

## Sessió 4

# Llegir i guardar fitxers de dades. Subdataframes i selecció amb condicions.

### 4.1 *Guardar i carregar objectes en format de R: les funcions save i load*

L'entorn de treball (*workspace*) conté tots els objectes creats durant la sessió. És recomanable tenir un espai de treball diferent per a cada tasca, per no barrejar objectes del mateix nom.

En aquesta secció i les dues properes seccions aprendrem a carregar `data.frames` de paquets de R, i a llegir i guardar els nostres objectes en diferents tipus de fitxer.

La funció `save` guarda en un fitxer `.RData` els objectes que especifiquem. El fitxer que es genera només es pot llegir amb la funció `load` de R, o bé des de RStudio clicant al fitxer des de la pestanya **files**.

```
a<-1:10
b<-LETTERS[4:10]
#guardem a i b al fitxer ab.RData (al nostre directori de treball)
save(a,b,file="ab.RData")
# podem recuperar-los:
load("ab.RData")
```

També podem guardar tots els objectes que tinguem al nostre entorn de treball amb `save.image('`nom.fitxer.RData`')`.

A continuació veurem altres maneres de llegir i guardar dades generades per R en fitxers que es puguin llegir amb altres programes.

### 4.2 *Importar fitxers de dades*

Un fitxer de dades és un objecte anomenat **data.frame**. A les llibreries de **R** hi ha molts fitxers de dades que ens poden ser útils. També podem llegir fitxers externs de dades de text, d'excel i d'altres paquets estadístics.

4.2.1 Dades dels paquets de R. La funció `data`

- Els fitxers de dades de R accessibles al paquet base (`datasets`) es carreguen amb la funció `data(nomdeldataset)`.

Per obtenir informació de les dades que conté una base de dades del paquet `datasets` (o d'algun altre paquet carregat) hi ha diverses funcions: ajuda (`?`), noms de les variables (`names`), visualització dels 6 primers casos (`head`), visualització dels 6 darrers casos (`tail`), nombre de files i de columnes (`dim`).

```
data()           # fitxers de dades dels paquets carregats
?ToothGrowth     # ajuda del data.frame ToothGrowth
dim(ToothGrowth) # dimensió
head(ToothGrowth) # capçalera
tail(ToothGrowth) # cua
names(ToothGrowth) # noms de les variables
```

Seguidament, carreguem el fitxer i les seves variables a l'espai de treball

```
# fem accessible el fitxer i li canviem el nom
ls() # no apareix com a objecte a l'entorn de treball
data(ToothGrowth) # carreguem el fitxer
ls() # ara sí que apareix ToothGrowth a la llista d'objectes
dents<-ToothGrowth # abreguem el nom
len # no reconeix la variable, perquè està dins de dents
dents$len # recordem com accedir a una variable d'un data frame
dents$supp
dents$dose
```

- Fitxers d'altres llibreries

```
data(package="AER") # fitxers de dades del paquet AER (si està
                    # instal.lat a l'ordinador, encara que no estigui carregat)
                    # si no el troba, cal instal.lar-lo abans
# un dels fitxers de dades és Affairs:
help('Affairs', package='AER')
# carreguem Affairs sense carregar tot el paquet AER:
data('Affairs', package='AER')
ls() # Affairs apareix al nostre entorn de treball.
search() # però el paquet AER no està carregat
# Informació dels fitxers de dades de TOTS els paquets instal.lats:
data(package = .packages(all.available = TRUE))
```

**Pràctica:**

- Treu el llistat de bases de dades del paquet `Lahman`.
- Carrega la base de dades `Salaries` (conté els salaris de jugadors de baseball) del paquet `Lahman` i reanomena-la `df.salaris`.
- Quants casos i quantes variables té aquesta base de dades?
- Com s'anomenen les variables que conté la base de dades `Salaries`?
- Quina és la descripció de cada una d'aquestes variables?
- Visualitza els 6 primers casos del data frame `df.salaris`.

