Tutorial 1 - Introducción a 

Tomás Pacheco

Econometría Avanzada

Maestría en Economía - Universidad de San Andrés

Introducción a R

¿Qué es R?

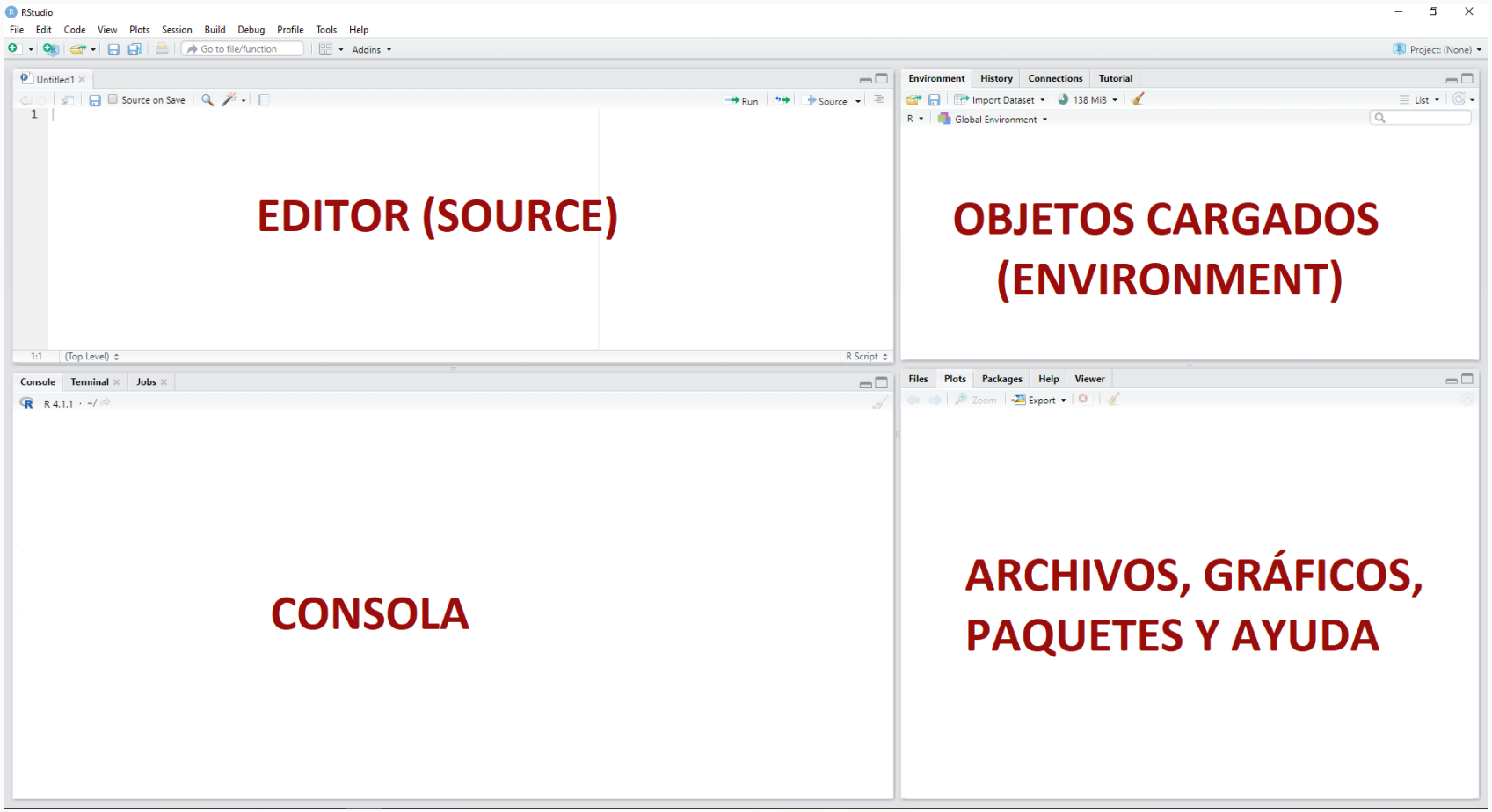
- R es un lenguaje de programación pensado para estadística
- Surgió como simplificación de S-PLUS
- ¡Gratis! Y *open source*. Gracias a eso, mucha información dando vuelta en la red (`https://stackoverflow.com`).
- Algunos libros sobre R acá, acá y acá.

Descarga

Van a tener que descargar dos cosas:

1. **R** desde `http://mirror.fcaglp.unlp.edu.ar/CRAN/` según el sistema operativo que tengan
2. La versión gratuita de **RStudio** desde `https://www.rstudio.com/` eligiendo la versión recomendada para su sistema. Es una interfaz gráfica para usar R que facilita el trabajo (si bien también es posible trabajar con RGui, que se descarga automáticamente en el paso anterior)

Básicos


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1
Source on Save
Run
Source
EDITOR (SOURCE)
Environment History Connections Tutorial
Import Dataset
138 MiB
List
Global Environment
OBJETOS CARGADOS (ENVIRONMENT)
Console Terminal Jobs
R 4.1.1 · ~/
CONSOLA
Files Plots Packages Help Viewer
Zoom
Export
ARCHIVOS, GRÁFICOS, PAQUETES Y AYUDA

