

Introducción a R

Junio 2022

Antes de iniciar

Siéntanse libre de preguntar.

R y R studio

R es un entorno y lenguaje de programación con un enfoque al análisis estadístico. R nació como una reimplementación de software libre del lenguaje S, adicionado con soporte para ámbito estático.

El lenguaje está compuesto por símbolos y reglas sintácticas y semánticas, expresadas en forma de instrucciones y relaciones lógicas, mediante las cuales se construye el código fuente de una aplicación o pieza de software determinado.

RStudio es un entorno de desarrollo integrado para el lenguaje de programación R, dedicado a la computación estadística y gráficos.

Ambos especializado en análisis estadístico y visualización de datos.

En resumen



- Software libre.
- Lenguaje de programación.
- Interfaz de una consola



- Software libre.
- Es un programa para manejar R - IDE (Entorno de desarrollo integrado).
- Interfaz de varios paneles.

O más claro :)

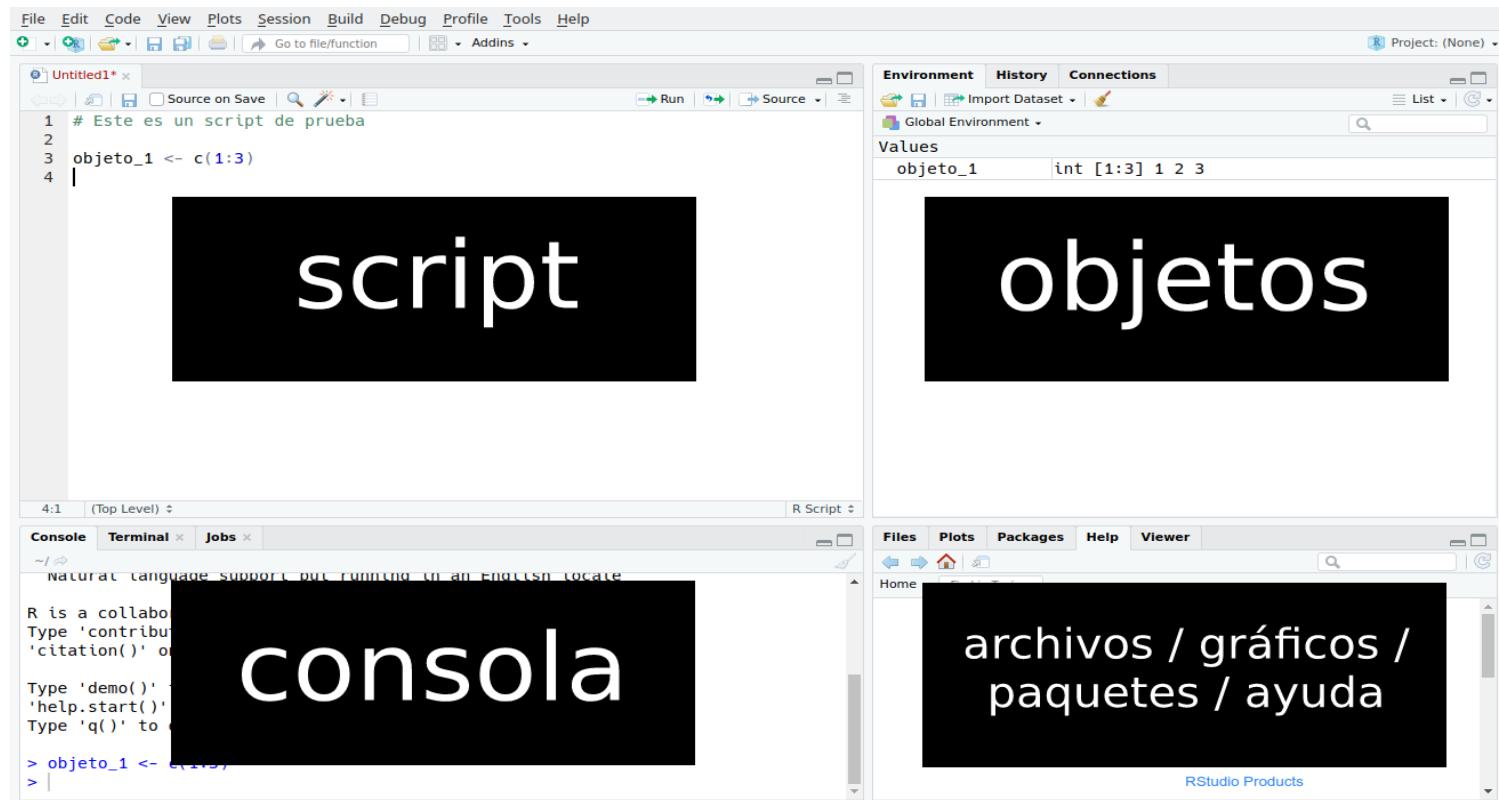


¿Por qué utilizar Rstudio?