### 4.2.2 Fitxers de text. Funció `read.table` i `read.csv`

Amb un editor de text, per exemple Notepad, obrim els fitxers `dades2.txt` i `dades3.txt` (els trobareu al Moodle), observem les diferències i els guardem al nostre directori de treball. Per importar-los, fem:

```
read.table('dades2.txt')
read.table('dades3.txt', header=T) # llegeix noms de les variables
```

#### Els arguments de les funcions `read.table` i `read.csv`

Les funcions següents serveixen per llegir dades d'un fitxer. En cada cas especifiquem quins són els arguments i els seus valors per defecte:

```
read.table('dades.txt', header=FALSE, sep=' ', dec='.', na.strings='NA')
read.csv('dades.csv', header=TRUE, sep=',', dec='.', na.strings='NA')
read.csv2('dades.csv', header=TRUE, sep=';', dec='.', na.strings='NA')
```

Entre cometes posem el nom de l'arxiu que volem carregar (com ara `dades.txt`). Si l'arxiu és al nostre directori de treball **només cal donar el nom**, si és a un altre directori li hem de donar l'adreça completa (com ara `C:\\...\\prac1\\dades.txt`).

**Important:** Abans de llegir un fitxer de text amb R cal que l'obriu amb un editor de textos com ara Notepad (bloc de notes) de Windows o Kate de Linux. Heu de fixar-vos en les característiques següents del fitxer i posar-les com a arguments a la funció `read.table`, `read.csv` o `read.csv2`:

- *Hi ha capçaleres de les columnes?*

L'opció **header** serveix per avisar quan hi ha capçaleres `header=TRUE` (també `header=T`) o bé, si no n'hi ha, `header=FALSE` (també `header=F`).

- *Quin és el símbol separador de columnes?*

L'opció **sep** serveix per indicar quin és el símbol utilitzat per separar columnes en el nostre fitxer. Si és espai o tabulador, posem `sep=' '`, si és coma, posem `sep=','`, si és punt i coma, `sep=';'`, ...

Quan des de excel o un altre full de càlcul guardem un fitxer en format CSV normalment ens demana com volem separar les columnes. Aquí podem decidir si utilitzar tabulador o “;”, que són els més recomanables.

- *Quina és la notació per a decimals?*

L'opció **dec** serveix per indicar quin és el símbol utilitzat per als decimals. Si els decimals estan escrits amb un punt, escriurem `dec='.'`. Si és un fitxer generat amb excel, es tractarà d'una coma, i escriurem `dec=','` (atenció que aleshores no podem utilitzar la coma com a separador!).

- *Hi ha valors perduts (falten dades en algunes columnes)?*

Més endavant veurem com tractar els valors perduts, però cal tenir present que si el fitxer té valors perduts, al llegir el fitxer cal avisar amb l'opció **na.strings**. Posant `na.strings=''` tots els espais en blanc s'ompliran amb NA, que R identificarà com a valor perdut.

A vegades en el nostre fitxer tenim més d'un símbol per denotar que no hi ha valor, com per exemple “-” i “?”. Aleshores hem d'escriure al llegir el fitxer `na.strings=c('-', '?')`.

La diferència entre les funcions `read.table`, `read.csv` i `read.csv2` és quines són les opcions per defecte. Podeu mirar l'ajuda per esbrinar-ho però el que sempre funciona és `read.table` amb totes les opcions ben especificades (**header**, **sep**, **dec** i **na.strings**).

#### Pràctica:

- a) Baixa el fitxer `Passatgers_ElPrat.csv` que trobaràs al Moodle i desa'l al teu directori de treball de R.
- b) Obre'l amb un editor de textos tipus Notepad++ o kate i mira'n les característiques. Llegeix-lo amb R utilitzant les funcions `read.csv` i `read.table` i comprova que s'ha llegit correctament.
- c) Quines són les variables? quants casos té? Visualitza els primers 6 casos.
- Baixa el fitxer `llegir-dades.zip` i descomprimeix-lo al teu directori de treball. Han d'aparèixer 11 fitxers de text. Contenen les mateixes dades però en formats lleugerament diferents. Obre'ls primer amb Notepad++ o Kate i carrega'ls a R amb `read.table` cadascun amb les opcions adequades. Si convé, fes petites modificacions del fitxer des de Notepad abans d'obrir-lo amb R. El resultat ha de ser en tots els casos el següent:

