

“Introducción a la Estadística, Probabilidad e Inferencia”

Maestría en Estadística Aplicada
Facultad de Ciencias Económicas y Estadística
UNR

Introducción a R

- ¿Qué es R? ¿Dónde se consigue?
- ¿Cómo se trabaja en R? Primeros pasos. Ayuda y Paquetes.
- Objetos. Tipo, modo y atributos. Estructuras.
- Manipulación, lectura y grabación de datos.

¿Qué es R?

- R es un entorno de programación, análisis estadístico y software gráfico derivado del lenguaje de programación S (Becker, Chambers y Wilks, 1988; Chambers, 1998; Chambers y Hastie, 1992; Venables y Ripley, 2000).
- R es un software libre que se alimenta y crece con los trabajos de investigadores provenientes de prácticamente todas las ramas del conocimiento.
- Este software se difundió rápidamente y su expansión es irrefrenable.

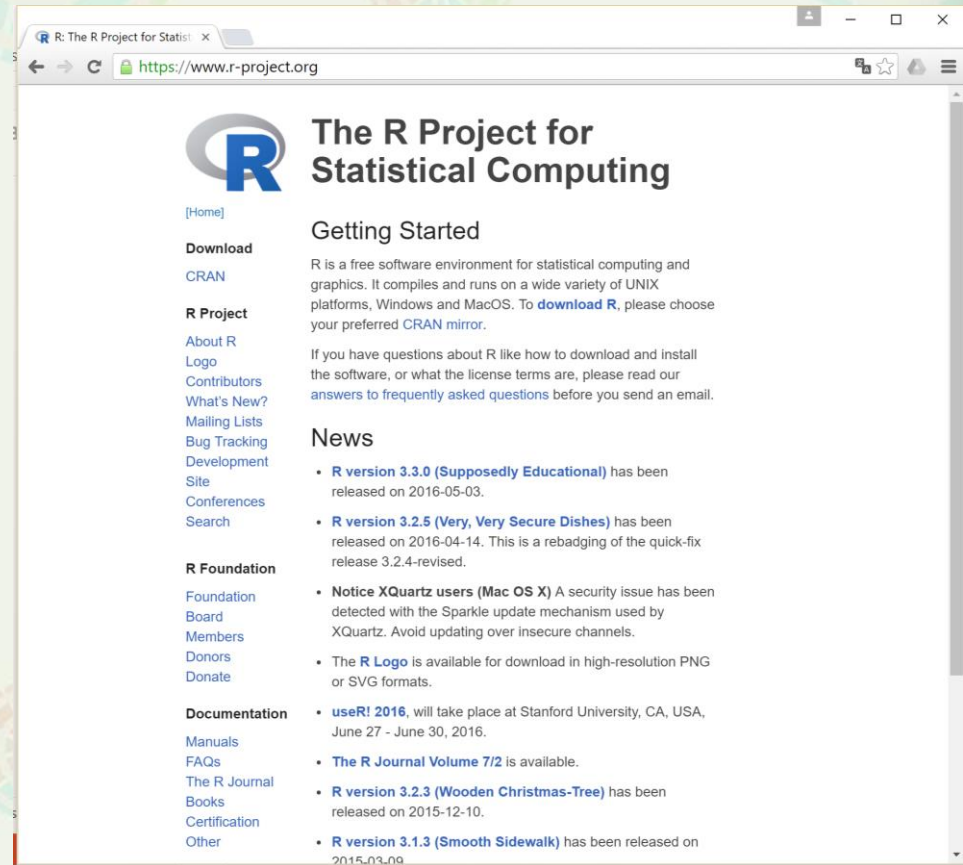
¿Qué es R?

R integra una multitud de paquetes cuya continua incorporación al entorno incrementan su capacidad y versatilidad.

- R dispone de funciones básicas relacionadas con los análisis descriptivos de datos y de los modelos más complejos y actuales concernientes con los últimos avances en el campo de la estadística, la econometría o el análisis de datos en áreas como psicología, economía, sociología, biología, medicina, informática, etc.
- Aparte de las capacidades de análisis estadístico, R es un potentísimo generador de gráficos.

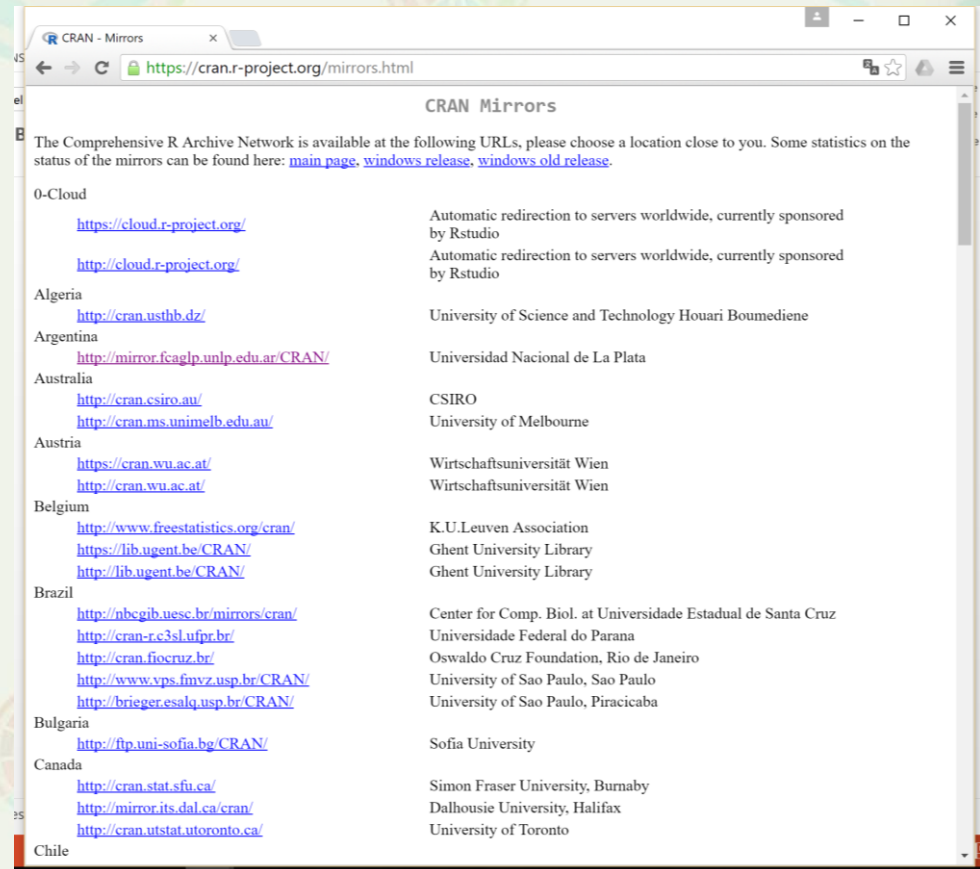
¿Dónde se consigue R?

El archivo de instalación
se encuentra en la página
<http://www.r-project.org>



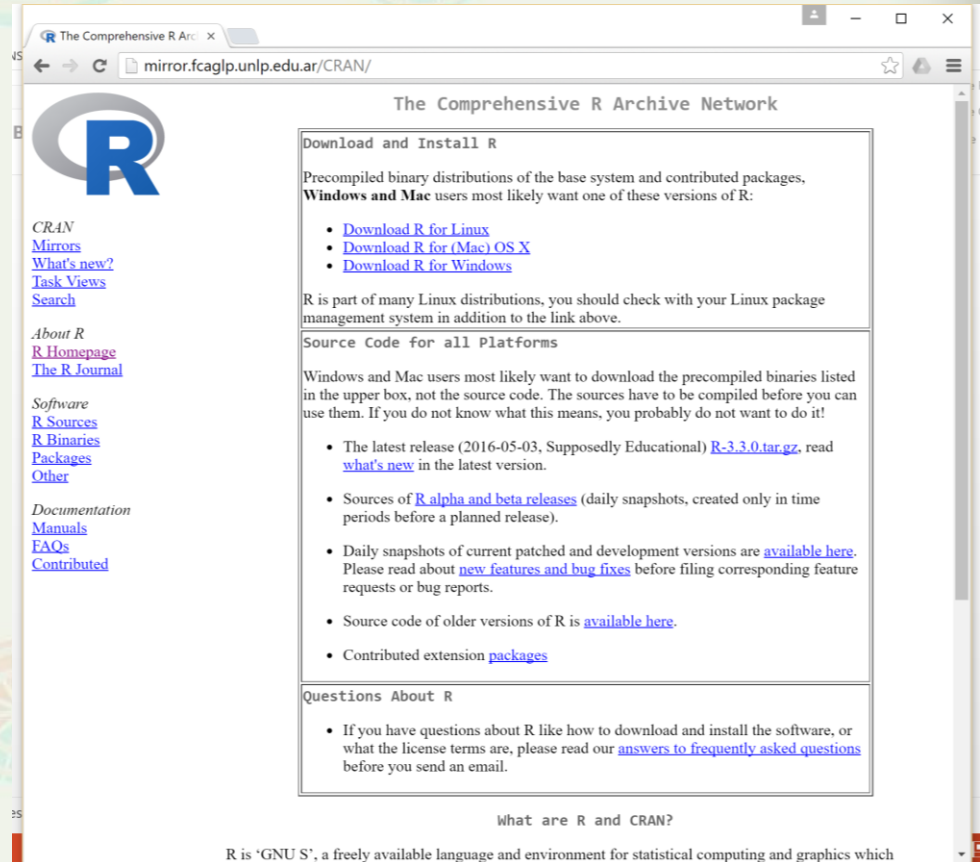
¿Dónde se consigue R?

