

Introducción a R

Aplicaciones a la enseñanza de la Estadística

IV - Encuentro Colombiano de Educación Estadística

ACEdEst

2021-05-31

IV - Encuentro Colombiano de Educación Estadística



Introducción a R con aplicaciones a la enseñanza de la Estadística

Daniel Enrique González Gómez



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS
Acreditación Institucional de Alta Calidad



UNIVERSIDAD PEDAGÓGICA
NACIONAL



LOS LIBERTADORES
FUNDACIÓN UNIVERSITARIA



Universidad
del Tolima



UNIVERSIDAD
DEL CAUCA



Universidad de la
Amazonia

Quienes somos

Agenda

Dia 1 : Introducción a 

Dia 2 :  apoyo a la enseñanza de la Estadística

Dia 3 :  apoyo a las labores de docencia e investigación

Contexto

Comunicación

Aprendizaje

Tecnología

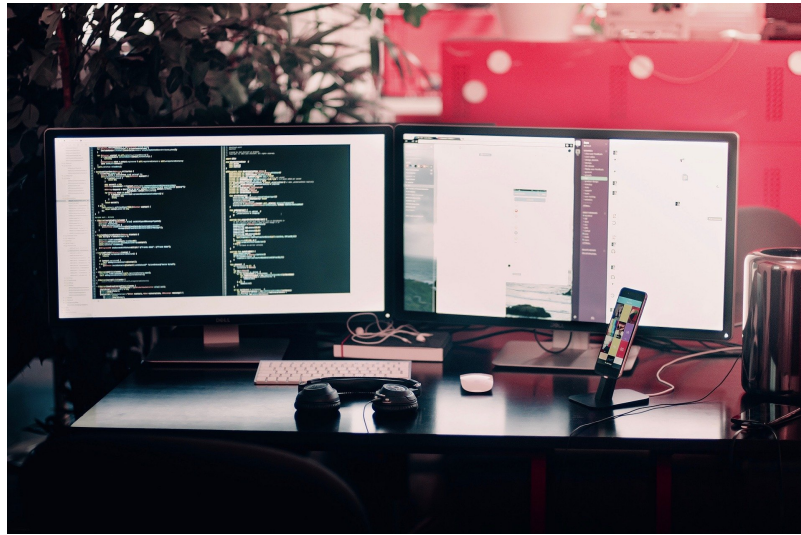
Ayer-Hoy

Enseñanza

Met.Estadística

PyE

La forma de comunicarnos cambia de manera acelerada



Que es R

- Es un lenguaje para la computación estadística
 - Licencia (GNU GPL) abierta y gratis
 - Creciente popularidad en ciencia de datos
 - Multipalataforma (UNIX, Windows, MacOS)
 - Ross Ihaka y Robert Gentleman (U.Auckland - Nueva Zelanda) 1993
 - Lenguaje multiparadigma
 - Código construido en C y Fortran
 - Gran comunidad muy activa
 - Mas de 7000 paquetes

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



Lenguajes utilizados en ciencia de datos



[*] Tomada de: <https://mappinggis.com/2019/07/lenguajes-de-programacion-para-realizar-ciencia-de-datos/>

