



Red Mexicana de
Bioinformática



Visualización de datos usando gramática de **ggplot2**

E. Ernestina Godoy Lozano

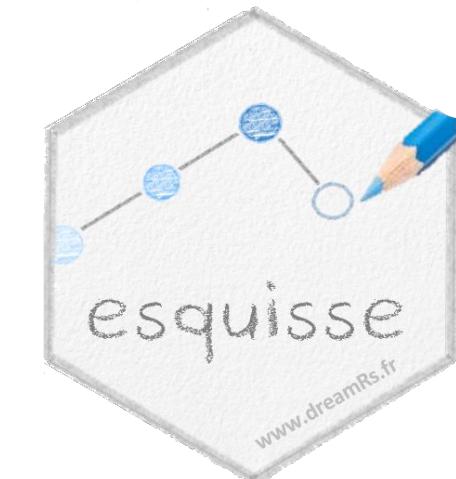
Departamento de Bioinformática en Enfermedades Infecciosas. CISEI-INSP

Carmina Barberena Jonas

Laboratorio de Genética Evolutiva y Poblaciones Humanas. Laboratorio Nacional
para la Diversidad Genómica

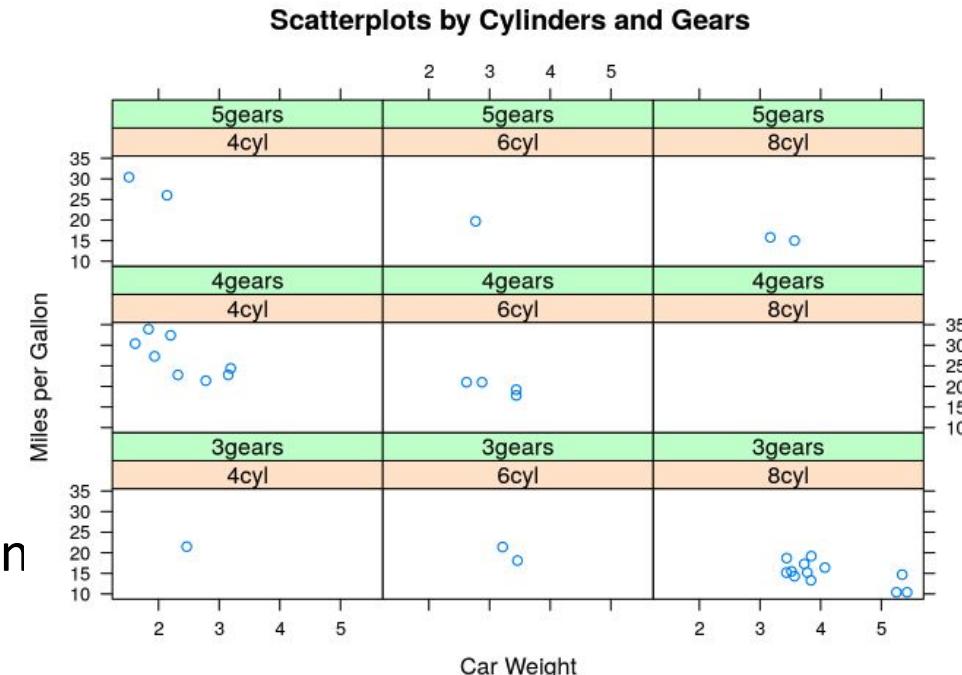
Orden de la plática

1. Introducción a [ggplot2](#)
2. Elementos de un gráfico en [ggplot2](#)
3. Sintaxis de [ggplot2](#)
4. Ejemplo de gráficos
 - Gráfica de barras
 - Scatterplots
5. Gráficos interactivos con [esquisse](#)
 - Histogramas
 - Boxplots
 - Densidad
6. Dudas



Introducción: Gráficos en *R*

- En la mayor parte de los lenguajes de programación, la capacidad de crear gráficos la proporcionan librerías adicionales, **ajenas a su núcleo**.
- En ***R***, los gráficos son **nativos**.
- En ***R*** existen dos motores gráficos:
 - Funciones de alto nivel (***plot***, ***hist***, ***barplot***, ***boxplot***) que a su vez invocan funciones de bajo nivel que nos permiten hacer modificaciones en los gráficos.
 - ***grid***. Creado por Paul Murrell para facilitar la generación de gráficos tipo Trellis, de celosía o de pequeños múltiples.
 - *lattice*, *ggplot2*



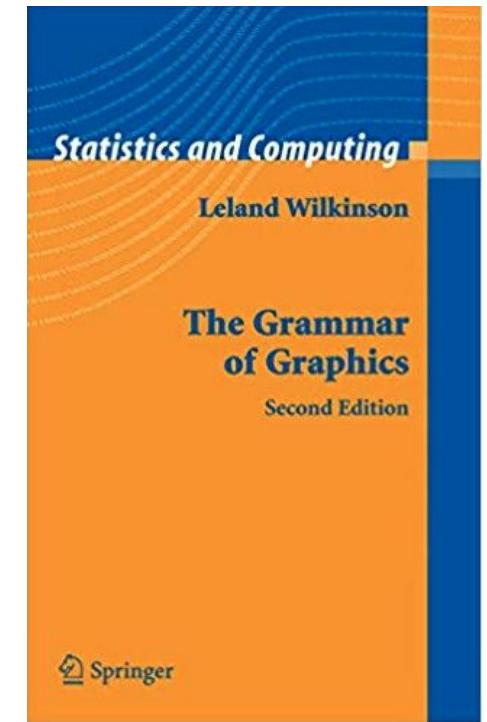


ggplot2

- Es una librería de R implementada por [Hadley Wickham](#) para [visualización de datos](#)
- Forma parte de un conjunto de librerías llamado [tidyverse](#)
- ggplot2 esta basado en “[The Grammar of Graphics](#)” de Leland Wilkinson (2000).
- Tiene más de 10 años y es usado por miles de personas en todo el mundo.
 - [Ventaja](#): Hay muchas comunidades que te ayudan a resolver dudas

The Grammar of Graphics

- Pone una serie de ideas novedosas de como se debe generar un gráfico.
- La idea central es:
 - Todos los gráficos pueden generarse mediante un lenguaje regular, con una sintaxis determinada.
 - Es posible construir una serie de **reglas comunes, conocidas y regulares** para crear representaciones visuales de datos de interés estadístico.
 - Es un marco que sigue un enfoque en capas para describir y construir visualizaciones o gráficos de **manera estructurada**.

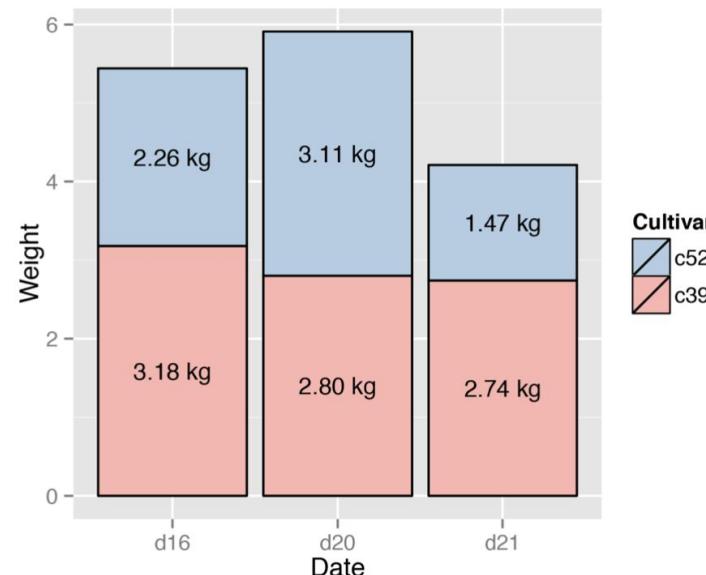


Instalación de ggplot2

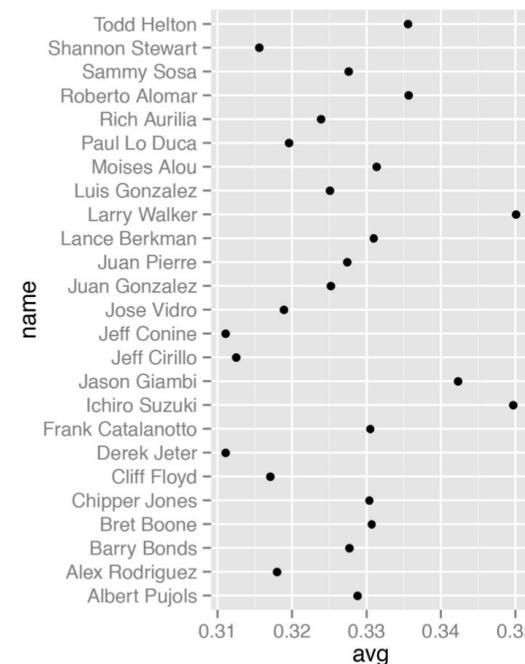
- Instalando la librería de *tidyverse* completa
 - `install.packages("tidyverse")`
- Instalandola desde su *repositorio de preferencia*
 - `install.packages("ggplot2", dependencies=TRUE)`
- O instalar la versión en desarrollo desde GitHub
 - `install.packages("devtools")`
 - `devtools:::install_github("tidyverse/ggplot2")`