- Es un lenguaje bastante adecuado para la estadística, ya que permite manipular los datos rápidamente y de forma precisa.
- Se puede automatizar fácilmente, gracias a la creación de scripts que automatizan procesos, por ejemplo, leer datos o hacer operaciones con los datos, y hacerlo siempre de forma automática.
- Puede leer prácticamente cualquier tipo de datos.
- Hasta cierto punto, es compatible con grandes conjuntos de datos.
- Es gratuito.
- Tiene capacidades avanzadas de gráficos, por lo que nos permite realizar gráficos y dashboards de forma que podamos presentar los resultados de forma vistosa.
- Se ejecuta en muchas plataformas.

Interfaz de Rstudio

RStudio está dividido en cuatro paneles:



Paneles

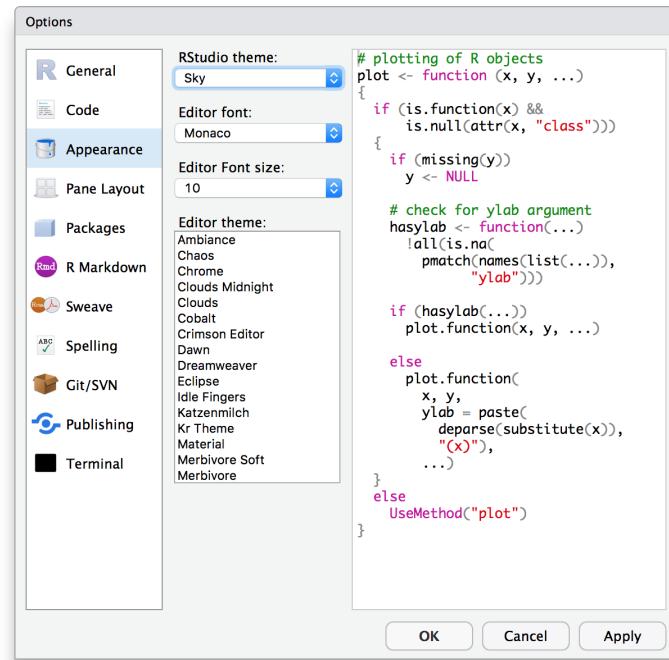
RStudio está dividido en cuatro paneles, que se presentarán a continuación. Vayamos a Rstudio y familiaríremos con estos componentes.

- Script: en este panel vas a escribir, editar, ver los R script y los datasets.
- Consola: también se conoce como terminal, aquí se ejecutan los comandos redactados en el script.
- Objeto / Environment: te muestra qué datasets y qué objetos (variables) que has creadas en la memoria.
- Archivo, gráficos, paqueterías y ayuda: es un panel multipropósito que devuelve información solicitada.

Recomendación:

Explora los subpaneles Files, Plots, Packages y Help observa que elementos por default Rstudio muestra.

Una vez conocido los paneles, puedes personalizar tu interfaz de R, selecciona el menú de "Tool", después clic en "Global options" y clic "Appearance".



Proyectos en Rstudio

Los **proyectos** hacen que sea más fácil dividir el trabajo en múltiples contextos, cada uno con su propio directorio de trabajo, espacio de trabajo, historial y los documentos de origen. Los proyectos de RStudio están asociados a los directorios de trabajo. Al trabajar en Rstudio se recomienda tener un orden de los elementos que crearás, que importarás y exportarás por eso la importancia de contar con un proyecto.

