Curso Introducción a R

Clase 1

Joaquin Cavieres G.

Estudiante doctorado

j.cavieres.g@gmail.com

3 de octubre de 2019

INTRODUCCIÓN

Lectura recomendada:

- An Introducción to R (R Development Core Team) ▸ Link
- Yet another R Introduction (Andreas Handel) ▸ Link

¿Que es R?

- R es un ambiente de programación para analisis estadísticos, construcción de gráficos, modelos, maquinas de aprendizaje, etc.
- Es un software "open source" (de libre acceso).
- Es un software que está en constante mejoramiento debido a la contribucuion de diversos autores.

¿Que es R?

- Es un efectivo software para el almacenamiento y manejo de datos.
- Permite la utilización de diversas herramientas para diferentes análisis de datos.
- Tiene propiedades gráficas amigables a los usuarios
- Lenguaje de programación simple y eficaz, permite la creación de "loops" y obtención de resultados de forma sencilla.

Manos a la obra...

Sitio web de R

- Sitio web: http://www.r-project.org ▸ Link
- Descarga del programa: http://cran.r-project.org ▸ Link

¿Que nos permite hacer R?

- Test estadísticos clásicos
- Análisis de cluster
- Modelado lineal y no lineal
- Análisis de series de tiempo
- Análisis de datos espaciales (latitud-longitud)
- Modelado bayesiano
- .... y muchas mas alternativas (Random Forest, Big Data, Redes Neuronales, Machine Learning, etc)

¿Que nos permite hacer R?