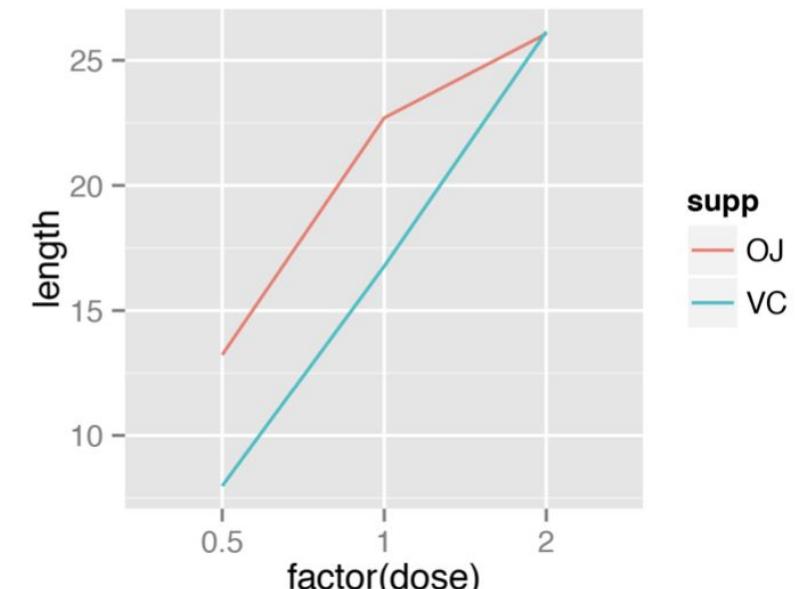
Repertorio de gráficos



Bar plots

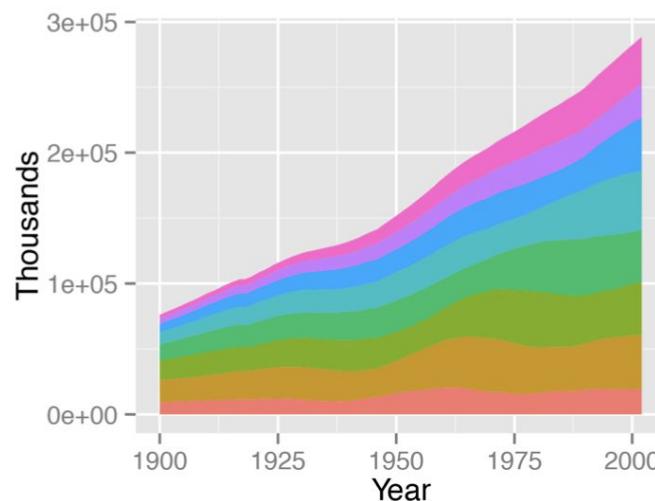


Dot plots

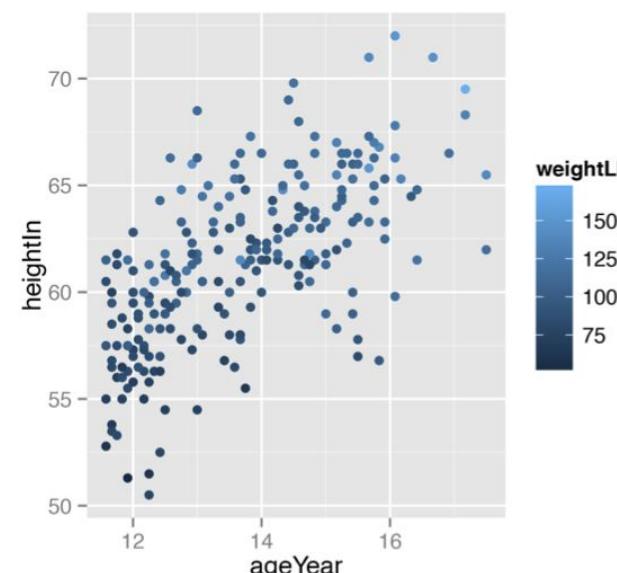


Line plots

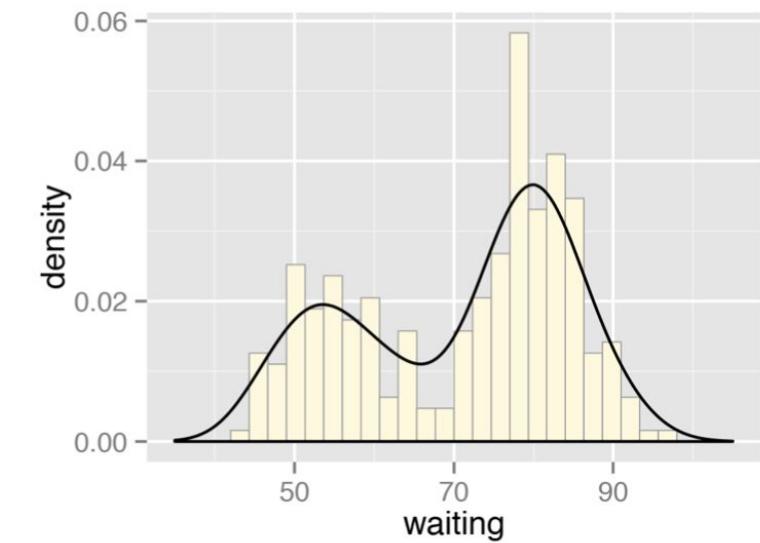
Repertorio de gráficos



Area graphs

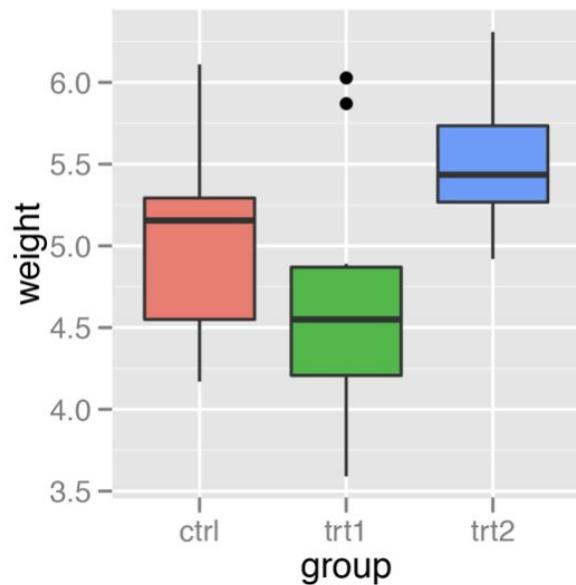


Scatter plots

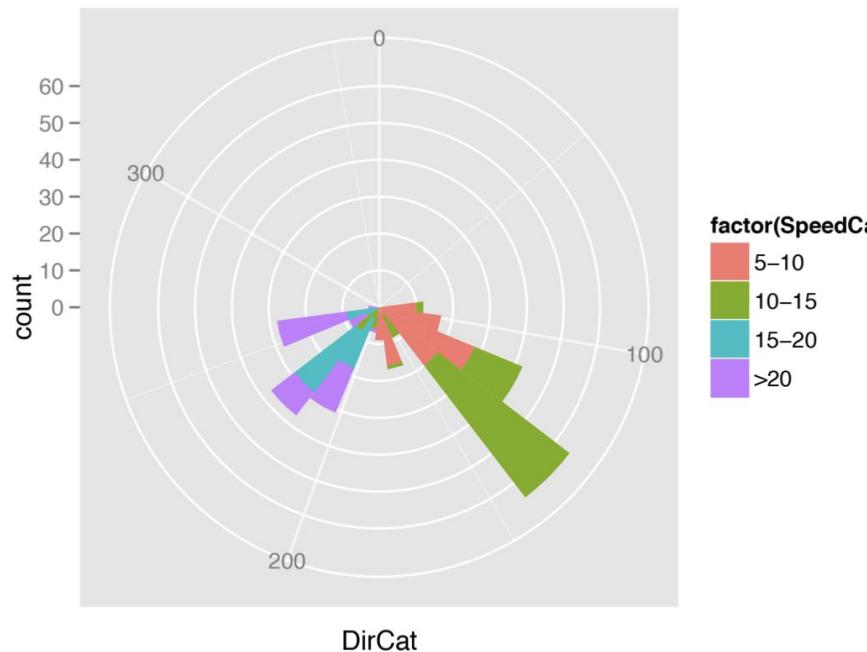


Histograms

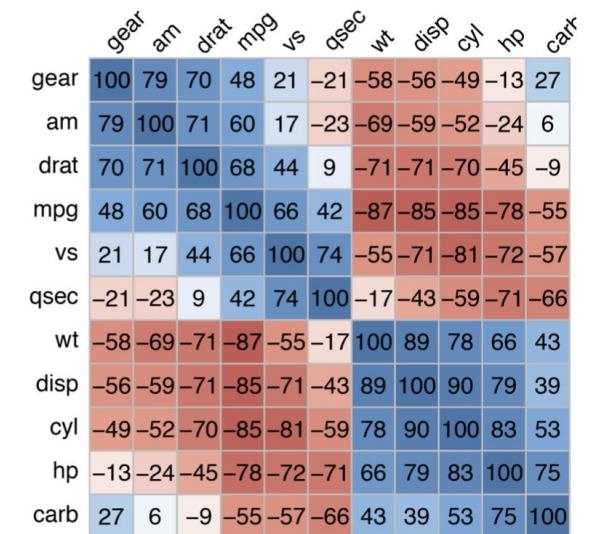
Repertorio de gráficos



Box plots

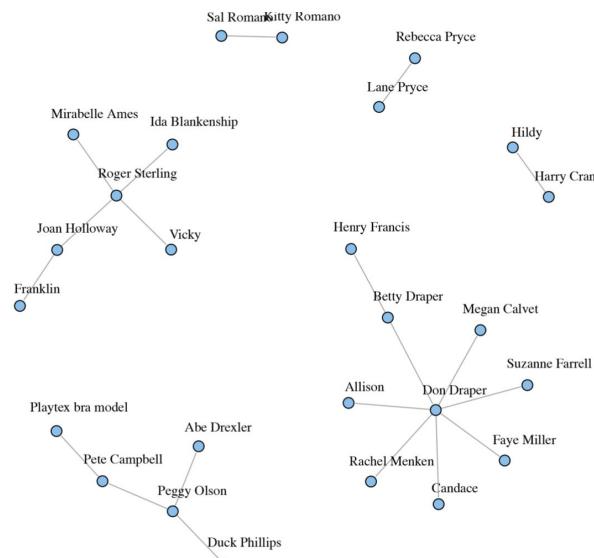


Polar plots

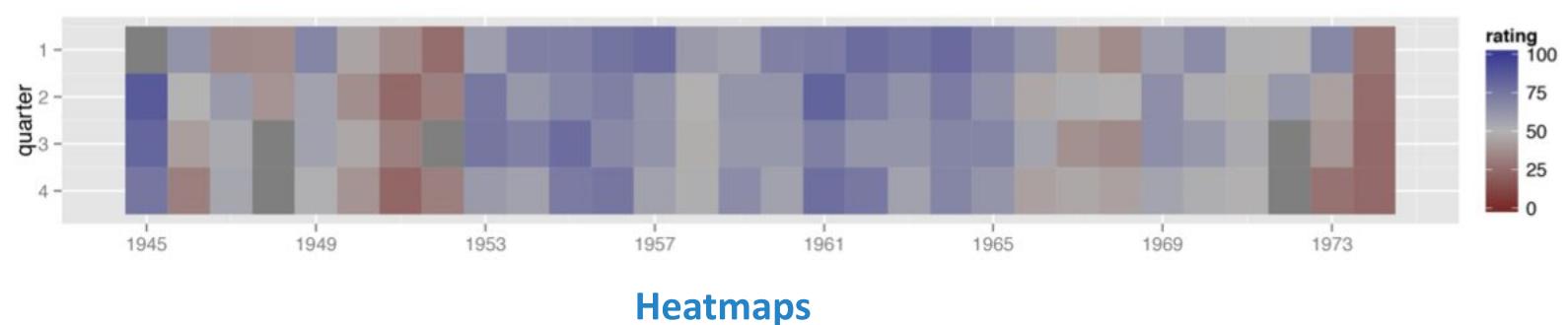


corplots

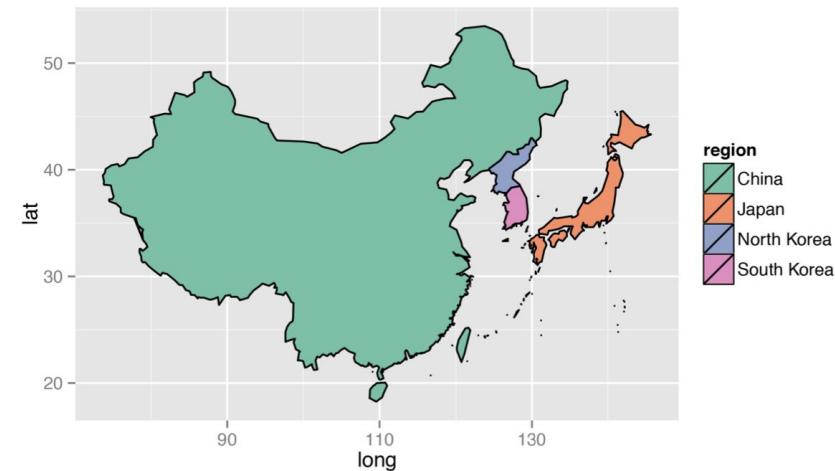
Repertorio de gráficos



Network plots



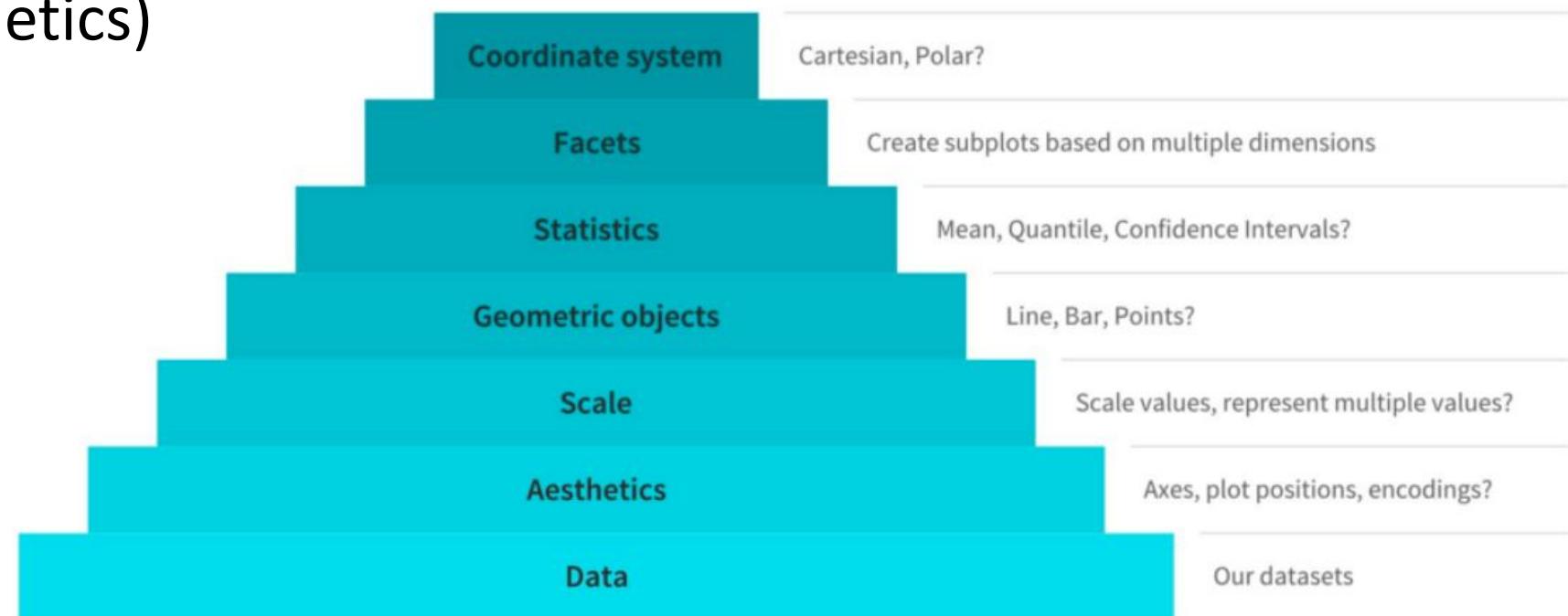
Heatmaps



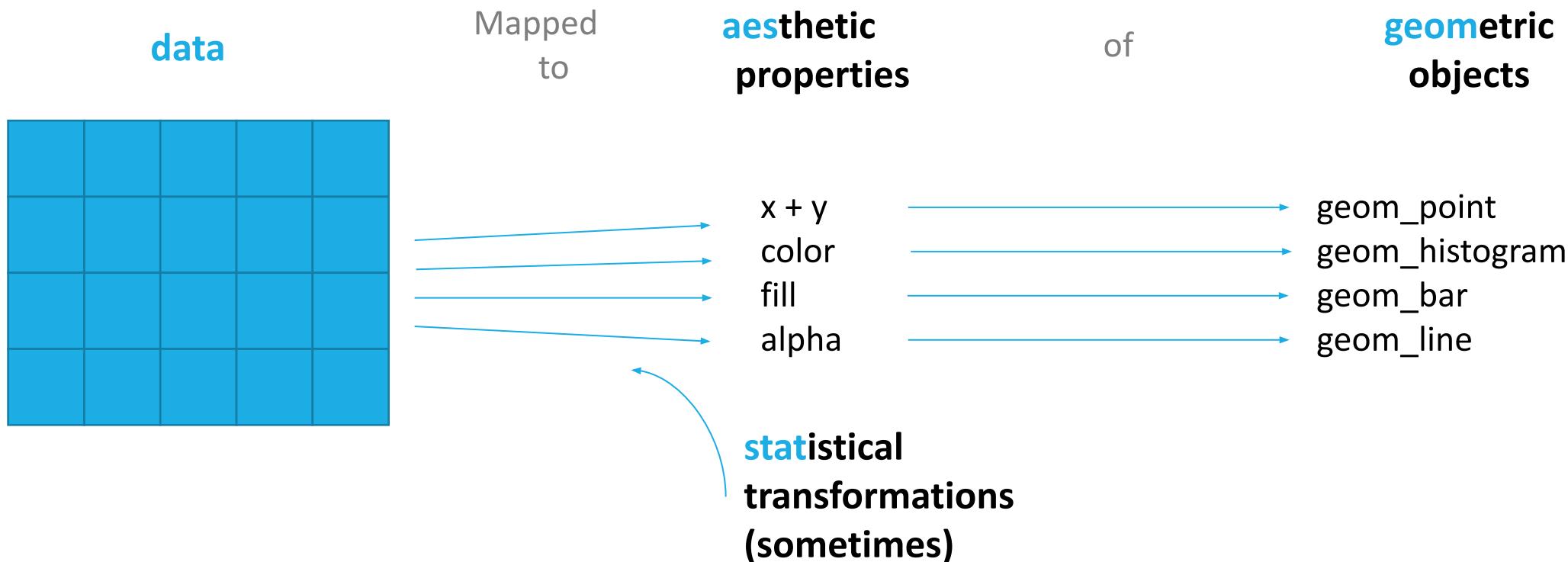
Maps

Elementos de un gráfico en ggplot

- Datos
- Estéticas (`aes`sthetics)
- Capas
- Facetas
- Temas



The grammar of the graphics and ggplot2

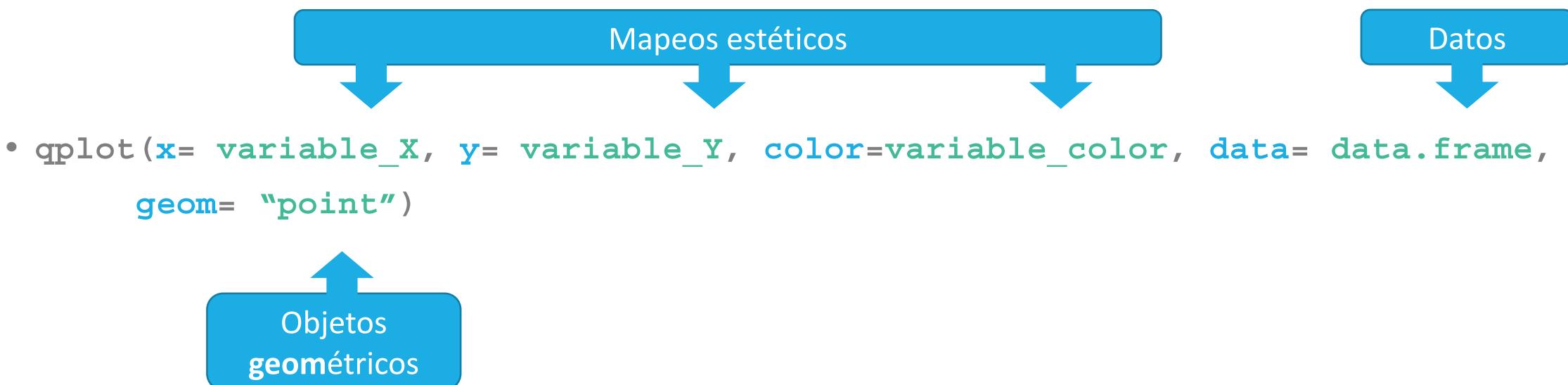


Funciones principales en ggplot2

- Hay dos funciones principales el ggplot2:
- Una función que nos permite echarle un vistazo rápido a los datos
 - `qplot()`
- Una función más compleja que nos va a permitir explorar más a fondo los datos
 - `ggplot()`

Sintaxis qplot()

- Crea un gráfico completo con los datos, geom y mapeos. Proporciona muchos valores por defecto.



qplot(

```
x,  
y,  
...,  
data,  
facets = NULL,  
margins = FALSE,  
geom = "auto",  
xlim = c(NA, NA),  
ylim = c(NA, NA),  
log = "",  
main = NULL,  
xlab = NULL,  
ylab = NULL,  
asp = NA,  
stat = NULL,  
position = NULL  
)
```

Arguments

x, y, ... Aesthetics passed into each layer

data Data frame to use (optional). If not specified, will create one, extracting vectors from the current environment.

facets faceting formula to use. Picks `facet_wrap()` or `facet_grid()` depending on whether the formula is one- or two-sided

