



DMC ONLINE

#YoMeQuedoEnCasa

SESIÓN 01:

Fundamentos de Programación en R

DOCENTE: LUIS FELIPE GARAYAR BURNEO

CORREO: LUIS.GARAYAR.DMC@GMAIL.COM



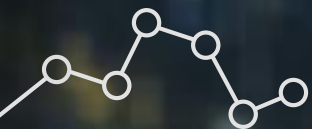


DMC ONLINE

#YoMeCapacitoEnCasa

Reglas e Itinerario

www.dmc.pe





DMC ONLINE



Reglas



Puntualidad



Mantener silenciado el micrófono durante la sesión



Las preguntas se realizarán por el chat/ en caso sea necesario se habilita el micrófono



Realizar las actividades encomendadas

#YoMeCapacitoEnCasa

www.dmc.pe

SOBRE EL DOCENTE

- ✓ Ingeniero Estadístico - UNI
- ✓ Ingeniero de Sistemas - UNMSM
- ✓ Candidato a Magister en Estadística Aplicada - UNALM
- ✓ Data Scientist Practitioner - UNIR (España)
- ✓ Más de 10 años de experiencia en Banca (Inteligencia Comercial: Campañas Dirigidas, Reportería, Análisis y Modelos Estadísticos)
- ✓ Actualmente laborando en el BCP.
- ✓ Campos de Interés: Excel, Macros (VBA), Dashboards en Excel, Power BI, Lenguaje SQL, Oracle PL/SQL, software estadístico R y Python



DMC ONLINE



Calificación

Asistencia (Curso) (10%)

Trabajos calificados (40%)

Examen Final (teoría y práctica) (50%)

#YoMeCapacitoEnCasa

www.dmc.pe



DMC ONLINE

#YoMeCapacitoEnCasa

Contenido del curso

www.dmc.pe



- I. Introducción al R y al R Studio
- II. Tipos y estructuras de datos en R
- III. Programación básica en R
- IV. Análisis Gráfico con R
- V. Manipulación Avanzada de Dataframes
- VI. Introducción al Análisis Estadístico con R
- VII. Publicación de Resultados



FECHA DE ENTREGAS DE TRABAJOS

- **Listado de Ejercicios Sesión 1** *Sábado 26/09 medianoche.*
- **Listado de Ejercicios Sesión 2** *Miércoles 30/09 medianoche.*
- **Listado de Ejercicios Sesión 3** *Sábado 03/10 medianoche.*
- **Listado de Ejercicios Sesión 4** *Jueves 08/10 medianoche.*
- **Listado de Ejercicios Sesión 5** *Domingo 11/10 medianoche.*
- **Examen Final Teórico y Práctico**

Formulario Activo entre Martes 06/10 y Domingo 11/10.

I. Introducción al R y al R Studio

II. Tipos y estructuras de datos en R

III. Programación básica en R

IV. Análisis Gráfico con R

V. Manipulación Avanzada de Dataframes

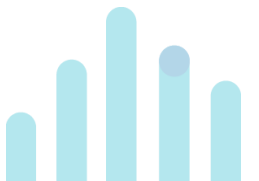
VI. Introducción al Análisis Estadístico con R

VII. Publicación de Resultados

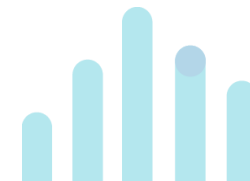
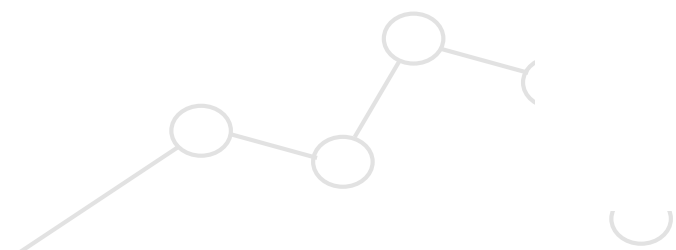


Un poco de historia...

- R es básicamente un lenguaje de programación que permite realizar comandos e implementar técnicas estadísticas en un entorno interactivo para el análisis estadístico y gráfico.
- *Es un lenguaje de programación con funciones orientadas a objetos.*
- R fue inicialmente diseñado por *Robert Gentleman y Ross Ihaka (1993)*, miembros del Departamento de Estadística de la Universidad de Auckland, en Nueva Zelanda.



Motivos para usar R



Motivos para usar R

- La sintaxis es simple e intuitiva.

```
#Creando vectores.
```

```
> x<-c(10.4,5.6,3.1,6.4,21.7)
```

```
> x
```

```
[1] 10.4  5.6  3.1  6.4 21.7
```

```
> w<-c("rojo","verde","azul")
```

```
> w
```

```
[1] "rojo"  "verde" "azul"
```

```
#Creando objetos y realizando cálculos aritméticos.
```

```
> a=10
```

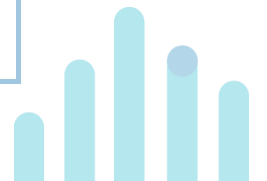
```
> b=25
```

```
> c=25
```

```
> y=a+b+c
```

```
> y
```

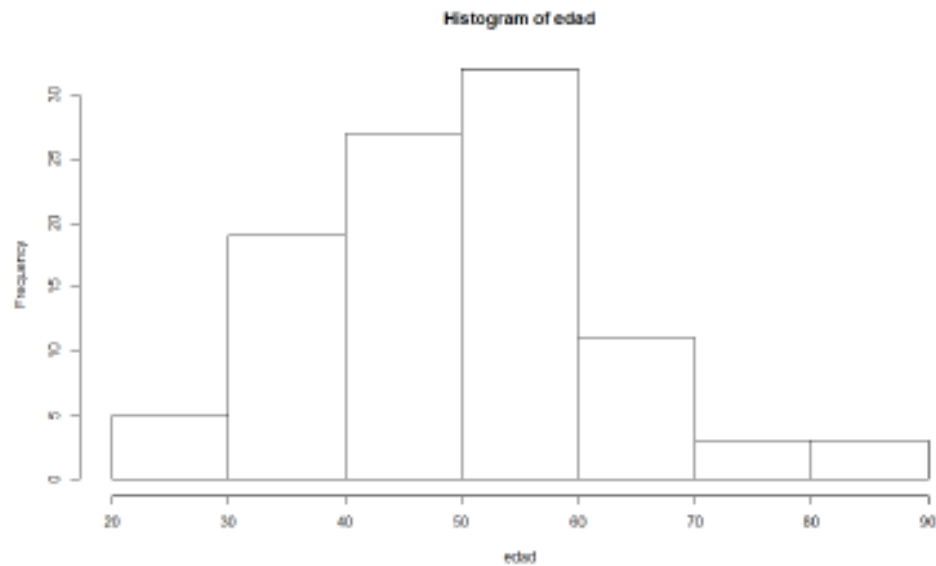
```
[1] 60
```



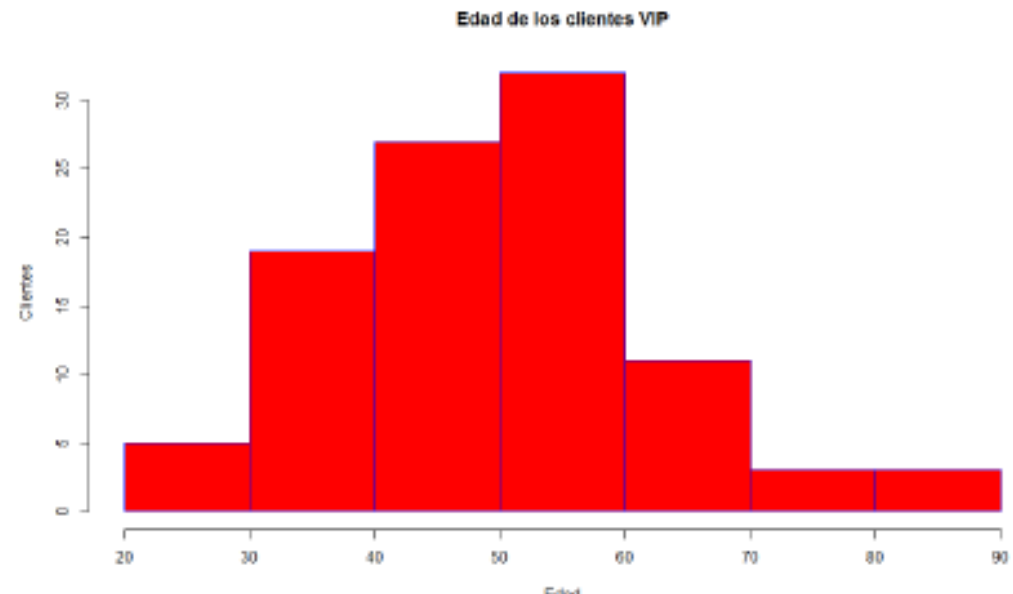
Motivos para usar R

- La estructura y facilidad de uso de R nos permite implementar nuestras propias funciones y rutinas a medida que aparecen nuestras necesidades.

```
>hist(edad)
```