Gráficas de alta calidad

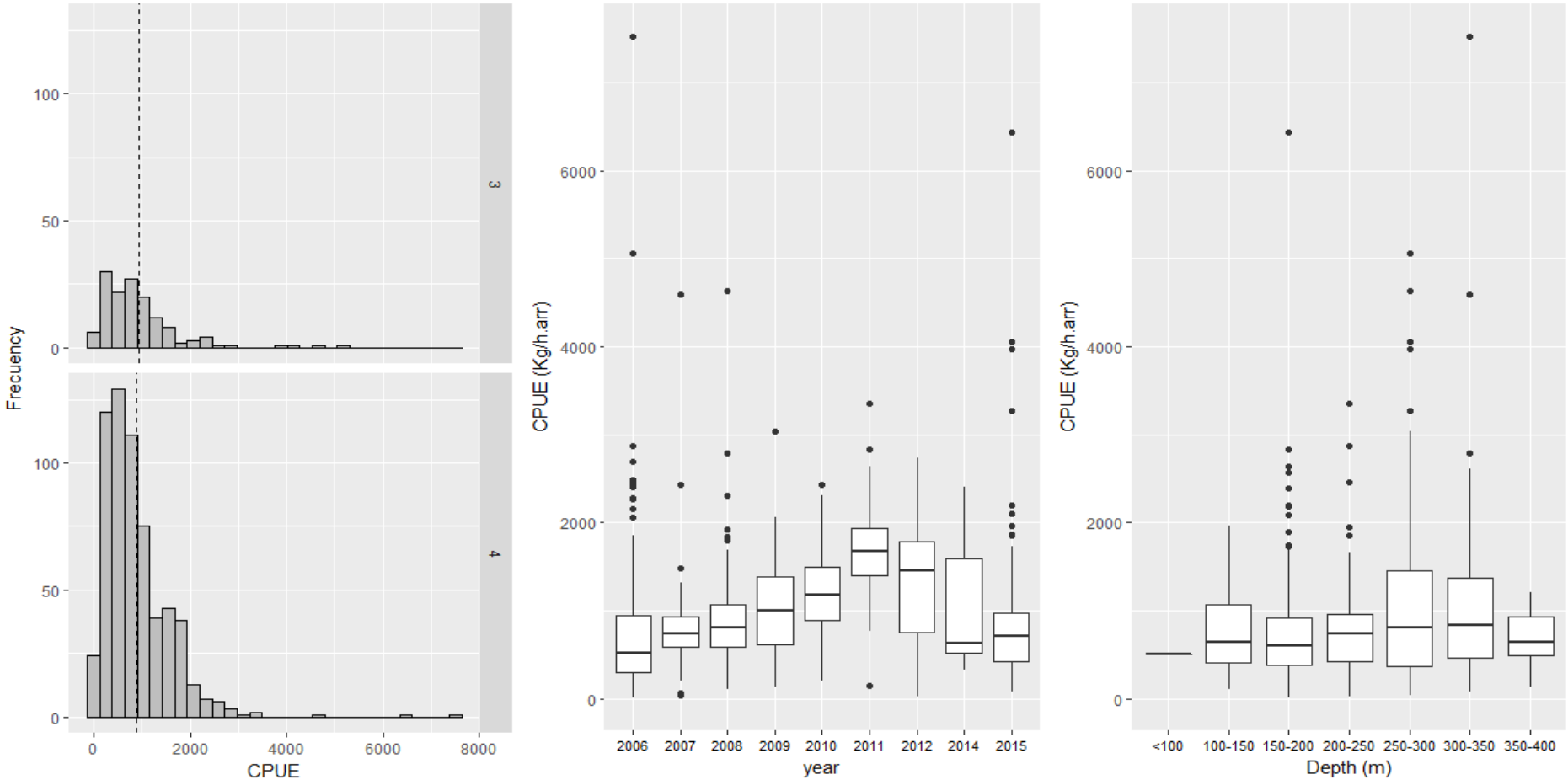

Figura 1: Histogramas y box-plot.

¿Que nos permite hacer R?

Mapas de datos espaciales

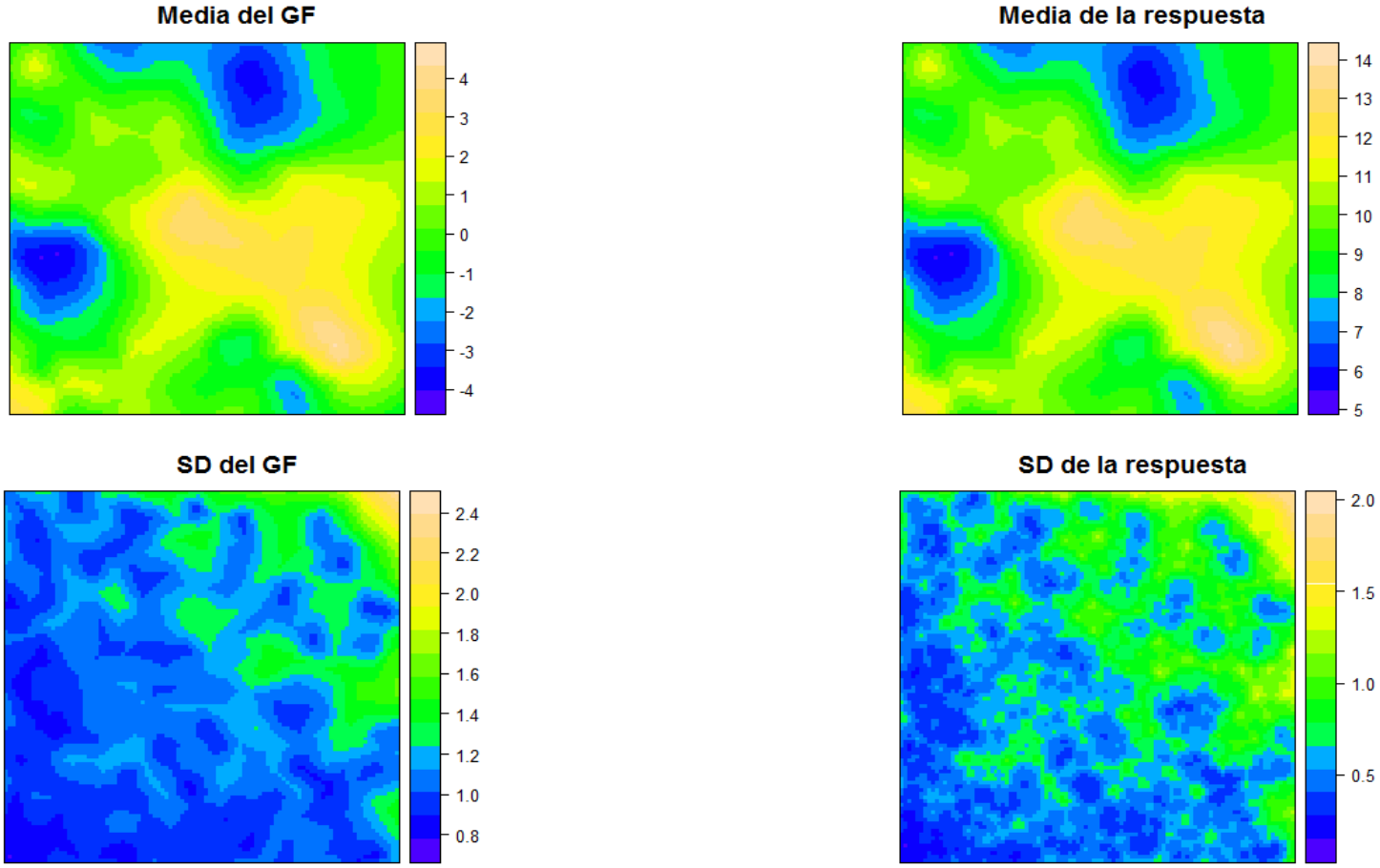

Figura 2: Histogramas y box-plot.