margins See `facet_grid`: display marginal facets?

geom Character vector specifying geom(s) to draw. Defaults to "point" if x and y are specified, and "histogram" if only x is specified.

xlim, ylim X and y axis limits

log Which variables to log transform ("x", "y", or "xy")

main, xlab, ylab Character vector (or expression) giving plot title, x axis label, and y axis label respectively.

asp The y/x aspect ratio

stat, position DEPRECATED.

Datos: mtcars

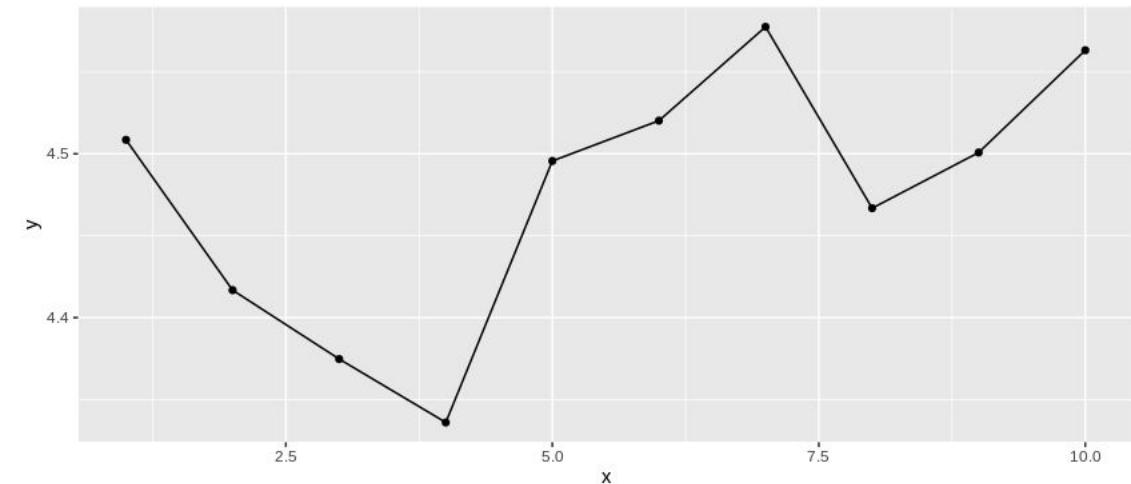
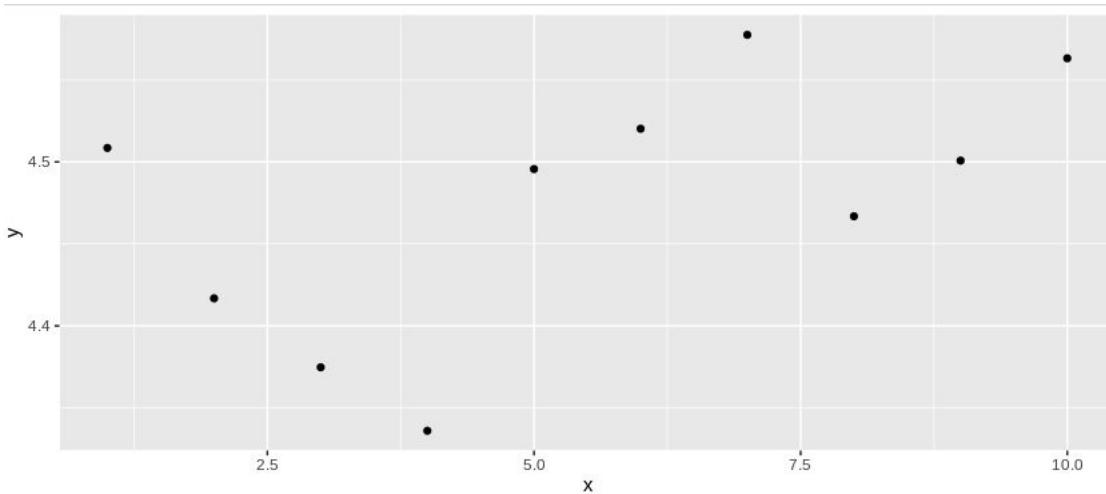
- Es un **data.frame** que viene con la librería de **ggplot2**
- Los datos se extrajeron de la revista **Motor Trend** de EE. UU. de 1974
- Comprenden el **consumo de combustible** y **10 aspectos del diseño** y el **rendimiento del automóvil** para **32 automóviles** (modelos 1973–74).

A data frame with 32 observations on 11 (numeric) variables.

```
[, 1] mpg Miles/(US) gallon  
[, 2] cyl  Number of cylinders  
[, 3] disp Displacement (cu.in.)  
[, 4] hp   Gross horsepower  
[, 5] drat Rear axle ratio  
[, 6] wt   Weight (1000 lbs)  
[, 7] qsec 1/4 mile time  
[, 8] vs    Engine (0 = V-shaped, 1 = straight)  
[, 9] am    Transmission (0 = automatic, 1 = manual)  
[,10] gear  Number of forward gears  
[,11] carb  Number of carburetors
```

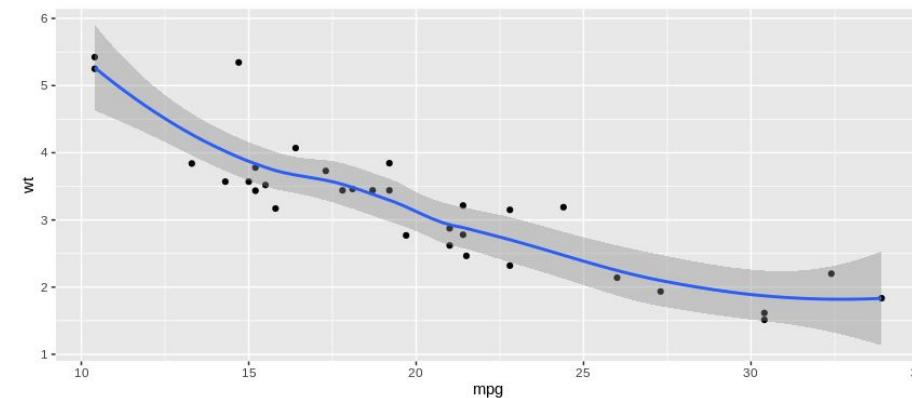
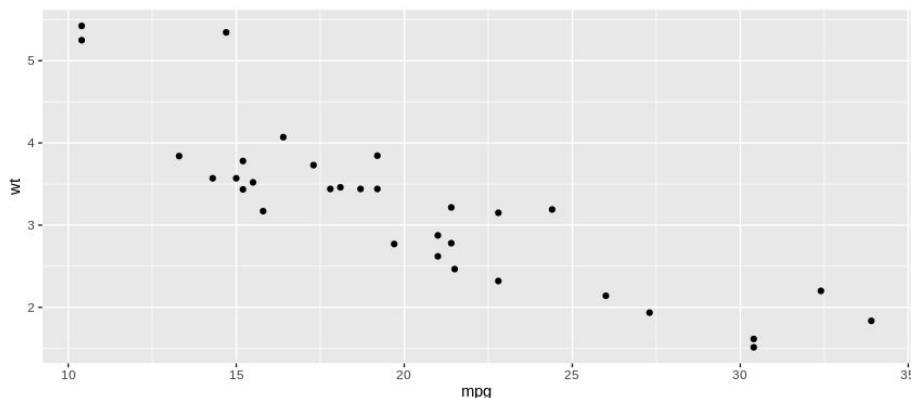
Ejercicio qplot()

```
# Use data from numeric vectors
x <- 1:10; y <- x*x
# Basic plot
qplot(x,y)
# Add line
qplot(x, y, geom=c("point", "line"))
```



Ejercicio qplot()

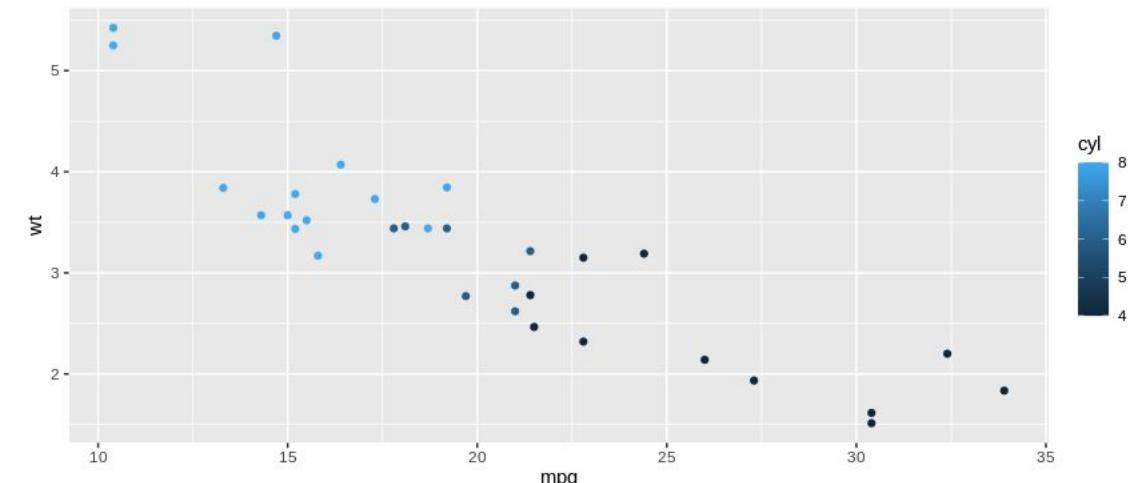
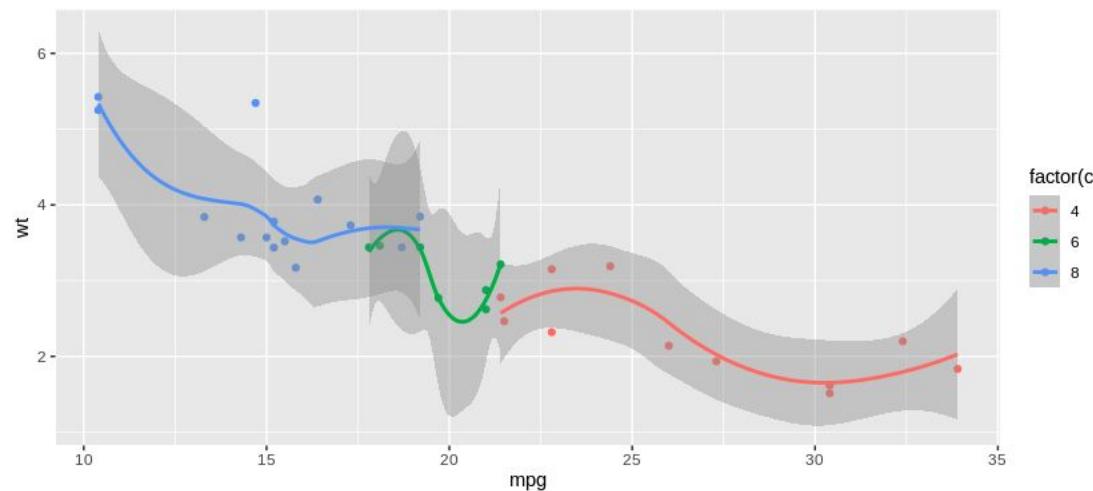
```
# Use data from a data frame  
# Explore data  
  
head(mtcars)  
  
qplot(mpg, wt, data=mtcars)  
  
# Smoothing (add a smoothed line with its standard error)  
qplot(mpg, wt, data = mtcars, geom = c("point", "smooth"))
```



