Introducción a R Markdown

Pablo Aguirre Hörmann

# R Markdown

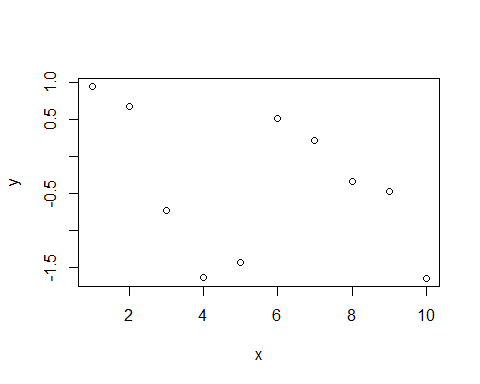
Este es un archivo de tipo *R Markdown* (.Rmd), en este tipo de archivos podemos de manera simple intercalar texto con código.

El código se define en los “pedazos de código” (*code chunks*):

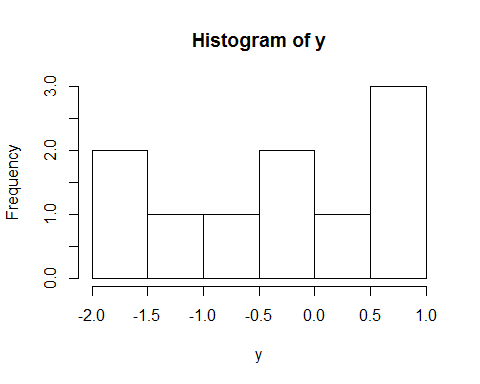
# Una suma por ejemplo  
2 + 2

## [1] 4

# o un gráfico simple  
x <- 1:10  
y <- rnorm(10)  
plot(x, y)



Y estos “pedazos de código” tienen distintas opciones que por ejemplos nos permiten solo motrar el resultado y no el código que lo genera (echo = FALSE):



# Formato de texto

Así como hay formas de manipular el código que trabajemos existe también una gran variedad de opciones para dar formato al texto que queramos incluir en un documento:

* *cursiva* o *cursiva*
* **negrita** o **negrita**
* ***cursiva y negrita*** o ***cursiva y negrita***
* ~~tachado~~
* [link](https://portal.sma.gob.cl/)
* Objetos de código
* entre otras..

También podemos definir distintos niveles de jerarquía respecto a títulos y subtítulos

# Título 1

## Título 2

### Título 3

#### Título 4

##### Título 5

###### Título 6

Por otro lado, este tipo de archivo también nos facilita escribir formulas/operaciones/objetos matemáticos (con sintáxis $\LaTeX$) tanto en linea con el texto () como en su propio párrafo:

Incluso podemos agregar imágenes directamente a nuestros documentos:  


# Código en línea

Una de las principales caractéristicas que nos facilitará la creación de reportes automatizados es que no solo podemos escribir código en los “pedazos de código” (*code chunks*) sino que también podemos escribir **código en línea con el texto**.

resultado <- 2 + 2  
rel <- round(cor(x,y), 2)

La suma del cálculo hecho arriba es igual a **4** y la correlación entre x e y es **-0.42**.

***Para más información sobre como trabajar con documentos en R Markdown ver*** [***la hoja de referencia***](https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-spanish.pdf) ***y el libro*** [***R Markdown: The Definitive Guide***](https://bookdown.org/yihui/rmarkdown/)

# Cargar datos a R

R puede cargar datos de manera bastante sencilla desde distintos origenes, en esta demostación trabajaremos con planillas. Por ejemplo, cargaremos información desde un archivo csv:

library(readr) # librería con funciones de carga de datos optimizadas  
Resumen\_UnidadFiscalizable <- read\_csv("../Datos/Resumen\_UnidadFiscalizable.csv")  
head(Resumen\_UnidadFiscalizable)