R y RStudio

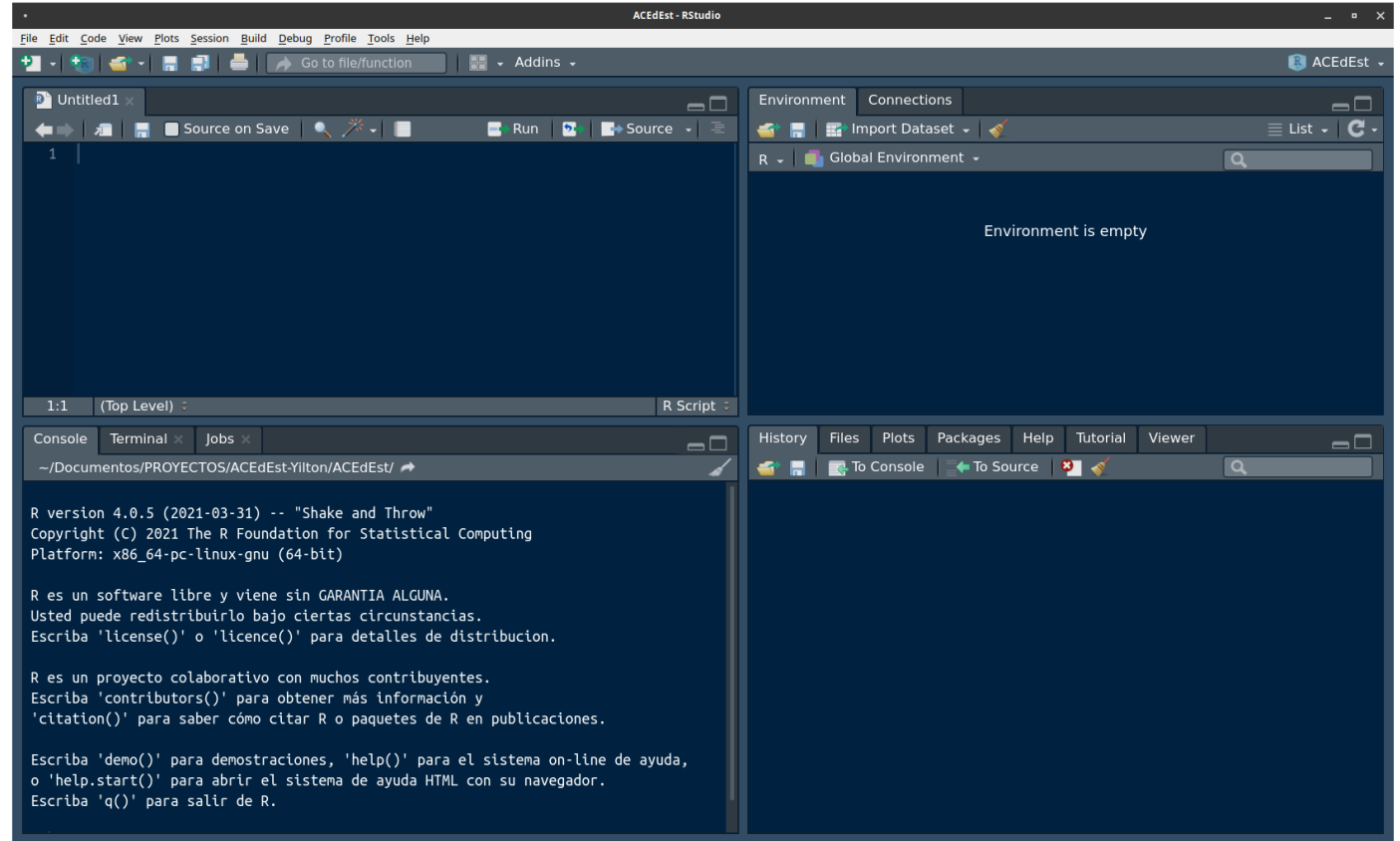
```
Terminal - deg@degE5450: ~/Escritorio
Archivo Editar Ver Terminal Pestañas Ayuda
deg@degE5450:~/Escritorio$ R
R version 4.0.5 (2021-03-31) -- "Shake and Throw"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

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

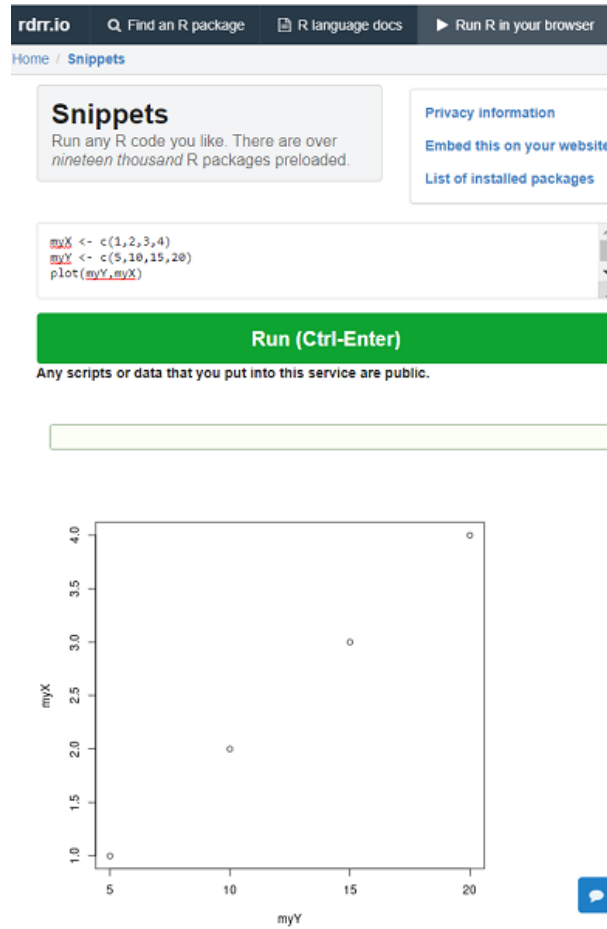
R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

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.

> 
```