Ejercicio qplot()

```
# Ajuste lineal por grupo
qplot(mpg, wt, data = mtcars, color = factor(cyl),
      geom=c("point", "smooth"))

# Cambiar el color por una variable numerica continua
qplot(mpg, wt, data = mtcars, colour = cyl)
```



Ejercicio qplot()

```
# Cambiar el color por grupos (factor)

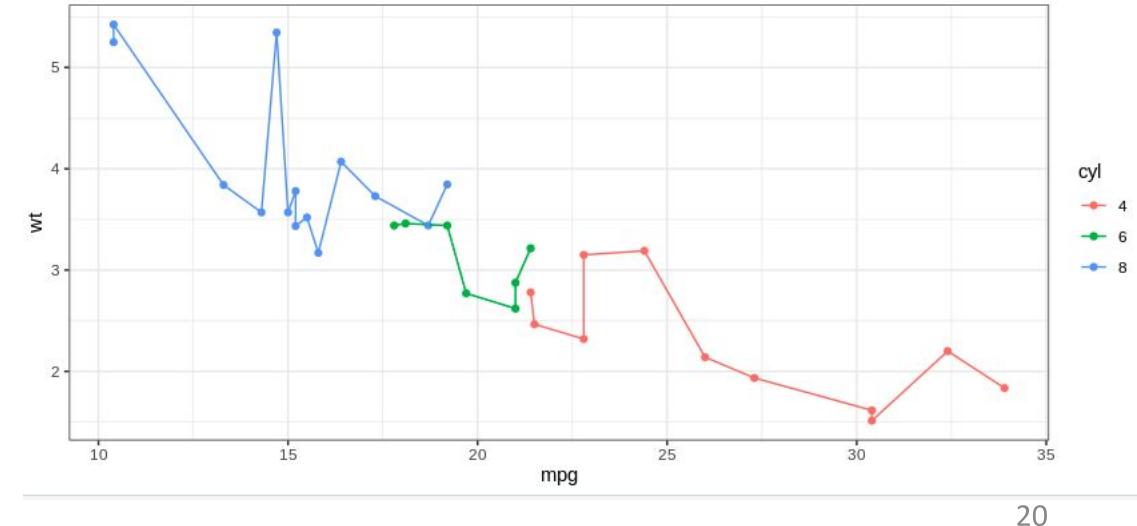
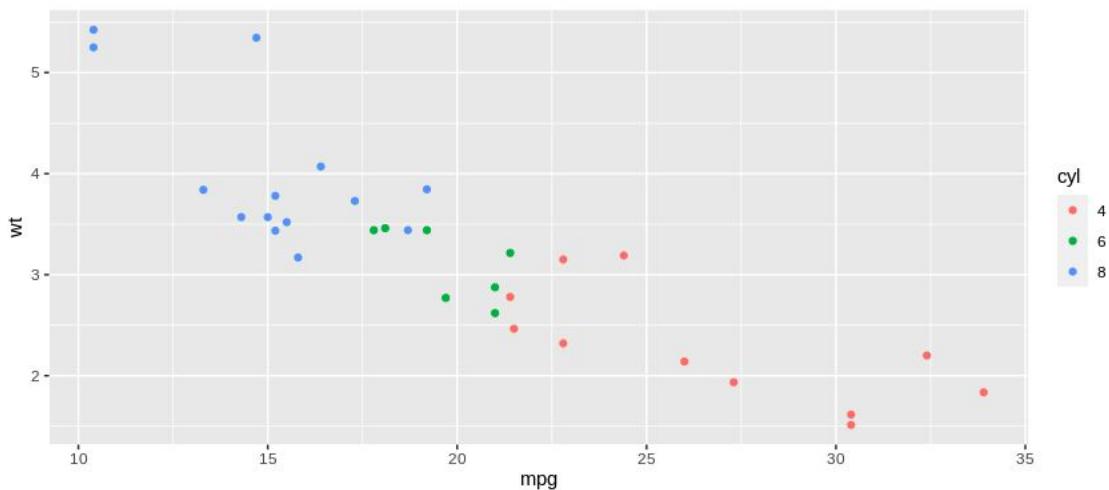
df <- mtcars

df[, 'cyl'] <- as.factor(df[, 'cyl'])

qplot(mpg, wt, data = df, colour = cyl)

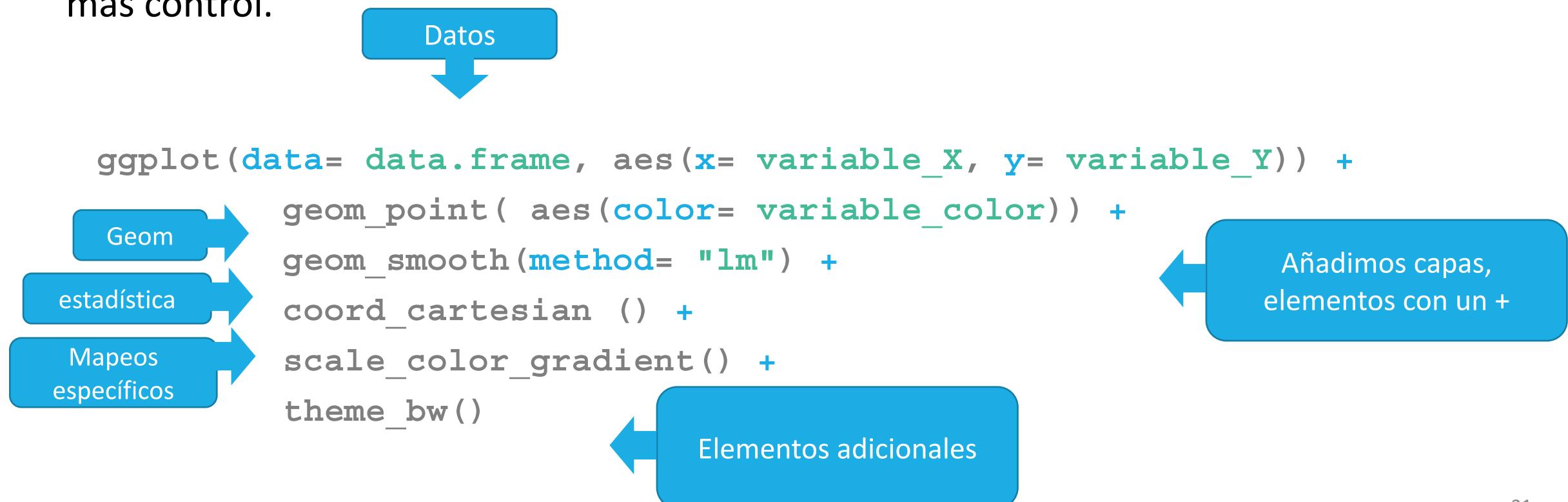
# Añadir lineas

qplot(mpg, wt, data = df, colour = cyl,
      geom=c("point", "line"))
```



Sintaxis ggplot()

- Crea un gráfico al que se le añaden capas. Sin valores por defecto, pero que proporciona más control.



`ggplot()` initializes a ggplot object. It can be used to declare the input data frame for a graphic and to specify the set of plot aesthetics intended to be common throughout all subsequent layers unless specifically overridden.

```
ggplot(data = NULL, mapping = aes(), ..., environment = parent.frame())
```

Arguments

data Default dataset to use for plot. If not already a `data.frame`, will be converted to one by `fortify()`.

If not specified, must be supplied in each layer added to the plot.

mapping Default list of aesthetic mappings to use for plot. If not specified, must be supplied in each layer added to the plot.

... Other arguments passed on to methods. Not currently used.

environment DEPRECATED. Used prior to tidy evaluation.

Datos -> Data

- Los datos son los elementos más importantes de un gráfico
- ggplot2 solo acepta un tipo de datos “**data.frames**”

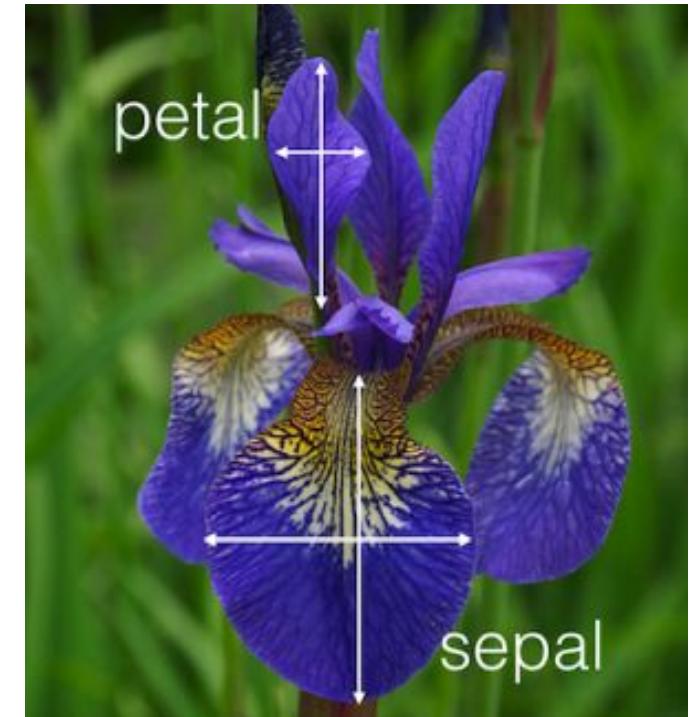
```
# Usaremos un set de datos que contiene la librería de ggplot2
```

```
# Explorar los datos
```

```
head(iris)
```

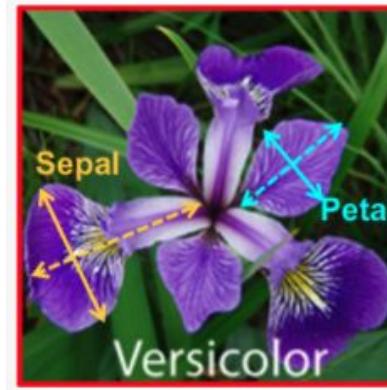
```
summary(iris)
```

```
p <- ggplot(data=iris)
```



Datos -> iris

- Este famoso conjunto de datos de **iris** (de Fisher o Anderson) da las medidas en centímetros de las variables:
 - Longitud y ancho del sépalo
 - Largo y ancho del pétalo
 - Para 50 flores de cada una de las 3 especies de iris.
 - Iris setosa, versicolor y virginica.



Estéticas -> aesthetics

- En un conjunto de datos hay muchas **variables** que asignamos a columnas dentro del data.frame
- Estas **variables** dentro de la terminología de ggplot2 se les llama **estéticas**
- Por lo que nosotros vamos a darle un significado a las variables dentro del gráfico.

```
p <- p + aes(x= Petal.Length, y = Petal.Width, colour = Species)
```

- Estamos añadiendo al objeto **p** la información sobre las estéticas que obtiene de los datos “**iris**”

Capas -> Layers

- Las capas o los objetos geométricos en ggplot (**geom**) indican que hacer con los datos y las estéticas que le indicaste previamente.
- Es la representación gráfica de tus datos
- ggplot2 nos permite ir añadiendo “**capas**” al lienzo para irlo modificando de acuerdo a nuestras necesidades.

```
# Add layers  
p <- p + geom_point()  
# Display the object "p"  
p  
p <- p + geom_smooth()
```

Facetas -> Faceting

- Las facetas implementan los gráficos de Trellis
- Las facetas nos permiten **hacer comparaciones** entre las diferentes variables
- Para poder hacer comparaciones adecuadas, una característica esencial es que se **comparten ejes**.

```
## Facet grid
### Vertical
ggplot(iris, aes(x = Petal.Length, y = Petal.Width)) + geom_point()
+ geom_smooth() + facet_grid(~ Species)
### Horizontal
ggplot(iris, aes(x = Petal.Length, y = Petal.Width)) + geom_point()
+ geom_smooth() + facet_grid(Species ~ .)
```

Temas -> Themes

- Los temas en ggplot2 permiten hacer **modificaciones a los aspectos estéticos** del gráfico.
- Lo que incluye son:
 - los ejes, etiquetas, colores de fondo, el tamaño de los márgenes, paletas de colores.
- Los temas son una colección de elementos modificables.
- El tema por defecto en ggplot es el `theme_grey`
- Podemos elegir entre una variedad de temas que tiene ggplot2
- Es posible construir temas propios y personalizados

```
## Change the Theme  
p <- p + theme_bw()  
p
```

Elementos modificables en Themes

Name	Description	Element type			
text	All text elements	element_text()	plot.background	Background of the entire plot	element_rect(fill = "white", colour = NA)
rect	All rectangular elements	element_rect()			element_text()
line	All line elements	element_line()	plot.title	Title text appearance	element_rect()
axis.line	Lines along axes	element_line()	strip.background	Background of facet labels	element_text()
axis.title	Appearance of both axis labels	element_text()	strip.text	Text appearance for vertical and horizontal facet labels	element_text()
axis.title.x	X-axis label appearance	element_text()	strip.text.x	Text appearance for horizontal facet labels	element_text()
axis.title.y	Y-axis label appearance	element_text()	strip.text.y	Text appearance for vertical facet labels	element_text()
axis.text	Appearance of tick labels on both axes	element_text()	panel.background	Background of plotting area	element_rect()
axis.text.x	X-axis tick label appearance	element_text()	panel.border	Border around plotting area	element_rect(linetype="dashed")
axis.text.y	Y-axis tick label appearance	element_rect()	panel.grid.major	Major grid lines	element_line()
legend.background	Background of legend	element_text()	panel.grid.major.x	Major grid lines, vertical	element_line()
legend.text	Legend item appearance	element_text()	panel.grid.major.y	Major grid lines, horizontal	element_line()
legend.title	Legend title appearance	element_text()	panel.grid.minor	Minor grid lines	element_line()
legend.position	Position of the legend	"left", "right", "bottom", "top" two-element numeric vector inside the plot area (for more see Recipe 10.2)	panel.grid.minor.x	Minor grid lines, vertical	element_line()
			panel.grid.minor.y	Minor grid lines, horizontal	element_line()

¿Cómo guardo mis gráficos?

- Hay **dos maneras** de guardar tus imágenes

1. Con funciones propias de la base de R

- **png()**
- **pdf()**
- Tu les indicas las especificaciones de tamaño y calidad así como la ruta y el nombre de la imagen.
- La función se abre antes de generar la gráfica y se tiene que cerrar el archivo con la función **dev.off()**

2. Con la función **ggsave()** de ggplot2

- Guarda la última imagen que creaste en el directorio en donde estás trabajando.