¿Ayuda adicional?

R presenta un comando que permite dar información adicional sobre algun tipo de instrucción que no se este declarando de la forma adecuada. Por ejemplo:

- `help(log)` ▸ Link
- `help(glm)` ▸ Link

R studio

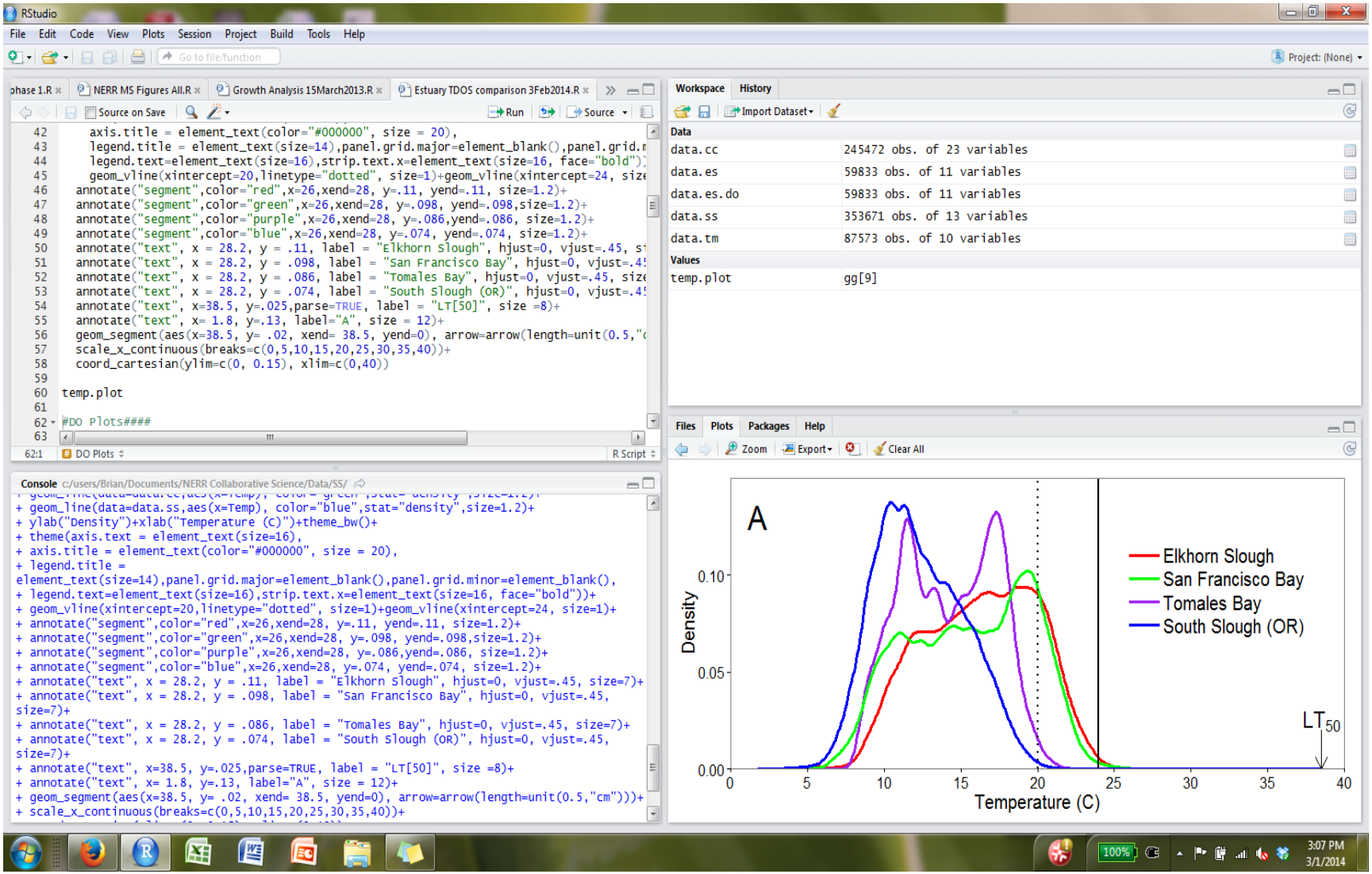

FIGURA 3: Software Rstudio.

