Importación/exportación de datos en R

Manuel (mramon@jccm.es), 23 a 26 de abril de 2021

- Como leer bases de datos: txt, csv, excel
- Como guardar bases de datos
- Como guardar y cargar objetos de R: RData

GUIÓN

- Formatos de datos. Recomendaciones
- Importar datos en R
- Exportar datos desde R
- Exportar espacio de trabajo: RData

Recomendaciones bases de datos

- Es preferible no usar formatos bajo licencia o poco comunes. Por ejemplo, todo el mundo tiene acceso a un editor de texto (TXT, CSV), pero podría no tener acceso a Excel o Access.
- Algunos formatos tienen limitación en el número de filas/columnas que se pueden almacenar. Los archivos de texto plano no.
- Prestar atención a los signos de puntuación usados para indicar decimales. En Excel español, los decimales se separan por comas; en otros programas (R, por ejemplo) se usa un punto.
- Carácter usado para separar campos/columnas: espacio, tabulador, punto y coma, ancho fijo, …

Recomendaciones bases de datos

Entonces, ¿cómo almacenamos los datos? Mis sugerencias

- Almacenar los datos en archivos de texto CSV con los campos separados por puntos y comas (;)
- Incluir en la primera fila el nombre de las variables
- Los nombres no deben contener espacios
- Usar puntos (.) para separar decimales
- No usar ningún signo para separar miles, millones,…
- No usar símbolos tipo #,*,!,? que pueden dar problemas de lectura

Importar datos en R

- R dispone de numerosas funciones para importar datos en función del formato: `scan()`, `read.table()`, `read.delim()`, `read.csv()`, `read.fortran()`
- Todas ellas leen archivos de texto y se diferencian principalmente en el delimitar que separa cada campo/variable
- Además, existen paquetes específicos que permiten leer otros formatos, como `foreign` o `readxl`, entre otros.

Importar datos en R

- La función más usada es `read.table()`. Vamos a ver la ayuda de la función

```
?read.table
```

```
read.table(file, header = FALSE, sep = "", quote = "\"'",
           dec = ".", numerals = c("allow.loss", "warn.loss", "no.loss"),
           row.names, col.names, as.is = !stringsAsFactors,
           na.strings = "NA", colClasses = NA, nrows = -1,
           skip = 0, check.names = TRUE, fill = !blank.lines.skip,
           strip.white = FALSE, blank.lines.skip = TRUE,
           comment.char = "#",
           allowEscapes = FALSE, flush = FALSE,
           stringsAsFactors = default.stringsAsFactors(),
           fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)
```

Importar datos en R

- `file` = ruta al archivo de datos que queremos cargar
- `header` = si tenemos (`TRUE`) o no (`FALSE`) encabezado en el archivo
- `sep` = que carácter separa las columnas/variables: espacio (“ ” por defecto), tabulado(“`\t`”), “`;`” etc
- `dec` = los decimales van separados por puntos (`.`) o comas (`,`)
- `skip` = número de filas que no queremos leer
- `na.strings` = como se codifican los datos faltantes (`NA` por defecto)

Importar datos en R

```
R> d1 <- read.table("datos/heightANDweight.txt",
+                   header = TRUE)

R> head(d1,3)
     id genero altura_cm peso_Kg
1 id001   male     165.6   86.36
2 id002   male     176.0   62.53
3 id003   male     175.1   65.00

R> str(d1)
'data.frame': 200 obs. of  4 variables:
 $ id        : chr  "id001" "id002" "id003" "id004" ...
 $ genero    : chr  "male" "male" "male" "male" ...
 $ altura_cm: num  166 176 175 171 179 ...
 $ peso_Kg   : num  86.4 62.5 65 76.5 72.4 ...
```

Ruta a la carpeta en donde tenemos los datos

Importar datos en R

- Es importante, por tanto, saber de antemano el formato de nuestro archivo de datos. Para saberlo, basta con abrir el archivo
- Sin embargo, en ocasiones el archivo puede ser muy pesado (ej: archivo de genotipos con 700.000 columnas y 1000 filas) y no se puede abrir. En estos casos es útil usar otra herramientas del sistema como el `terminal`
- Es aconsejable incluir un archivo `README.txt` en nuestra carpeta de proyecto indicados además de los análisis, qué bases de datos tenemos, con qué formato, que variables se incluyen y en qué unidades