¿Cómo guardo mis gráficos?

```
# Primera opción

#### PNG con una resolución de 300 px

png("dot_plot_iris_data.png", width=10*300, height=8*300, res= 300, units =
"px")

p

dev.off()

#### PDF (imágenes vectorizadas)

pdf("dot_plot_iris_data.pdf", width=10, height=8)

p

dev.off()

# Segunda opción con una función propia de ggplot2

ggsave("dot_plot_iris_data.jpg", device= "jpg")
```

Ejercicio: Bar plots

```
# Crear datos

cabbage_exp <- data.frame(Cultivar=c(rep("c39", 3), rep("c52", 3)), Date=rep(c("d16",
"d20", "d21"), 2), Weight=c(3.18, 2.80, 2.74, 2.26, 3.11, 1.47), sd= c(0.9566144,
0.2788867, 0.9834181, 0.4452215, 0.7908505, 0.2110819), n=rep(10,6), se=c(0.30250803,
0.08819171, 0.31098410, 0.14079141, 0.25008887, 0.06674995))

cabbage_exp
# plot

ggplot(data= cabbage_exp, aes(x=Date, y= Weight)) +
  geom_bar(stat="identity")
ggplot(data= cabbage_exp, aes(x=Date, y= Weight)) +
  geom_bar(stat="identity", fill="dodgerblue")
ggplot(data= cabbage_exp, aes(x=Date, y= Weight)) +
  geom_bar(stat="identity", fill="dodgerblue") +
  theme_bw()

### Graph with grouped bars
ggplot(cabbage_exp, aes(x=Date, y=Weight, fill=Cultivar)) +
  geom_bar(stat="identity", position="dodge")
```



Ejercicio: Bar plots

`cabbage_exp` -> Es el recopilado de un experimento de un cultivo de "Col o repollo (cabbage)".

- Fueron 2 condiciones C39 y C52;
- Se recolectaron las coles a los días 16, 20 y 21
- se pesaron 10 coles de la condición
- Se obtuvo la desviación estandar y el error estandar de la media del peso.



Crear datos

```
cabbage_exp <- data.frame(Cultivar=c(rep("c39", 3), rep("c52", 3)), Date=rep(c("d16", "d20", "d21"), 2), Weight=c(3.18, 2.80, 2.74, 2.26, 3.11, 1.47), sd= c(0.9566144, 0.2788867, 0.9834181, 0.4452215, 0.7908505, 0.2110819), n=rep(10,6), se=c(0.30250803, 0.08819171, 0.31098410, 0.14079141, 0.25008887, 0.06674995))
```

Paletas de colores en R

- R tiene paletas de colores ya preestablecidas
- Y tiene una librería que se dedica exclusivamente al color
 - `library(RColorBrewer)`



Use another color palette

```
ggplot(data=cabbage_exp, aes(x=Date, y=Weight, fill=Cultivar)) +  
  geom_bar(stat="identity", position="dodge") +  
  scale_fill_brewer(palette="Set1")
```

Add text

```
ggplot(data=cabbage_exp, aes(x=Date, y=Weight, fill=Cultivar)) +  
  geom_bar(stat="identity") +  
  scale_fill_brewer(palette="Set1") +  
  geom_text(aes(label=Weight), vjust=1.5)
```

position dodge

```
ggplot(data=cabbage_exp, aes(x=Date, y=Weight, fill=Cultivar)) +  
  geom_bar(stat="identity", position="dodge") +  
  scale_fill_brewer(palette="Set1") +  
  geom_text(aes(label=Weight), vjust=-0.4, position=position_dodge(0.9), size=3,  
colour="green4")
```

Add error bars

```
ggplot(data=cabbage_exp, aes(x=Date, y=Weight, fill=Cultivar)) +  
  geom_bar(stat="identity", position="dodge") +  
  scale_fill_brewer(palette="Set1") +  
  geom_errorbar(aes(ymin=Weight-se, ymax=Weight+se), width=0.2,  
position=position_dodge(0.9))
```

Ejercicio: scatterplots

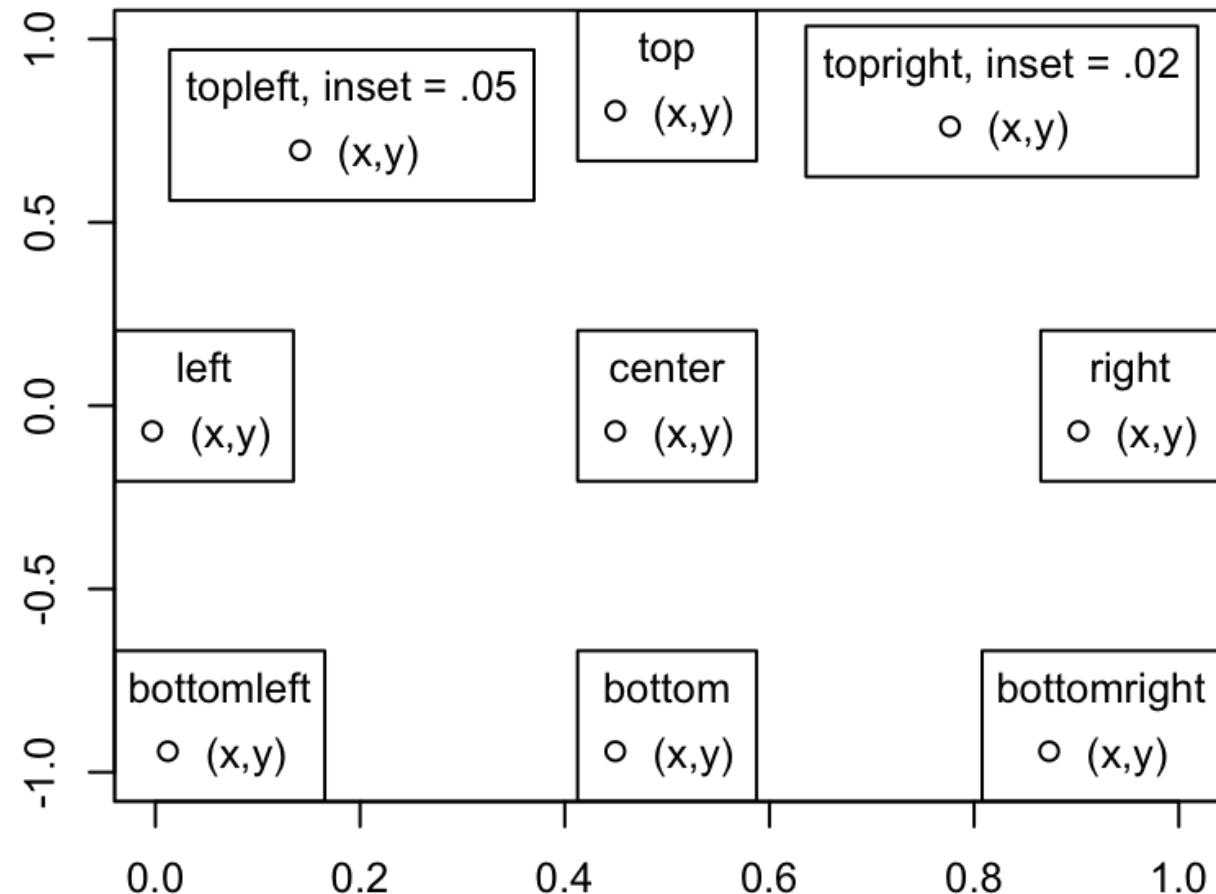
```
# Basic scatterplot
ggplot(data= iris, aes(x= Petal.Length, y= Petal.Width)) +
  geom_point()
# split by color
ggplot(data= iris, aes(x= Petal.Length, y= Petal.Width, colour=Species)) +
  geom_point()
#split by shape
ggplot(data= iris, aes(x= Petal.Length, y= Petal.Width, shape=Species)) +
  geom_point()
#split by shape and color
ggplot(data= iris, aes(x= Petal.Length, y= Petal.Width, shape=Species, colour=Species)) +
  geom_point()
# change the values of shape
ggplot(data= iris, aes(x= Petal.Length, y= Petal.Width, shape=Species, colour=Species)) +
  geom_point() +
  scale_shape_manual(values=c(1,2,3))
# change manual colors
ggplot(data= iris, aes(x= Petal.Length, y= Petal.Width, shape=Species, colour=Species)) +
  geom_point() +
  scale_shape_manual(values=c(1,2,3)) +
  scale_colour_manual(values=c("deepskyblue", "seagreen2", "salmon")) +
  theme_bw()
```

Shapes in R

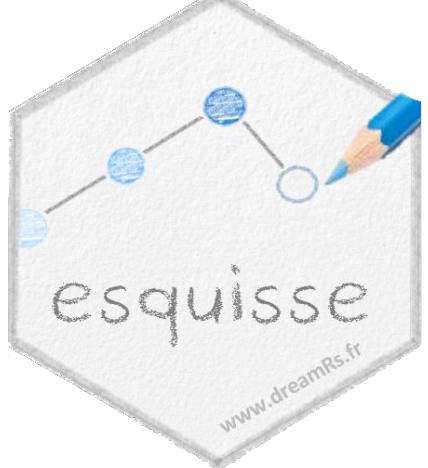
- R tiene un código de **números y símbolos** que pueden cambiar la forma de los objetos que graficamos.

0	□	6	▽	12	田	18	◆	24	△	0	0
1	○	7	⊗	13	⊗	19	●	25	▽	+	+
2	△	8	*	14	☒	20	●	*	*	-	-
3	+	9	◊	15	■	21	○	.	.		
4	×	10	⊕	16	●	22	□	0	○	%	%
5	◇	11	✡	17	▲	23	◊	0	○	#	#

Legend positions

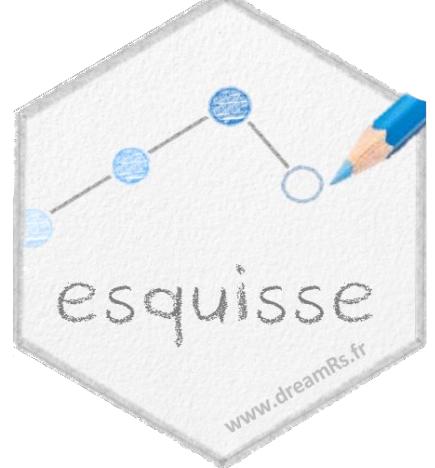


Gráficos Interactivos de ggplot: esquisse



- Paquete de R que permite realizar análisis exploratorios graficando con ggplot de manera interactiva.
- Genera una interfaz gráfica donde puedes seleccionar con botones las opciones del plot
- Genera el código de ggplot para poder reproducir las graficas
- Permite de manera rápida e intuitiva explorar los datos.

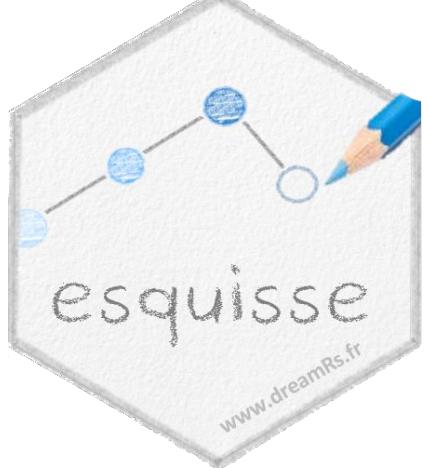
Gráficos Interactivos de ggplot: esquisse



- Instalandola desde su **repositorio** de preferencia
 - `install.packages("esquisse")`
- O instalar la versión en desarrollo desde **GitHub**
 - `remotes::install_github("dreamRs/esquisse")`

```
install.packages("esquisse")
library(esquisse)
```

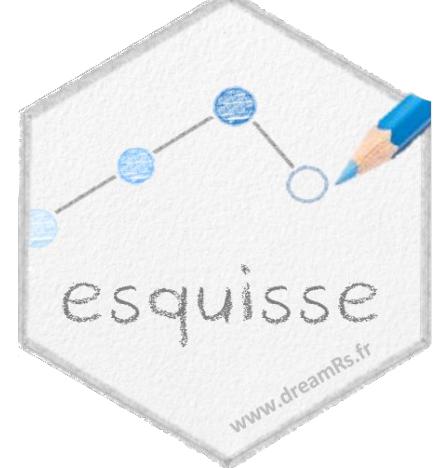
Gráficos Interactivos de ggplot: esquisse



- Cargar datos
- Para lanzar la aplicación se utiliza el comando “**esquisser()**”, dentro del cual se pone el **data.frame** a explorar

```
esquisser(iris)
```

Gráficos Interactivos de ggplot: esquisse - Iris data.frame



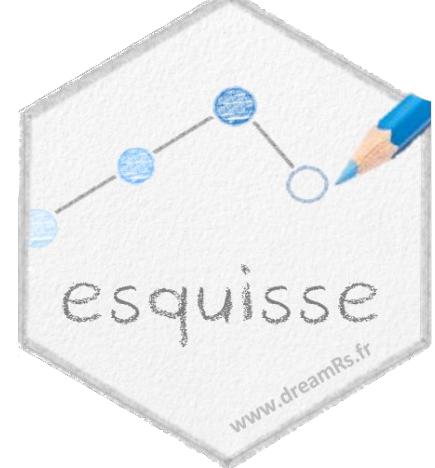
```
esquisser(iris)
```



- En este caso vamos a cargar el data.frame iris que se encuentra cargado en R
- Da las mediciones en centímetros de ancho y largo de sépalo y pétalos de 3 especies de flores iris

Gráficos Interactivos de ggplot: esquisse

esquisser(iris)



The screenshot shows the ggplot2 builder interface. A red arrow points to the "Geom" section, labeled "Tipo de gráfico". Another red arrow points to the "Variables" section, labeled "Variables". A third red arrow points to the "Mapeos" section, labeled "Mapeos". A fourth red arrow points to the bottom right corner, labeled "Controles".