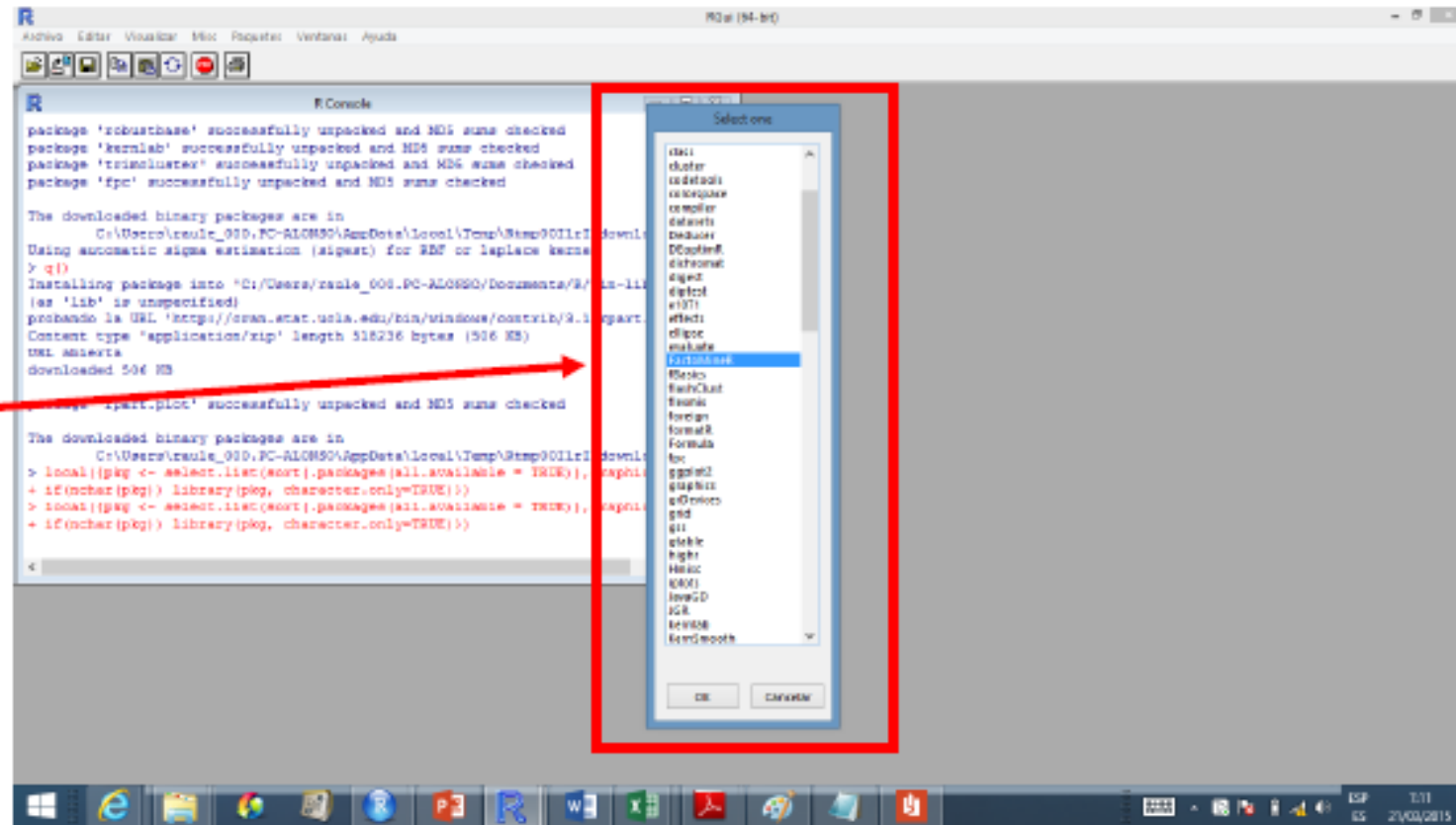
```
>hist(edad,col="red", xlab="Edad",ylab="Clientes",  
main="Edad de los clientes VIP",border="blue")
```



Motivos para usar R

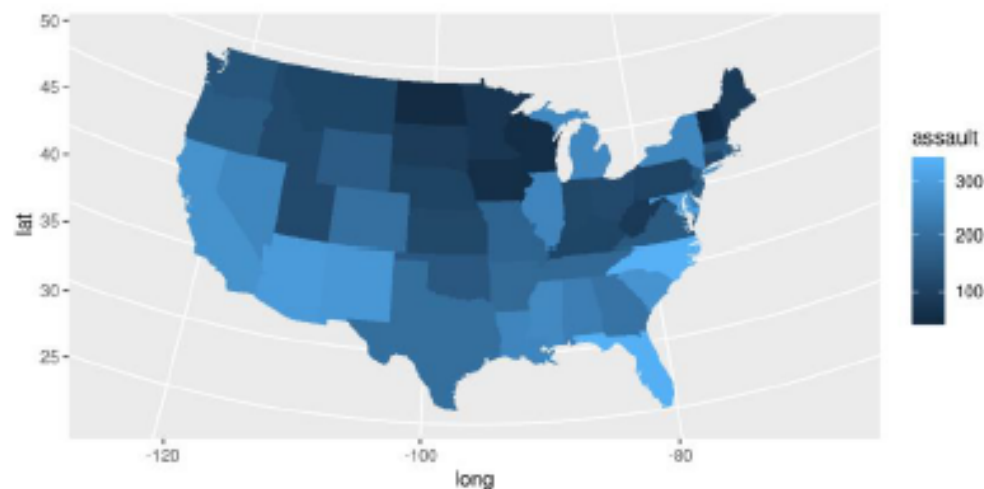
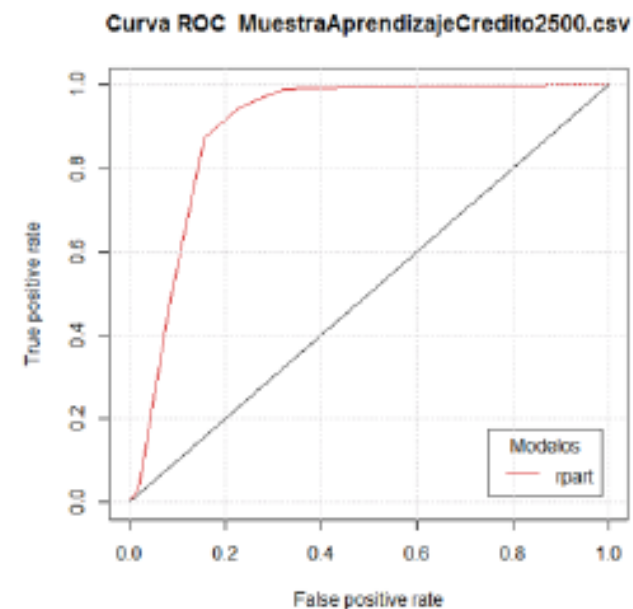
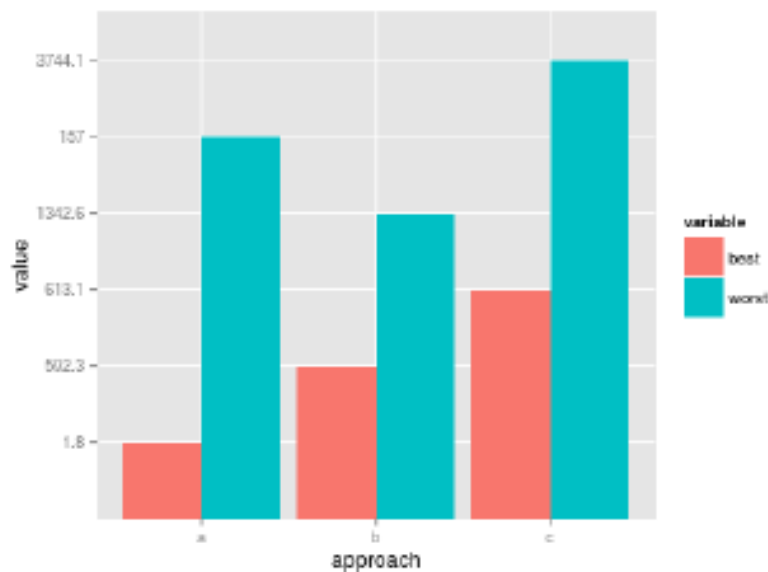
- La distribución de R viene acompañada de un numeroso conjunto de librerías base. Así mismo, es posible añadir librerías adicionales.