R online

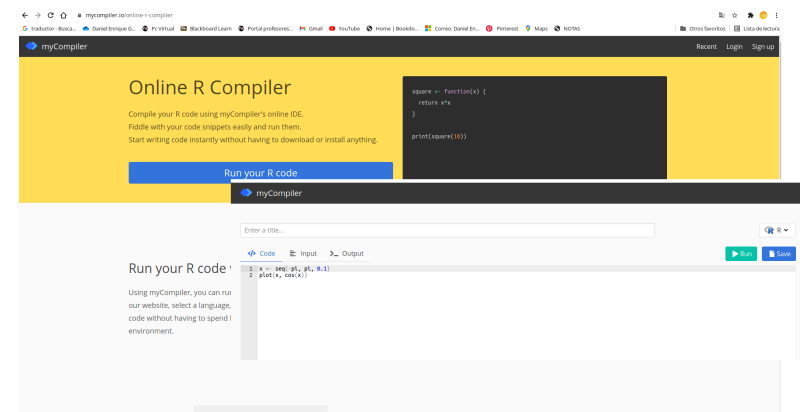


The screenshot shows the rdrv.io website with a dark header containing navigation links: "Find an R package", "R language docs", and "Run R in your browser". The main content area is titled "Snippets" and includes a sub-header "Run any R code you like. There are over nineteen thousand R packages preloaded." Below this is a code editor with the following R code:

```
myX <- c(1,2,3,4)
myY <- c(5,10,15,20)
plot(myY, myX)
```

A green button labeled "Run (Ctrl-Enter)" is positioned below the code. A disclaimer states: "Any scripts or data that you put into this service are public." Below the disclaimer is a light green bar. At the bottom, a scatter plot is displayed with "myY" on the x-axis (ranging from 5 to 20) and "myX" on the y-axis (ranging from 1.0 to 4.0). The plot shows four data points at (5, 1), (10, 2), (15, 3), and (20, 4). A small blue chat icon is in the bottom right corner.