Geom
(Tipo de gráfico)

Variables

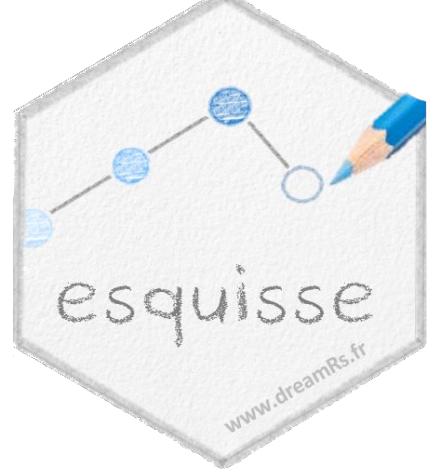
Mapeos

Área del plot

Controles

43

Gráficos Interactivos de ggplot: esquisse

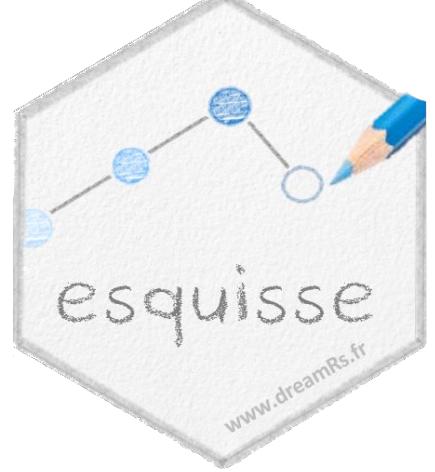


- Tomamos las variables y las llevamos al área de mapeo

The screenshot shows the ggplot2 builder interface. The 'Variables' section has 'Sepal.Length', 'Sepal.Width', 'Petal.Length', and 'Petal.Width' selected. The 'X' field is highlighted with a dashed border. The 'Color' field is also highlighted with a dashed border. The 'Play' button is visible at the bottom right.

- Si tomamos
“Sepal.Length” Y
colocamos en la
sección de mapeo al
eje **X**

Gráficos Interactivos de ggplot: esquisse



- Tomamos las variables y las llevamos al área de mapeo

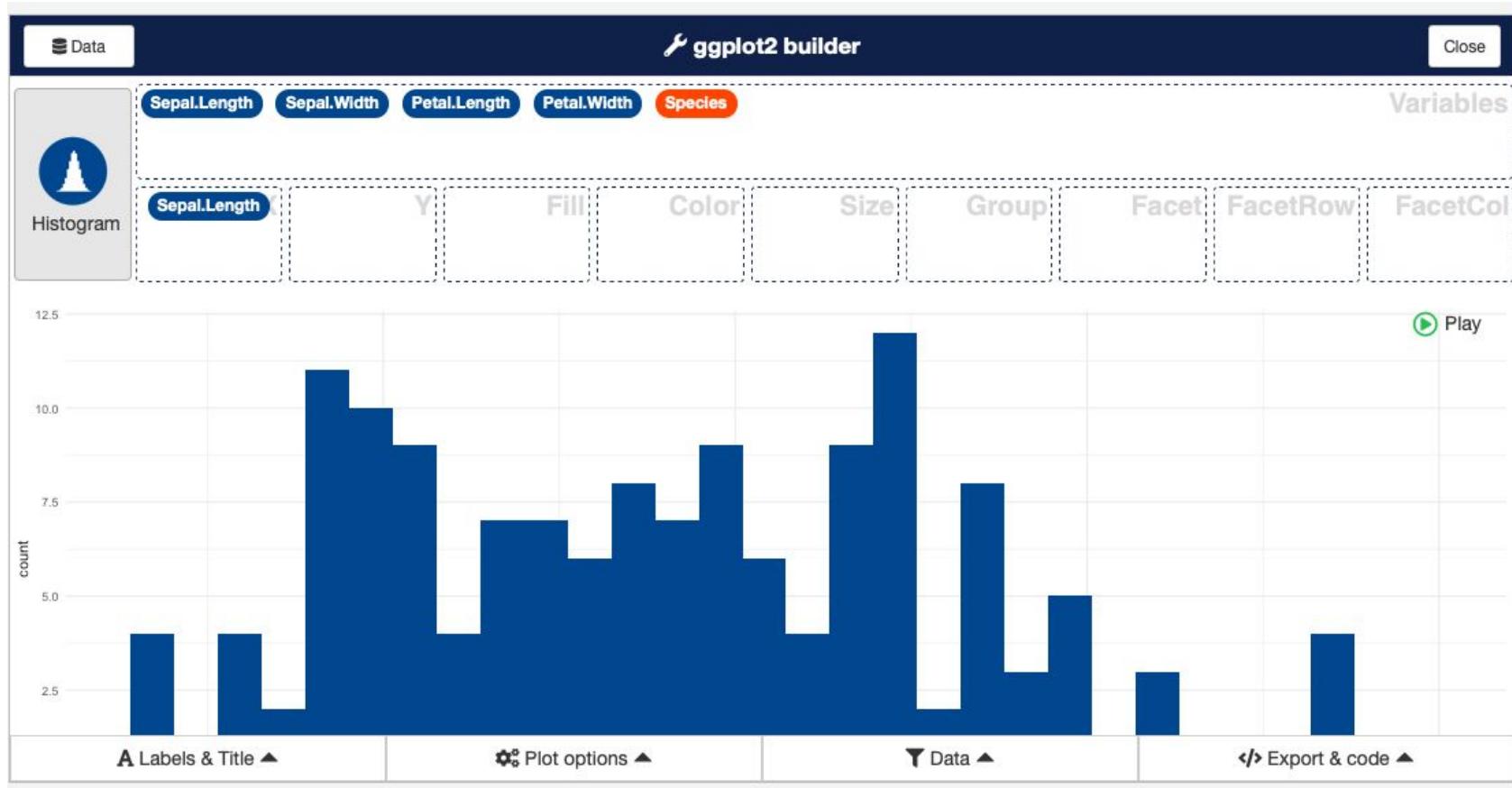
The screenshot shows the ggplot2 builder interface. In the 'Variables' section, 'Sepal.Length' is selected for the X variable. The 'Data' tab is active. A green box highlights the 'X' input field. A yellow circle labeled 'auto' is selected in the dropdown. A 'Play' button is visible at the bottom right.

- Si tomamos
“Sepal.Length” Y
colocamos en la
sección de mapeo al
eje **X**

Gráficos Interactivos de ggplot: esquisse

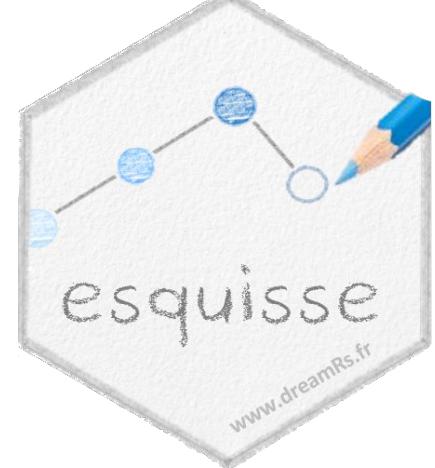


- Tomamos las variables y las llevamos al área de mapeo



- Automáticamente selecciona la clase de plot y genera el plot

Gráficos Interactivos de ggplot: esquisse

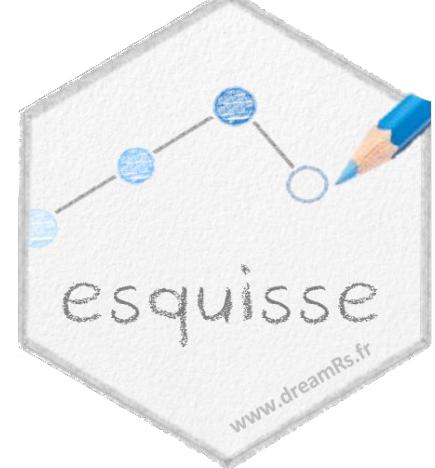


- Tomamos las variables y las llevamos al área de mapeo



- Código

Gráficos Interactivos de ggplot: esquisse

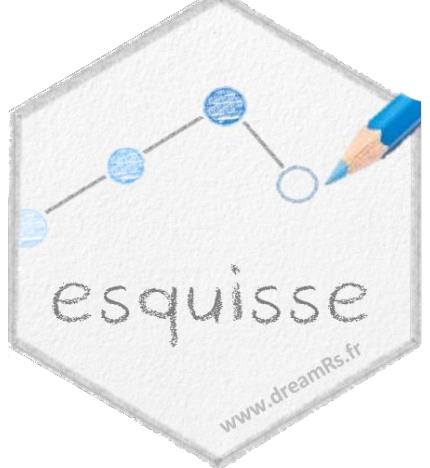


- Tomamos las variables y las llevamos al área de mapeo



- Podemos ahora tomar la variable **Species** y utilizarla para darle color llevandola a la región de mapero de **color**

Gráficos Interactivos de ggplot: esquisse



- Tomamos las variables y las llevamos al área de mapeo



- Código

Gráficos Interactivos de ggplot: esquisse



- Podemos Cambiar el tipo de gráfico



- Nos muestra las opciones que se adaptan a los datos seleccionados

Gráficos Interactivos de ggplot: esquisse

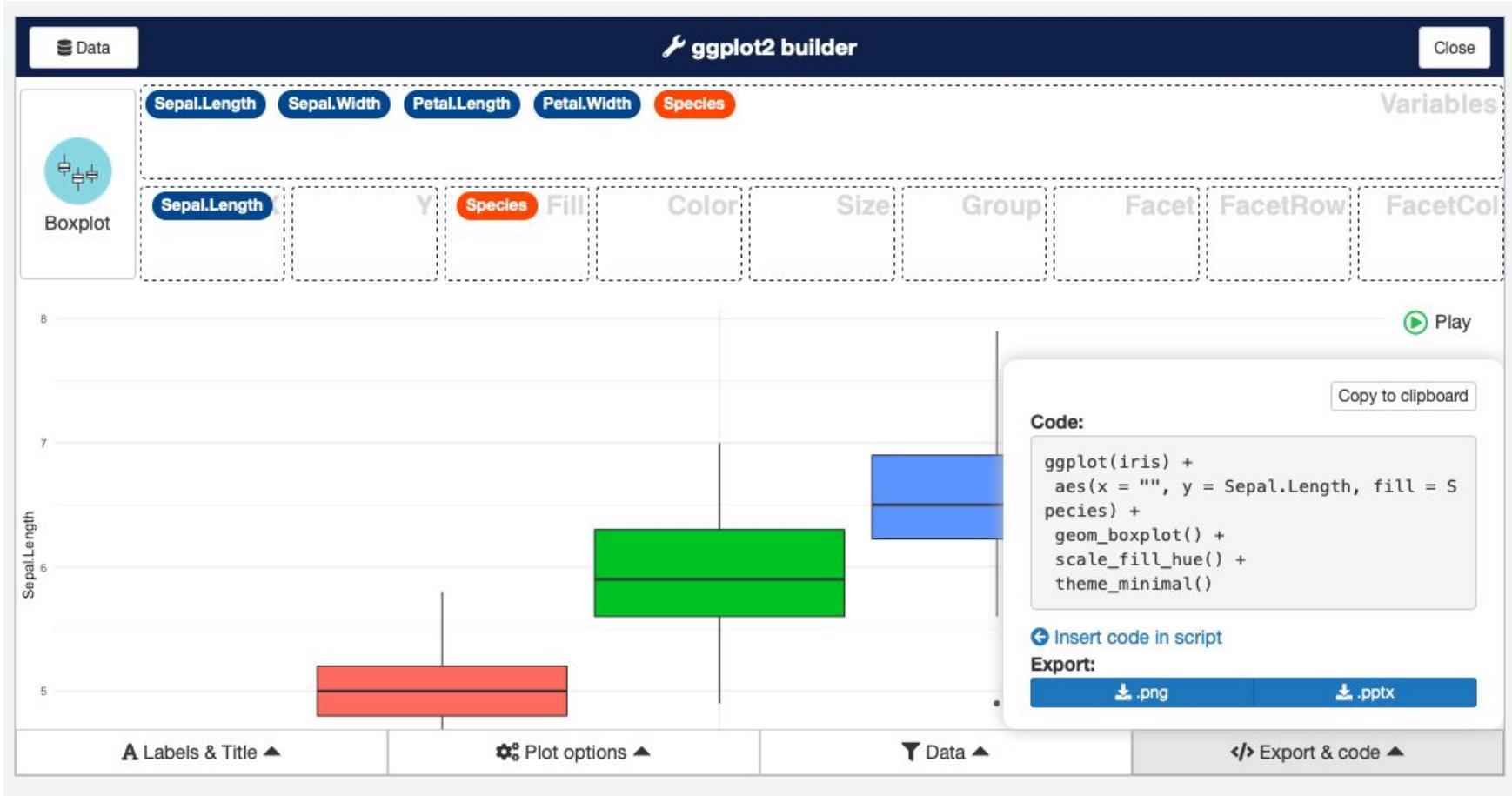
- BoxPlot



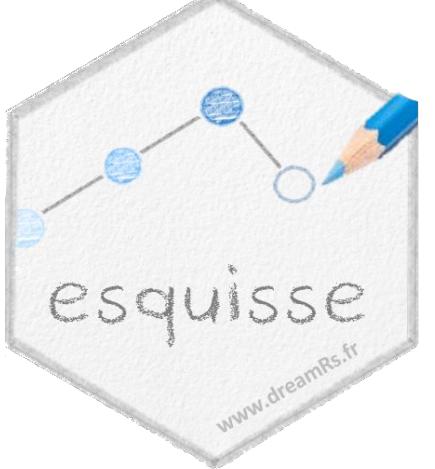
- Boxplot o diagrama de caja y bigote nos permite representar gráficamente los datos a través de sus cuartiles