Al hacer click en download, la página redirecciona a otra en donde se listan todos los URL donde se puede descargar R.



¿Dónde se consigue R?

En la siguiente página se debe elegir la plataforma adecuada (Linux, MacOS X, Windows).



The screenshot shows the CRAN (Comprehensive R Archive Network) website. The browser address bar shows 'mirror.fcaglp.unlp.edu.ar/CRAN/'. The page title is 'The Comprehensive R Archive Network'. The main content area is titled 'Download and Install R' and contains the following text: 'Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:'. Below this, there are three bullet points with links: 'Download R for Linux', 'Download R for (Mac) OS X', and 'Download R for Windows'. The text continues: 'R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.' Below this, there is a section titled 'Source Code for all Platforms' with the text: 'Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!'. Below this, there are four bullet points with links: 'The latest release (2016-05-03, Supposedly Educational) [R-3.3.0.tar.gz](#), read [what's new](#) in the latest version.', 'Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).', 'Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.', and 'Source code of older versions of R is [available here](#).' Below this, there is a section titled 'Questions About R' with one bullet point: 'If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.' At the bottom of the page, there is a footer that says 'What are R and CRAN?' and 'R is 'GNU S', a freely available language and environment for statistical computing and graphics which'.

CRAN
[Mirrors](#)
[What's new?](#)
[Task Views](#)
[Search](#)

About R
[R Homepage](#)
[The R Journal](#)

Software
[R Sources](#)
[R Binaries](#)
[Packages](#)
[Other](#)

Documentation
[Manuals](#)
[FAQs](#)
[Contributed](#)

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2016-05-03, Supposedly Educational) [R-3.3.0.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

Questions About R

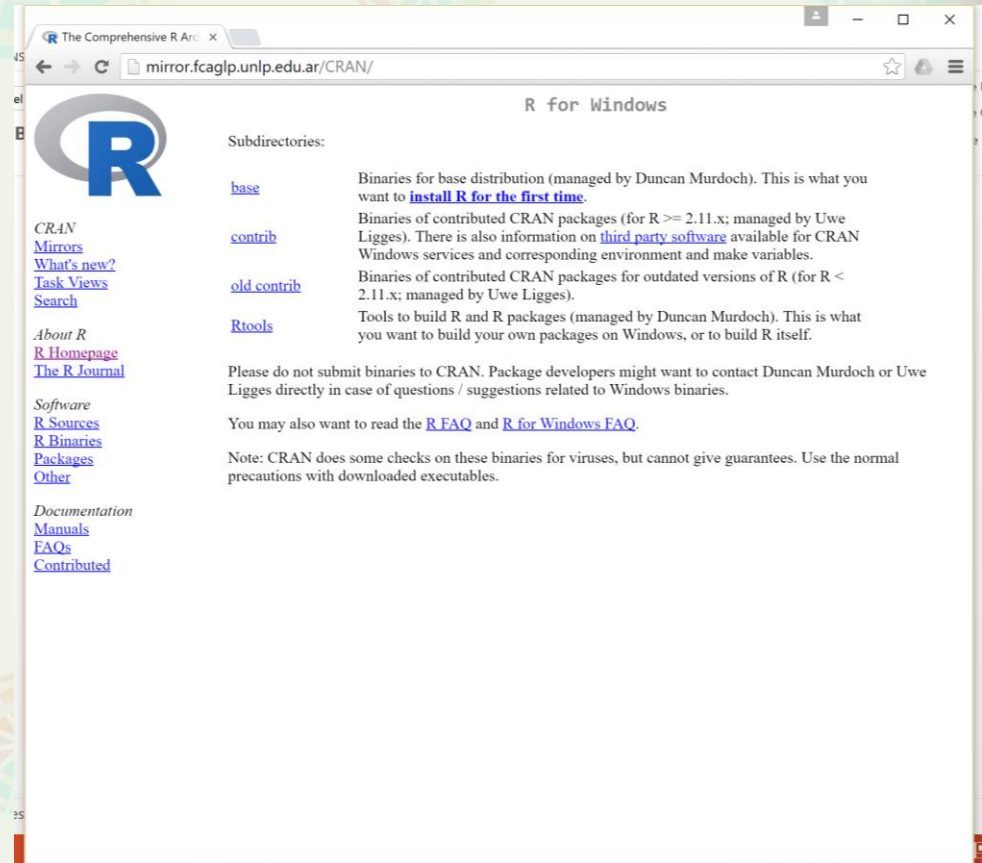
- If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

What are R and CRAN?

R is 'GNU S', a freely available language and environment for statistical computing and graphics which

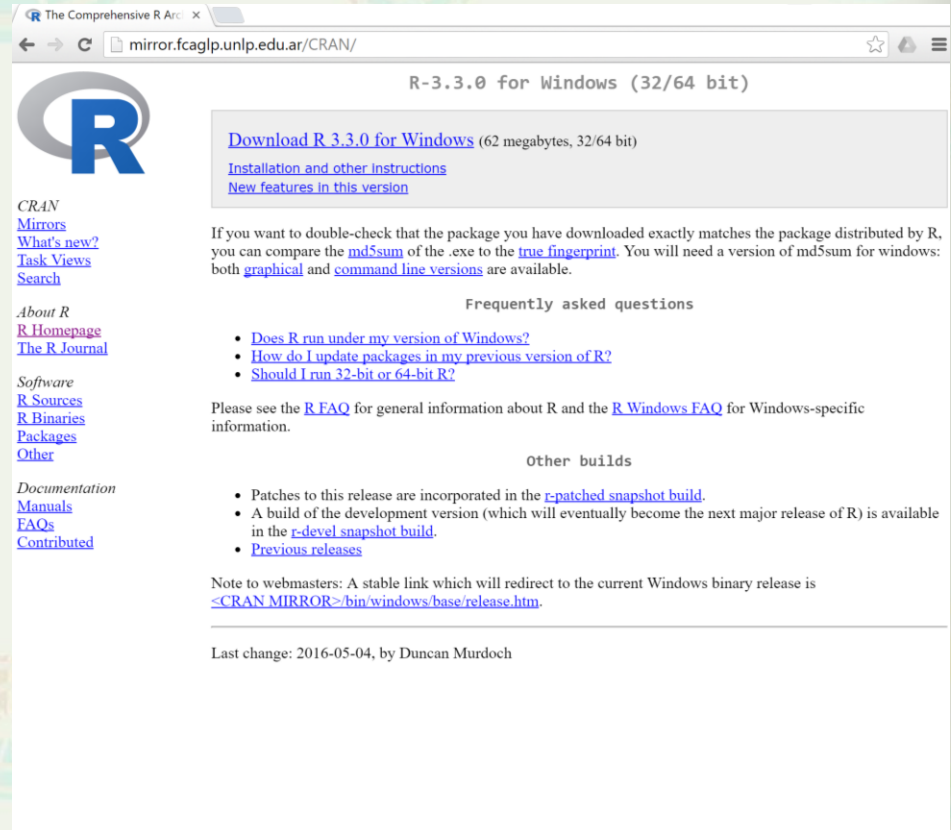
¿Dónde se consigue R?

Se accede a la siguiente pantalla en la que debe seleccionarse la opción base.



¿Dónde se consigue R?

Finalmente se accede a la pagina desde la que se descarga la versión base de R.



The screenshot shows a web browser window with the address bar displaying 'mirror.fcaglp.unlp.edu.ar/CRAN/'. The page title is 'The Comprehensive R Archive Network'. The main heading is 'R-3.3.0 for Windows (32/64 bit)'. Below this, there is a section for downloading R 3.3.0 for Windows (62 megabytes, 32/64 bit), with links for 'Download R 3.3.0 for Windows', 'Installation and other instructions', and 'New features in this version'. A paragraph explains how to double-check the package download by comparing the md5sum of the .exe file to the true fingerprint, and mentions that graphical and command line versions are available. A 'Frequently asked questions' section lists three questions: 'Does R run under my version of Windows?', 'How do I update packages in my previous version of R?', and 'Should I run 32-bit or 64-bit R?'. Below this, a note refers to the R FAQ for general information and the R Windows FAQ for Windows-specific information. An 'Other builds' section lists three items: patches to this release, a build of the development version, and previous releases. A note to webmasters provides a stable link to the current Windows binary release. The page footer indicates the last change was on 2016-05-04 by Duncan Murdoch.