Comandos simples

- El signo "#" permite "comentar" la línea para que no sea leída.

```
#En esta linea se declara que los datos..... etc...
```

- Si se quiere escribir un algoritmo entonces:
  - Utilizar nombres recordables
  - No usar líneas repetidas en el codigo
  - No asumir que se recordaran comandos mas adelante en las líneas de codigo.

Comandos simples

$> 2 + 2$

```
[1] 4
```

$> 2^2$

```
[1] 4
```

$> 2 * (1 + 1)$

```
[1] 4
```

Ejercicios simples

- Escriba en R y calcule:

$$> 1 + 2(3 + 4)$$

Ejercicios simples

- Escriba en R y calcule:

$> 1 + 2(3 + 4)$

$> log(4^3 + 3^{2+1})$

Ejercicios simples

- Escriba en R y calcule:

$> 1 + 2(3 + 4)$

$> log(4^3 + 3^{2+1})$

$> \sqrt{(4 + 3)(2 + 1)}$

Ejercicios simples

- Escriba en R y calcule:

$> 1 + 2(3 + 4)$

$> log(4^3 + 3^{2+1})$

$> \sqrt{(4 + 3)(2 + 1)}$

$> (\frac{1+2}{2+4})^2$

CREACION DE OBJETOS

- Cada resultado obtenido de cualquier operación matemática puede almacenarse en un **objeto**.
    - Numeros
    - Caracteres
    - Tablas
    - Vectores/Matrices
    - Gráficos
    - Resultados estádisticos

Es necesario almacenar los objetos con nombres que sean recordables.

Asignación

Podemos asignar un valor a una variable, por ejemplo:

```
variable <- x
```

A "variable" le hemos asignado el valor de x.

Pero también podemos asignar con el signo "=", por ejemplo:

```
variable = x
```

Asignación

También podemos asignar caracteres a una variable, esto utilizando “” para la asignación:

```
nombre <- ‘‘Juan’’
```

A “nombre’ le hemos asignado el caracter “Juan”.

Si queremos otorgar un nombre con espacio, debemos escribir:

```
nombre <- ‘‘Juan Cavieres’’
```

Ver objetos creados

Existen diferentes formas de visualizar los objetos creados.

```
print(nombre)
```

```
[1] Juan Cavieres.
```

```
nombre
```

```
[1] Juan Cavieres.
```

Si queremos manipular un objeto creado podemos hacer: `x <- 2`

```
x*2
```

```
[1] 4.
```

Vectores

Operaciones con vectores

```
> x <- 1:3

log(x)

[1] 0.0000000 0.6931472 1.0986123.

> x + 1

[1] 2 3 4.

> x*2

[1] 2 4 6.
```

Vectores

Operaciones con vectores

```
> x <- 1:3

log(x)

[1] 0.0000000 0.6931472 1.0986123.

> x + 1

[1] 2 3 4.

> x*2

[1] 2 4 6.
```

Vectores

Operaciones con vectores

```
> x <- 1:3

log(x)

[1] 0.0000000 0.6931472 1.0986123.

> x + 1

[1] 2 3 4.

> x*2

[1] 2 4 6.
```

Eliminar objetos creados

- Podemos ejecutar el comando `list()` para ver los objetos creados.

```
ls()
```

Eliminar objetos creados

- Podemos ejecutar el comando `list()` para ver los objetos creados.

```
ls()
```

```
[1] ‘‘name’’ ‘‘variable’’
```

Eliminar objetos creados

- Podemos ejecutar el comando `list()` para ver los objetos creados.

```
ls()
```

```
[1] ‘‘name’’ ‘‘variable’’
```

- Para remover un objeto en particular ejecutamos `rm()`, por ejemplo, si queremos eliminar el objego “name”, escribimos:

```
rm(name)
```

Eliminar objetos creados

- Podemos ejecutar el comando `list()` para ver los objetos creados.

```
ls()
```

```
[1] ‘‘name’’ ‘‘variable’’
```