## # A tibble: 6 x 16  
## UnidadFiscaliza~ Direccion ComunaId ComunaNombre RegionId RegionNombre  
## <dbl> <chr> <dbl> <chr> <dbl> <chr>   
## 1 1 Mostazal~ 49 Mostazal 8 Región del ~  
## 2 2 Requínoa~ 41 Requínoa 8 Región del ~  
## 3 3 Requínoa~ 41 Requínoa 8 Región del ~  
## 4 4 Las Cabr~ 30 Las Cabras 8 Región del ~  
## 5 5 Requínoa~ 41 Requínoa 8 Región del ~  
## 6 6 Camino S~ 39 Rancagua 8 Región del ~  
## # ... with 10 more variables: RegionSigla <chr>,  
## # UnidadFiscalizableEstadoId <dbl>, UnidadFiscalizableEstado <chr>,  
## # Latitud <dbl>, Longitud <dbl>, SubCategoriaEconomicaPpal <lgl>,  
## # SubCategoriaEconomicaId <dbl>, SubCategoriaEconomicaNombre <chr>,  
## # CategoriaEconomicaId <dbl>, CategoriaEconomicaNombre <chr>

*Muchas de las funcionalidades que ocupes en R tendrán que ser cargadas desde librerías/paquetes. Estas se deben instalar una sola vez (install.packages("NombreLibreria")) y cargadas en cada nueva sesión de R (library(NombreLibreria))*

Ahora bien, en el caso de la **Superintendencia de Medio Ambiente** trabajamos directamente desde los motores de base de datos (en específico desde *SQL Server*). En ese caso, para cargar datos, primero se debe generar una conexión con la base de datos para luego “llamar” las tablas que querramos ir usando.

Una conexión a un motor de base de datos se ve más o menos así:

# Cargamos librerías para la conexión con motores de bases de datos  
library(DBI)  
library(odbc)  
  
con <- dbConnect(odbc(),  
 Driver = "SQL Server",  
 Server = "ServidorX",  
 Database = "BBDD",  
 Trusted\_Connection = "True",  
 Port = 1433)

Y dos formas de cargar estos datos en R son:

head(dbReadTable(con, "Resumen\_UnidadFiscalizable"))  
## UnidadFiscalizableId  
## 1 1  
## 2 2  
## 3 3  
## 4 4  
## 5 5  
## 6 6  
## Direccion  
## 1 Mostazal, Región del Libertador General Bernardo O'Higgins  
## 2 Requínoa, Región del Libertador General Bernardo O'Higgins  
## 3 Requínoa, Región del Libertador General Bernardo O'Higgins  
## 4 Las Cabras, Región del Libertador General Bernardo O'Higgins  
## 5 Requínoa, Región del Libertador General Bernardo O'Higgins  
## 6 Camino Santa Elena 0, Parcela 6 S/N, Rancagua, Región del Libertador General Bernardo O'Higgins  
## ComunaId ComunaNombre RegionId  
## 1 49 Mostazal 8  
## 2 41 Requínoa 8  
## 3 41 Requínoa 8  
## 4 30 Las Cabras 8  
## 5 41 Requínoa 8  
## 6 39 Rancagua 8  
## RegionNombre RegionSigla  
## 1 Región del Libertador General Bernardo O'Higgins VI  
## 2 Región del Libertador General Bernardo O'Higgins VI  
## 3 Región del Libertador General Bernardo O'Higgins VI  
## 4 Región del Libertador General Bernardo O'Higgins VI  
## 5 Región del Libertador General Bernardo O'Higgins VI  
## 6 Región del Libertador General Bernardo O'Higgins VI  
## UnidadFiscalizableEstadoId UnidadFiscalizableEstado Latitud Longitud  
## 1 1 Vigente -34.01789 -70.70131  
## 2 1 Vigente -34.25937 -70.80287  
## 3 1 Vigente -34.23775 -70.77586  
## 4 1 Vigente -34.16642 -71.45372  
## 5 1 Vigente NA NA  
## 6 1 Vigente -34.16400 -70.77373  
## SubCategoriaEconomicaPpal SubCategoriaEconomicaId  
## 1 1 94  
## 2 1 47  
## 3 1 126  
## 4 1 111  
## 5 1 94  
## 6 1 47  
## SubCategoriaEconomicaNombre CategoriaEconomicaId  
## 1 Estación de servicio 11  
## 2 Planta procesadora de productos agrícolas 6  
## 3 Instrumentos de Planificación Territorial 15  
## 4 Conjunto habitacional 12  
## 5 Estación de servicio 11  
## 6 Planta procesadora de productos agrícolas 6  
## CategoriaEconomicaNombre  
## 1 Equipamiento  
## 2 Agroindustrias  
## 3 Otras categorías  
## 4 Vivienda e Inmobiliarios  
## 5 Equipamiento  
## 6 Agroindustrias  
head(dplyr::tbl(con, "Resumen\_UnidadFiscalizable"))  
## # Source: lazy query [?? x 16]  
## # Database: sqlite 3.29.0 [:memory:]  
## UnidadFiscaliza~ Direccion ComunaId ComunaNombre RegionId RegionNombre  
## <dbl> <chr> <dbl> <chr> <dbl> <chr>   
## 1 1 Mostazal~ 49 Mostazal 8 Región del ~  
## 2 2 Requínoa~ 41 Requínoa 8 Región del ~  
## 3 3 Requínoa~ 41 Requínoa 8 Región del ~  
## 4 4 Las Cabr~ 30 Las Cabras 8 Región del ~  
## 5 5 Requínoa~ 41 Requínoa 8 Región del ~  
## 6 6 Camino S~ 39 Rancagua 8 Región del ~  
## # ... with 10 more variables: RegionSigla <chr>,  
## # UnidadFiscalizableEstadoId <dbl>, UnidadFiscalizableEstado <chr>,  
## # Latitud <dbl>, Longitud <dbl>, SubCategoriaEconomicaPpal <int>,  
## # SubCategoriaEconomicaId <dbl>, SubCategoriaEconomicaNombre <chr>,  
## # CategoriaEconomicaId <dbl>, CategoriaEconomicaNombre <chr>