CRAN
[Mirrors](#)
[What's new?](#)
[Task Views](#)
[Search](#)

About R
[R Homepage](#)
[The R Journal](#)

Software
[R Sources](#)
[R Binaries](#)
[Packages](#)
[Other](#)

Documentation
[Manuals](#)
[FAQs](#)
[Contributed](#)

R-3.3.0 for Windows (32/64 bit)

[Download R 3.3.0 for Windows](#) (62 megabytes, 32/64 bit)
[Installation and other instructions](#)
[New features in this version](#)

If you want to double-check that the package you have downloaded exactly matches the package distributed by R, you can compare the [md5sum](#) of the .exe to the [true fingerprint](#). You will need a version of md5sum for windows: both [graphical](#) and [command line versions](#) are available.

Frequently asked questions

- [Does R run under my version of Windows?](#)
- [How do I update packages in my previous version of R?](#)
- [Should I run 32-bit or 64-bit R?](#)

Please see the [R FAQ](#) for general information about R and the [R Windows FAQ](#) for Windows-specific information.

Other builds

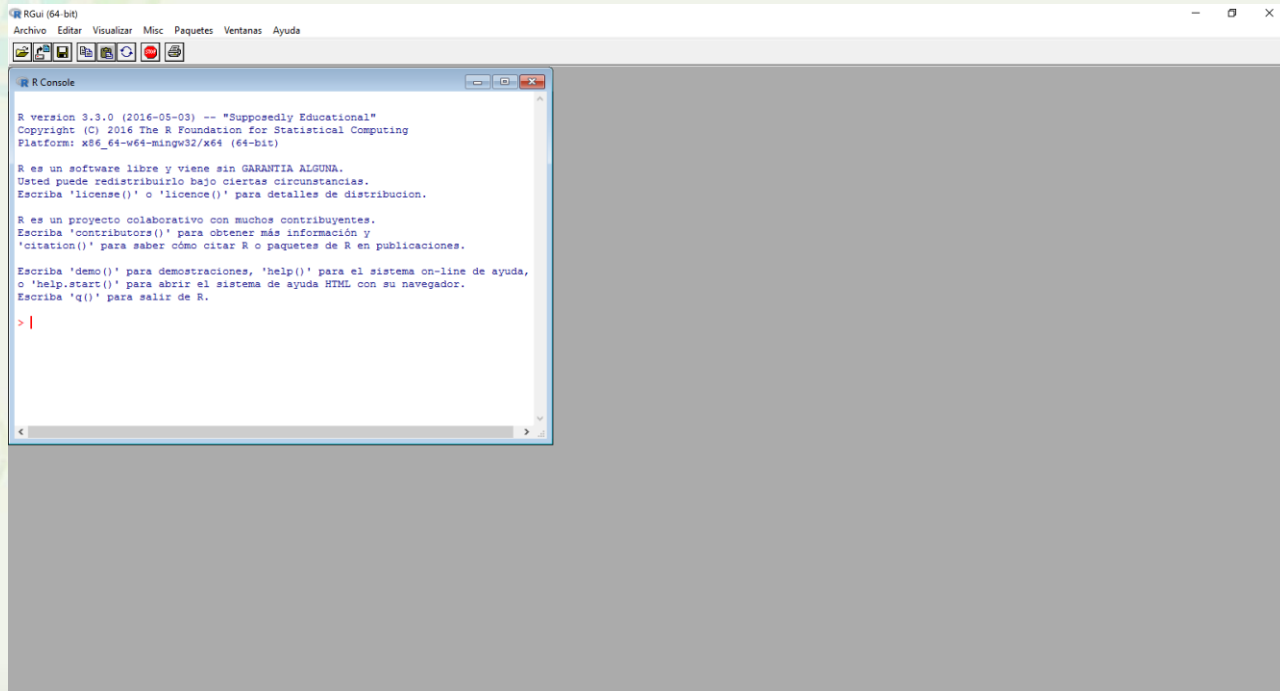
- Patches to this release are incorporated in the [r-patched snapshot build](#).
- A build of the development version (which will eventually become the next major release of R) is available in the [r-devel snapshot build](#).
- [Previous releases](#)

Note to webmasters: A stable link which will redirect to the current Windows binary release is [<CRAN MIRROR>/bin/windows/base/release.htm](#).

Last change: 2016-05-04, by Duncan Murdoch

¿Cómo se trabaja con R?

Al ejecutar R, aparece una pantalla llamada *consola de R*. En la misma puede comprobarse la versión instalada. Además muestra el símbolo del sistema o *prompt* en color que indica que R está listo para recibir comandos.



```
RGui (64-bit)
Archivo  Editar  Visualizar  Misc.  Paquetes  Ventanas  Ayuda

R Console

R version 3.3.0 (2016-05-03) -- "Supposedly Educational"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribución.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> |
```

¿Cómo se trabaja con R?

- La interfaz de R provee escasas opciones. R no dispone de menús desplegables para el análisis de datos o para la generación de gráficos. A través a la barra de menús se accede a funciones rudimentarias.
- El entorno R está configurado en principio sobre una interfaz de comandos donde el usuario escribe las instrucciones y comandos que desea ejecutar.
- Sin dudas que las interfaces gráficas compuestas por sistemas de ventanas y menús desplegables son útiles y convenientes. Sin embargo, las posibilidades que ofrecen las interfaces de comandos son mayores y no están limitadas más que por la habilidad del programador.

¿Cómo se trabaja con R?

- El uso de comandos exige mayor esfuerzo para el aprendizaje que el requerido por las interfaces gráficas pero las ganancias en términos de independencia, control y creatividad no son comparables.
- Escribir un código supone una comprensión más profunda de aquello que se desea aplicar.
- La redacción de códigos se puede realizar de varias maneras:

¿Cómo se trabaja con R?

- Trabajar directamente sobre la consola de R tecleando los comandos en la línea de «símbolo de sistema».
 - Aquí sólo puede ejecutarse una línea por vez por lo tanto sólo resulta útil para ejecutar acciones simples.
 - La flecha hacia arriba permite acceder a comandos previos y la flecha hacia abajo a comandos posteriores al actual.
 - Las flechas hacia la izquierda y hacia la derecha mueven el cursor en la respectiva dirección.
 - Los comandos pueden copiarse y pegarse como en cualquier editor de textos con las funciones Ctrl+c y Ctrl+v.

¿Cómo se trabaja con R?

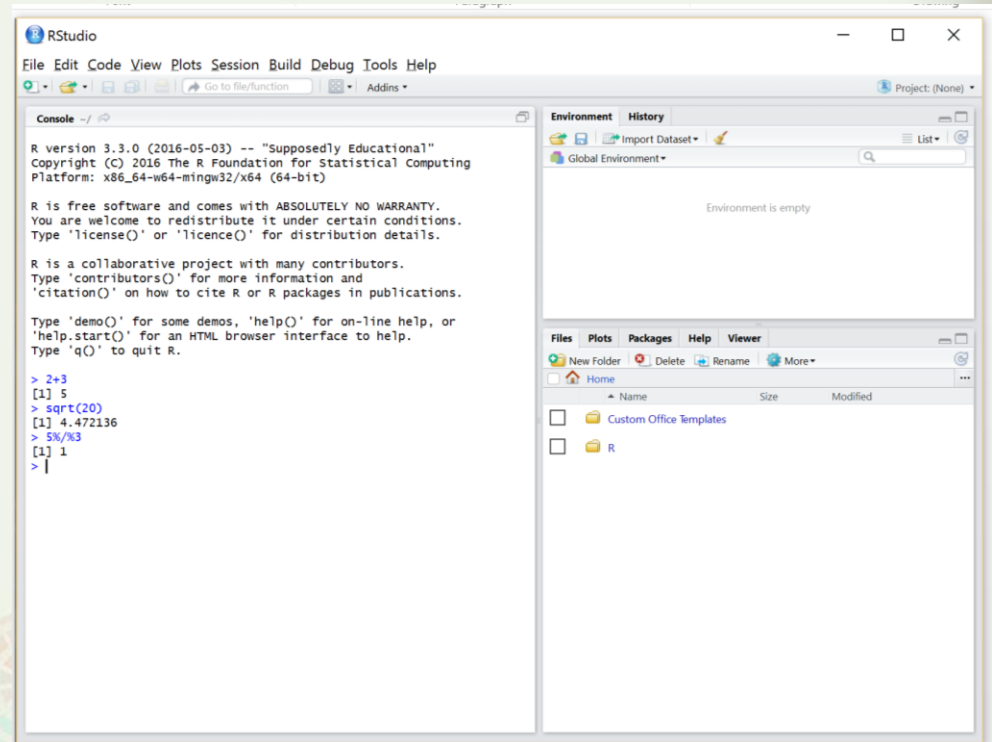
- Trabajar en una ventana de escritura (Script) a la que se accede mediante *Archivo>>Nuevo Script*. Esta opción permite crear códigos más complejos.
 - Los códigos se teclean directamente en esta ventana o puede ser copiados de un archivo existente en formato ASCII.
 - Los códigos se ejecutan por línea o por bloques para lo que debe seleccionarse y pulsar Ctrl+r o el ícono de ejecución.
 - Las salidas se muestran en la consola.