Librerías
propias y
librerías
adicionales
como el
FactoMineR



Motivos para usar R

- Gran variedad de librerías gráficas.



Motivos para usar R

- Gran red de apoyo y soporte disponible en foros, blogs, Facebook, etc.

El mejor foro: <http://stackoverflow.com/>

El mejor blog: <http://www.r-bloggers.com/>



Motivos para usar R



Entorno R y R Studio

Podemos definir R desde dos perspectivas distintas:

R es un entorno *software*

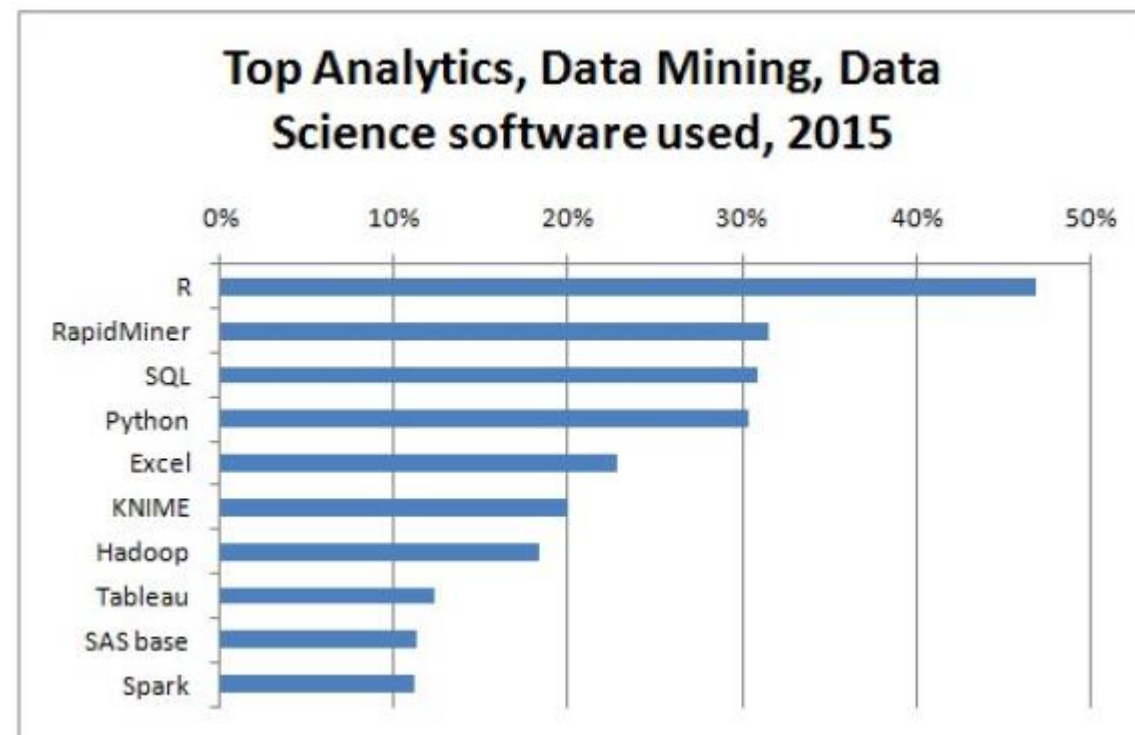
R es un lenguaje de programación

R Permite:

- Cargar y Manipular datos.
- Realizar Análisis.
- Presentar los resultados gráficamente

Ventajas de R

- Herramienta libre y gratuita(GNUGPL).
- Evolución continua y rápida.
- Versiones en múltiples sistemas operativos.
- Flexible y potente.
- Mucha documentación/ayuda. Multitud de librerías y extensiones.



Entorno R y R Studio

Instalación R

<https://cran.r-project.org>



The R Project for Statistical Computing

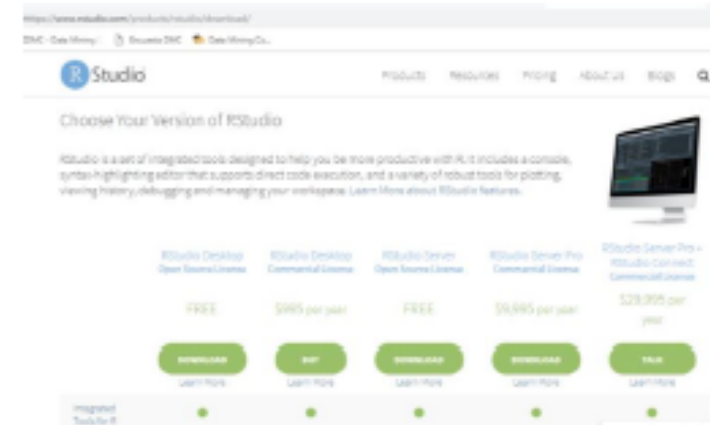
Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To [download R](#), please choose your preferred CRAN mirror.

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.

Instalación RStudio

<https://www.rstudio.com/products/rstudio/download/>



Choose Your Version of RStudio

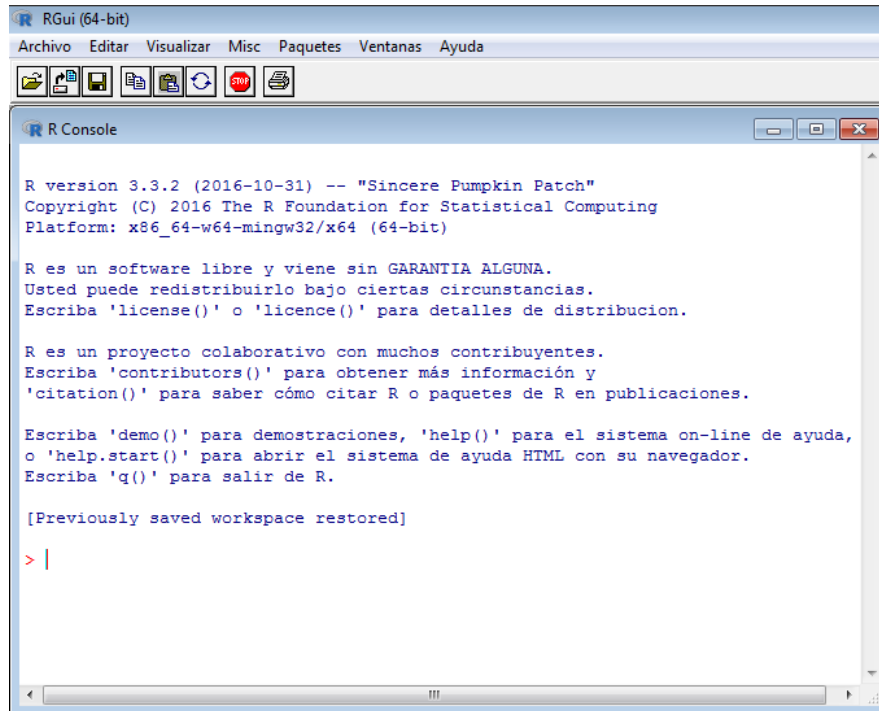
RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, and a variety of robust tools for plotting, viewing history, debugging and managing your workspace. [Learn More about RStudio features.](#)

RStudio Desktop <small>Open Source License</small>	RStudio Desktop <small>Commercial License</small>	RStudio Server <small>Open Source License</small>	RStudio Server Pro <small>Commercial License</small>	RStudio Server Pro + RStudio Connect <small>Commercial License</small>
FREE	\$995 per year	FREE	\$9,995 per year	\$29,995 per year
Download	Buy	Download	Download	Buy
Learn More	Learn More	Learn More	Learn More	Learn More