Crear un proyecto nuevo

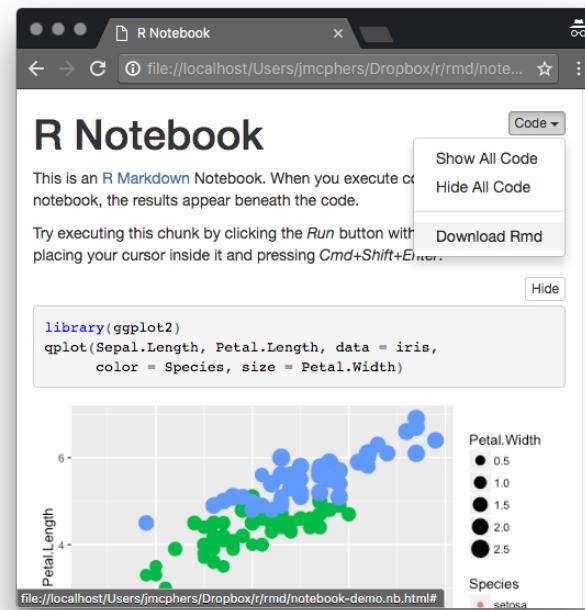
- Hacer clic en el menú **Archivo**, luego en **Nuevo proyecto**.
- Hacer clic en **Nuevo directorio** .
- Hacer clic en **Nuevo proyecto**.
- Introducir el nombre del directorio para guardar tu proyecto, por ejemplo: **feminismo_datos**.

R Notebook

Un R Notebook es un documento R Markdown con fragmentos que se pueden ejecutar de forma independiente e interactiva, con la salida visible inmediatamente debajo de la entrada.

Con este archivo trabajaremos, para abrirlo debemos dar:

- Clic en el menú principal **File**.
- Despues clic en **New file**.
- Por ultimo clic en **R Notebook**.



Directorio de trabajo

El directorio o carpeta de trabajo es el lugar en nuestra computadora en el que se encuentran los archivos con los que estamos trabajando en R. Este es el lugar donde R buscara archivos para importarlos y al que serán exportados, a menos que indiquemos otra cosa.

Puedes encontrar cuál es tu directorio de trabajo con la función `getwd()`. Sólo tienes que escribir la función en la consola y ejecutarla.

Para ejecutar parte del código: presiona **Ctrl + Enter**

```
#Muestra la ruta o path de mi directorio  
getwd()
```

```
## [1] "C:/Users/Paola Viridiana/Downloads"
```

Tambien, puedes cambiar el directorio de trabajo usando la función `setwd()`, dando como argumento la ruta del directorio que quieras usar.

```
setwd("~/")
```

Antes de iniciar

Debes conocer algunas herramientas en R.

Comentarios

En R puedes añadir comentarios en tu script, lo que esta junto al **#** es comenario.

```
# Este es un comentario y no interfiere en los comandos de R.  
# Son anotaciones para nosotros.
```

¿Cómo ejecutar comando o líneas?

presiona **Ctrl + Enter** | Seleccionar y dar clic en la viñeta de **Run** el cual se ejecutará en el panel de la consola.

Asignación de variables

Una variable puede almacenar un objeto.

```
# Mi variable x tiene un objeto de valor de 1 | tres manera de asignar nombre  
x <- 1  
x = 1  
1-> x
```

Tipo y estructura de datos

Tipo numérico:

```
#Números decimales  
x<-1.0  
1.0 + 2.0
```

```
## [1] 3
```

Tipo enteros o integer:

```
#Números enteros  
int <- 1
```

Tipo character:

```
# Cadenas de texto  
texto<- "Hello, world!"  
print(texto)
```

```
## [1] "Hello, world!"
```

Tipo y estructura de datos

Tipo factor:

```
#Una variable factor es una variable categórica. Los vectores de caracteres a menudo  
#se almacenan como factores para explotar funciones para tratar datos categóricos  
"Femenino, Masculino, Otro"
```

```
## [1] "Femenino, Masculino, Otro"
```

Tipo logical:

```
# Verdadero (TRUE) o falso (FALSE). Es a menudo el resultado de operaciones lógicas  
a <- 1  
b <- 2  
a < b
```

```
## [1] TRUE
```

Tipo de datos

NA y NULL

En R, usamos **NA** para representar datos perdidos, mientras que **NULL** representa la ausencia de datos.

La diferencia entre las dos es que un dato **NULL** aparece sólo cuando R intenta recuperar un dato y no encuentra nada, mientras que **NA** es usado para representar explícitamente datos perdidos, omitidos o que por alguna razón son faltantes.

Por ejemplo, si tratamos de recuperar la edad de una persona encuestada que no existe, obtendríamos un **NULL**, pues no hay ningún dato que corresponda con ello. En cambio, si tratamos de recuperar su estado civil, y la persona encuestada no contestó esta pregunta, obtendríamos un **NA**.

NA además puede aparecer como resultado de una operación realizada, pero no tuvo éxito en su ejecución.