¿Cómo se trabaja con R?

- Trabajar a través de otra interfaz. Existen programas específicos para la creación/edición de códigos. Entre los más utilizados:
 - Tinn-R
 - Rstudio
 - WinEdit
- La construcción de códigos permite al usuario diseñar y programar sus propias funciones. La programación significa controlar el ordenador por medio de instrucciones que indican que operaciones llevar a cabo sobre los datos, que imprimir, que guardar, que resolver y como diseñar los gráficos.

Primeros pasos en R

- El modo más simple o primitivo de trabajar con R sería utilizarlo como una potente calculadora. En este sentido, R evalúa y devuelve los resultados de cualquier expresión introducida en la línea de comandos.



Primeros pasos en R

- Algunas funciones algebraicas

Función	Operación
<code>+</code> , <code>-</code> , <code>*</code> , <code>/</code>	Suma, Resta, Multiplicación, División
<code>sin</code> , <code>cos</code> , <code>tan</code>	Funciones trigonométricas
<code>asin</code> , <code>acos</code> , <code>atan</code>	Funciones trigonométricas inversas
<code>exp</code> , <code>log</code>	Exponencial y logaritmo natural
<code>sqrt()</code> , <code>^</code>	Raíz cuadrada, potencia
<code>abs</code>	Valor absoluto
<code>round</code>	Redondeo
<code>%/%</code>	División entera
<code>%%</code>	Resto de la división

Primeros pasos en R

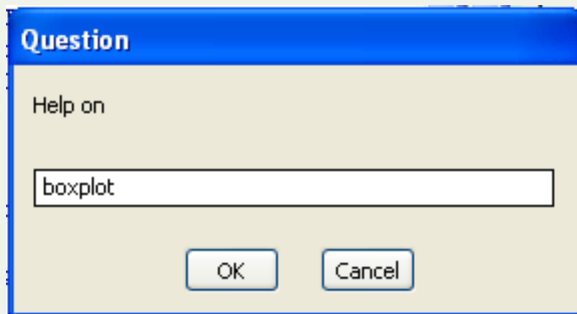
- Algunas observaciones:
 - Si en lugar del símbolo `>`, la consola de R muestra el símbolo `+`, indica que la instrucción dada a R está incompleta y no se puede ejecutar sin antes finalizar correctamente la secuencia de comandos.
 - Ante la presencia del símbolo `+`, si se pulsa la tecla Esc permite volver al símbolo del sistema.
 - Es posible escribir más de una función en la misma línea de comandos utilizando el carácter `«;»` para delimitarlas.
 - El símbolo `«#»` permite añadir a los códigos comentarios no ejecutables.

Primeros pasos en R

- Para abandonar R tras una sesión de trabajo se puede:
 - Teclear en la consola `q()`
 - Desde la barra de menús Archivo >> Salir
 - Pulsando el icono de salida.

Primeros pasos en R: Ayuda

- R dispone de varias fuentes de ayuda sobre procedimientos, comandos, paquetes o sobre la aplicación de modelos formales.
- Ayuda sobre funciones
 - *Ayuda* >> *funciones R* (texto) Abre una ventana donde se tipea la función acerca de la que se busca información.



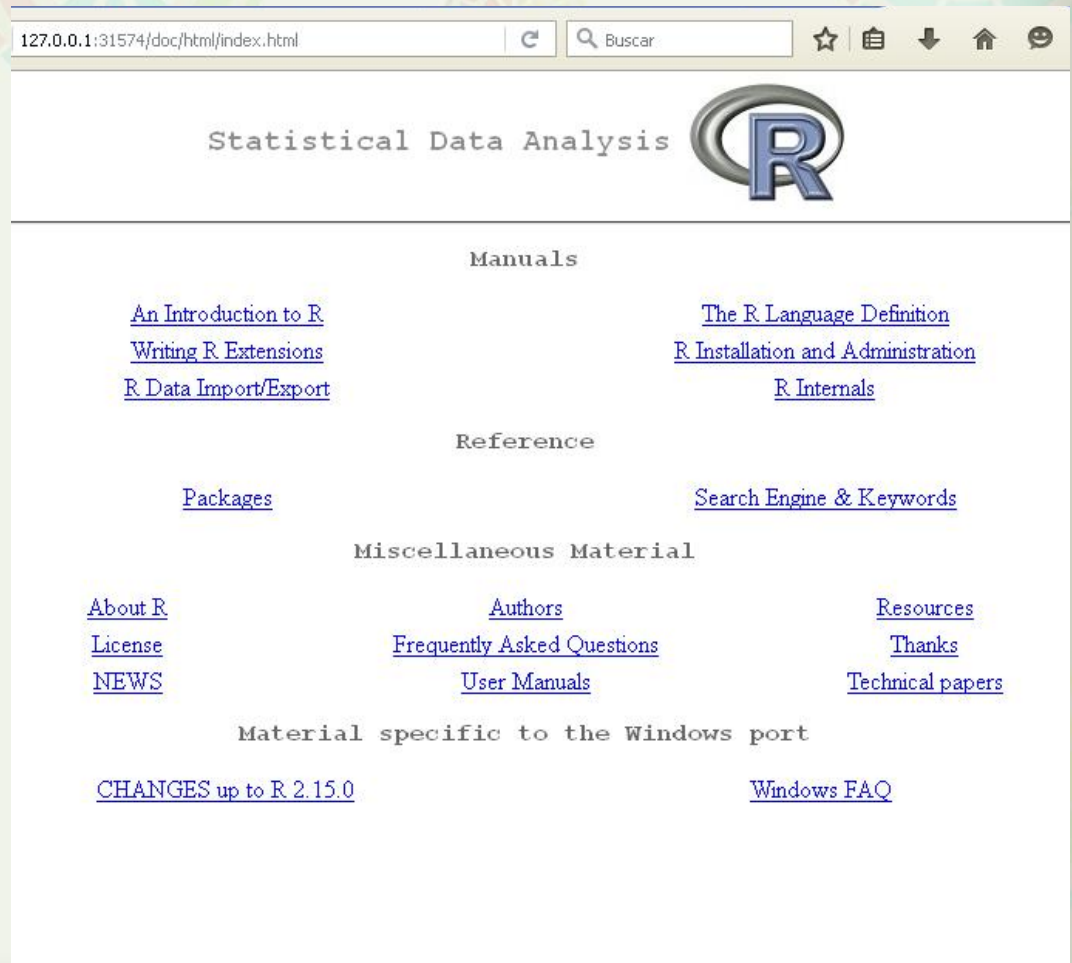
- Este modo de obtener información es equivalente a tipear «help(boxplot)» o «?boxplot» en la consola

Primeros pasos en R: Ayuda

– *Ayuda >> Html help*

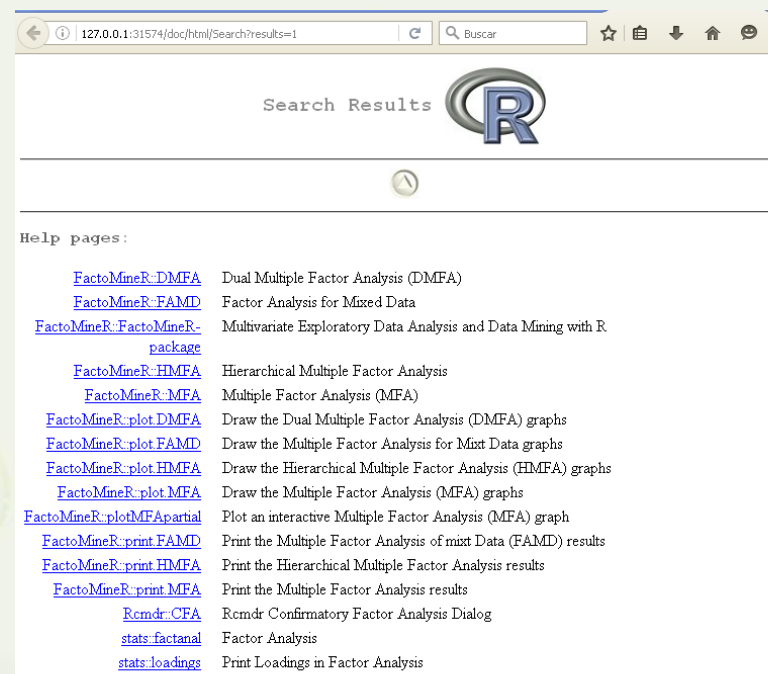
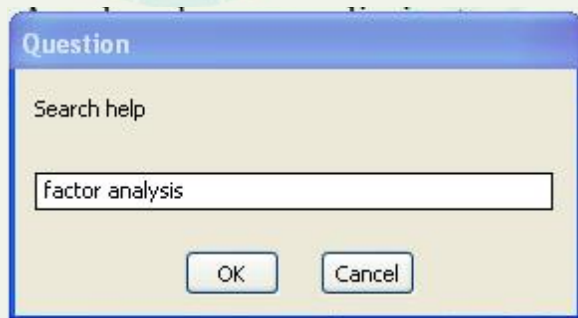
Abre el explorador definido por defecto y muestra una pantalla a través de la cual se accede a la información almacenada en la memoria.

