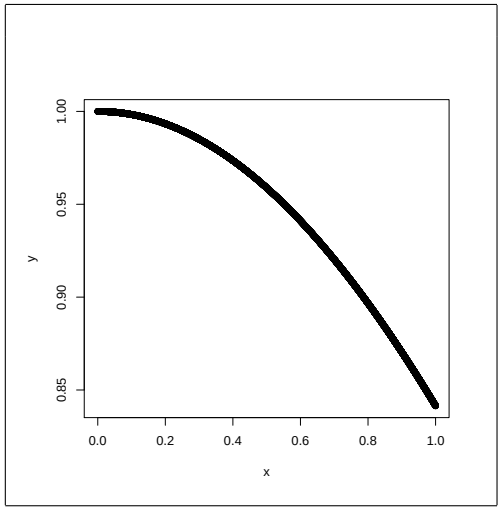
i el gràfic que s'obté



5. Les comandes són

```
> x<-seq(0,1,by=0.0001)
> y<-sin(x)/x
> plot(x,y)
```

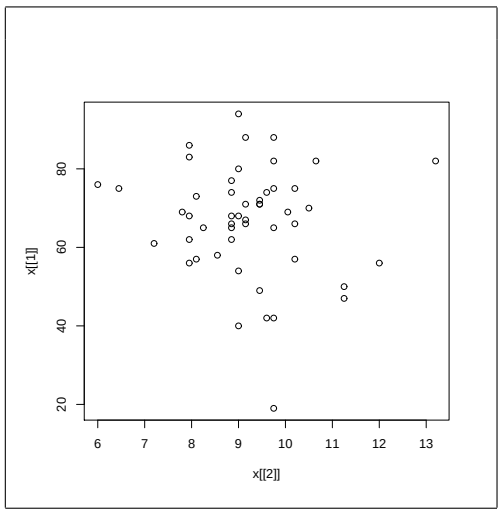
i el gràfic que s'obté



6. Ara hem de fer

```
> x<-read.table("d:/vida.txt")
> plot(x[[2]],x[[1]])
```

i el gràfic que s'obté és


x[[1]]
20
40
60
80
6
7
8
9
10
11
12
13
x[[2]]