<https://rdrv.io/snippets/>



The screenshot shows the myCompiler.io website with a yellow header. The main content area is titled "Online R Compiler" and includes a sub-header "Compile your R code using myCompiler's online IDE. Fiddle with your code snippets easily and run them. Start writing code instantly without having to download or install anything." Below this is a blue button labeled "Run your R code". To the right of the button is a code editor with the following R code:

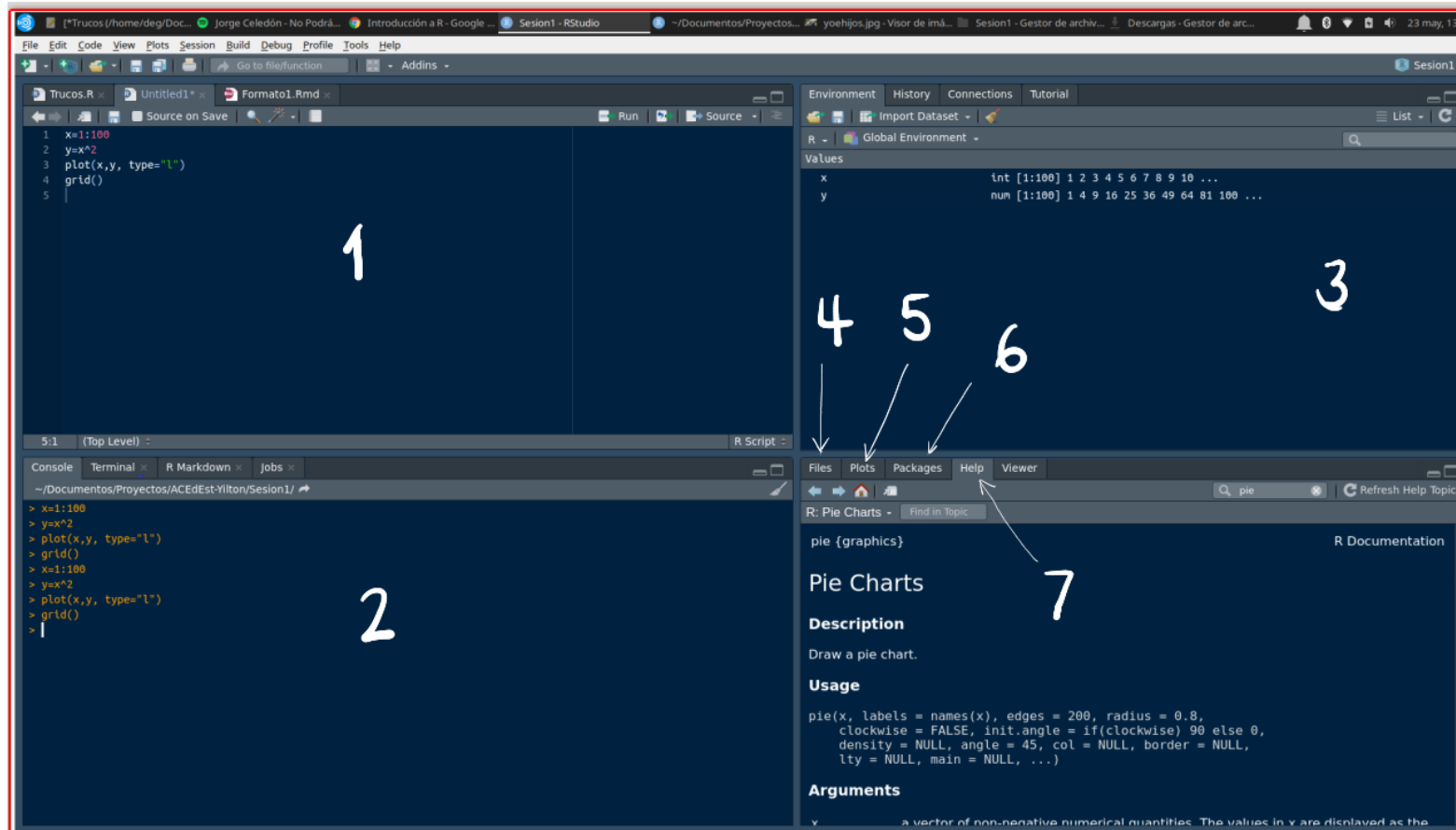
```
square <- function(x) {
  return x*x
}
print(square(5))
```

A green button labeled "Run" is positioned below the code. Below the code editor is a section titled "Run your R code" with a sub-header "Using myCompiler, you can run our website, select a language, code without having to spend an environment." Below this is a light green bar.