Se puede acceder a la misma página a través del comando «`help.start()`»



Primeros pasos en R: Ayuda

- Ayuda sobre procedimiento o modelos
 - *Help*>>*Search help* o tipear «`help.search()`» en la consola: ofrece una lista de todas las funciones cuyas páginas de ayuda contienen la palabra buscada.



Primeros pasos en R: Ayuda

- Si se hace click sobre el nombre de alguna de dichas funciones

The screenshot shows a web browser window displaying the R help page for the `factanal` function. The address bar shows the URL `127.0.0.1:15293/library/stats/html/factanal.html`. The page title is "Factor Analysis". The content includes a "Description" section, a "Usage" section with the function signature, and an "Arguments" section. Annotations with blue circles and arrows highlight the `factanal` function name and the `{stats}` package name, with labels "Paquete" and "Función" pointing to them respectively.

← → ↻ 127.0.0.1:15293/library/stats/html/factanal.html

`factanal` `{stats}` → Paquete

Función

Factor Analysis

Description

Perform maximum-likelihood factor analysis on a covariance matrix or data matrix.

Usage

```
factanal(x, factors, data = NULL, covmat = NULL, n.obs = NA,
         subset, na.action, start = NULL,
         scores = c("none", "regression", "Bartlett"),
         rotation = "varimax", control = NULL, ...)
```

Arguments

x
A formula or a numeric matrix or an object that can be coerced to a numeric matrix.

factors
The number of factors to be fitted.

data

Primeros pasos en R: Ayuda

- Los comandos anteriores ofrecen información sobre funciones que están incluidas en los paquetes que tengamos instalados. Sin embargo, en muchas ocasiones el interés es saber si R dispone de funciones para ejecutar determinado tipo de análisis.
- La base de la exploración puede ser ampliada a una búsqueda en la red. Para ello hay que utilizar la opción *Ayuda*>>*search.r.project.org* o tipear «Rsite-Search()» en la consola.

Primeros pasos en R: Paquetes

- R es un sistema integrado al que se añaden complementos que reciben el nombre de paquetes. Un paquete es un conjunto de funciones que mantienen algún tipo de relación entre ellas y que usualmente vienen acompañadas de archivos de ayuda y de ficheros de datos.
- La pluralidad de paquetes va pareja a la pluralidad de áreas de conocimiento, aplicaciones y modelos estadísticos.
- Algunos de los paquetes están incluidos en la instalación básica de R. Al resto de los paquetes se accede a través de repositorios públicos.

Primeros pasos en R: Paquetes

- La instalación de paquetes puede realizarse:
 - desde la barra de menús: *Paquetes>>Instalar paquetes*
 - o desde la consola tipeando: `install.packages('nombre del paquete')`
- Ambas opciones dan paso a una ventana que ofrece un listado de sitios imagen. Una vez seleccionado uno de ellos, se mostraran en pantalla todos los paquetes disponibles ordenados alfabéticamente. Al pulsar sobre el paquete y presionar OK, R descargará e instalará el paquete.

Primeros pasos en R: Paquetes

- Para utilizar las funciones incluidas en un paquete es necesario instalarlo y cargarlo. Un paquete se instala una sola vez pero es necesario cargarlo en cada sesión de trabajo en el que se desee utilizar.
- Para cargar un paquete:
 - desde la barra de menús: *Paquetes>>Cargar paquete*
 - o desde la consola tipeando: `library('nombre del paquete')`
- Si se intenta utilizar una función incluida en un paquete que no ha sido cargado, R devuelve un mensaje de error.

Primeros pasos en R: Paquetes

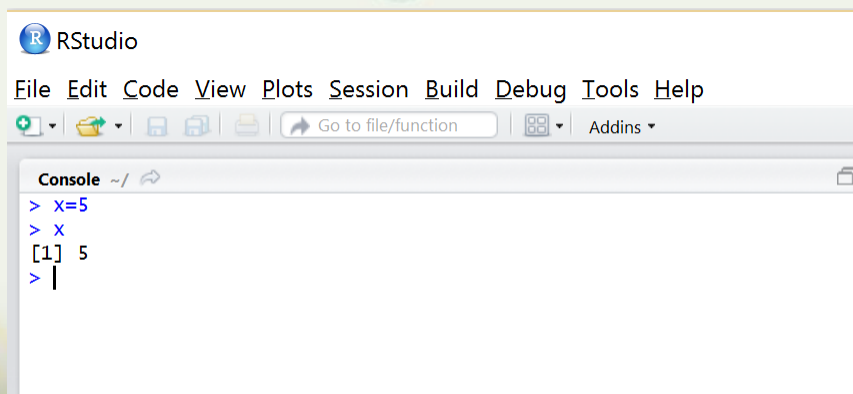
- Algunos comandos para el manejo de paquetes:
 - *installed.packages()*: lista todos los paquetes instalados (excepto los incluidos en la instalación por defecto)
 - *get()option*: lista todos los paquetes instalados por defecto
 - *(.packages())*: lista los paquetes cargados
- El rápido incremento en el número de paquetes disponibles para R ha hecho necesario crear un sistema de clasificación que facilite orientarse entre ellos. Los paquetes han sido catalogados en varios grupos a los que se puede acceder a través de la página CRAN (<http://www.cran.r-project.org/web/views>)

Primeros pasos en R: Paquetes

- La estructura de la información ha sido reducida a categorías:
 - Bayesian
 - ChemPhys
 - ClinicalTrials
 - Cluster
 - Distributions
 - Econometrics
 - Environmetrics
 - ExperimentalDesign
 - Finance
 - Genetics
 - Etc.

Objetos

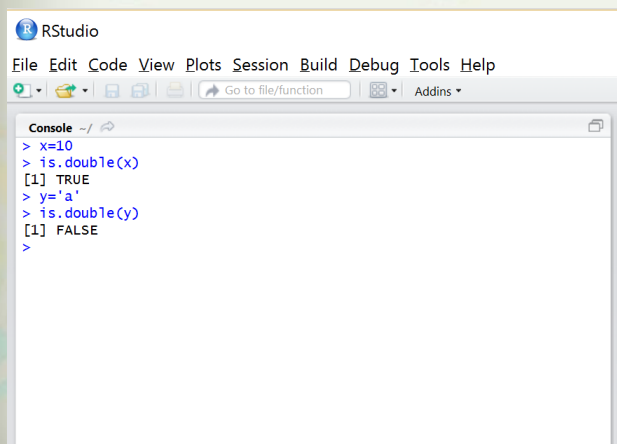
- En R prácticamente todo es definido como un objeto: un dato numérico, un vector, una matriz o una función. Es decir, un objeto es la forma en la que R almacena la información.
- R opera sobre objetos y distingue entre los distintos tipos.
- Cada objeto tiene un nombre, es de un determinado tipo y tiene ciertos atributos.
- Para almacenar un objeto es necesario utilizar la función de asignación: `-<` o `=`



```
RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function Addins
Console ~/
> x=5
> x
[1] 5
> |
```

Objetos

- Los nombres de objetos pueden contener letras, números, guiones, punto pero no pueden comenzar con un número.
- R distingue entre mayúsculas y minúsculas.
- Un objeto puede ser almacenado bajo diferentes formas. Los tipos de almacenamiento más comunes son dobles, enteros, complejos, lógicos y carácter.



```
RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function Addins
Console ~/
> x=10
> is.double(x)
[1] TRUE
> y='a'
> is.double(y)
[1] FALSE
>
```

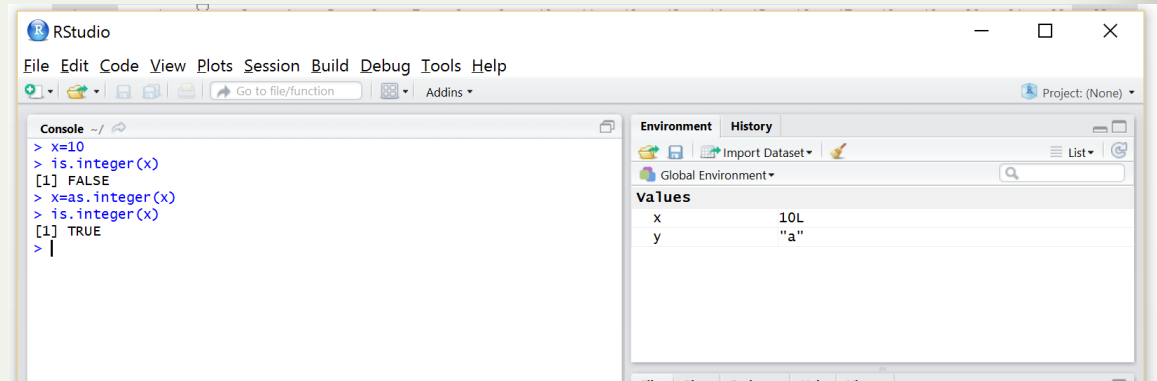
– **Dobles:** valores numéricos continuo. Es el tipo almacenamiento utilizado por defecto por R para representar números.