Integrated Tools for R

Entorno R

Consola R



```

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

R Console

R version 3.3.2 (2016-10-31) -- "Sincere Pumpkin Patch"
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 distribucion.

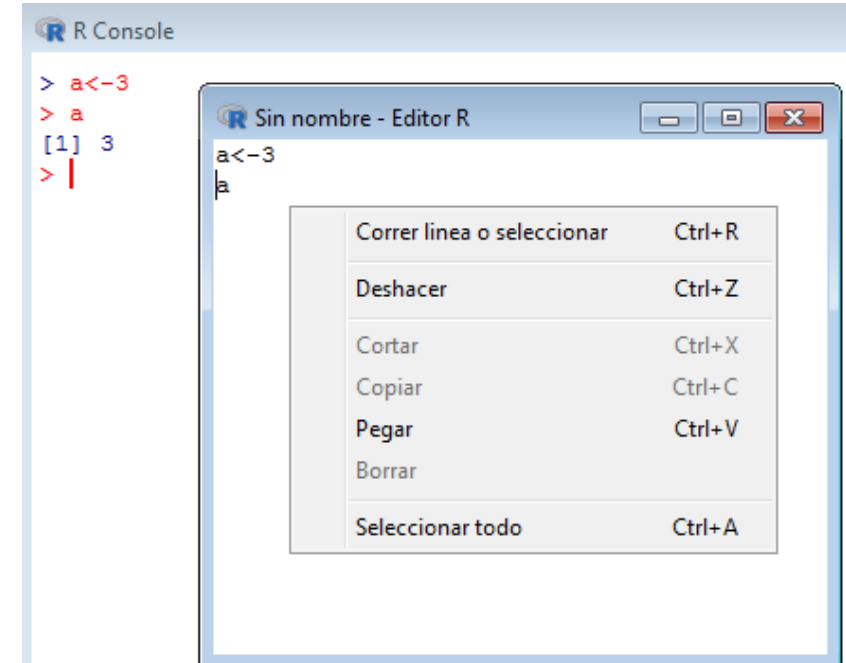
R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

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.

[Previously saved workspace restored]

> |
  
```

Editor de programas



```

R Console

> a <- 3
> a
[1] 3
> |

Sin nombre - Editor R

a <- 3
a

Correr línea o seleccionar  Ctrl+R
Deshacer                    Ctrl+Z
Cortar                      Ctrl+X
Copiar                      Ctrl+C
Pegar                      Ctrl+V
Borrar
Seleccionar todo            Ctrl+A
  
```

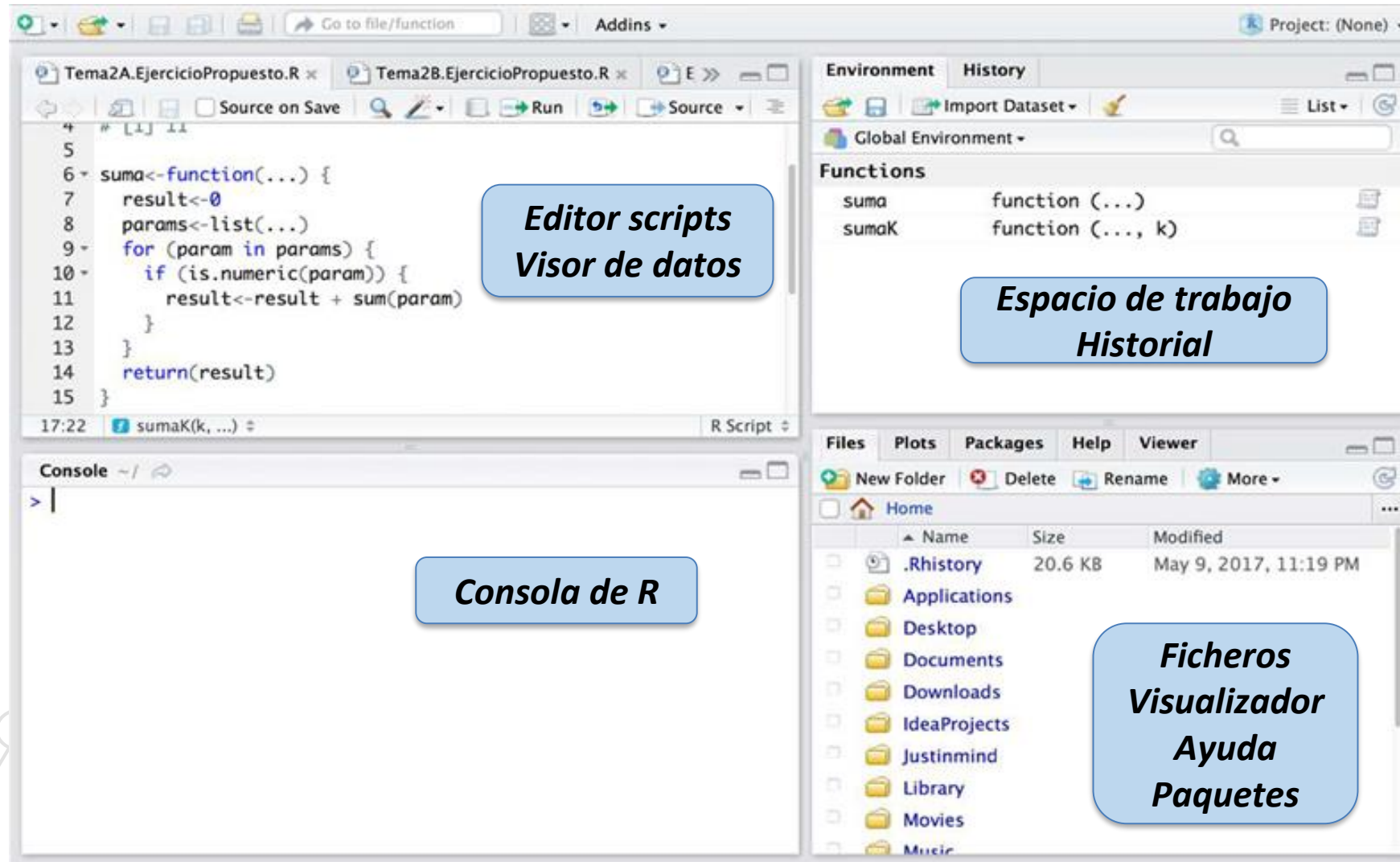
Atajo Ejecutar : **CTRL+ R**

- **Prompt (>):** Indica que R está esperando comandos
- Se ingresa comandos y se ejecuta en consola con un salto de línea.
- Otra forma de ingresar comandos es mediante un **Editor de scripts R**.
- En este caso, se ejecuta por línea completa (donde se encuentra el cursor) o una selección (puede ser parte de una línea o seleccionando varias líneas).

Entorno R Studio

R Studio

Atajo Ejecutar : **CTRL + Enter**



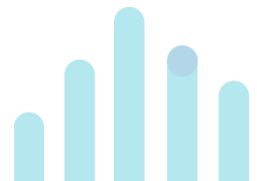
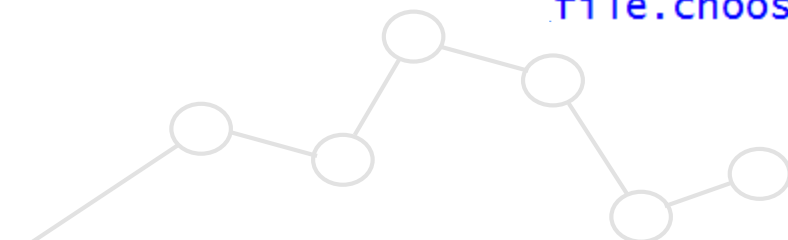
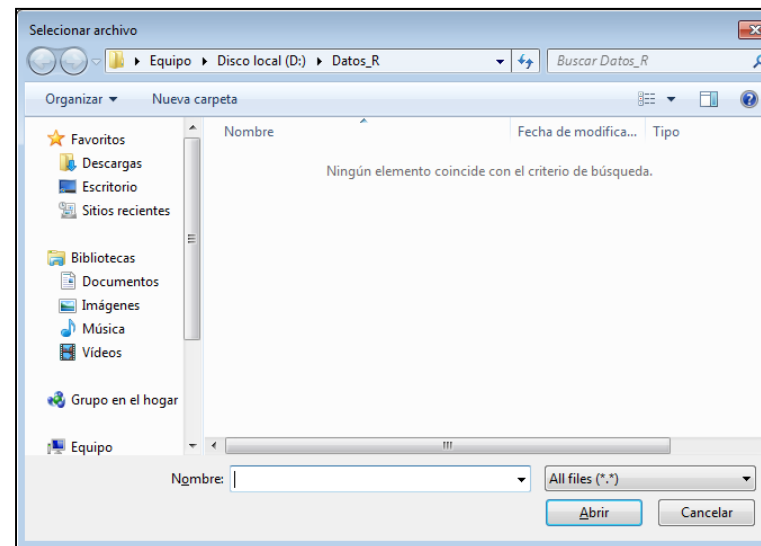
Workspace y Objetos en memoria