- Para remover un objeto en particular ejecutamos `rm()`, por ejemplo, si queremos eliminar el objego “name”, escribimos:

```
rm(name)
```

```
ls()
```

Eliminar objetos creados

- Podemos ejecutar el comando `list()` para ver los objetos creados.

```
ls()
```

```
[1] ‘‘name’’ ‘‘variable’’
```

- Para remover un objeto en particular ejecutamos `rm()`, por ejemplo, si queremos eliminar el objego “name”, escribimos:

```
rm(name)
```

```
ls()
```

```
[1] ‘‘variable’’
```

Eliminar objetos creados

Si queremos eliminar todos los objetos escribimos: `rm(list=ls())`

Eliminar objetos creados

Si queremos eliminar todos los objetos escribimos: `rm(list=ls())`

Revisamos el contenido de los objetos creados con `list()`...

Tipos de datos

1. Existen distintos tipos de datos los cuales describen como son guardados en el computador.
2. Cuando se guarda un objeto en R, no es necesario especificar el tipo de dato guardado.
3. Los diferentes tipos de datos en R son:
   - Numericos (Enteros, puntos flotantes, etc)
   - Lógicos (Boleanos: Verdadero/Falso)
   - Caracteres (textos de datos)
4. Los tipos de datos no son obvios, sobretodo cuando son leídos de fuentes externas, por eso es necesario saber a que tipo de datos pertenecen.

Tipos de datos

```
>variable2 <- 2

>variable2

[1] 2.

>mode(variable2)

[1] ‘‘numeric’’.

>is.numeric(variable2)

[1] ‘‘TRUE’’.
```

Tipos de datos

Funciones similares pueden ser aplicadas a los objetos declarados como “caracter”

```
>is.character(variable2)
```

```
[1] ‘‘FALSE’’.
```

```
>is.character(nombre)
```

```
[1] ‘‘TRUE’’.
```

Las clases “numeric” y “character” son las mas comunmente encontradas en las declacarión de objetos

Vectores

Un vector es un colección de valores ordenados

```
>length(vector)
```

```
[1] 1 2 3 4 5.
```

Vectores

Se puede crear vectores de diferentes formas, por ejemplo, si queremos crear un vector de 1 a 10 escribimos:

```
>c(1,2,3,4,5,6,7,8,9,10)
```

```
[1] 1 2 3 4 5 6 7 8 9 10.
```

donde la letra "c" significa "concatenación" (combina los elementos en forma de vector)

Podemos escribir lo anterior escribiendo una secuencia de numeros como:

```
>1:10
```

o

```
>seq(from=1, to=10, by=1)
```

o también,

```
seq(1,10,1)
```

Vectores

Otra opción de crear vectores es mediante la función `rep()`.

```
> x <- 1:3                    # a "x" le otorgo los valores de 1 a 3.
> x
[1] 1 2 3.
> rep(x, length=10)      # rep replica los valores en "x" n veces
[1] 1 2 3 1 2 3 1 2 3 1.
> rep(x, times=2)
[1] 1 2 3 1 2 3.
> rep(x, each=2)
[1] 1 1 2 2 3 3.
```

Vectores

Operaciones con vectores

```
> x <- 1:3

log(x)

[1] 0.0000000 0.6931472 1.0986123.

> x + 1

[1] 2 3 4.

> x*2

[1] 2 4 6.
```

Vectores

Operaciones con vectores

```
> y <- 4:6
```

```
> x + y
```

```
[1] 5 7 9
```

```
> x - y
```

```
[1] -3 -3 -3.
```

```
> x / y
```

```
[1] 0.25 0.40 0.50.
```

Vectores

Operaciones con vectores

```
> y <- 4:6
```

```
> x + y
```

```
[1] 5 7 9
```

```
> x - y
```

```
[1] -3 -3 -3.
```

```
> x / y
```

```
[1] 0.25 0.40 0.50.
```

Vectores

Operaciones con vectores

```
> y <- 4:6
```

```
> x + y
```

```
[1] 5 7 9
```

```
> x - y
```

```
[1] -3 -3 -3.
```

```
> x / y
```

```
[1] 0.25 0.40 0.50.
```

Ejercicios

Cree vectores usando las funciones `seq()`, `rep()` y `c()` en los siguientes casos:

- Enteros positivos desde 1 a 50
- Impares positivos de 1 a 20
- El vector 1,1,1,2,2,2,3,3,3
- El vector 1,2,3,4,5,1
- Los numeros 1,8,27,64,125,216