Para verificar si un determinado dato es de tipo doble utilizar la instrucción: *is.double()*.

Tipos de Objetos

- **Enteros:** valores numéricos no continuo.

Para definir un valor numérico como entero se puede utilizar la instrucción:
as.integer().



The screenshot shows the RStudio interface. The console on the left contains the following code and output:

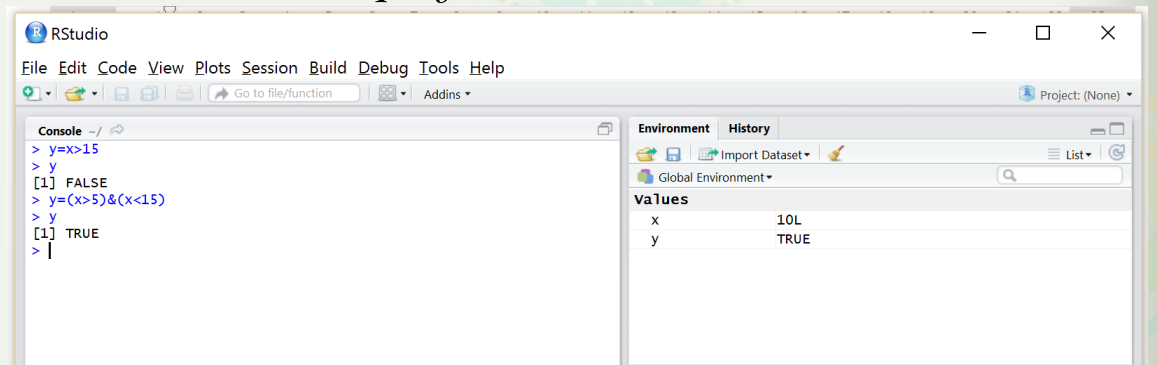
```
> x=10
> is.integer(x)
[1] FALSE
> x=as.integer(x)
> is.integer(x)
[1] TRUE
> |
```

The Environment pane on the right shows the Global Environment with two variables:

Values	
x	10L
y	"a"

- **Complejos:** valores numéricos complejos.

- **Lógicos:** valores ‘verdadero’ o ‘falso’ (son generados por R tras evaluar expresiones lógicas).



The screenshot shows the RStudio interface. The console on the left contains the following code and output:

```
> y=x>15
> y
[1] FALSE
> y=(x>5)&(x<15)
> y
[1] TRUE
> |
```

The Environment pane on the right shows the Global Environment with two variables:

Values	
x	10L
y	TRUE

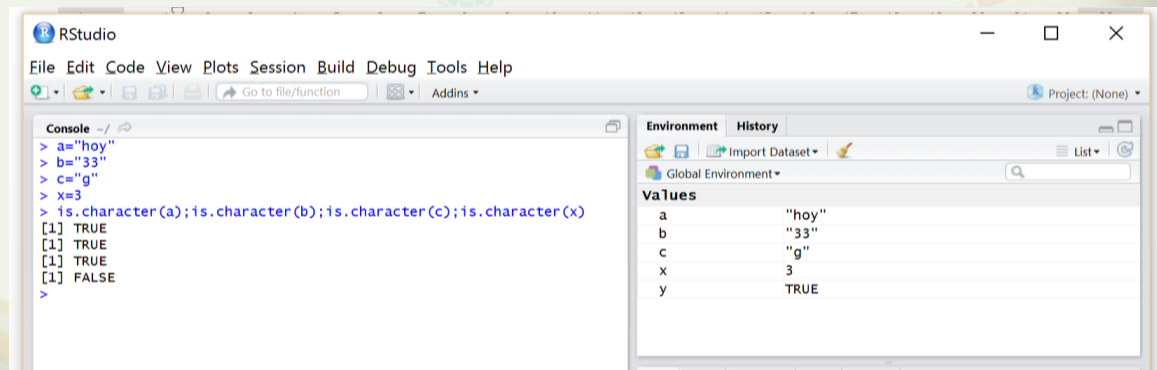
Tipos de Objetos

- Algunos operadores lógicos para construir expresiones son:

Expresiones lógicas		Operadores lógicos	
<, <=	Menor que, Menor o igual que	&	“y”
>, >=	Mayor que, Mayor o igual que		“o”
==	Igual a	!	“no”
!=	No igual a		

- **Carácter:** valores no numéricos

Se escriben y representan entre comillas.



The screenshot shows the RStudio interface. The Console window displays the following code and output:

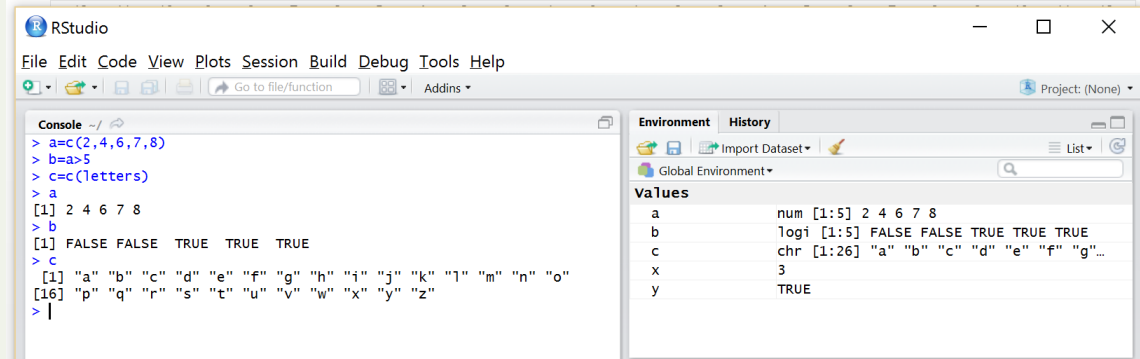
```
> a="hoy"
> b="33"
> c="g"
> x=3
> is.character(a);is.character(b);is.character(c);is.character(x)
[1] TRUE
[1] TRUE
[1] TRUE
[1] FALSE
>
```

The Environment window on the right shows the values of the variables:

Variable	Value
a	"hoy"
b	"33"
c	"g"
x	3
y	TRUE

Estructura de Objetos: Vectores

- Las estructuras sobre las que trabaja R son: vectores; factores; matrices; arrays; marcos de Datos; listas; funciones.
- Un **vector** es un objeto unidimensional constituido por elementos del mismo tipo. Los elementos de un vector o modo pueden ser de tipo numérico, lógico o carácter.



The screenshot shows the RStudio interface. The console on the left displays the following commands and output:

```
> a=c(2,4,6,7,8)
> b=a>5
> c=c(letters)
> a
[1] 2 4 6 7 8
> b
[1] FALSE FALSE TRUE TRUE TRUE
> c
[1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o"
[16] "p" "q" "r" "s" "t" "u" "v" "w" "x" "y" "z"
> |
```

The Environment pane on the right shows the following objects:

Object	Mode	Values
a	num [1:5]	2 4 6 7 8
b	logi [1:5]	FALSE FALSE TRUE TRUE TRUE
c	chr [1:26]	"a" "b" "c" "d" "e" "f" "g"...
x		3
y		TRUE

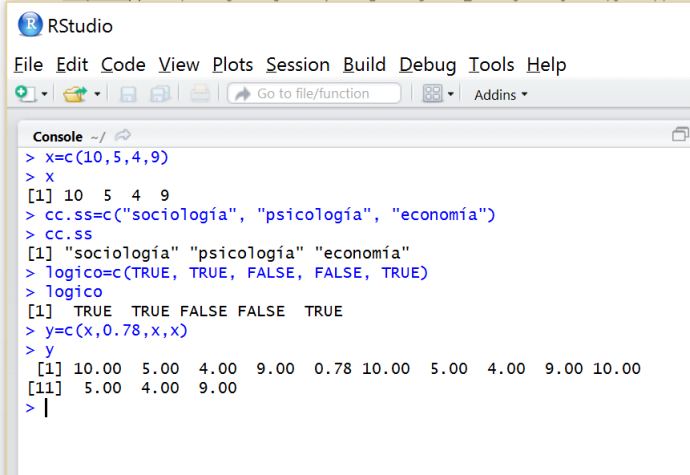
- Los atributos del vector son su modo y longitud



```
> is.vector(a)
[1] TRUE
> mode(a); length(a)
[1] "numeric"
[1] 5
> |
```