Workspace (espacio de trabajo)

- Es el lugar donde se almacena todo lo que se realiza en una sesión de R.
- Para obtener la dirección actual del Workspace se usa la función ***getwd()***.
- Para cambiar la ruta que usaremos para el Workspace, usaremos ***setwd()***.
- En general, si no tenemos la ruta de la carpeta que seleccionaremos, podemos ubicarla con ***file.choose()***, la que nos abrirá un navegador de archivos.

```
> getwd()
[1] "C:/Users/Garayar/Documents"
> setwd("d:/Datos_R")
> getwd()
[1] "d:/Datos_R"
```

`file.choose()`



Workspace y Objetos en memoria

Objetos en R

- Son todas las entidades que manipula R. Por ejemplo, vectores de números, vectores lógicos, vectores de caracteres, matrices, etc.
- Los atributos de un objeto suministran información específica sobre el propio objeto.

Atributo	Descripción
Modo	Cualquier tipo de entidad que maneja R. Se usa la función: <code>mode(objeto)</code> .
Tipo	Tipo de datos de los objetos: entero, carácter, double, etc. Se usa la función: <code>typeof(objeto)</code> .
Nombre	Etiquetas de los elementos individuales de un vector o lista. Se usa la función: <code>names(objeto)</code> .
Dimensiones	Dimensiones de las matrices (alguna puede ser cero). Se usa la función: <code>dim(objeto)</code> .
Dimnames	Nombres de las dimensiones de la matriz. Se usa la función: <code>dimnames(objeto)</code> .
Clase	Vector alfanumérico con la lista de las clases del objeto. Se usa la función: <code>class(objeto)</code> .

Workspace y Objetos en memoria

Creación de Objetos y Variables

- Se usa el operador de asignación (<-)
- También se puede usar el operador (=)

```
> n<-5
> n
[1] 5
> a=15
> a
[1] 15
```

Reglas para la creación de nombres de variables



R distingue entre mayúsculas y minúsculas.



Usualmente se permite utilizar toda clase de símbolos alfanuméricos (incluyendo caracteres acentuados, ñ, etc.) además del punto «.» y del guion bajo «_». Eso sí, los nombres deben empezar por «.» o por una letra y si empiezan por «.» entonces el segundo carácter no debería ser un número..



El entorno no impone límites a la longitud de los nombres de los símbolos (aunque por motivos prácticos no deberíamos hacerlos demasiado largos).

Ejemplos:

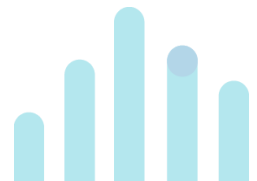
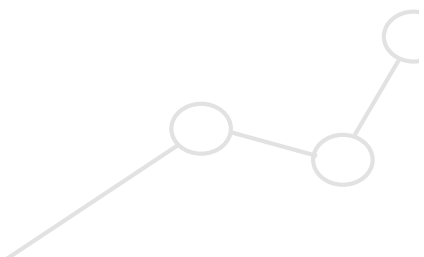
```
data_año1
Data_año1
.var9
tabla.final..piloto.PLD_2
```

Workspace y Objetos en memoria

Algunas consideraciones:

- Para obtener el valor de una variable u objeto por consola, tecleamos su nombre seguido de un fin de línea. También se puede usar la función *print*.
- Para modificar el valor de un objeto o variable simplemente lo reasignamos.
- Los caracteres se colocan entre comillas simples (' ') o dobles (" ").
- Se pueden realizar varios comandos en una línea separados por punto y coma (;)

```
> name <- "Carmen"; n1 <- 10; n2 <- 100; m <- 0.5
> n1
[1] 10
> name
[1] "Carmen"
> name <- 'Luis Felipe'
> name
[1] "Luis Felipe"
> print(name)
[1] "Luis Felipe"
```



Workspace y Objetos en memoria

➤ La función *ls* nos permite listar los objetos existentes en memoria.

También se puede usar la función *objects*.

➤ La opción *pattern (pat)* nos permite realizar búsquedas entre las variables.

➤ La función *ls.str* nos da algunos detalles de las variables y objetos en memoria.

➤ La función *rm* borra objetos de la memoria.

```
> ls()
[1] "m"      "n1"     "n2"     "name"
```

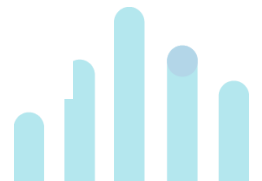
```
> objects()
[1] "m"      "n1"     "n2"     "name"
```

```
> ls(pat = "m") #que contenga m
[1] "m"      "name"
```

```
> ls(pat = "^m") #que empiece con m
[1] "m"
```

```
> ls.str()
m :   num 0.5
n1 :   num 10
n2 :   num 100
name :  chr "Luis Felipe"
```

```
> rm(name)
> ls()
[1] "m"      "n1"     "n2"
```



Workspace y Objetos en memoria

- R almacena los valores con el máximo número de dígitos, hasta 17, pero muestra solo 7, si uno desea que muestre mayor número, debemos cambiar esta opción.
- La función **options** determina como R calcula y muestra los resultados.

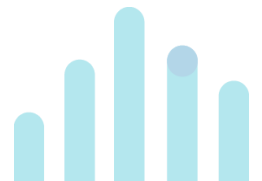
```
> pi                # R por defecto muestra 7 dígitos
[1] 3.141593
> options(digits=15) # Cambia a 15 dígitos
> pi
[1] 3.14159265358979
```

- En consola, se puede navegar en comandos anteriores con las teclas direccionales.
- En cualquier momento se puede guardar la sesión con **save.image()** (si no se especifica ruta, se guardará en la ruta definida para Workspace)

Limpiar consola: **CTRL + L**

Salir de la sesión: **q()**

```
> q()
Save workspace image to ~/.RData? [y/n]: y
```



Instalación y carga de paquetes

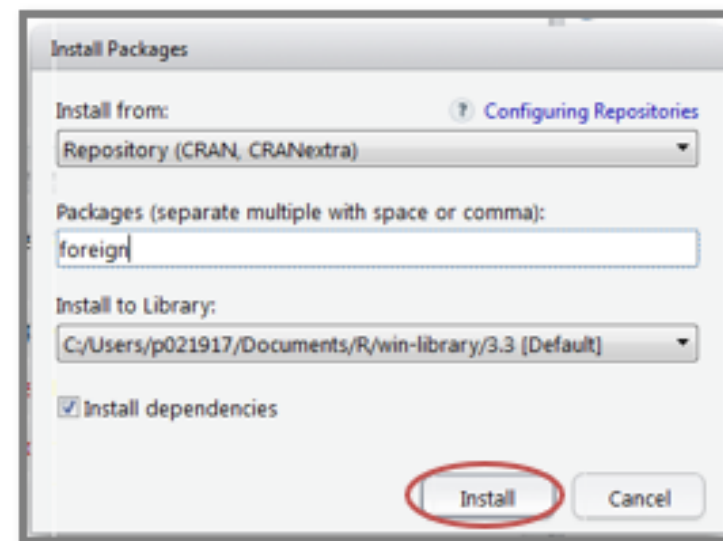
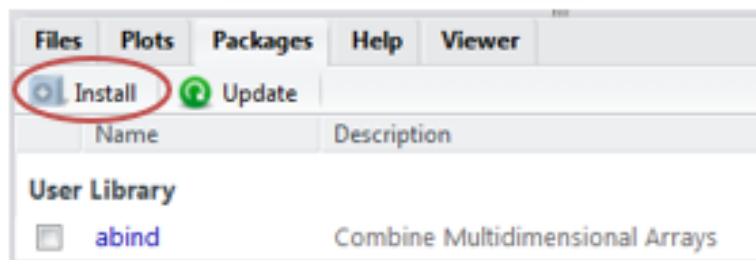
- Una ventaja de R es que es fácilmente expandible usando paquetes.
- Existen más de 8000 paquetes actualmente.
- Estos paquetes se obtienen de servidores CRAN esparcidos geográficamente.
- En consola, instalamos un paquete con **install.packages**

> install.packages("foreign", dependencies=TRUE)

- Esta sentencia instala el paquete "foreign" con otros paquetes que dependen de este. Luego cargamos el paquete para su uso: **> library("foreign")**

- En R Studio hay otra forma de instalar paquetes:

En el Visualizador



<https://cran.r-project.org/web/views/>

- Las vistas de tareas de CRAN tienen como objetivo proporcionar una guía sobre qué paquetes en CRAN son relevantes para las tareas relacionadas con un tema determinado.
- Ofrecen una breve descripción general de los paquetes incluidos.
- Las vistas están destinados a tener un enfoque claro para qué paquetes deben ser incluidos (o excluidos) - y son *no* destinado a apoyar los "mejores" paquetes para una tarea determinada.