	ID	Sexe	Pes	Talla	AnyNaixement	Edat	Franja
1	365	home	76.40	186.00	71	30	30-34
2	306	dona	49.00	167.00	82	19	18-19
3	280	home	49.20	165.00	41	60	60-65
4	278	<NA>	70.00	177.30	NA	NA	<NA>
5	395	dona	58.00	162.00	53	48	45-49
6	320	home	90.94	180.45	47	54	50-54
7	270	home	63.00	174.00	80	21	20-24
8	291	home	75.00	170.00	60	41	40-44
9	369	dona	55.00	161.00	78	23	20-24

### 4.2.3 Fitxers Excel

#### Mètode directe

Si volem obrir el fitxer directament d'Excel (.xlsx o .xls), podem utilitzar la funció `read.xlsx()` de la llibreria `xlsx` (sistema Windows) o bé de la llibreria `openxlsx` (sistema Linux). Hi ha moltes altres funcions que permeten obrir un fitxer Excel però `read.xlsx()` és de les més bàsiques.

Per desgràcia, la funció `read.xlsx` no es crida de la mateixa manera si es treballa amb el paquet `xlsx` o si es fa amb el paquet `openxlsx`. Vegem un exemple amb el fitxer de Excel `Noms_nadons_2017.xlsx` (descarregueu-lo de Moodle i guardeu-lo al vostre directori de treball).

Amb `xlsx`:

```
install.packages("xlsx") # només si no el tens instal·lat
library(xlsx)
df.noms17<-read.xlsx(file="Noms_nadons_2017.xlsx", sheetIndex = 1,
                     startRow = 10, endRow = 29, header = F, encoding="UTF-8")
df.noms17<-df.noms17[, -6] # Eliminem la darrera variable perquè està buida
names(df.noms17) <- c("Nom", "Ranquing2017", "Quantitat", "Taxax1000",
                     "Ranquing2016")
#no podíem aprofitar els noms de l'Excel pq no estaven en una mateixa fila
# Per veure tots els arguments (i el seu ús) de la funció read.xlsx():
? read.xlsx
```

Amb openxlsx:

```
install.packages("openxlsx") # per a linux
library(openxlsx)
df.noms17 <- read.xlsx("Noms_nadons_2017.xlsx", rows = 10:30, colNames = F)
names(df.noms17) <- c("Nom", "Ranquing2017", "Quantitat", "Taxax1000",
                      "Ranquing2016")
# no podíem aprofitar els noms de l'Excel pq no estaven en una mateixa fila
# Per veure tots els arguments (i el seu ús) de la funció read.xlsx():
? read.xlsx
```

### Creant abans un fitxer csv i després obrint amb read.table()

Una manera alternativa i que sempre funciona d'obrir amb **R** fitxers Excel és transformant-los primer en fitxers de text des de Excel, de la manera següent:

1. Obre amb Excel el fitxer `ExerPr1Enq.xlsx` i posiciona't al full **dades**, on veuràs les variables ***Id, Edat, Alt, Pes, Sexe, Tabac i Oci***. Aquestes dades provenen d'una enquesta.
2. Des de l'Excel, guarda el full en format “text separat per tabulacions” i anomena'l `ExerPr1Enq.txt`
3. Desa'l al directori de treball i fes

```
enq<-read.table('ExerPr1Enq.txt', header=T, dec=',')
# al fitxer excel els decimals estan separats per ','
head(enq)
```

Si hi ha valors perduts (*missing values*), com al fitxer `notes.xlsx`, és millor fer

1. Obre amb Excel el fitxer `notes.xlsx`
2. Des de l'Excel, guarda el full com a “text separat per comes” i anomena'l `notes.csv` (és de fet text separat per ; com pots comprovar obrint-lo amb Notepad+)
3. Desa'l al directori de treball i fes

```
notes<-read.csv2('notes.csv', header=T, dec=',', na.strings=c('na', 'np', ''))
notes # el fitxer és curt i el veiem complet
```

Nota: El valor 'np' és ara un valor perdut.