Validación de datos

Se puede verificar si un dato es de un tipo específico con la familia de funciones `is()`

Función	Tipo que verifican
<code>is.integer()</code>	Entero
<code>is.numeric()</code>	Numerico
<code>is.character()</code>	Cadena de texto
<code>is.factor()</code>	Factor
<code>is.logical()</code>	Lógico
<code>is.na()</code>	NA
<code>is.null()</code>	NULL

Además, con la función `class` podemos saber que tipo de dato es una variable.

Coerción de datos

También podemos hacer coerciones explícitas usando la familia de funciones `as()`.

Función	Tipo al que hace coerción
<code>as.integer()</code>	Entero
<code>as.numeric()</code>	Numerico
<code>as.character()</code>	Cadena de texto
<code>as.factor()</code>	Factor
<code>as.logical()</code>	Lógico
<code>as.na()</code>	NA
<code>as.null()</code>	NULL

Vectores

- Elemento más básico en R.
- Se crea con la función `c()`, que significa ‘concatenar’ o ‘combinar’.
- Un vector puede almacenar varios objetos diferentes en orden en r un vector es un espacio de memoria, estas últimas dos representan estructuras rectangulares de datos.

Creamos vectores con información de algunos estados de México.

Estados con mayor población en Mx:

```
abr_may <- c("EdoMex", "CDMX", "Ver", "Jal") #Vector numeric  
entidad_may <- c("Estado México", "Ciudad de México", "Veracruz", "Jalisco") #Vect  
pob_mil_may <-c(17363387, 8811266, 8163963, 8110943)
```

Estados con menor población en Mx

```
abr_men <- c("Nay", "Camp", "BCS", "Col")  
entidad_men <- c("Nayarit", "Campeche", "Baja California S", "Colima")  
pob_mil_men <-c(1268460, 935047, 809833, 747801)
```

Funciones de vectores

```
length(entidad_may) #Longitud del vector  
## [1] 4  
  
entidad_may[2:3] #Extraer información del vector  
## [1] "Ciudad de México" "Veracruz"  
  
entidad_may[c(2:3)] #Extraer información del vector | Otra forma  
## [1] "Ciudad de México" "Veracruz"  
  
pob_mil_may - pob_mil_men #Transformar vectores  
## [1] 16094927  7876219  7354130  7363142  
  
is.vector(entidad_may) #Validar vector  
## [1] TRUE
```

Matrices

Una matriz es una forma de acomodar los datos que tiene renglones o filas y columnas. Continuando con nuestro ejemplo.

```
matrix_ent_may <- matrix(c(abr_may, entidad_may, pob_mil_may),  
                           nrow = 4,  
                           ncol = 3)  
matrix_ent_may
```

```
##      [,1]     [,2]      [,3]  
## [1,] "EdoMex"  "Estado México"  "17363387"  
## [2,] "CDMX"    "Ciudad de México" "8811266"  
## [3,] "Ver"      "Veracruz"      "8163963"  
## [4,] "Jal"      "Jalisco"       "8110943"
```

```
colnames(matrix_ent_may)<-c("Abreviatura", "Entidad", "Población")  
rownames(matrix_ent_may)<-c(1, 2, 3, 4)  
matrix_ent_may
```

```
##   Abreviatura Entidad      Población  
## 1 "EdoMex"    "Estado México"  "17363387"  
## 2 "CDMX"      "Ciudad de México" "8811266"  
## 3 "Ver"        "Veracruz"      "8163963"  
## 4 "Jal"        "Jalisco"       "8110943"
```

Selección de elementos de una matriz

```
matrix_ent_may[4, ] #Selección de una fila (pob)
```

```
## Abreviatura      Entidad    Población  
##      "Jal"      "Jalisco"  "8110943"
```

```
matrix_ent_may[, 3] #Selección de una fila (pob)
```

```
##           1           2           3           4  
## "17363387"  "8811266"  "8163963"  "8110943"
```

```
matrix_ent_may [4,2] # Seleccionar un elemento en específico
```

```
## [1] "Jalisco"
```

Hemos visto:

- **Variables:** espacio para guardar un objeto.
- **Vectores:** una o más variable del mismo tipo de datos.
- **Matrices:** varias columnas/vectores del mismo tipo de datos.
¿qué sigue?
- **Dataframes:** tabla o columna de diferente tipo de datos.

Dataframes:

En un dataframe podemos tener una columna con caracteres otra con números y otra con variables lógicas.