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
HighPerformanceComputing	High-Performance and Parallel Computing with R
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis

MetaAnalysis	Meta-Analysis
ModelDeployment	Model Deployment with R
Multivariate	Multivariate Statistics
NaturalLanguageProcessing	Natural Language Processing
NumericalMathematics	Numerical Mathematics
OfficialStatistics	Official Statistics & Survey Methodology
Optimization	Optimization and Mathematical Programming
Pharmacokinetics	Analysis of Pharmacokinetic Data
Phylogenetics	Phylogenetics, Especially Comparative Methods
Psychometrics	Psychometric Models and Methods
ReproducibleResearch	Reproducible Research
Robust	Robust Statistical Methods
SocialSciences	Statistics for the Social Sciences
Spatial	Analysis of Spatial Data
SpatioTemporal	Handling and Analyzing Spatio-Temporal Data
Survival	Survival Analysis
TimeSeries	Time Series Analysis
WebTechnologies	Web Technologies and Services
gR	gRaphical Models in R

Primeros pasos en R

- **Expresión**

- Se evalúa, se muestra y el valor se pierde
- Ejemplo: `10 * 3`

- **Asignación**

- Evalúa la expresión, no se muestra y guarda el resultado en una variable u objeto
- Ejemplo: `x <- 10 * 3`

- Operaciones matemáticas `+`, `-`, `*`, `/`, `^`
- Operadores de comparación `<`, `==`, `>`, `<=`, `>=`, `!=`
- Operadores lógicos (and, or, not) `&`, `|`, `!`

Funciones para clases de datos

- `class()`
- `is.classname()`
 - `is.numeric()` `is.integer()` `is.character()`
- `as.classname()`
 - `as.numeric()` `as.integer()` `as.character()`

Operaciones con valores lógicos TRUE FALSE

- | | |
|-------------------------|---------------------------|
| • ¿Es una clase lógica? | <code>is.logical()</code> |
| • Conectiva lógica AND: | <code>&</code> |
| • Conectiva lógica OR: | <code> </code> |
| • Conectiva lógica NOT: | <code>!</code> |

Primeros pasos en R

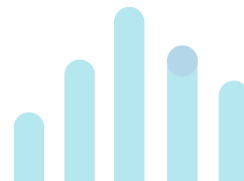
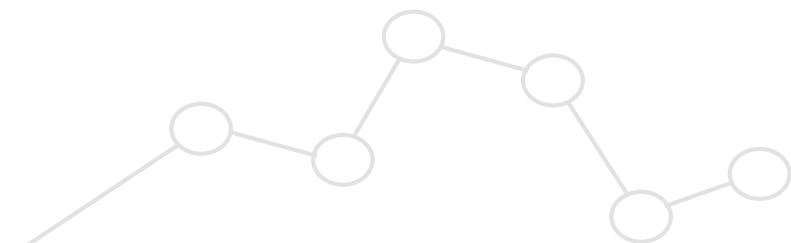
Realizando operaciones básicas

```
> 2+3
[1] 5
> 2*3
[1] 6
> 2/3
[1] 0.6666667
> 2^3
[1] 8
> 2**3
[1] 8
> 3%%2
[1] 1
> (2.2+3.5)*2.1
[1] 11.97
> 1+2+3-4
[1] 2
> 5%%2
[1] 1
```

Llamando a funciones

```
> exp(1)
[1] 2.718282
> log(exp(1))
[1] 1
> log10(10)
[1] 1
> log2(2)
[1] 1
> sqrt(4)
[1] 2
> cos(pi)
[1] -1
> sin(pi)
[1] 1.224647e-16
> tan(pi)
[1] -1.224647e-16
> abs(-2.4)
[1] 2.4
> ceiling(-2.4)
[1] -2
> floor(-2.4)
[1] -3
> round(-2.3894, 1)
[1] -2.4
```

```
> round(-2.3894, digits=1)
[1] -2.4
> round(digits=1, x=-2.3894)
[1] -2.4
```



Primeros pasos en R

Operador	Definición	Ejemplo
<code>abs(a)</code>	Valor absoluto de a	<code>> abs(-4)</code> [1].4
<code>sign(a)</code>	Devuelve el signo de a: 1 \equiv Positivo 0 \equiv Cero -1 \equiv Negativo	<code>> sign(-4)</code> [1].-1 <code>> sign(11-11)</code> [1].0 <code>> sign(8)</code> [1].1
<code>round(a, dec)</code>	Redondea un número según el número de decimales	<code>> round(13.68,1)</code> [1] 13.7
<code>sqrt(a)</code>	Raíz cuadrada de a	<code>> sqrt(4)</code> [1].2
<code>log(a)</code>	Logaritmo exponencial	<code>> log(4)</code> [1].1.386294
<code>log10(a)</code>	Logaritmo de a de base 10	<code>> log10(10)</code> [1].1
<code>exp(a)</code>	Función exponencial de a	<code>> exp(1)</code> [1].2.718282

Uso de la ayuda en R

➤ La función **help** nos permite acceder a la ayuda en R. Es igual si se antecede ? a la función.

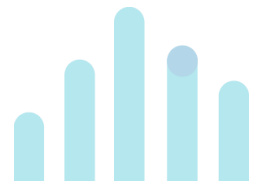
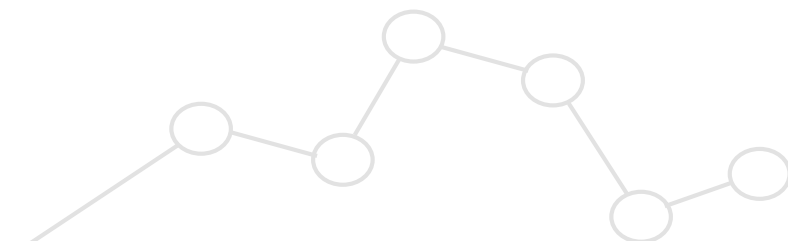
```
> help(mean)
> help("mean")
> ?mean
```

➤ Si se busca ayuda sobre una función especial (como un operador matemático) se busca con comillas (simples o dobles)

```
> ?*
Error: unexpected '*' in "?*"
> ?'*'
> help("*")
```

➤ La función **help.search** busca toda la ayuda que exista con una palabra clave entre comillas (también se puede usar ??)

```
> ??mean
> help.search("mean")
```