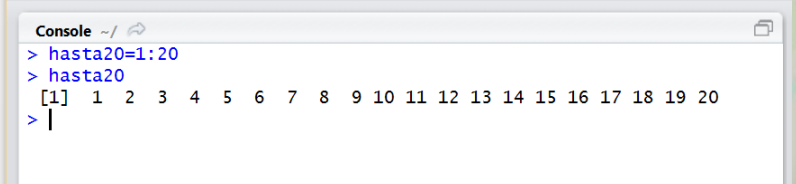
Estructura de Objetos: Vectores

- La generación de vectores puede realizarse utilizando la función concatenar: $c()$.



```
RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function Addins
Console ~/
> x=c(10,5,4,9)
> x
[1] 10 5 4 9
> cc.ss=c("sociología", "psicología", "economía")
> cc.ss
[1] "sociología" "psicología" "economía"
> logico=c(TRUE, TRUE, FALSE, FALSE, TRUE)
> logico
[1] TRUE TRUE FALSE FALSE TRUE
> y=c(x,0.78,x,x)
> y
[1] 10.00 5.00 4.00 9.00 0.78 10.00 5.00 4.00 9.00 10.00
[11] 5.00 4.00 9.00
> |
```



- Se puede generar secuencias numéricas que se almacenarán como vectores utilizando el operador :





```
Console ~/
> hasta20=1:20
> hasta20
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
> |
```

Estructura de Objetos: Vectores

- La función *seq()* genera secuencias de un modo más general. Los argumentos de esta función son el valor inicial, el valor final y el incremento.


```
Console ~/    
> a=seq(-5,5,1.5)  
> a  
[1] -5.0 -3.5 -2.0 -0.5 1.0 2.5 4.0  
> b=seq(10,3,-2)  
> b  
[1] 10 8 6 4  
> |
```

- La función *rep()* repite un vector dado. Los argumentos de esta función son el vector y el número de veces a repetirse el mismo.


```
Console ~/    
> rep(2,3)  
[1] 2 2 2  
> rep(1:4,3)  
[1] 1 2 3 4 1 2 3 4 1 2 3 4  
> rep(1:3,3:5)  
[1] 1 1 1 2 2 2 2 2 3 3 3 3 3  
> |
```



Vectores. Operaciones

- Los cálculos ejecutados sobre vectores se llevan a cabo sobre cada uno de sus elementos.

```
Console ~/   
> x=c(3,4,5,1)  
> x*x  
[1] 9 16 25 1  
> |
```

- Si los vectores involucrados en una misma expresión no son de la misma longitud, el vector de menor longitud se repetirá hasta que alcance la longitud del mayor.

```
Console ~/   
> x=c(3,4,5,1)  
> x*x  
[1] 9 16 25 1  
> y=3  
> x+y  
[1] 6 7 8 4  
> |
```

```
Console ~/   
> y=c(1:5)  
> y  
[1] 1 2 3 4 5  
> z=x+y  
Warning message:  
In x + y : longer object length is not a multiple of shorter object length  
> z  
[1] 4 6 8 5 8  
> |
```

- Observar que R genera un mensaje de aviso. En estos casos, es aconsejable revisar el procedimiento que lo ha originado y el resultado final.

Vectores. Operaciones

- Existen varias funciones específicas que trabajan sobre vectores algunas de las cuales se presentan a continuación.

Función	Salida
<i>length()</i>	Longitud del vector
<i>sum()</i>	Suma de los elementos del vector
<i>prod()</i>	Producto de los elementos del vector
<i>max()</i> , <i>min()</i>	Máximo valor del vector, Mínimo valor del vector
<i>cumsum()</i>	Vector de suma acumulada de los elementos del vector
<i>cumprod()</i>	Producto acumulado de los elementos
<i>diff()</i>	Vector de diferencias entre elementos
<i>unique()</i>	Vector de valores únicos
<i>duplicated()</i>	Vector lógico que indica si los elementos están duplicados


Vectores. Operaciones

- Algunas funciones estadísticas aplicables sobre vectores son:

Función	Salida
median()	Mediana del vector
mean()	Media de vector
quantile()	Cuantiles de los vectores
IQR()	Rango intercuartil del vector
range()	Rango del vector
sd()	Desviación estándar
var()	Variancia de los elementos
summary()	Resumen descriptivo

Vectores. Operaciones

- La función `sort()` ordena los elementos de un vector de forma ascendente y `rev(sort())` ordena de modo descendente.
- La función `order()` genera un vector cuyos elementos indican el orden que ocupa los elementos en el vector.
- El comando `outer()` realiza cálculos cruzados entre dos vectores. El resultados de esta función es una matriz de dimensiones $(dim(x), dim(y))$ cuyos elementos son el resultado de aplicar una determinada función sobre x e y.

```
Console ~/   
> x=1:3; names(x)=x  
> y=10:15; names(y)=y  
> outer(x,y,"*")  
      10 11 12 13 14 15  
1 10 11 12 13 14 15  
2 20 22 24 26 28 30  
3 30 33 36 39 42 45  
> outer(x,y,"-")  
      10 11 12 13 14 15  
1 -9 -10 -11 -12 -13 -14  
2 -8 -9 -10 -11 -12 -13  
3 -7 -8 -9 -10 -11 -12  
> |
```


Estructura de Objetos: Factores

- Las variables categóricas se almacenan en factores.
- La función *factor()* permite convertir un vector con variantes categóricas en un factor. Y la función *levels()* permite visualizar los niveles de un factor.

```
Console ~/
> sexo=c(rep('mujer',5),rep('varon',10))
> sexo=factor(sexo)
> levels(sexo)
[1] "mujer" "varon"
> sexo
[1] mujer mujer mujer mujer mujer varon varon varon varon varon
[11] varon varon varon varon varon
Levels: mujer varon
>
```


- Los niveles de un factor pueden ser combinados.

Otra forma es asignarle la misma etiqueta a las categorías a combinar.


```
Console ~/
> edad=c(rep('niño', 30), rep('adolescente', 20),rep('adulto', 2), rep('mayor', 3))
> edad=factor(edad)
> levels(edad)
[1] "adolescente" "adulto" "mayor" "niño"
> levels(edad)=list(niño='niño',adolescente='adolescente',adulto=c('adulto','mayor'))
> levels(edad)
[1] "niño" "adolescente" "adulto"
> edad
[1] niño niño niño niño niño
[6] niño niño niño niño niño
[11] niño niño niño niño niño
[16] niño niño niño niño niño
[21] niño niño niño niño niño
[26] niño niño niño niño niño
[31] adolescente adolescente adolescente adolescente adolescente
[36] adolescente adolescente adolescente adolescente adolescente
[41] adolescente adolescente adolescente adolescente adolescente
[46] adolescente adolescente adolescente adolescente adolescente
[51] adulto adulto adulto adulto adulto
Levels: niño adolescente adulto
>
```

Estructura de Objetos: Factores

- Se pueden generar factores a través de la categorización de un factor continuo mediante las funciones *cut()* y *breaks()*.

```
Console ~/   
> x=1:15  
> x  
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15  
> y=cut(x,breaks=c(0,5,10,15))  
> y  
[1] (0,5] (0,5] (0,5] (0,5] (0,5] (5,10] (5,10]  
[8] (5,10] (5,10] (5,10] (10,15] (10,15] (10,15] (10,15]  
[15] (10,15]  
Levels: (0,5] (5,10] (10,15]  
> |
```

- Una de las operaciones más comunes con factores es la generación de tablas de frecuencias. Las funciones *table()*, *xtabs()* y *ftable()* crean tablas. La primera es la más básica.

```
Console ~/   
> table(y)  
y  
 (0,5] (5,10] (10,15]  
      5      5      5  
> table(sexo)  
sexo  
mujer varon  
      5     10  
> |
```

Estructura de Objetos: Factores

- El objeto que se genera se puede almacenar bajo un nombre. Las funciones *rownames()* y *colnames()* permiten asignar nombres a las filas y columnas respectivamente de cualquier tabla.
- La función *t()* permite transponer una tabla.
- Algunas de las funciones específicas para el manejo de tablas son *margin.table()* y *prop.table()*. Uno de los argumentos es un valor igual a 1 o 2 si se quiere frecuencias (o proporciones) por filas o por columnas respectivamente.