<https://www.mycompiler.io/online-r-compiler>



RStudio



1.Fuente/ 2.Consola/ 3.Ambiente/ 4.Archivos/ 5.Graficos/ 6.Paquetes/ 7.Ayudas

Instalación R y RStudio



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

Descargar e instalar R y RStudio

<https://www.youtube.com/watch?v=Nmu4WPdJBRO>



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

Ayuda

```
?pie # abre la ventana de ayudas
```

```
example(pie) # muestra ejemplos de la funcion
```

```
help.start()
```



RStudio cheatsheets

String manipulation with stringr : : CHEAT SHEET

The **stringr** package provides a set of internally consistent tools for working with character strings, i.e. sequences of characters surrounded by quotation marks.



Detect Matches



str_detect(string, pattern) Detect the presence of a pattern match in a string. `str_detect(fruit, "a")`



str_which(string, pattern) Find the indexes of strings that contain a pattern match. `str_which(fruit, "a")`



str_count(string, pattern) Count the number of matches in a string. `str_count(fruit, "a")`



str_locate(string, pattern) Locate the positions of pattern matches in a string. Also **str_locate_all**. `str_locate(fruit, "a")`

Subset Strings



str_sub(string, start = 1L, end = -1L) Extract substrings from a character vector. `str_sub(fruit, 1, 3); str_sub(fruit, -2)`



str_subset(string, pattern) Return only the strings that contain a pattern match. `str_subset(fruit, "b")`



str_extract(string, pattern) Return the first pattern match found in each string, as a vector. Also **str_extract_all** to return every pattern match. `str_extract(fruit, "[aeiou]")`



str_match(string, pattern) Return the first pattern match found in each string, as a matrix with a column for each () group in pattern. Also **str_match_all**. `str_match(sentences, "[a][the] ([^]+)")`

Manage Lengths



str_length(string) The width of strings (i.e. number of code points, which generally equals the number of characters). `str_length(fruit)`



str_pad(string, width, side = c("left", "right", "both"), pad = " ") Pad strings to constant width. `str_pad(fruit, 17)`



str_trunc(string, width, side = c("right", "left", "center"), ellipsis = "...") Truncate the width of strings, replacing content with ellipsis. `str_trunc(fruit, 3)`



str_trim(string, side = c("both", "left", "right")) Trim whitespace from the start and/or end of a string. `str_trim(fruit)`

Mutate Strings



str_sub() <- value. Replace substrings by identifying the substrings with **str_sub**() and assigning into the results. `str_sub(fruit, 1, 3) <- "str"`



str_replace(string, pattern, replacement) Replace the first matched pattern in each string. `str_replace(fruit, "a", ".")`



str_replace_all(string, pattern, replacement) Replace all matched patterns in each string. `str_replace_all(fruit, "a", ".")`



str_to_lower(string, locale = "en")¹ Convert strings to lower case. `str_to_lower(sentences)`



str_to_upper(string, locale = "en")¹ Convert strings to upper case. `str_to_upper(sentences)`



str_to_title(string, locale = "en")¹ Convert strings to title case. `str_to_title(sentences)`

Join and Split



str_c(..., sep = "", collapse = NULL) Join multiple strings into a single string. `str_c(letters, LETTERS)`



str_c(..., sep = "", collapse = "") Collapse a vector of strings into a single string. `str_c(letters, collapse = "")`



str_dup(string, times) Repeat strings times times. `str_dup(fruit, times = 2)`



str_split_fixed(string, pattern, n) Split a vector of strings into a matrix of substrings (splitting at occurrences of a pattern match). Also **str_split** to return a list of substrings. `str_split_fixed(fruit, "i", n=2)`



str_glue(..., sep = "", envir = parent.frame()) Create a string from strings and (expressions) to evaluate. `str_glue("Pi is {pi}")`



str_glue_data(x, ..., sep = "", envir = parent.frame(), na = "NA") Use a data frame, list, or environment to create a string from strings and (expressions) to evaluate. `str_glue_data(mtcars, "{rownames(mtcars)} has {hp} hp")`