Añadimos columna TRUE - FALSE

```
ubicacion_may <- c(TRUE, T, F, F)
```

```
df_ent_may <- data.frame(abr_may,
                           entidad_may,
                           pob_mil_may,
                           ubicacion_may)
df_ent_may
```

```
##   abr_may      entidad_may  pob_mil_may  ubicacion_may
## 1 EdoMex     Estado México    17363387      TRUE
## 2 CDMX  Ciudad de México     8811266      TRUE
## 3 Ver        Veracruz       8163963     FALSE
## 4 Jal        Jalisco        8110943     FALSE
```

Comenzamos a utilizar propiedades de los dataframes

```
names(df_ent_may) #Nombre de las columnas

## [1] "abr_may"      "entidad_may"    "pob_mil_may"   "ubicacion_may"

head(df_ent_may, 2) #Muestra el encabezado del df la ",," después del objeto es el

##   abr_may      entidad_may pob_mil_may ubicacion_may
## 1 EdoMex     Estado México    17363387        TRUE
## 2 CDMX Ciudad de México     8811266        TRUE

#Muestra la parte final del df la ",," después
#del objeto es el número de obsevaciones que deseas ver.

tail(df_ent_may, 1)

##   abr_may entidad_may pob_mil_may ubicacion_may
## 4     Jal     Jalisco     8110943        FALSE
```

```
dim(df_ent_may) #Muestra la dimension del df
```

```
## [1] 4 4
```

```
class(df_ent_may)
```

```
## [1] "data.frame"
```

La función `summary()` provee salidas para cada variable dependiendo del tipo de datos. Cuando los valores son numéricos, como en nuestro caso, `summary()` muestra el minimo, 1er cuartil, mediana, media mean, entre otroe. Para variables categóricas, muestra el número de veces que cada valor aparece en los datos (esto es llamado “level”).

```
summary(df_ent_may) #Muestra la clase del df
```

```
##     abr_may      entidad_may      pob_mil_may      ubicacion_may
##  Length:4      Length:4      Min.   : 8110943      Mode  :logical
##  Class :character  Class :character  1st Qu.: 8150708      FALSE:2
##  Mode   :character  Mode   :character  Median : 8487614      TRUE :2
##                                         Mean   :10612390
##                                         3rd Qu.:10949296
##                                         Max.   :17363387
```

En resumen

Qué hemos visto:

- R y R studio .
 - Tipos y estructura de datos.
 - Extracción de información.

Hasta ahorita nos hemos introducido en dataframes, continuaremos con ellos y más de sus propiedades en complemento con otras paqueterías.



Todo con funciones y sintaxis de R base (funciones por default de Rstudio sin necesidad de instalar paqueterías externas)

Tidyverse

Tidyverse es un conjunto de paquetes en R diseñados para ciencia de datos. Ayuda en todo el proceso de importar transformar visualizar modela y comunicar toda la información que normalmente utilizamos en procesos de ciencia de datos.