I. Introducción al R y al R Studio

II. Tipos y estructuras de datos en R

III. Programación básica en R

IV. Análisis Gráfico con R

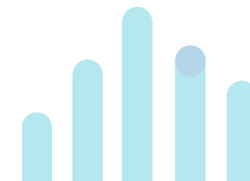
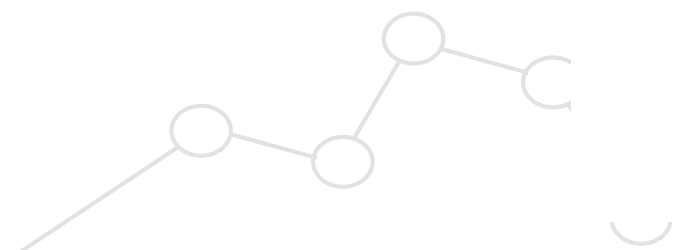
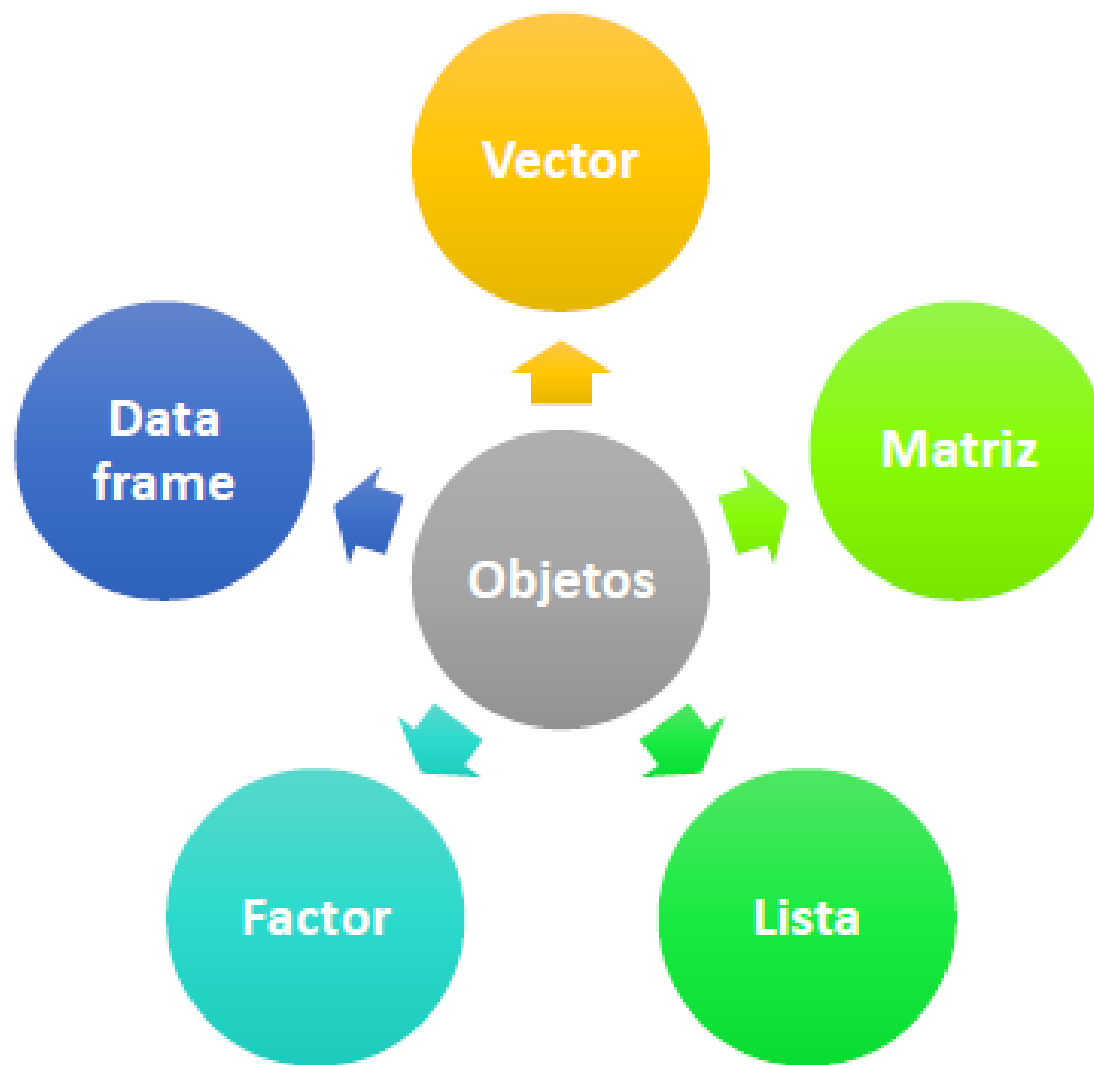
V. Manipulación Avanzada de Dataframes

VI. Introducción al Análisis Estadístico con R

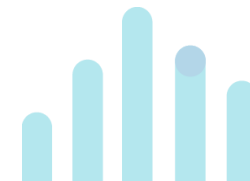
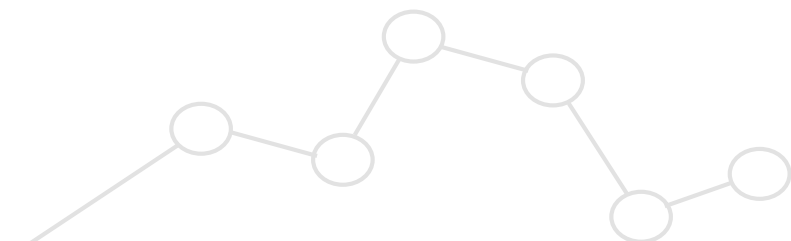
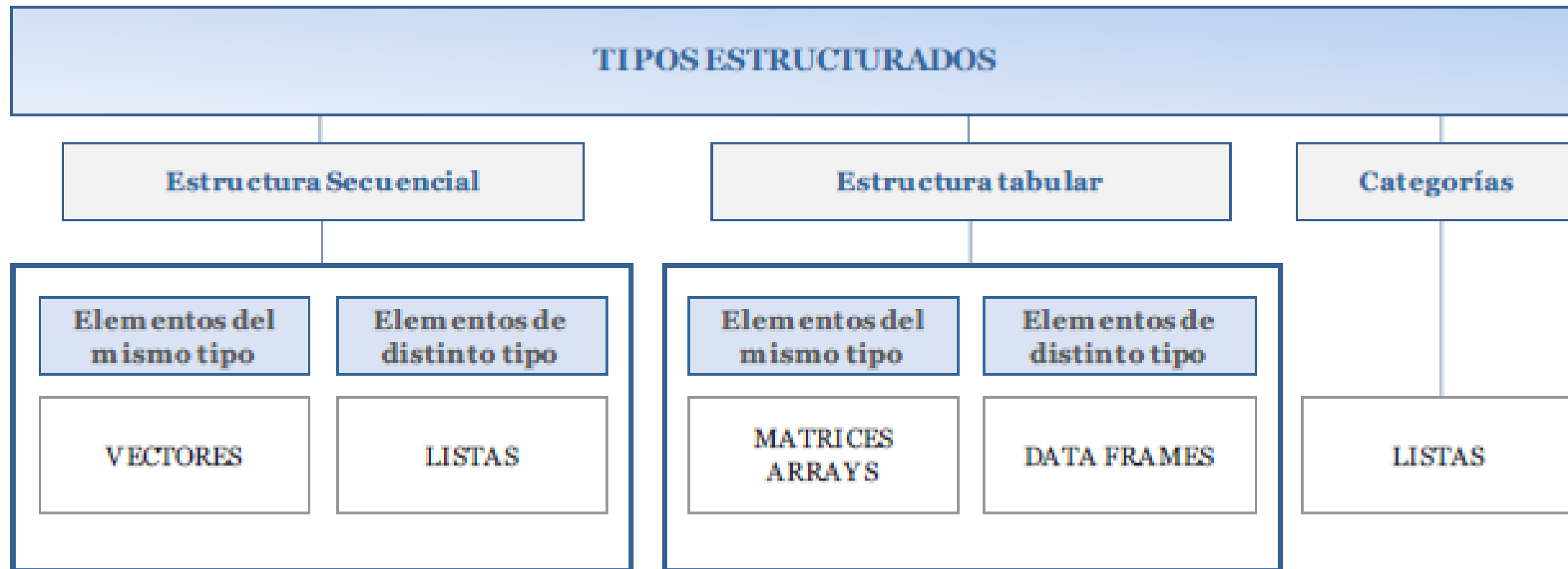
VII. Publicación de Resultados