Order Strings



str_order(x, decreasing = FALSE, na_last = TRUE, locale = "en", numeric = FALSE, ...) Return the vector of indexes that sorts a character vector. `str_order(x)`



str_sort(x, decreasing = FALSE, na_last = TRUE, locale = "en", numeric = FALSE, ...) Sort a character vector. `str_sort(x)`

Helpers



str_conv(string, encoding) Override the encoding of a string. `str_conv(fruit, "ISO-8859-1")`



str_view(string, pattern, match = NA) View HTML rendering of first regex match in each string. `str_view(fruit, "[aeiou]")`



str_view_all(string, pattern, match = NA) View HTML rendering of all regex matches. `str_view_all(fruit, "[aeiou]")`



str_wrap(string, width = 80, indent = 0, exdent = 0) Wrap strings into nicely formatted paragraphs. `str_wrap(sentences, 20)`



RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more at stringr.tidyverse.org • Diagrams from @LVAudor • stringr 1.2.0 • Updated: 2017-10

Tipos de datos

- **Vectores** : arreglo unidimensional

```
x=c(1,2,3,4,5)      #<<
```

- **Matrices** : arreglo bidimensional

```
x=1:9  
m=matrix(x,nrow=3)   #<<
```

- **Arrays** : arreglos multidimensionales

```
x=1:9  
y=10:19  
mn=array(c(x,y),dim=c(3,3,2)) #<<
```

- **Factores** : vector de variables categóricas

```
x=letters[1:3]  
y=rep(x, times=3)  
z=rep(x, each = 3)
```

Tipos de datos

- **Listas** : colección de objetos cada uno de tipos diferentes

```
h=hist(rnorm(100)) #<<
```

- **Data Frames** : estructura de datos de dos dimensiones - filas y columnas - base de datos

```
data=data(iris) #<<
```

- **Funciones**

```
fx=function(x){1/(x-1)^2} #<<  
fx(100)
```

Resumen

objeto	tipos	varios tipos posibles en el mismo objeto?
vector	numérico, caracter, complejo o lógico	No
factor	numérico o caracter	No
arreglo	numérico, caracter, complejo o lógico	No
matriz	numérico, caracter, complejo o lógico	No
data.frame	numérico, caracter, complejo o lógico	Si
ts	numérico, caracter, complejo o lógico	Si
lista	numérico, caracter, complejo, lógico función, expresión, ...	Si

[*] Tomado de R para principiantes

Operadores

Operadores					
Aritméticos		Comparativos		Lógicos	
+	adición	<	menor que	! x	NO lógico
-	substracción	>	mayor que	x & y	Y lógico
*	multiplicación	<=	menor o igual que	x && y	id.
/	división	>=	mayor o igual que	x y	O lógico
^	potencia	==	igual	x y	id.
%%	módulo	!=	diferente de	xor(x, y)	O exclusivo
%/ %	división de enteros				

[*] Tomado de R para principiantes

Actividad

Instalación de R y RStudio :

- [\href{https://www.r-project.org/}](https://www.r-project.org/)
- <https://rstudio.com/products/rstudio/download/>.

Créditos

- Imágenes
 - <https://pixabay.com/es/images/>
 - <https://medium.com/@gabriela.solera05/el-aula-invertida-en-ingl%C3%A9s-flipped-classroom-es-una-modalidad-de-blended-learning-aprendizaje-86170628d95b>
- R para principiantes, J.A. Ahumada (2003)
- The Book R, T.M. Davies (2016)
- R para profesionales de los datos, C.J. Gil Vellosta (2018)
- Beginning Data Science with R , M. A. Pathak (2014)
- R for Data Science - H.Wickham - G. Grolemund (2016)

Práctica