Gráficos Interactivos de ggplot: esquisse

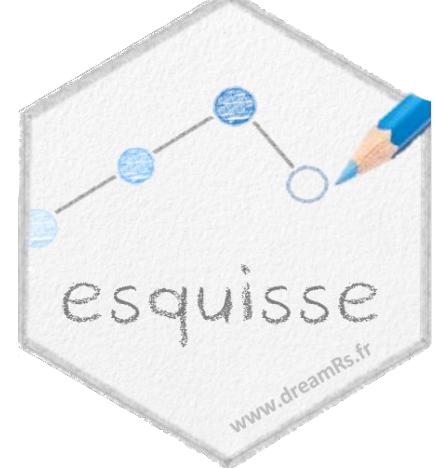
- BoxPlot



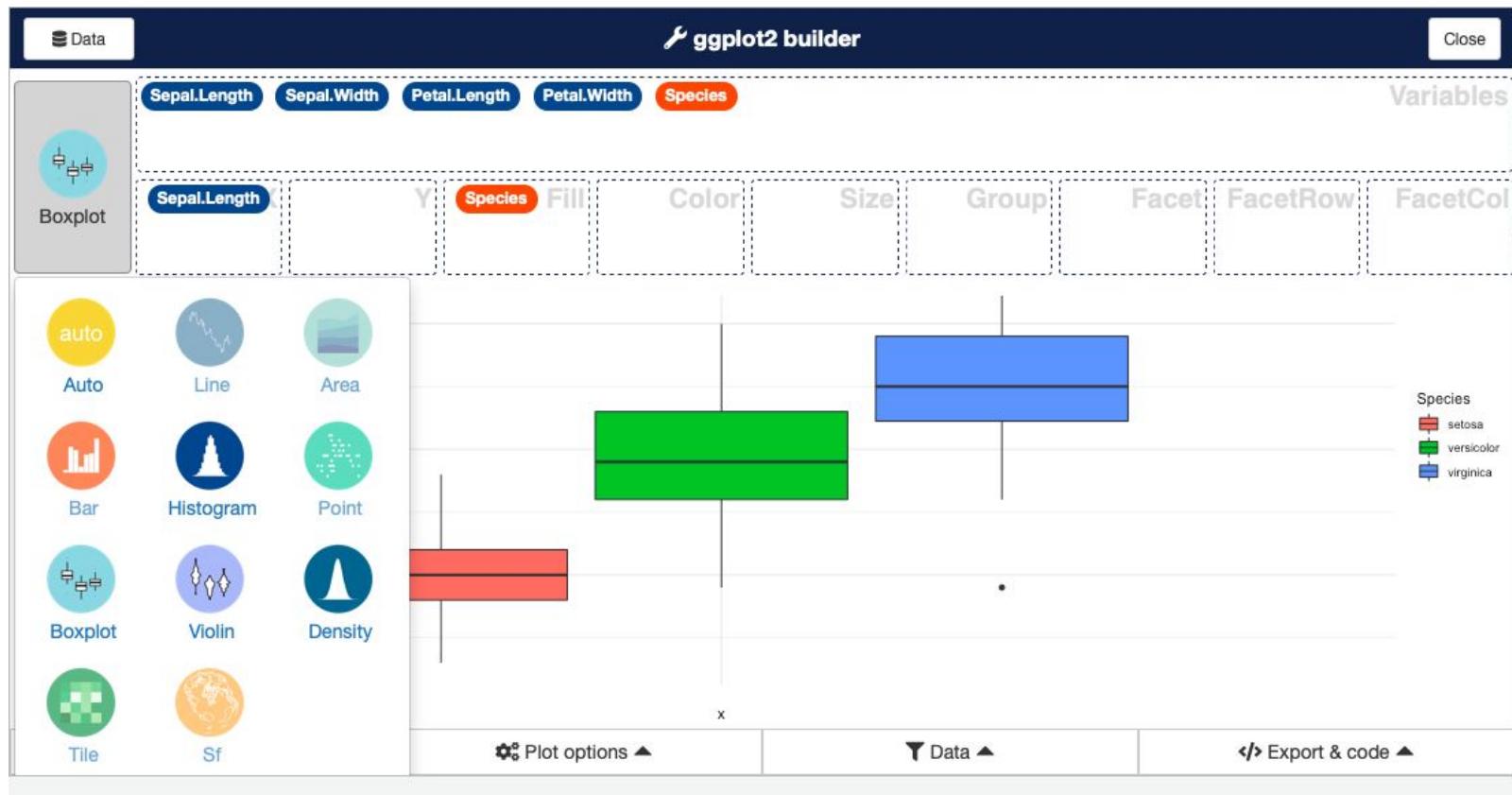
- Código



Gráficos Interactivos de ggplot: esquisse

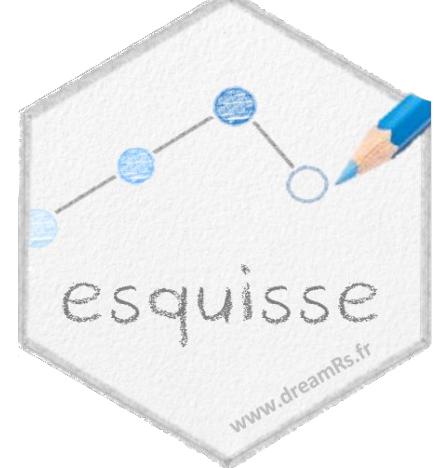


- Tomamos las variables y las llevamos al área de mapeo

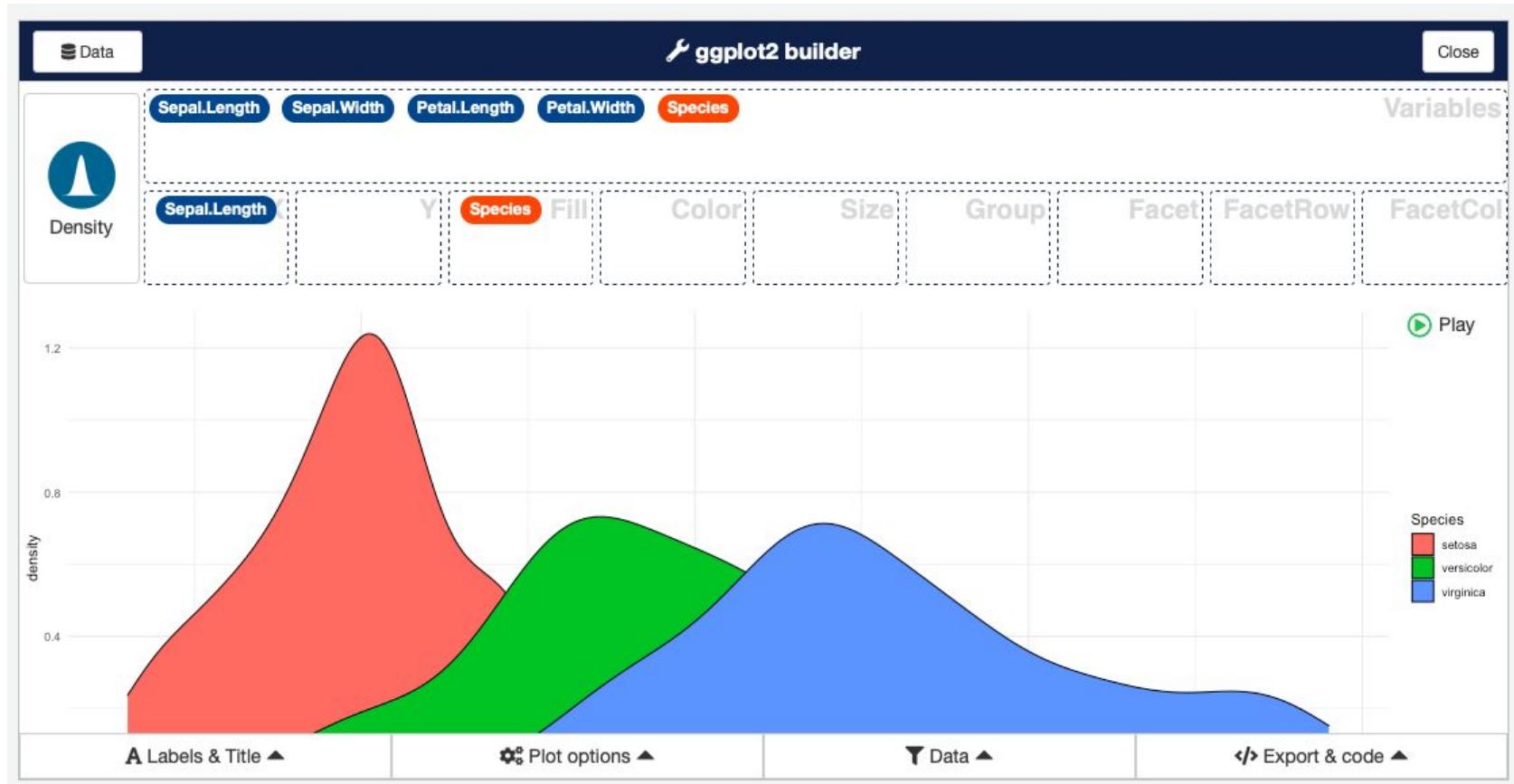


- Esto podemos cambiarlo y seleccionar el de nuestra preferencia

Gráficos Interactivos de ggplot: esquisse

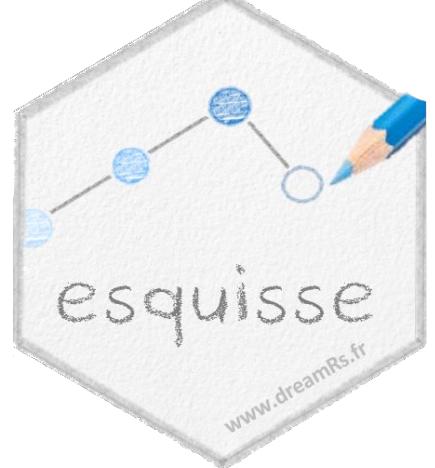


- Diagrama de densidad



- Los diagrama de densidad nos permiten representar la distribución de una variable numérica

Gráficos Interactivos de ggplot: esquisse

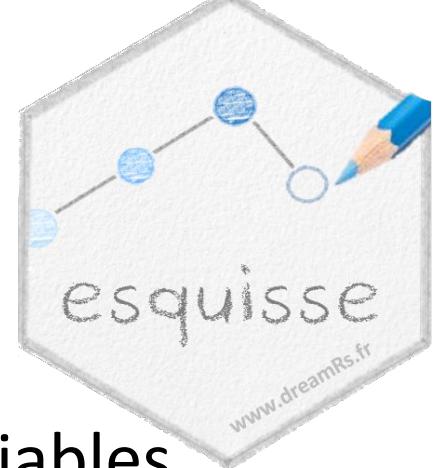


- Diagrama de densidad

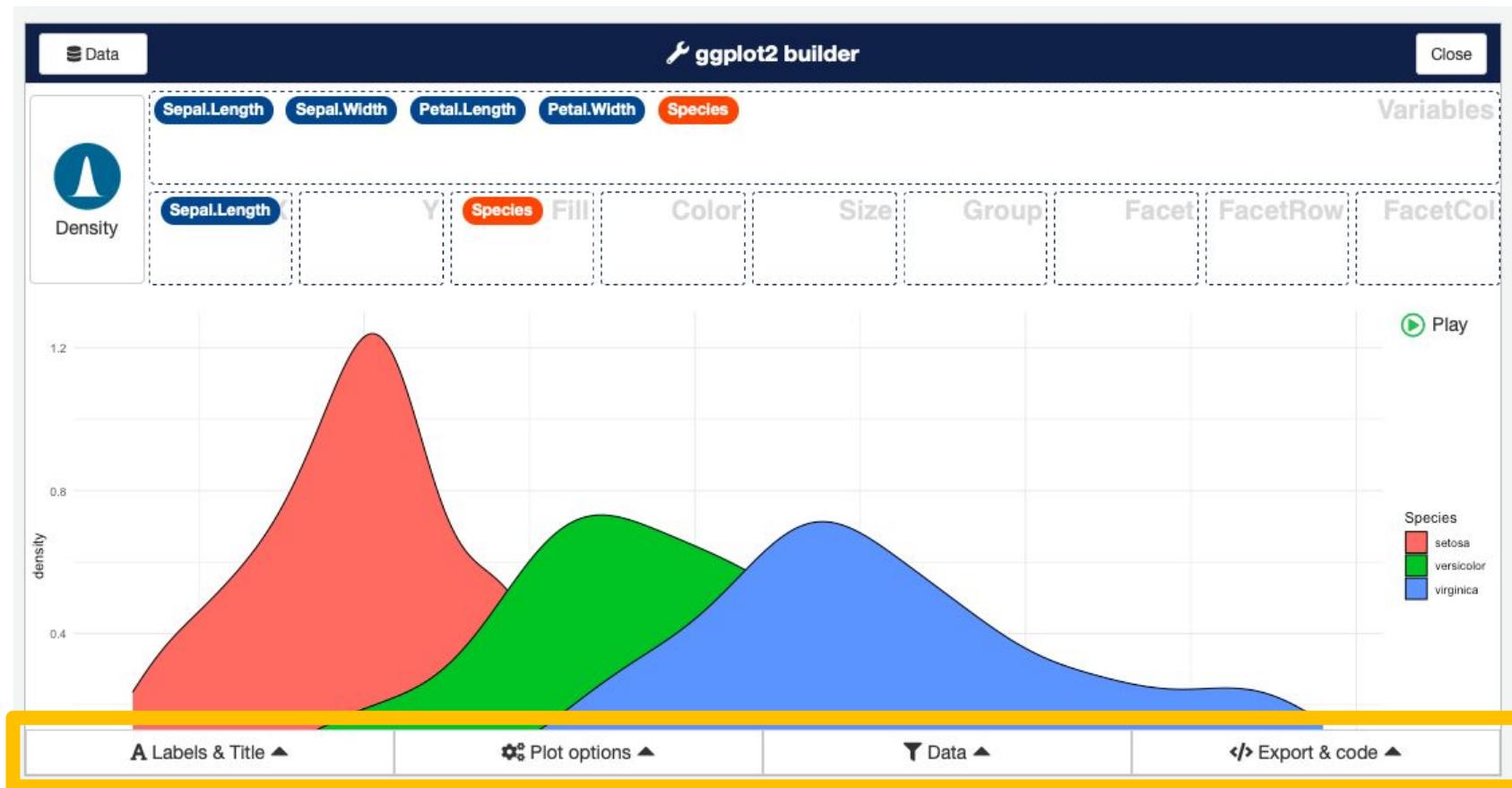


- Código

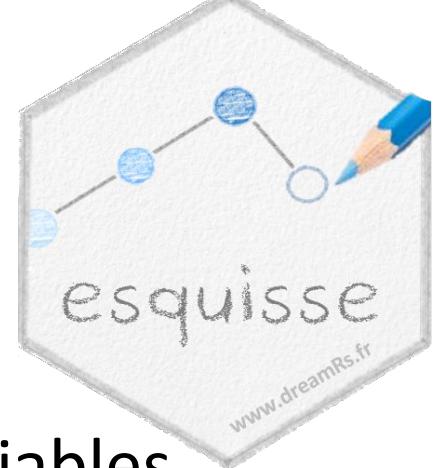
Gráficos Interactivos de ggplot: esquisse