Importar datos en R

- Es muy probable que el archivo de datos que con más frecuencia trabajéis es una hoja Excel
- Para leer datos de Excel, usamos funciones del paquete `readxl`
- Como ya hemos visto, primero tenemos que instalar el paquete (si no lo tenemos), y…
- luego lo cargamos con la función `library()`, o usando los menús de la pestaña packages de RStudio
- Una vez cargado podemos usar la función `read_excel()`

Importar datos en R

```
?read_excel


read_excel(path, sheet = NULL, range = NULL, col_names = TRUE,
  col_types = NULL, na = "", trim_ws = TRUE, skip = 0,
  n_max = Inf, guess_max = min(1000, n_max),
  progress = readxl_progress(), .name_repair = "unique")
```

- `path` =es la ruta al directorio donde está la hoja excel
- `sheet` = es la hoja que queremos leer
- `col_names` = (`TRUE` o `FALSE`) si debe leer la primera fila como nombre de variables
- `na` = que caracter se usa para los datos faltantes (celda vacía por defecto)

Importar datos en R

```
R> R> d3 <- read_excel("datos/heightANDweight.xlsx")

R> head(d3,3)
# A tibble: 6 x 4
  id    genero altura_cm peso_Kg
  <chr> <chr>      <dbl>   <dbl>
1 id001 male        166.    86.4
2 id002 male        176.    62.5
3 id003 male        175.    65.0

R> str(d3)
tibble [200 × 4] (S3: tbl_df/tbl/data.frame)
 $ id       : chr [1:200] "id001" "id002" "id003" "id004" ...
 $ genero   : chr [1:200] "male" "male" "male" "male" ...
 $ altura_cm: num [1:200] 166 176 175 171 179 ...
 $ peso_Kg  : num [1:200] 86.4 62.5 65 76.5 72.4 ...
```

Es un tipo de base de datos

TIDYVERSE

Exportar datos desde R

- Para guardar datos en el disco vamos a usar la función `write.table()`
- A igual que cuando importábamos datos, tenemos que especificar el formato que queremos usar
- Este formato vendrá determinado por el uso que le vayamos a dar a la base de datos que guardamos: usar en otros programas, crear una tabla para un informe, …
- Además podemos guardar en otros formatos. Para guardar hojas de excel, vamos a usar la función `write_xlsx()` del paquete `writexl`

Exportar datos desde R

- Vamos con la ayuda de la función write.table()

```
?write.table


write.table(x, file = "", append = FALSE, quote = TRUE, sep = " ",
            eol = "\n", na = "NA", dec = ".", row.names = TRUE,
            col.names = TRUE, qmethod = c("escape", "double"),
            fileEncoding = "")
```

Exportar datos desde R

- `file` = ruta al archivo de datos que queremos cargar
- `append` = si escribe (`TRUE`) o no (`FALSE`) a continuación de un archivo que ya existe. Es decir, añade datos a un archivo existente
- `quote` = si usa comillas (`TRUE`) o no (`FALSE`) para los caracteres
- `sep` = que separador de campo usar: espacio (“” por defecto), tabulado(“`\t`”), “`;`”…
- `dec` = los decimales van separados por puntos (`.`) o comas (`,`)
- `row.names/col.names` = se escribe los nombres de filas/columnas
- `na` = como se codifican los datos faltantes (`NA` por defecto)

Guardar el espacio de trabajo. RData

- R permite guardar todo el espacio de trabajo (lo que tenemos cargado en R en ese momento), o parte de él
- Las funciones que usaremos serán `save.image()` y `save()`
- Se crea un archivo de extensión .RData
- Para cargar el espacio de trabajo, usamos la función `load()`
- Estas funciones son muy interesantes para salvar pasos intermedios de una análisis