



dcc

CIENCIAS DE LA COMPUTACIÓN
UNIVERSIDAD DE CHILE

CC1000

Herramientas Computacionales para Ingeniería y Ciencias

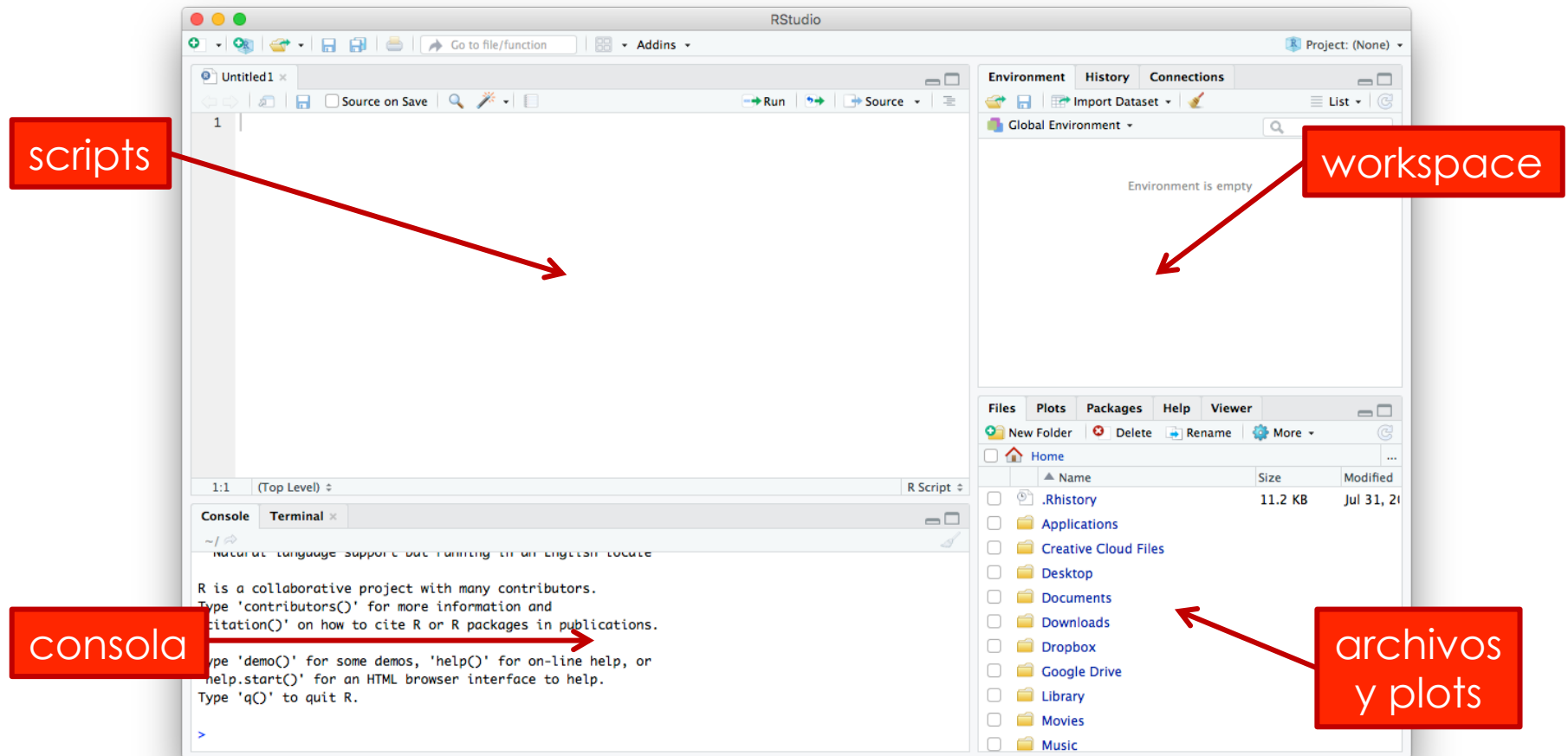
Laboratorio #10 – Introducción a R / RStudio