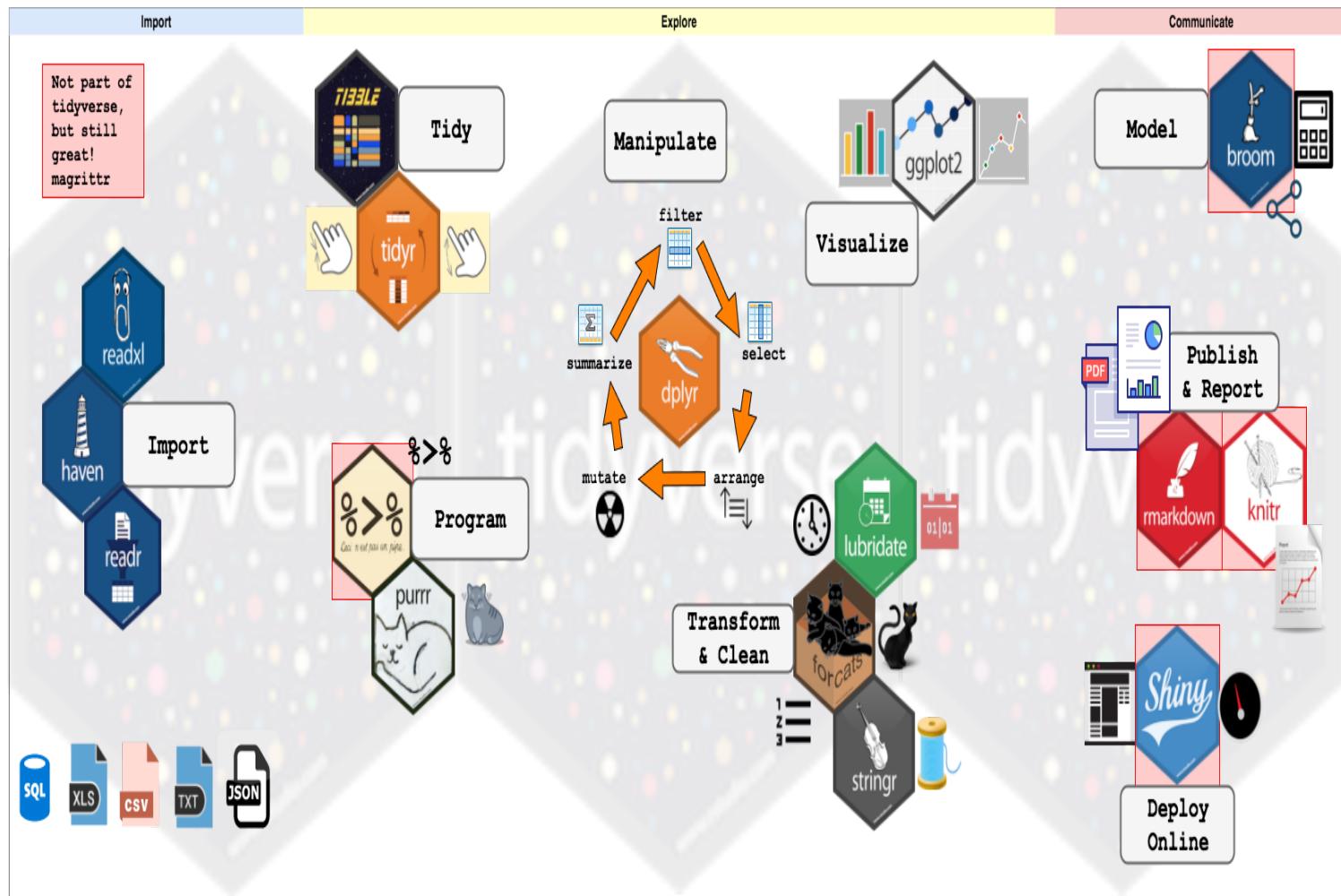
Ventajas

- Facilita el análisis y manipulación de datos y más rápido.
- Comparten estructura y nombre comunes.
- Sixtaxis, nombre y característica similares compatible entre paquetes.

Desventaja

- Deja de lado la forma usual de programación.
- Uso de pipes %>%.

Tidyverse



Código con R base y tidyverse

Instalamos y cargamos la paquetería tidyverse

```
#Se instala paqueteria una sola vez  
#install.packages("tidyverse")  
  
#Se carga la paqueteria cada vez que deseamos trabajar con ella  
library(tidyverse)
```

```
## — Attaching packages ————— tidyverse 1.3.1 —  
  
## ✓ ggplot2 3.3.6      ✓ purrr    0.3.4  
## ✓ tibble   3.1.7      ✓ dplyr    1.0.9  
## ✓ tidyr    1.2.0      ✓ stringr  1.4.0  
## ✓ readr    2.1.2      ✓forcats  0.5.1  
  
## — Conflicts ————— tidyverse_conflicts()  
## ✗ dplyr::filter() masks stats::filter()  
## ✗ dplyr::lag()    masks stats::lag()
```

Ojo: identificar mensajes de error (color rojo) y avisos de R

Cargar bases de datos formato csv

```
victimas<-read.csv("https://docs.google.com/spreadsheets/d/e/2PACX-1vQr_U2qTUJHQGQ
```