***Para más información sobre el uso de bases de datos con R ver el*** [***siguiente link***](https://db.rstudio.com/)

# Manejo de datos

Ahora bien, cuando queramos realizar reportes tendrémos que probablemente manipular la información que tenemos. Esto dependerá de cada caso y habrán distintas formas de hacerlo. Por ejemplo, a continuación tomaremos los datos recién cargados y:

1. Seleccionaremos solo algunas de las variables/columnas para trabajar (función select())
2. Filtraremos (seleccionaremos filas) los datos para eliminar valores nulos respecto a dos variables (función filter())
3. Agruparemos las observaciones respecto a Región y Sector económico para luego calcular en número de UFs para cada grupo (funciónes group\_by() y summarise())
4. Los pasos anteriores generarán una “planilla” de varias filas y tres columnas que luego será modificada a un formato “más cuadrado” (función spread())

*Todos estos pasos son distintas operaciones que serán unidas a través de un operador llamado “la pipa” (%>%) que permite escribir código con acciones de izquierda a derecha.*

library(dplyr) # librería de manejo de datos (seleccionar, filtrar, agrupar, etc)  
library(tidyr) # librería para modificación de "forma" de datos ("pivot", "unpivot")  
  
# Generamos un nuevo data set "Resumen\_UnidadFiscalizable\_cuadro" luego de hacer toda la manipulación  
  
Resumen\_UnidadFiscalizable\_cuadro <-   
 # datos originales  
 Resumen\_UnidadFiscalizable %>%  
 # seleccionamos cuatro variables/columnas  
 select(UnidadFiscalizableId, RegionNombre, CategoriaEconomicaNombre) %>%   
 # dejamos solo aquellas observaciones/filas que no tienen valores nulos para RegionNombre ni CategoriaEconomicaNombre  
 filter(!is.na(RegionNombre) & !is.na(CategoriaEconomicaNombre)) %>%   
 # Agrupamos las observaciones/filas respecto a las variables RegionNombre y CategoriaEconomicaNombre  
 group\_by(RegionNombre, CategoriaEconomicaNombre) %>%   
 # Calculamos el número de filas/observaciones para cada uno de los grupos  
 summarise("Número de UFs" = n()) %>%   
 # Modificamos la forma de los datos (a "más cuadrado")  
 spread(CategoriaEconomicaNombre, `Número de UFs`)  
  