Ejecutar y guardar código

- R funciona a través de funciones o comandos. Se pueden ejecutar directamente en la **consola**, escribiéndolos despúes del símbolo $>$ y oprimiendo "Enter".
- Pero en general los escribiremos en el **editor** para poder guardarlos en un archivo **RScript** y ejecutarlos más tarde, o de a muchos juntos. Para ejecutar un comando desde el editor se puede oprimir **Run** o directamente **Ctrl+Enter**. Para ejecutar varias líneas de comando a la vez, seleccionarlas.
- Se puede tener abiertos muchos Scripts a la vez e ir ejecutando comandos de cada uno de ellos alternadamente
- Para **comentar** código (y que esa línea no sea ejecutada), se escribe # antes de la línea

Directorio de trabajo

Antes que nada, conviene definir nuestro **directorio de trabajo**, para que R sepa dónde guardar y desde dónde abrir los archivos que usemos y genermos.

La función para hacerlo es `setwd()` y debemos escribir el directorio alguna de las siguientes formas:

- `setwd("C:/ejemplo/de/directorio")` con barra INVERTIDA
- `setwd("G:\\ejemplo2\\de\\directorio")` con barra DOBLE

Para chequear en qué directorio estamos actualmente, correr `cwd()`.

Objetos

- R tiene distintos objetos. La estructura básica de datos es un **vector** y sus elementos pueden ser números (`numeric`), cadenas de texto (`character`), verdaderos o falso (`logical`), todos del mismo tipo. Las listas (`list`) pueden contener distintos tipos de datos. Las bases de datos son los `DataFrame` o `tibble`. Las funciones también son objetos.
- Podemos usar muchos objetos a la vez (¡incluyendo muchas bases de datos a la vez!)
- Cada objeto tiene un nombre que se asigna con = o <- `nombre = objeto` ; `nombre <- objeto`
  - Sobreescribe (si guardo una cosa nueva con el mismo nombre, reemplaza lo anterior)
  - Es case-sensitive (no es lo mismo *Esto* que *esto*)
  - Muchos errores son por escribir mal nombres sin darnos cuenta

Funciones o comandos

- Para pedir ayuda sobre un comando, ejecutar `help(comandoencuestión)` o `?comandoencuestión`
- R incluye muchas funciones de base por ejemplo `setwd()`
- Además, hay librerías o paquetes extra con grupos de funciones (u otros objetos). Se instalan por primera vez con `install.packages("nombrelibería")` y antes de usar las funciones en esa librería, debe ser abierta ejecutando `library("nombrelibrería")`
- Además, podemos armar y definir nuestras propias funciones
- Las funciones pueden (o no) tener agumentos. Por ejemplo, `cwd()` no tiene y el directorio es un argumento de `setwd()`. Los argumentos van entre paréntesis: `nombrefuncion(argumento1, argumento2, ...)`

¡Ejemplos en el Script **T1 intro a R.R**!

Subsetting

- En las bases de datos, podemos usar el símbolo $ para llamar a una columna particular de un dataset (`data$columna`)
- Además, podemos indicar con qué números de filas y/o columnas quedarnos. Para elegir subconjuntos de datos (*subsetting*) se usan corchetes, con la estructura: `data[qué filas, qué columnas]`. Lo mismo para matrices, o vectores con `vector[qué elementos]`
  - `data[c(1,2),1:5]` filas 1 y 2 de las columnas 1 a 5
  - `data[2,]` fila 2 y **todas las columnas**
  - `data[,c("nombretalcolumna","nombreotracolumna")]` todas las filas y las columnas que quiera (sólo en los dataframes las columnas tienen nombre)

Importar bases de datos

- Podemos abrir distintos tipos de archivos (guardaremos la base con en nombre data, puede ser cualquier otro)
  - `data <- read.csv("archivo.csv")` y si los datos están separados por punto y coma, usar `read.csv2`
  - `data <- read.table("archivo.txt", header = TRUE)`
  - para abrir archivos de Excel, primero hay que instalar la librería readxl usando `install.package("readxl")`. Luego, abrirla corriendo`library(readxl)`. Una vez hecho esto, procedemos a abrir la base con `data <- read.excel("archivo.xlsx")`
- También lo podemos hacer con la interfaz gráfica

Exportar bases de datos

Si quieren guardar una base de datos, lo mejor es hacerlo en formato csv (Excel tiene la desventaja de cambiar el formato de los datos en algunas ocasiones). Como ya definieron el directorio, el archivo se guardará en esa carpeta elegida previamente-.

- `write.csv(data, file = "data.csv")`
- En otros formatos, con otros separadores o decimales `write.table(data, "data.txt", sep=" ", dec=",")`
- `library(xlsx)` seguido de `write.xlsx(data, "data.xlsx")` para Excel

Regresión Lineal

Regresión lineal

- Código general para estimar el modelo lineal: `lm(y ~ x1+ x2 + data = basededatos)`. Ver `?lm` para más opciones.
- En el R script que acompaña la tutorial, guardamos el resultado de la regresión en un objeto con el nombre "lm.fit". Para ver qué información se guarda en ese objeto, correr `names(lm.fit)`. Por ejemplo, podemos ver los coeficiente corriendo `lm.fit$coefficients`
- Alternativamente, para ver el valor estimado de los coeficientes podemos correr directo `lm.fit` o `coef(lm.fit)`
- Información más detallada sobre la regresión corriendo `summary(lm.fit)`

¡Más detalles en el script!