#### Pràctica:

Al Moodle pots trobar un fitxer anomenat `batx-a-12.xls` que conté el nombre d'alumnes avaluats de 2on de batxillerat el curs 2018–19. Importa a R les dades de les noies que es van avaluar del batxillerat d'Humanitats i Ciències Socials (nom comarca, nombre d'avaluades, promocionen sense pendents a juny, promocionen sense pendents setembre i han de repetir) per comarques. Anomena el data.frame `Avaluades_18_19`. (Pista: potser hauràs d'utilitzar l'argument `colIndex`, o bé llegir tot el fitxer i després eliminar files i columnes que no siguin necessàries). Posa les etiquetes de les variables que has importat.

#### 4.2.4 Fitxers d'altres paquets estadístics

L'SPSS és un paquet estadístic comercial, molt utilitzat en ciències socials. Per llegir fitxers de l'SPSS (`.sav`) podem fer:

```
library(foreign) # potser cal instal·lar-la abans
pesedat <- read.spss('pesedat.sav', to.data.frame = T)
head(pesedat)
```

*Nota:* La mateixa llibreria `foreign` permet importar fitxers creats amb altres paquets comercials, SAS, STATA, etc.

##### **Pràctica:**

- a) El fitxer `Percepcio_Turisme_BCN_2017.sav` conté les respostes a les enquestes realitzades als ciutadans de Barcelona sobre la percepció que tenen del turisme de la ciutat. Baixa el fitxer `Percepcio_Turisme_BCN_2017.sav` que trobaràs al Moodle i posa'l en un `data.frame` anomenat `PerTur17`.
- b) Quantes enquestes conté aquest fitxer?
- c) Cerca l'argument per a que les variables que són factors no s'importin com a tals i aplica'l per fer una nova importació del fitxer. Què passa amb la variable `LLENGUA_ENQ`?

#### 4.2.5 Importar fitxers directament per menús de RStudio

Una altra de les ajudes que ens proporciona RStudio és la importació de fitxers utilitzant menús. A la pestanya **Environment** del quadrant de dalt a la dreta hi ha una pestanya per importar que s'anomena **Import Dataset**. També podeu utilitzar la pestanya **files**.

El primer que s'ha d'escollir és quin tipus de fitxer volem importar. En aquest cas importarem el fitxer "Padro2018" que ens baixarem del Moodle. Com és un fitxer csv escollirem l'opció `From Text` (`readr`) (també podríem importar amb `From text (base)` però va millor la primera opció).

Surt una finestra com aquesta:

The screenshot shows the 'Import Text Data' dialog box in RStudio. It includes a 'File/Url' field with a file path, a 'Data Preview' table, and 'Import Options' for naming the data frame, delimiters, and skip rows. A 'Code Preview' shows the generated R code using the `read_delim` function. Red circles and arrows with numbers 1-7 provide a step-by-step guide:

1. Busqueu on està el vostre fitxer (Find where your file is).
2. Nom que li voleu donar al vostre data.frame (Name you want to give to your data.frame).
3. Nombre de files en el fitxer amb informació que no volem exportar. A vegades pot ser 0. (Number of rows in the file with information we don't want to export. Sometimes it can be 0).
4. Indiquem que la primera fila conté el nom de les variables (We indicate that the first row contains the names of the variables).
5. Indiquem que el separador de columnes és ";" (We indicate that the column separator is ";").
6. Aquí ens dona l'escript que podem guardar per executar quan sigui necessari (Here it gives us the script we can save to execute when necessary).
7. Ja podem importar (We can now import).

**Nota:** Com és necessària la llibreria `readr`, comprova primer si la tens instal·lada i si no és així caldrà que l'instal·lis.

#### Pràctica:

Utilitzant el mateix fitxer que a la pràctica d'importar fitxers amb xls. En aquest cas importa les dades dels nois que es van avaluar del batxillerat d'Humanitats i Ciències Socials (nom comarca, nombre d'avaluats, promocionen sense pendents a juny, promocionen sense pendents setembre i han de repetir) per comarques. Anomena el data.frame `Avaluats_18_19`. Com no deixa posar un rang compost exporteu per finestres només les variables d'avaluació i després afegiu la columna de les comarques del data.frame `Avaluades_18_19`. Posa els mateixos noms a les variables. Potser t'hauràs d'instal·lar el paquet `readxl`.