Base de dato Secretariado Ejecutivo del Sistema Nacional de Seguridad Pública La incidencia delictiva se refiere a la presunta ocurrencia de delitos registrados en averiguaciones previas iniciadas o carpetas de investigación, reportadas por las Procuradurías de Justicia y Fiscalías Generales de las entidades federativas en el caso del fuero común y por la Fiscalía General de la República en el fuero federal. Los datos presenta informacion de [Cifras de Víctimas del Fuero Común, 2015 - abril 2022](#)

```
#Vemos las columnas que integra la base  
colnames(victimas)
```

```
## [1] "Año"                      "Clave_Ent"                 "Entidad"  
## [4] "Bien.jurídico.afectado"  "Tipo.de.delito"           "Subtipo.de.delito"  
## [7] "Modalidad"                "Sexo"                     "Rango.de.edad"  
## [10] "Enero"                    "Febrero"                  "Marzo"  
## [13] "Abril"                    "Mayo"                     "Junio"  
## [16] "Julio"                    "Agosto"                   "Septiembre"  
## [19] "Octubre"                 "Noviembre"               "Diciembre"
```

```
#Leemos los delitos que integra la base de datos  
unique(victimas$Tipo.de.delito)
```

```
## [1] "Homicidio"  
## [2] "Lesiones"  
## [3] "Feminicidio"  
## [4] "Otros delitos que atentan contra la vida y la integridad corporal"  
## [5] "Secuestro"  
## [6] "Tráfico de menores"  
## [7] "Rapto"  
## [8] "Otros delitos que atentan contra la libertad personal"  
## [9] "Extorsión"  
## [10] "Corrupción de menores"  
## [11] "Trata de personas"  
## [12] "Otros delitos contra la sociedad"  
## [13] "Aborto"
```

```
#Concatenamos dos delitos de interés  
#Sólo trabajaremos con Feminicidios y Homicidios dolosos -filtramos delitos  
delitos<- c("Feminicidio", "Homicidio doloso")
```

Transformación y manipulación de datos con pipes (%>%)

```
library(dplyr)

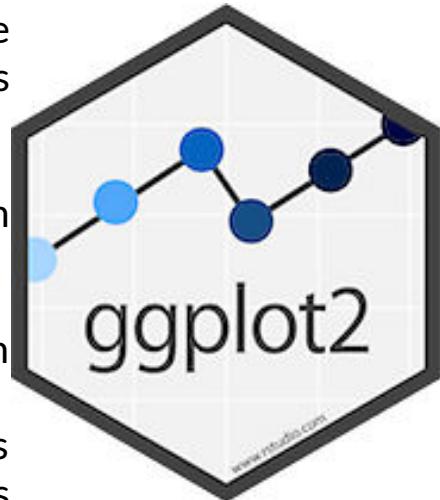
#Hacemos sumatoria de los meses por los delitos que corresponden a mujeres
victimas %>%
  filter(Sexo=="Mujer",
        Subtipo.de.delito %in% delitos) %>%
  group_by(Año, Subtipo.de.delito) %>%
  summarise(ene=sum(Enero, na.rm = T),
            feb=sum(Febrero, na.rm = T),
            mar=sum(Marzo, na.rm = T),
            abr=sum(Abril, na.rm = T),
            may=sum(Mayo, na.rm = T),
            jun=sum(Junio, na.rm = T),
            jul=sum(Julio, na.rm = T),
            ago=sum(Agosto, na.rm = T),
            sep=sum(Septiembre, na.rm = T),
            oct=sum(Octubre, na.rm = T),
            nov=sum(Noviembre, na.rm = T),
            dic=sum(Diciembre, na.rm = T),
            Total=sum(ene+feb+mar+abr+
                      may+jun+jul+ago+
                      sep+oct+nov+dic)) %>%
  select(Año, Subtipo.de.delito, Total)->victimas_anual
```

Paquetería ggplot2

La librería ggplot2 de R es un sistema organizado de visualización de datos. Forma parte del conjunto de librerías llamado **tidyverse**.

Los elementos necesarios para representar un gráfico con ggplot2 son los siguientes:

- Un **data frame** que contiene los datos que se quieren visualizar.
- Los **aesthetics**, es decir, una lista de relaciones entre las variables del fichero de datos y determinados aspectos del gráfico (como por ejemplo coordenadas, formas o colores).
- Los **geoms**, que especifican los elementos geométricos (puntos, líneas, círculos, etc) que se van a representar.
- Esta basado en [The grammar of graphics](#) de Leland Wilkinson.



