uc3m | Universidad Carlos III de Madrid

Curso OpenCourseWare

Aprendizaje del Software Estadístico R: un entorno para simulación y computación estadística

Alberto Muñoz García

---

7. Lectura y escritura de datos en R

Introducción de datos desde terminal y fichero

Para introducir datos desde la propia ventana de comandos se utiliza la función **scan():**

> scan()

1: 1

2: 2

3: 3

4:

Read 3 items

[1] 1 2 3

Como vemos, cuando queremos dejar de introducir datos, pulsamos un ENTER de más.

Si queremos guardar el resultado en una variable, escribiríamos algo como:

> datos = scan()

La funcion scan toma la siguiente forma:

scan(file=" ", what = numeric(), n, sep, ...)

Veamos algunos ejemplos:

datos = scan("c:\\datos\\datos.txt")	Recuperaría un hipotético fichero de datos, denominado datos.txt situado en el directorio C:\DATOS, y lo almacenaría en la variable datos
> nombres = scan(,what=character(),3) **1: pepe** **2: paco** **3: pipo** **Read 3 items** **> nombres** **[1] "pepe" "paco" "pipo"**	Crea un vector de 3 cadenas de caracteres.

Supongamos que tenemos un fichero denominado pocosdatos.txt y que lo queremos leer y convertir en matriz:

```
> datos = scan("c:\\cursoada\\pocosdatos.txt",sep=",")
Read 9 items
> datos
[1] 1 2 3 0 1 2 4 5 6
```

Observemos que hemos puesto **sep=","** para indicar que los datos vienen separados por comas.

Para darle forma de matriz:

```
> datos = matrix(datos,ncol=3,byrow=T)
> datos
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    0    1    2
[3,]    4    5    6
```

Lectura de tablas desde ficheros

A veces nos será de utilidad la función **read.table().**

Supongamos que tenemos un fichero llamado edadaltura.txt, y que queremos leerlo con una sola función.

```
> datos = read.table("c:\\cursoada\\edadaltura.txt")
> datos
     edad altura
paco   20    174
pepe   22    180
kiko   19    170
```

> colnames(datos)

[1] "edad" "altura"

> rownames(datos)

[1] "paco" "pepe" "kiko"

Como vemos, el sistema automáticamente asigna nombres a filas y columnas.

La estructura general de la orden es**:**

read.table(file, header = F, sep = "", ...)

Si deseamos especificar que la primera fila contiene los nombres de las variables pondremos header=T.

La orden análoga para escribir una tabla en disco es:

write.table(x, file = "", append = FALSE, quote = TRUE, sep = " ", row.names = TRUE, col.names = TRUE)

Veamos algunos ejemplos sobre la tabla datos creada anteriormente:

> write.table(datos,"c:\\cursoada\\datos1.txt")	**C:\cursoADA>more datos1.txt** **"edad" "altura"** **"paco" 20 174** **"pepe" 22 180** **"kiko" 19 170**
> write.table(datos,"c:\\cursoada\\datos1.txt",quote=F)	**C:\cursoADA>more datos1.txt** **edad altura** **paco 20 174** **pepe 22 180** **kiko 19 170**
>write.table(datos,"c:\\cursoada\\datos1.txt",quote=F,row.names=F)	**C:\cursoADA>more datos1.txt** **edad altura** **20 174** **22 180** **19 170**

Escritura de ficheros de texto

Aparte de utilizar la función **write.table()** , podemos utilizar la función **write()**.

Supongamos que queremos guardar en un fichero el contenido de un vector:

```
> x<-c(1,2,3,4,5)
> write(x,"c:\\cursoada\\x.txt")
```

El contenido del fichero **x.txt** será el vector 1 2 3 4 5

Si lo que queremos es escribir una matriz con este procedimiento, hay que ser más cuidadosos.

Creamos una matriz para probar:

```
> x<-matrix(1:9,ncol=3,byrow=T)
> x
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

La escribimos en el fichero x.txt:

```
> write(t(x),"c:\\cursoada\\x.txt",ncol=ncol(x))
```

La función write() actúa escribiendo una columna de la matriz cada vez, por lo que primero deberemos trasponer. El número de columnas en este caso es 3, pero indicando ncol(x) nos aseguramos de acertar siempre.