Estructura de Objetos: Matrices

- Los objetos bidimensionales constituidos por filas y columnas de elementos del mismo tipo se almacenan en matrices. Los elementos de una matriz pueden ser numéricos, lógicos o caracteres.
- La función *dim()* aplicada sobre un vector da como resultado una matriz con un numero de filas y de columnas especificado.

```
Console ~/
> x=1:8
> dim(x)=c(2,4)
> x
      [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
>
```

La función *matrix()* genera matrices a partir de un vector

```
Console ~/
> x=matrix(1:8,2,4,byrow=F)
> x
      [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
> x=matrix(1:8,2,4,byrow=T)
> x
      [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
> a=1:8
> x=matrix(a,2,4)
> x
      [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
>
```


Estructura de Objetos: Matrices

- Otra forma de crear matrices es concatenando varios vectores o matrices. La función `cbind()` lo hace por columnas y la función `rbind()` por filas.

```
Console ~/
> cbind(c(1,2,3),c(4,5,6))
  [,1] [,2]
[1,]  1  4
[2,]  2  5
[3,]  3  6
> rbind(c(1,2,3),c(4,5,6))
  [,1] [,2] [,3]
[1,]  1  2  3
[2,]  4  5  6
> |
```

- Las funciones aplicables sobre vectores son también válidas para matrices.

```
Console ~/
> x=matrix(1:9,ncol=3)
> y=c(5:7)
> x
  [,1] [,2] [,3]
[1,]  1  4  7
[2,]  2  5  8
[3,]  3  6  9
> y
[1] 5 6 7
> x+y
  [,1] [,2] [,3]
[1,]  6  9 12
[2,]  7 10 13
[3,]  8 11 14
```

```
> x*x
  [,1] [,2] [,3]
[1,]  1 16 49
[2,]  4 25 64
[3,]  9 36 81
> |
```

Estructura de Objetos: Matrices

- El producto $x*x$ da como resultado el cuadrado de cada uno de los elementos de la matriz.
- La función `%*%` es la que realiza el producto matricial.
- Otras funciones específicas para matrices son

Función	Salida
<i>det()</i>	Determinante de la matriz
<i>solve()</i>	Inversa de la matriz
<i>diag()</i>	Elementos diagonals de la matriz
<i>ncol()</i>	Número de columnas de la matriz
<i>nrow()</i>	Número de columnas de la matriz
<i>t()</i>	Transpuesta de la matriz

Estructura de Objetos: Matrices

- Además de las operaciones relacionadas directamente con el cálculo matricial, se pueden aplicar funciones sobre las filas o columnas de una matriz. Entre ellas:

Función	Salida
<i>colnames()</i>	Nombre de las columnas de la matriz
<i>rownames()</i>	Nombre de las filas de la matriz
<i>colSums()</i>	Suma los elementos de las columnas
<i>rowSums()</i>	Suma los elementos de las filas
<i>dim()</i>	Dimensiones de la matriz
<i>length()</i>	Número de elementos de la matriz

Estructura de Objetos: Arrays

- Los arrays son generalizaciones de las matrices, es decir pueden estar compuestos por 3 o más dimensiones. Los elementos de los arrays pueden ser numéricos, lógicos o caracteres.
- La generación de arrays es similar a la generación de matrices.
- Las funciones u operaciones que pueden aplicarse sobre las matrices también pueden aplicarse sobre los arrays.

Estructura de Objetos: Data frames

- El data frame es el tipo de estructura más común en R para el análisis de datos.
- A diferencia de vectores, matrices y arrays, en los que todos los elementos son del mismo tipo, los data frame pueden contener datos de distinto tipo.
- Los data frame incorporan nombres para cada una de las columnas y pueden incorporar nombres para cada una de las filas.

Estructura de Objetos: Data frames

- La generación de data frame puede ser realizada concatenando vectores, matrices o arrays.

```
Console ~/
> x=1:10;y=round(rnorm(10,5,2),0);z=letters[10:1]
> juntos=cbind(x,y,z)
> juntos
      x      y      z
[1,] "1"    "4"    "j"
[2,] "2"    "3"    "i"
[3,] "3"    "5"    "h"
[4,] "4"    "8"    "g"
[5,] "5"    "5"    "f"
[6,] "6"    "5"    "e"
[7,] "7"    "4"    "d"
[8,] "8"    "2"    "c"
[9,] "9"    "6"    "b"
[10,] "10"  "8"    "a"
> class(juntos)
[1] "matrix"
> mode(juntos)
[1] "character"
> |
```

```
> bien=data.frame(x,y,z)
> bien
   x y z
1  1 4 j
2  2 3 i
3  3 5 h
4  4 8 g
5  5 5 f
6  6 5 e
7  7 4 d
8  8 2 c
9  9 6 b
10 10 8 a
> attributes(bien)
$names
[1] "x" "y" "z"

$row.names
[1] 1 2 3 4 5 6 7 8 9 10

$class
[1] "data.frame"
>
```

- La función *as.data.frame()* transforma cualquier objeto en un data frame.

Estructura de Objetos: Data frames

- Las funciones *names()* y *rownames()* asignan o modifican los nombres de las columnas.
- La adición de columnas a un data frame se puede realizar mediante la función *cbind()* y la de filas mediante la función *rbind()*
- La función *head()* permite visualizar las primeras filas de un data frame y la función *summary()* permite obtener un resumen de las variables que lo integran.
- La función *merge()* permite fusionar data frames de acuerdo a algún criterio de emparejamiento, que habitualmente es un conjunto de columnas que tienen el mismo nombre.

Estructura de Objetos: Listas

- Una lista es una colección ordenada de elementos de distinto tipo.
- En general, la lista es la estructura que utiliza R para almacenar las salidas de las funciones estadísticas.
- El comando *list()* permite crear listas especificando los nombres de los componentes.

```
> x=1:7
> y=c('ana', 'marcos', 'juan', 'pedro', 'ramon')
> z=list(sequencia=x, nombres=y)
> z
$sequencia
[1] 1 2 3 4 5 6 7

$nombres
[1] "ana"      "marcos"   "juan"     "pedro"    "ramon"
```

- La función *names()* permite extraer los nombres de los componentes o cambiar sus etiquetas.
- Y `[[]]` o el símbolo `$` permiten añadir elementos a la lista.

Manipulación de Datos

- Una vez creada una estructura de datos, R permite manipularlos muy fácilmente. Cada uno de los datos puede ser seleccionado para operar sobre el.
- En el caso de vectores, la extracción o selección de un elemento puede realizarse especificando entre corchetes el lugar que ocupa el elemento dentro del vector
- En el caso de matrices, habrá que identificar tanto el subíndice de la fila como el de la columna.

Manipulación de Datos

- Ejercicio 1.
 - Generar un vector x con los números naturales de 1 a 9 y otro vector con los primeros 3 elementos de x.
 - Generar un vector lógico de la misma longitud que x que indique si los elementos de x son mayores a 6 y otro vector que contenga solo los elementos de x que cumplen la condición descripta.
- Ejercicio 2.
 - Generar una matriz x con 6 filas que contenga los números naturales de 1 a 36. Seleccionar el elemento en la fila 5 y columna 5. Seleccionar todos los elementos de la columna 5. Seleccionar todos los elementos de las columnas 3, 4 y 5 de la fila 4.
 - Generar una matriz lógica de la misma dimensión que la matriz original.

Manipulación de Datos

- En el caso de data frame, la extracción de elementos se puede realizar mediante los mismos procedimientos que para las matrices. Pero además se puede utilizar otro procedimiento.
- Supongamos que nuestro data frame se llama Datos. Luego la sentencia `Datos$Edad` extraerá la columna Edad del data frame Datos.
- La selección de un subconjunto de elementos que cumplan una determinada condición se puede realizar mediante la función *subset()*.

Lectura y Grabación de Datos

- R utiliza por defecto un directorio de trabajo en el que se guardan las sesiones de trabajo o los objetos que se desee almacenar. La función *getwd()* permite comprobar cual es ese directorio y la función *setwd()* permite modificarlo.
- Tener en cuenta que en R los path se definen a través del símbolo ‘/’ o ‘\\’
- Si el conjunto de datos con los que se va a operar es pequeño, los datos pueden introducirse a través de la consola o del editor de R.
- Sin embargo, en la mayoría de las situaciones se dispondrá de los datos en algún archivo a partir del cual se deberán importar a R.

Lectura y Grabación de Datos

- La función *read.table()* lee archivos externos en los que las líneas son casos y las columnas son variables. El formato básico de esta función es:
 - *read.table('path',sep=',',dec=',', header=T)*
- Esta función asume que los caracteres ausentes en el fichero están representados por NA. En el caso de no ser así, hay dos opciones. Una es modificar los valores antes de importar los datos. La otra es utilizar la función *na.string* para definir los valores ausentes.
- Esta función es especialmente útil para la importación de ficheros cuyos separadores han sido bien definidos.

Lectura y Grabación de Datos

- La función *read.fwf()* lee ficheros de datos y los transforma en data frame. La utilización de esta función exige indicar el ancho de las columnas que contienen las variables que se realiza a través del argumento *widths()*.
- La función *scan()* permite la lectura de ficheros de datos externos que no tienen un formato fijo.
- Tanto *read.table()* y *scan()* permiten la utilización del argumento *skip()* para especificar el número de líneas del principio que no deben importarse.
- La diferencia fundamental entre *read.table()* y *scan()* es que esta última es más flexible respecto al tipo de fichero de datos a importar.

Lectura y Grabación de Datos

- R también contiene algunas opciones para importar datos provenientes de otro software.
- La grabación externa de un vector o una matriz de datos generados por R se realiza con el comando *write()*:
 - *write(x,file='misdatos')*
- Esta función permite especificar el numero de columnas por medio del argumento *ncolumns()*.
- Si el conjunto de datos es un data frame, la función a utilizar es *write.table()*:
 - *write.table(dataframe,file=" ", sep=" ", row.names=F, col.names=T)*