Este enlace te llevará a conocer más gráficos en R.

class: left, bottom background-image:
url(https://miro.medium.com/max/700/1*MMZuYgeC_YjXNC1r4D4sog.png)
background-position: background-size:

Gramática de gráficos

7 capas

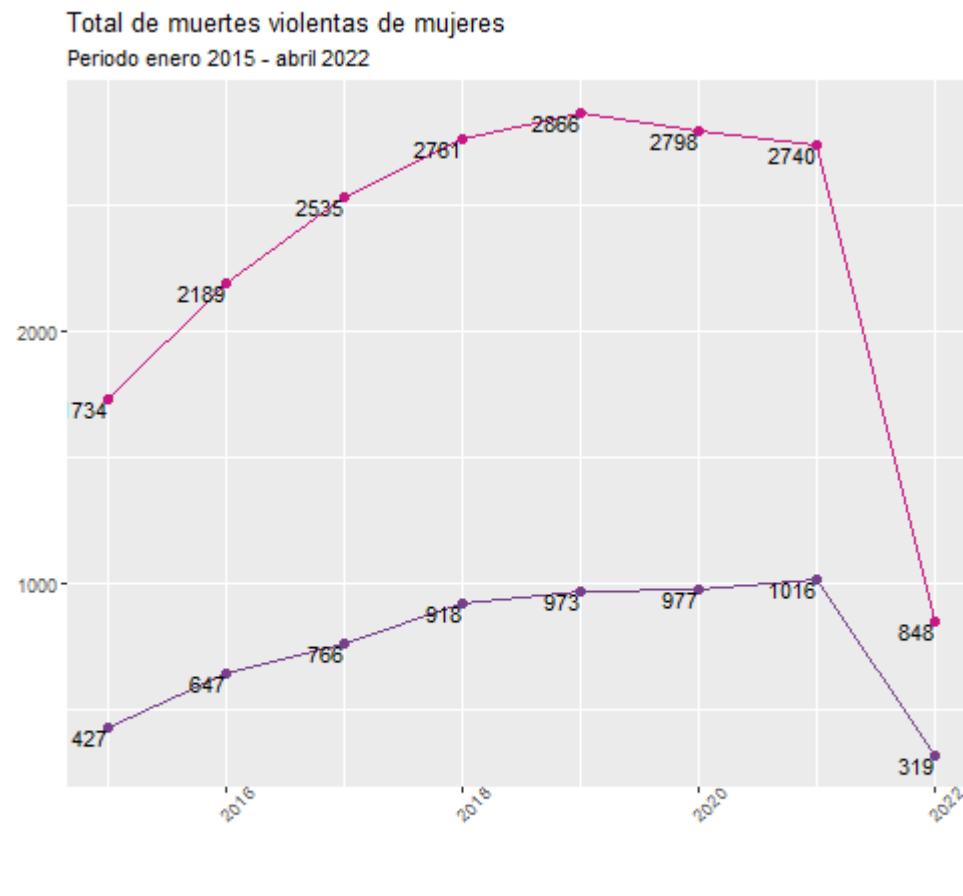
1. **Datos:** Todo comienza aquí, con las variables trazadas que desea visualizar.
2. **Estética:** una capa de escalas (por ejemplo, ejes xey) en la que se asignan los datos.
3. **Geometría:** formas utilizadas para representar sus datos, por ejemplo, barras, líneas, puntos, etc.
4. **Facetas:** cuando hay demasiados puntos de datos en un gráfico, puede dividir las variables para obtener una vista más clara.
5. **Estadísticas:** datos agregados (resúmenes) o tendencias utilizando modelos estadísticos.
6. **Coordenadas:** describe el espacio de trazado que está utilizando.
7. **Temas:** describe la tinta sin datos. La capa de tema le permite diseñar con una identidad visual particular utilizando fuentes, colores y otros elementos de diseño.

La idea central de esta es que se puedan generar gráficos a partir de una sintaxis determinada.

Generación de gráficos

```
#Visualización anual de líneas y puntos de muertes violentas
victimas_anual %>% ggplot() +
  geom_line(aes(x=Año, y=Total, group=Subtipo.de.delito, color=Subtipo.de.delito))
  geom_point(size=2, aes(x=Año, y=Total, group=Subtipo.de.delito, color=Subtipo.de
scale_color_manual(values = c("#78408B", "#C91682", "#ca97db"))+
  labs(title = "Total de muertes violentas de mujeres",
       subtitle = "Periodo enero 2015 - abril 2022",
       x= "",
       y="",
       caption = "Elaborado a partir de los datos abiertos del Secretariado Ejecut
@Paomrom")+
  theme(axis.text.x = element_text(angle = 45), legend.position = "bottom") +
  geom_text(aes(x=Año, y=Total, label=Total, hjust=1, vjust=1), colour="black")-> .
```

gráfico



Elaborado a partir de los datos abiertos del Secretariado Ejecutivo del Sistema Nacional de Seguridad Pública
@Paomrom