### 4.3 Escriure dades des de R a un fitxer de text

#### 4.3.1 Les funcions `write.table` i `write.csv`

Durant una sessió de R creem o modifiquem data.frames i en finalitzar a vegades volem guardar-les per llegir-los amb un full de càlcul o amb un altre paquet. Per a guardar els data.frames de R com a fitxers `.txt` o `.csv` podem usar les funcions `write.csv` i `write.table`, de manera similar a com utilitzàvem `read.csv` i `read.table`:

```
write.table(x, file='nom.csv', sep=';', dec='.', row.names=T, col.names=NA)
```

on `x` és el nom que hem donat dins de R a la taula que volem guardar. Les opcions **sep** i **dec** són les mateixes que per a llegir fitxers. L'opció **row.names** s'utilitza quan les files tenen nom.

Per exemple primer llegim el fitxer "dades1.csv" amb R i el guardem en la variable 'x':

```
x<-read.csv('dades1.csv', sep=';', dec='.', header=F)
```

i després guardem `x` en un altre fitxer, per exemple `data_de_R.csv`:

```
write.table(x, file='data_de_R.csv', sep=';', dec='.', row.names=F)
write.table(x, "dadesX.txt", quote=FALSE, row.names=F)
write.table(x, "dadesY.txt", quote=FALSE, row.names=T)
write.table(x, "dadesZ.txt", quote=FALSE, row.names=T, col.names=NA)
# fixeiu-vos amb els diferents arguments i com queden els fitxers de text.
```

**Pràctica:**

- Crea un data.frame anomenat `TotalAvaluacio1819` amb tres vectors: `Total`, `dones` i `homes`. Cada vector ha de contenir la suma d'avaluats, promocionen sense pendents a juny, promocionen sense pendents setembre i han de repetir del total de nois/es que es van avaluar del batxillerat d'Humanitats i Ciències Socials (potser hauràs de tornar a importar el fitxer `batx-a-12.xls`)  
Posa noms a les fileres i a les columnes del data.frame.
- Exporta aquesta taula (o data.frame) a un fitxer `Curs1819.csv` que puguem obrir més endavant.
- Guarda el data.frame també en el fitxer `TotalAvaluacio1819.RData`

## 4.4 Sub-dataframes

Una part molt important en la gestió dels fitxers de dades és el filtratge de casos i la selecció de variables, que veurem a continuació.

### 4.4.1 La funció `subset`

Apliquem la funció `subset` a un data.frame i construïm un nou data frame, subconjunt de l'original. Seleccionem els casos (les files) que satisfan una condició, que involucra a una o més variables, i amb l'argument addicional `select` seleccionem unes quantes columnes del data frame original.

```
df<-data.frame(x1=c(2,4,6,8),x2=c(1,0,0,1),x3=c("a","b","b","a"))
df # volem un nou data.frame amb els casos que x2 sigui 0
(df1<-subset(df,x2==0)) # df1 només els casos en què x2 és 0
# també podem utilitzar <, >, <=, >= , o amb altres
# condicions compostes, amb AND i OR
# podem seleccionar alhora només alguna de les variables
df2<-subset(df,x2==0,select=c("x1","x3"))
df2 # només conté les variables x1 i x3 dels casos amb x2 igual a 0
df3 <-subset(df,x2==0 | x1>3)
df3 # només conté els casos amb x2 igual a 0 o x1 és més gran que 3
```

### 4.4.2 Sub-dataframes amb indexació

També podem filtrar casos i seleccionar variables amb indexació i condicions sobre les variables.

Recordem la notació per seleccionar files i columnes d'una matriu o un data.frame vista a la pràctica anterior. Per exemple, `df[3:10, c(2,4,8)]` ens dona un nou data.frame amb les files 3 a 10 i les columnes 2,4 i 8 de `df`. En comptes d'escriure 3:10 per referir-nos a les files hi podem posar una condició lògica. Aquesta filosofia també es pot aplicar a vectors.