Francisco J. Gutiérrez
frgutier@dcc.uchile.cl

COMPONENTES DE R

- **Consola:** ejecutar comandos de forma interactiva
- **Scripts:** programas que pueden ser ejecutados una y otra vez
- **Workspace:** conjunto de objetos creados por el usuario, a través de la consola o ejecutando un script
- **Plots:** gráficos creados para visualizar datos

RSTUDIO



TRABAJANDO CON LA CONSOLA

Podemos usar la consola como una calculadora simple:

```
Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.
```

```
> 12 * 23  
[1] 276  
> 5 ^2  
[1] 25  
>
```

El **[1]** indica la posición dentro del vector de respuestas

TRABAJANDO CON LA CONSOLA

Podemos guardar cálculos asignándolos a variables:

```
> a <- 12 * 23  
> b <- 5 ^2  
> a  
[1] 276  
> b  
[1] 25  
> a * b -> a  
> a  
[1] 6900
```

VECTORES

Los **vectores** son objetos que permiten guardar información

Son el tipo de dato más comúnmente usado en R

```
> x <- c(1, 2, 3.5, 3.6, 5, 29, 1.4)
> x #todo lo que viene despues del # es un comentario
[1] 1.0 2.0 3.5 3.6 5.0 29.0 1.4
>
> vocales <- c("a", "e", "i", "o", "u")
> vocales
[1] "a" "e" "i" "o" "u"
```

La función **c()** toma un número arbitrario de argumentos y retorna un nuevo vector con dichos elementos

VECTORES

Podemos usar el operador : (*dos puntos*) para generar secuencias de números

```
> x <- 1:15
> x
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
> y <- 30:1
> y
[1] 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
> z <- 1.2 : 3.2
> z
[1] 1.2 2.2 3.2
```

VECTORES

Podemos acceder a los elementos de un vector por su índice:

```
> e <- c(22, 33, 44, 55)
> e
[1] 22 33 44 55
>
> e[1]
[1] 22
> e[2]
[1] 33
> e[length(e)]
[1] 55
```

- En R, los índices parten en 1
- La función **length(x)** retorna el largo del vector **x**

VECTORES

Podemos realizar cálculos con vectores:

```
> x <- 1:10
> x
[1] 1 2 3 4 5 6 7 8 9 10
> x2 <- c(0,2)
> z <- x + x2
> z
[1] 1 4 3 6 5 8 7 10 9 12
>
> y <- 1/x
> y
[1] 1.0000000 0.5000000 0.3333333 0.2500000 0.2000000 0.1666667 0.1428571 0.1250000 0.1111111
[10] 0.1000000
```

FUNCIONES PREDEFINIDAS

R incluye varias funciones que podemos aplicar a variables y vectores:

```
> log(1)
[1] 0
> log(1:10)
[1] 0.0000000 0.6931472 1.0986123 1.3862944 1.6094379 1.7917595 1.9459101 2.0794415 2.1972246
[10] 2.3025851
>
> x <- c(2,9,1,6,7,8,4,5,3,10)
> x
[1] 2 9 1 6 7 8 4 5 3 10
> max(x)
[1] 10
> min(x)
[1] 1
> range(x)
[1] 1 10
```

Formato: **salida** <- **nombreFuncion**(parametros)

Otras funciones: **exp**, **sin**, **cos**, **sqrt**, ...

FUNCIONES PREDEFINIDAS

En particular, tenemos funciones para calcular estadísticos sobre vectores numéricos:

```
> x <- 1:5
> sum(x)
[1] 15
> prod(x)
[1] 120
> mean(x)      #equivalente a sum(x) / length(x)
[1] 3
> var(x)       #equivalente a sum((x - mean(x))^2) / (length(x) - 1)
[1] 2.5
> median(x)
[1] 3
```

Pueden usar el comando **help(...)** para ver todos los parámetros que recibe una función

SECUENCIAS DE NÚMEROS

Podemos usar la función **seq()** para generar secuencias no triviales de números:

```
> seq(from=4, to=10, by=0.5)
[1] 4.0 4.5 5.0 5.5 6.0 6.5 7.0 7.5 8.0 8.5 9.0 9.5 10.0
>
>
> seq(from=4, to=10, length=10)
[1] 4.000000 4.666667 5.333333 6.000000 6.666667 7.333333 8.000000 8.666667 9.333333
[10] 10.000000
```

from = **x**, significa *desde x*

to = **y**, significa *hasta y*

by = **p**, significa *con paso p*

length = **n**, significa *con n elementos*

} No se pueden usar
simultáneamente

VALORES LÓGICOS

R reconoce tres valores lógicos:

TRUE, verdadero

FALSE, falso

NA, dato no disponible

Operadores relacionales: <, >, <=, >=, ==, !=

```
> 1 > 0
[1] TRUE
> c(1,4,8) > 0
[1] TRUE TRUE TRUE
> c(1,4,8) >= 4
[1] FALSE TRUE TRUE
```

VECTORES LÓGICOS

```
> x <- seq(1,10)
> x
[1] 1 2 3 4 5 6 7 8 9 10
> pares <- x %% 2 == 0
> pares
[1] FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE TRUE
> sum(pares)
[1] 5
```

TRUE y FALSE toman los valores 1 y 0 en cualquier cálculo numérico.
Por ejemplo, TRUE + TRUE = 2.

Operadores Lógicos

- `c1 & c2`: conjunción (**c1 y c2**)
- `c1 | c2`: disyunción (**c1 o c2**)
- `!c`: negación (**no c**)

DATOS NO DISPONIBLES

Al operar con valores no disponibles (**NA**), el resultado es NA:

```
> 5 + NA
[1] NA
>
> a <- 5
> a + NA
[1] NA
>
> x <- c(2, 6, 2, NA, 6, 2, NA, 3, 4, NA, 9)
> x + 4
[1] 6 10 6 NA 10 6 NA 7 8 NA 13
```

La función **is.na(x)**, devuelve TRUE si el elemento pasado como parámetro es NA (y FALSE en caso contrario)

```
> is.na(x)
[1] FALSE FALSE FALSE TRUE FALSE FALSE TRUE FALSE FALSE TRUE FALSE
```

SECUENCIAS DE ÍNDICES

Podemos usar `[]` para seleccionar varios elementos de un vector al mismo tiempo:

```
> x <- 0:10
> x
[1] 0 1 2 3 4 5 6 7 8 9 10
> x[x > 6]
[1] 7 8 9 10
> x[seq(1, length(x), by=2)]
[1] 0 2 4 6 8 10
> x[-3]      #quitamos el elemento de indice 3
[1] 0 1 3 4 5 6 7 8 9 10
> x[-(0:5)]   #quitamos los elementos de indices 0 a 5
[1] 5 6 7 8 9 10
>
>
> x <- c(2, 6, 2, NA, 6, 2, NA, 3, 4, NA, 9)
> x[!is.na(x)] #quitamos los elementos NA
[1] 2 6 2 6 2 3 4 9
```

SECUENCIAS DE ÍNDICES

También podemos usar `[]` para modificar varios elementos de un vector al mismo tiempo:

```
> x <- 0:10
> x
[1] 0 1 2 3 4 5 6 7 8 9 10
> x[x %% 2 == 0] <- 0    #dejar en 0 los elementos pares
> x
[1] 0 1 0 3 0 5 0 7 0 9 0
>
> x <- 0:10
> x[x > 4] <- x[x > 4] + 4    #sumarle 4 a los elementos mayores a 4
> x
[1] 0 1 2 3 4 9 10 11 12 13 14
```

ACTIVIDAD DE PREPARACIÓN

1. ¿Qué comandos de R permitirían realizar las siguientes operaciones?
 - a) Calcular 123 por 11, dejando el valor en la variable `a`
 - b) Calcular 4 elevado a 5, dejando el valor en la variable `b`

2. ¿Qué comandos permiten realizar las siguientes operaciones?
 - a) Crear la secuencia: -5 -4 -3 -2 -1 0 1 2 3 4 5
 - b) Crear el vector: 1 2 3 3 2 1 1 2 3

3. Sin usar la instrucción `c()` :
 - a) Cree el vector de los primeros 15 números naturales al cuadrado (1, 4, 9, 16, ..., 225)
 - b) Cree el vector con los números pares entre 10 y 40, ambos inclusive