Estructuras de datos básicos en R



Estructuras de datos básicos en R



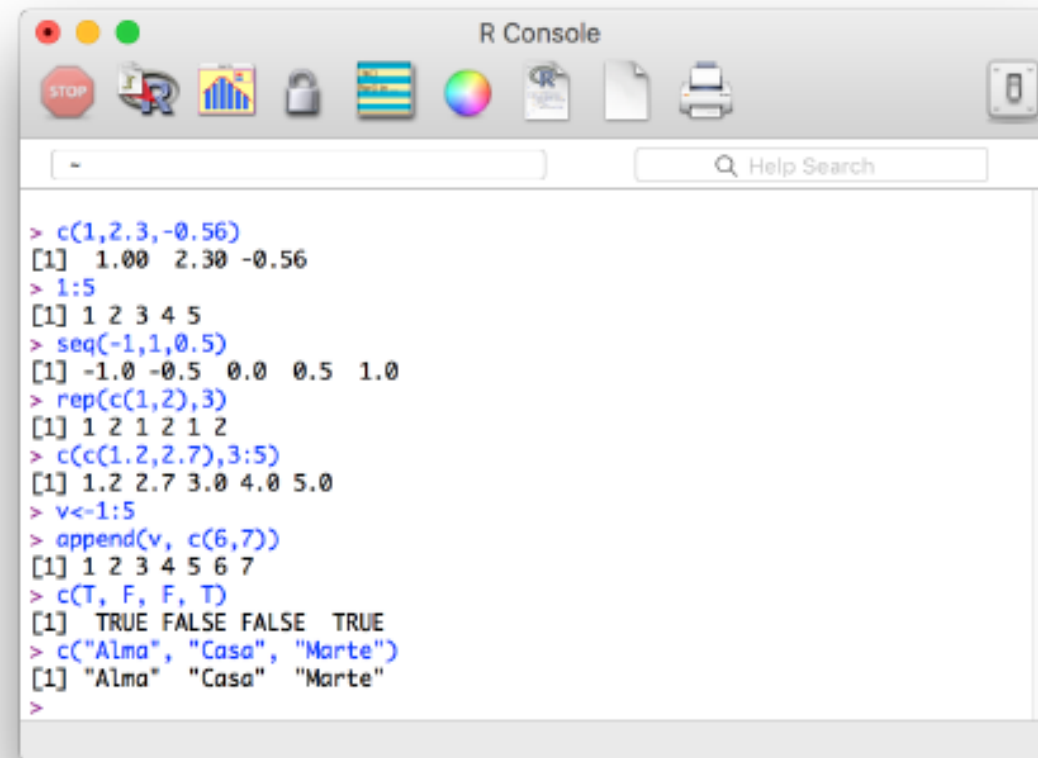
Estructuras de datos en R: Vectores

Estructura de datos: vectores

► Secuencias de elementos del mismo tipo

► Se crean con:

- Función c
- Operador :
- Función seq
- Función rep
- Función append



```

> c(1,2.3,-0.56)
[1] 1.00 2.30 -0.56
> 1:5
[1] 1 2 3 4 5
> seq(-1,1,0.5)
[1] -1.0 -0.5 0.0 0.5 1.0
> rep(c(1,2),3)
[1] 1 2 1 2 1 2
> c(c(1.2,2.7),3:5)
[1] 1.2 2.7 3.0 4.0 5.0
> v<-1:5
> append(v, c(6,7))
[1] 1 2 3 4 5 6 7
> c(T, F, F, T)
[1] TRUE FALSE FALSE TRUE
> c("Alma", "Casa", "Marte")
[1] "Alma" "Casa" "Marte"
>
  
```


Estructuras de datos en R: Vectores

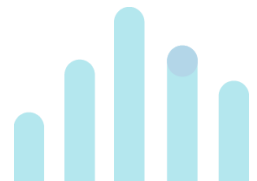
- Un vector es una colección ordenada de elementos del mismo tipo.
- Utilizamos la función `c` para generar vectores.

```
#Creando vectores.

> x<-c(10.4,5.6,3.1,6.4,21.7)
> x
[1] 10.4  5.6  3.1  6.4 21.7

> y<-c(1:5)
> y
[1] 1  2  3  4  5

> w<-c("rojo","verde","azul")
> w
[1] "rojo"  "verde" "azul"
```



Estructuras de datos en R: Vectores

- Generación de sucesiones

```
> 1:20
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
> 1:3*5+2#Se realiza la sucesión primero y luego las operaciones aritméticas
[1] 7 12 17
> seq(1,10,by=2)#Genera una secuencia que inicia en 1 y termina en 10 con elementos que van de 2 en 2
[1] 1 3 5 7 9
```

- Los vectores lógicos aparecen en condiciones lógicas.
- Sólo pueden tomar dos valores TRUE(verdadero) y FALSE(falso).

```
> temp<-x>7
> temp
[1] TRUE FALSE FALSE FALSE TRUE
```

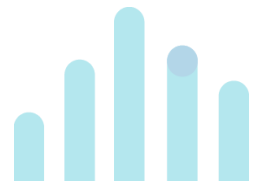
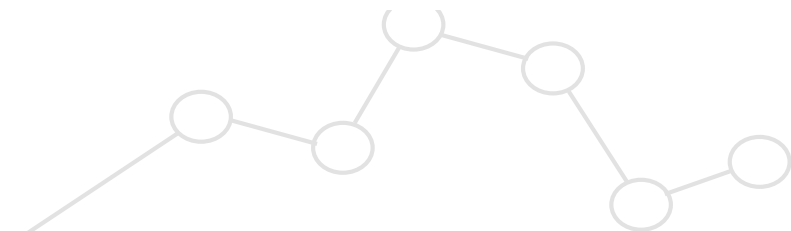


Vectores índices y sub-vectores

- Para seleccionar un sub-vector o un elemento de un vector se utiliza un vector de índices.
- Estructura: **nombre del vector[vector de índices]**

```

> x
[1] 10.4  5.6  3.1  6.4 21.7
> x[4]#Devuelve el elemento de la posición 4
[1] 6.4
> x[1:3]#Devuelve los elementos de x de las posiciones 1,2,3
[1] 10.4  5.6  3.1
> x[-(3:4)]#Devuelve los elementos de x sin considerar los elementos de la posición 3 y 4
[1] 10.4  5.6 21.7
  
```

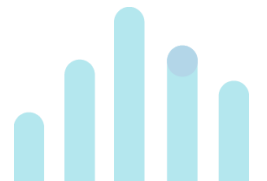
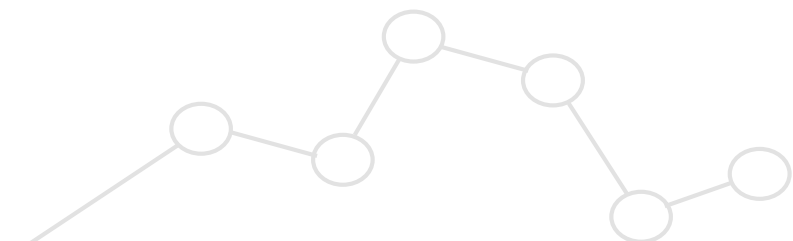


Funciones con Vectores

- Operadores aritméticos elementales +, -, *, / y para la potencia ^.
- Las operaciones se efectúan elemento a elemento.

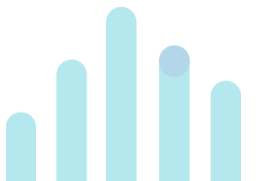
```
> v<-4*x-y+2
> v
[1] 34.2 20.8 9.3 20.2 66.1
```

- Funciones conocidas como `log()`, `exp()`, `sin()`, `cos()`, `tan()`, `sqrt()`, `max()`, `min()`, `length()`, `mean()`, `var(x)`, `sd()`, `sort()`.



Funciones con Vectores

<code>sum(x)</code>	<code>prod(x)</code>	Sum/product of the elements of <code>x</code>
<code>cumsum(x)</code>	<code>cumprod(x)</code>	Cumulative sum/product of the elements of <code>x</code>
<code>min(x)</code>	<code>max(x)</code>	Minimum/Maximum element of <code>x</code>
<code>mean(x)</code>	<code>median(x)</code>	Mean/median of <code>x</code>
<code>var(x)</code>	<code>sd(x)</code>	Variance/standard deviation of <code>x</code>
<code>cov(x,y)</code>	<code>cor(x,y)</code>	Covariance/correlation of <code>x</code> and <code>y</code>
<code>range(x)</code>		Range of <code>x</code>
<code>quantile(x)</code>		Quantiles of <code>x</code> for the given probabilities
<code>fivenum(x)</code>		Five number summary of <code>x</code>
<code>length(x)</code>		Number of elements in <code>x</code>
<code>unique(x)</code>		Unique elements of <code>x</code>
<code>rev(x)</code>		Reverse the elements of <code>x</code>
<code>sort(x)</code>		Sort the elements of <code>x</code>
<code>which()</code>		Indices of TRUEs in a logical vector
<code>which.max(x)</code>	<code>which.min(x)</code>	Index of the max/min element of <code>x</code>
<code>match()</code>		First position of an element in a vector
<code>union(x, y)</code>		Union of <code>x</code> and <code>y</code>
<code>intersect(x, y)</code>		Intersection of <code>x</code> and <code>y</code>
<code>setdiff(x, y)</code>		Elements of <code>x</code> that are not in <code>y</code>
<code>setequal(x, y)</code>		Do <code>x</code> and <code>y</code> contain the same elements?



• ¡GRACIAS!