En el apartado de **Controles** podemos modificar diversas variables



Gráficos Interactivos de ggplot: esquisse

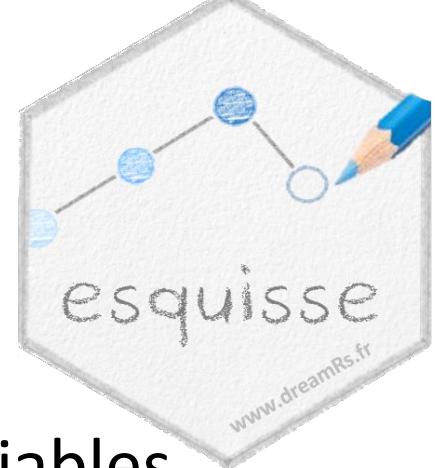


En el apartado de **Controles** podemos modificar diversas variables

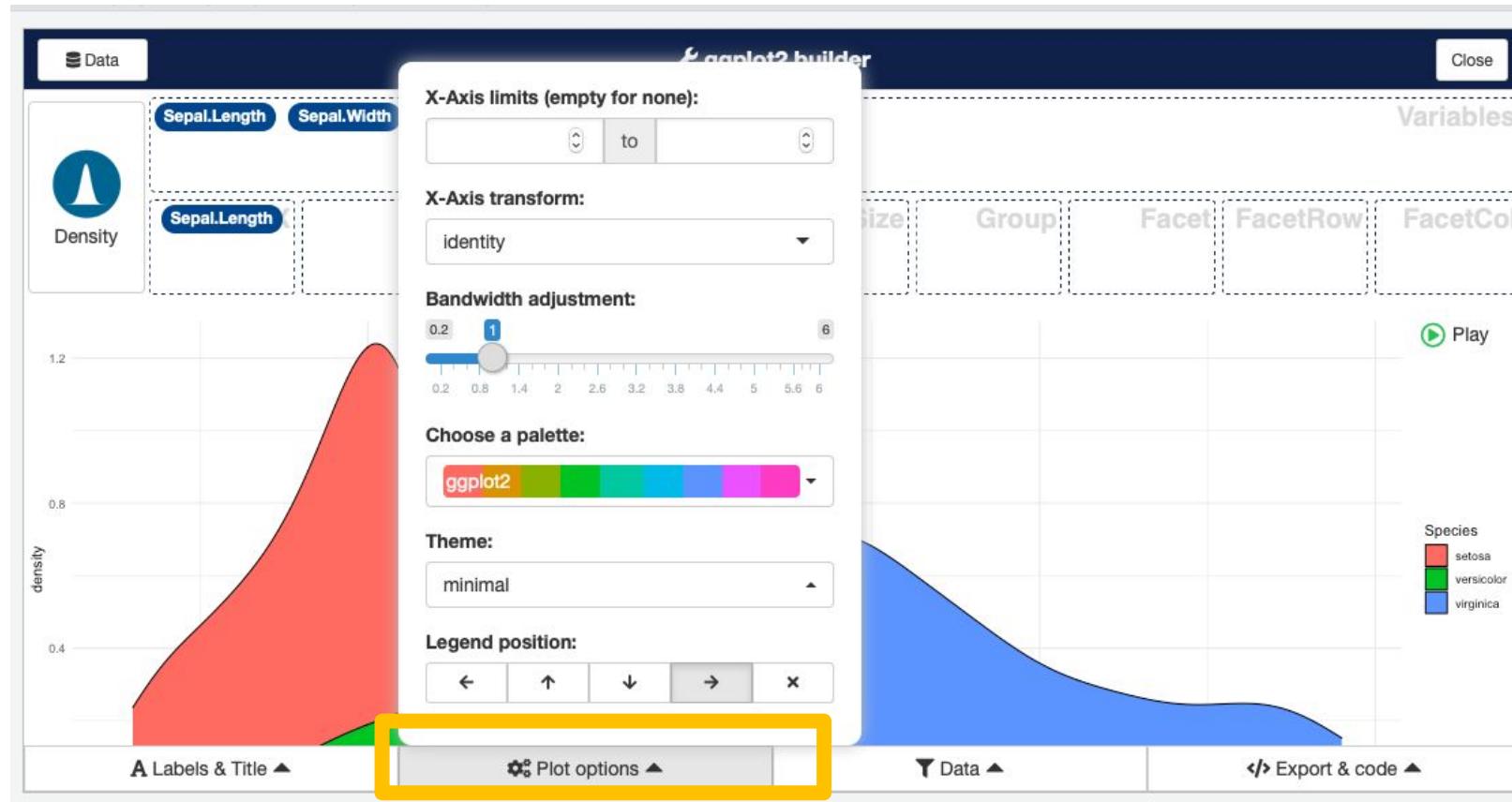


Podemos cambiar
Titulos,subtitulos,
etiquetas de los ejes,
etcétera.

Gráficos Interactivos de ggplot: esquisse



En el apartado de **Controles** podemos modificar diversas variables



Podemos cambiar los límites de los ejes, la paleta de colores, la posición de la legenda, etc.

Gráficos Interactivos de ggplot: esquisse

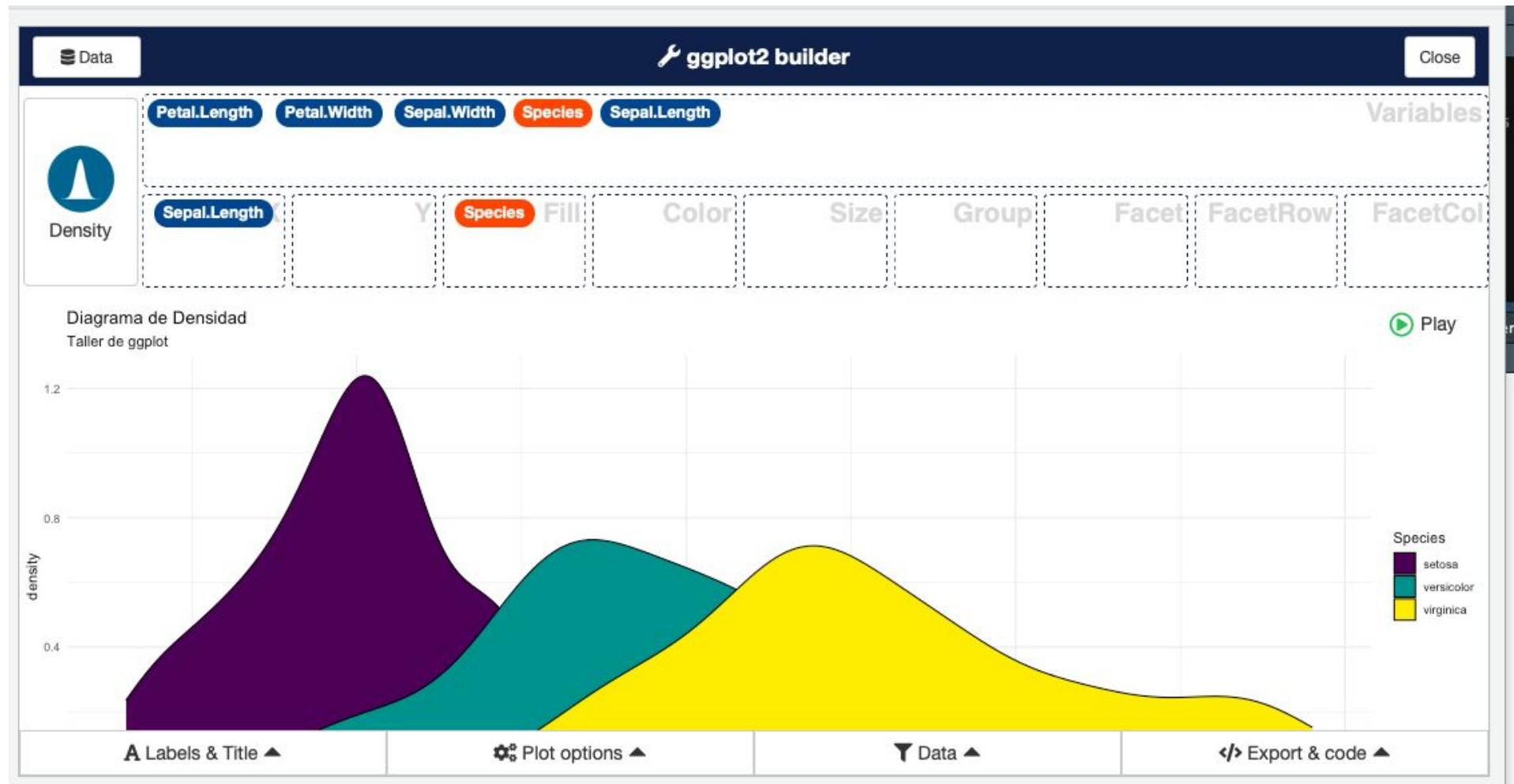


En el apartado de **Controles** podemos modificar diversas variables

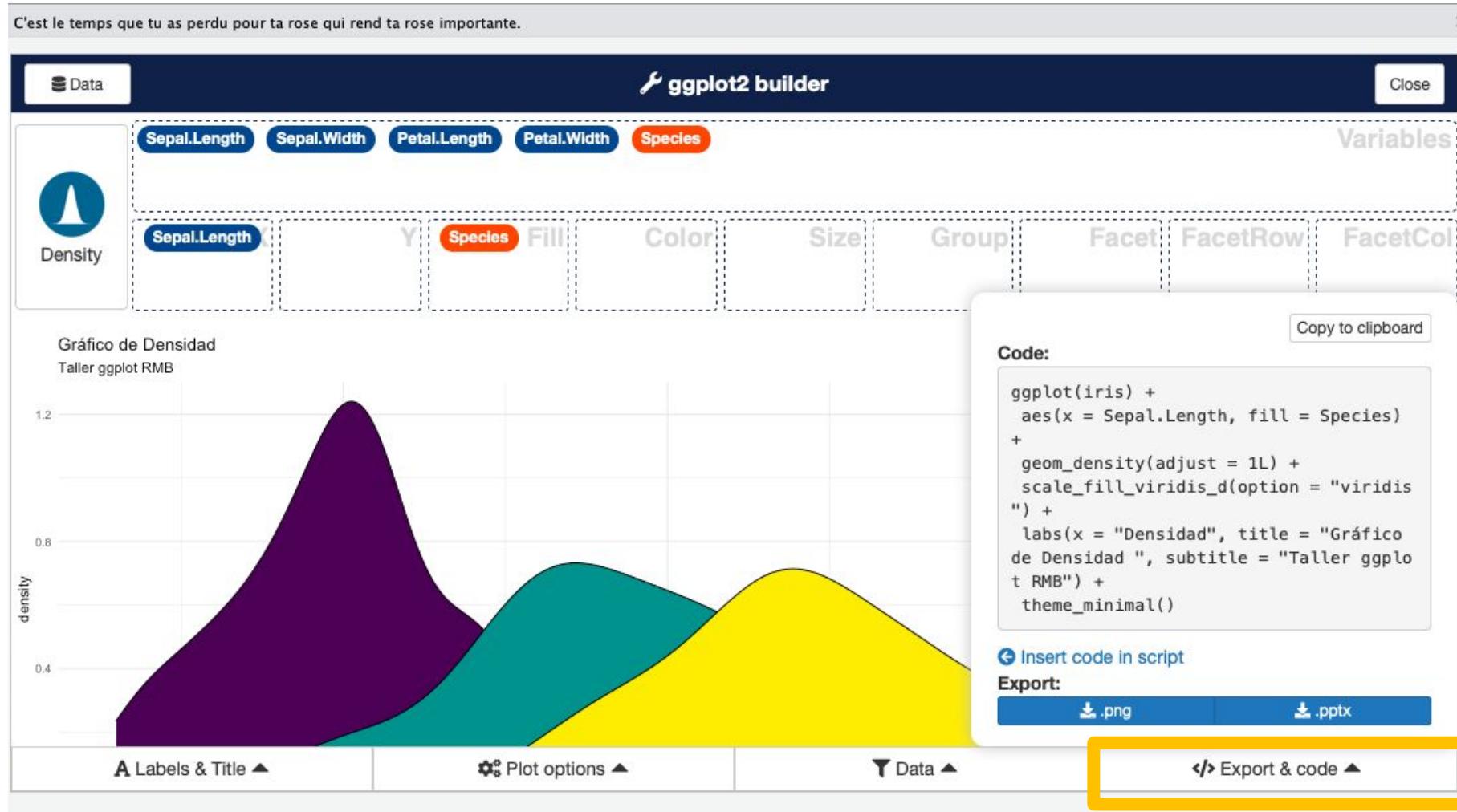
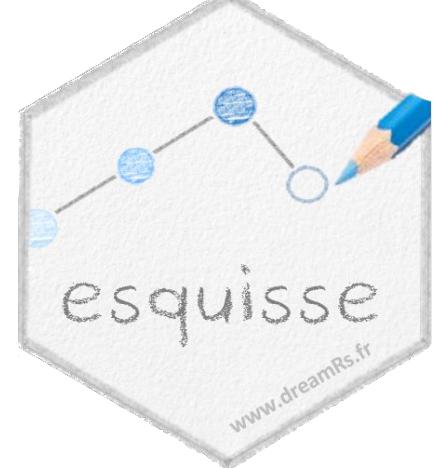


Podemos hacer manipular
los datos haciendo subsets
de variables particulares

Gráficos Interactivos de ggplot: esquisse



Gráficos Interactivos de ggplot: esquisse



Podemos exportar el plot y ver el código que lo genera.

Recursos

- R Graphics Cookbook de Winston Chang
 - <https://r-graphics.org/>
- Página de “tidyverse”
 - <https://ggplot2.tidyverse.org/>
- STHDA (Statistical tools for hig-throughput data analysis)
 - <http://www.sthda.com/english/wiki/ggplot2-essentials>
- Tutoriales en línea
 - <http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html>
 - <http://r-statistics.co/ggplot2-Tutorial-With-R.html>
- Cheatsheets
 - <https://rstudio.com/wp-content/uploads/2015/04/ggplot2-spanish.pdf>
- Google
 - “How to in ggplot2”



Red Mexicana de
Bioinformática



¿Dudas?



elizabeth.godoy@insp.mx
car.barjon@gmail.com



Red Mexicana de
Bioinformática



Estamos en Pausa



Continuamos en unos minutos

elizabeth.godoy@insp.mx
car.barjon@gmail.com