Resumen\_UnidadFiscalizable\_cuadro

## # A tibble: 16 x 17  
## # Groups: RegionNombre [16]  
## RegionNombre Agroindustrias Energía Equipamiento ETFAS Forestal  
## <chr> <int> <int> <int> <int> <int>  
## 1 Región de A~ NA 147 67 4 NA  
## 2 Región de A~ 4 24 58 NA NA  
## 3 Región de A~ 13 98 47 1 NA  
## 4 Región de A~ 15 8 176 NA 1  
## 5 Región de C~ 40 59 80 3 NA  
## 6 Región de l~ 39 51 226 1 19  
## 7 Región de l~ 72 74 237 6 5  
## 8 Región de L~ 31 32 105 1 13  
## 9 Región de M~ 10 17 60 NA 1  
## 10 Región de Ñ~ 40 26 99 NA 12  
## 11 Región de T~ 1 40 86 1 NA  
## 12 Región de V~ 77 60 101 8 3  
## 13 Región del ~ 32 118 149 9 38  
## 14 Región del ~ 197 43 155 1 2  
## 15 Región del ~ 206 69 160 NA 17  
## 16 Región Metr~ 180 79 676 35 3  
## # ... with 11 more variables: `Infraestructura de Transporte` <int>,  
## # `Infraestructura Hidráulica` <int>, `Infraestructura Portuaria` <int>,  
## # `Instalación fabril` <int>, Minería <int>, `Monitoreo de Calidad  
## # Ambiental` <int>, `Otras categorías` <int>, `Pesca y  
## # Acuicultura` <int>, `Saneamiento Ambiental` <int>, `Transportes y  
## # almacenajes` <int>, `Vivienda e Inmobiliarios` <int>

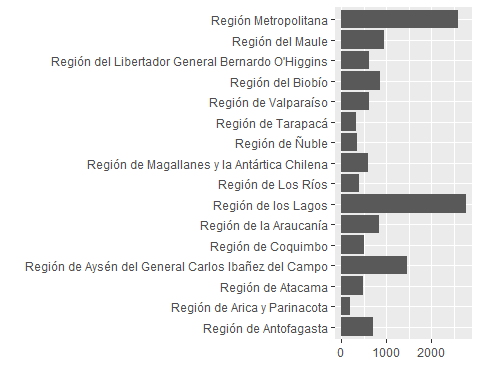
# Visualización

De forma similar podemos también

Resumen\_UnidadFiscalizable\_reg <-  
 # datos originales  
 Resumen\_UnidadFiscalizable %>%  
 # seleccionamos cuatro variables/columnas  
 select(UnidadFiscalizableId, RegionNombre, CategoriaEconomicaNombre) %>%   
 # dejamos solo aquellas observaciones/filas que no tienen valores nulos para RegionNombre ni CategoriaEconomicaNombre  
 filter(!is.na(RegionNombre) & !is.na(CategoriaEconomicaNombre)) %>%   
 # Agrupamos las observaciones/filas respecto a las variables RegionNombre y CategoriaEconomicaNombre  
 group\_by(RegionNombre, CategoriaEconomicaNombre) %>%   
 # Calculamos el número de filas/observaciones para cada uno de los grupos  
 summarise("Número de UFs" = n())  
  
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.6.1

Resumen\_UnidadFiscalizable\_reg %>%   
 ggplot(aes(RegionNombre, `Número de UFs`)) +   
 geom\_col() +   
 coord\_flip() +  
 xlab("") +  
 ylab("")



***En estos ejemplos usamos principalmente librerías del denominado tidyverse que es un conjunto de librerías especializadas en* data science*. Para más información pueden revisar el*** [***siguiente link***](https://www.tidyverse.org/)