Comencem amb un exemple de com podem eliminar d'un vector els valors perduts utilitzant la funció `is.na` i la indexació:

```
v<-c(3,4,NA,NA,2,10,1,NA)      # vector de longitud 8
is.na(v)      # generem un vector lògic
[1] FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE
!is.na(v)     # el vector lògic contrari a l'anterior
[1]  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE
# podem guardar-lo en un objecte que anomenem seleccio
seleccio <- !is.na(v)  # seleccio és un vector lògic de longitud 8
# què passa si agafem les components que indica seleccio??
v[seleccio]
[1]  3  4  2 10  1  # Tenim les components de v que NO són NA

# el que hem obtingut és el mateix que si haguéssim escrit
v[c(1,2,5,6,7)]
```

El mateix s'aplica amb `data.frames`, per seleccionar files. Vegem com aconseguir els tres `data.frames` `df1`, `df2`, `df3` del requadre d'exemples de la funció `subset`, d'una altra manera.

```
df
  x1 x2 x3
1  2  1  a
2  4  0  b
3  6  0  b
4  8  1  a

# volem un nou data.frame amb els casos que x2 sigui 0
(df1<-df[df$x2==0, ])
  x1 x2 x3
2  4  0  b
3  6  0  b
# volem que df2 tingui els casos anteriors i les variables x1 i x3
df2<-df[df$x2==0,c('x1','x3')]
df2<-df[df$x2==0,c(1,3)]      # equivalent a l'anterior
# volem df3 amb els casos tals que x2 és 0 o x1 és més gran que 3
df3<-df[df$x2==0 | df$x1>3, ]  # equivalent a l'anterior
```

Vegem un altre exemple, treballant amb el fitxer de dades anorexia del paquet `MASS`. Aquest `data.frame` té tres variables: `Treat` (tractament), `Prewt` (pes abans), `Postwt` (pes després). `Treat` té tres modalitats (`Cont`, `CBT`, `FT`). Volem estudiar l'augment relatiu de pes per als diferents tractaments. Per això hem de separar les nostres dades en tres `data.frames` corresponents a cada un dels tres tractaments. Abans afegim l'augment relatiu percentual de pes com a nova variable al `data.frame`. Es calcula com

$$100 * (\text{pes després} - \text{pes abans}) / \text{pes abans}$$

```
library(MASS)
data(anorexia)  # hem carregat el data.frame del paquet MASS
names(anorexia); dim(anorexia)
# afegim l'augment relatiu percentual de pes com a nova variable
anorexia$AugmentRelatiu<-100*(anorexia$Postwt-anorexia$Prewt)/anorexia$Prewt
```

```
# si visualitzem la capçalera del fitxer veiem que té una variable més
head(anorexia)

# construïm tres sub-dataframes segons tractament:
d.FT<-anorexia[anorexia$Treat=="FT",c("Treat", "AugmentRelatiu")]
d.Cont<-anorexia[anorexia$Treat=="Cont",c("Treat", "AugmentRelatiu")]
d.CBT<-anorexia[anorexia$Treat=="CBT",c("Treat", "AugmentRelatiu")]

# utilitzant la funció subset equivaldria a
d.FT<-subset(anorexia,Treat=="FT",select=c("Treat", "AugmentRelatiu"))
d.Cont<-subset(anorexia,Treat=="Cont",select=c("Treat", "AugmentRelatiu"))
d.CBT<-subset(anorexia,Treat=="CBT",select=c("Treat", "AugmentRelatiu"))
```

**Pràctica:** En aquesta pràctica treballarem amb el data.frame que has generat per la pràctica d'importar fitxers d'altres paquets estadístics: data.frame anomenat PerTur17.

- Genera un sub-data.frame anomenat PerTur17.dones que contingui les respostes de les dones.
- Genera un sub-data.frame anomenat PerTur17.seniors que contingui les respostes dels majors de 65 anys.
- Genera un sub-data.frame anomenat PerTur17.part1 que contingui totes les respostes de les variables de freqüència de visita.
- Genera un sub-data.frame anomenat PerTur17.part2 que contingui la variable SEXE, LLENGUA\_ENQ i totes les respostes de les variables de freqüència de visita només dels casos enquestats en català.
- Genera un sub-data.frame anomenat PerTur17.part3 amb el sexe, l'edat i la nacionalitat actual dels 500 primers casos.