Visualización de datos en R

Elena Quintero

13/01/2025

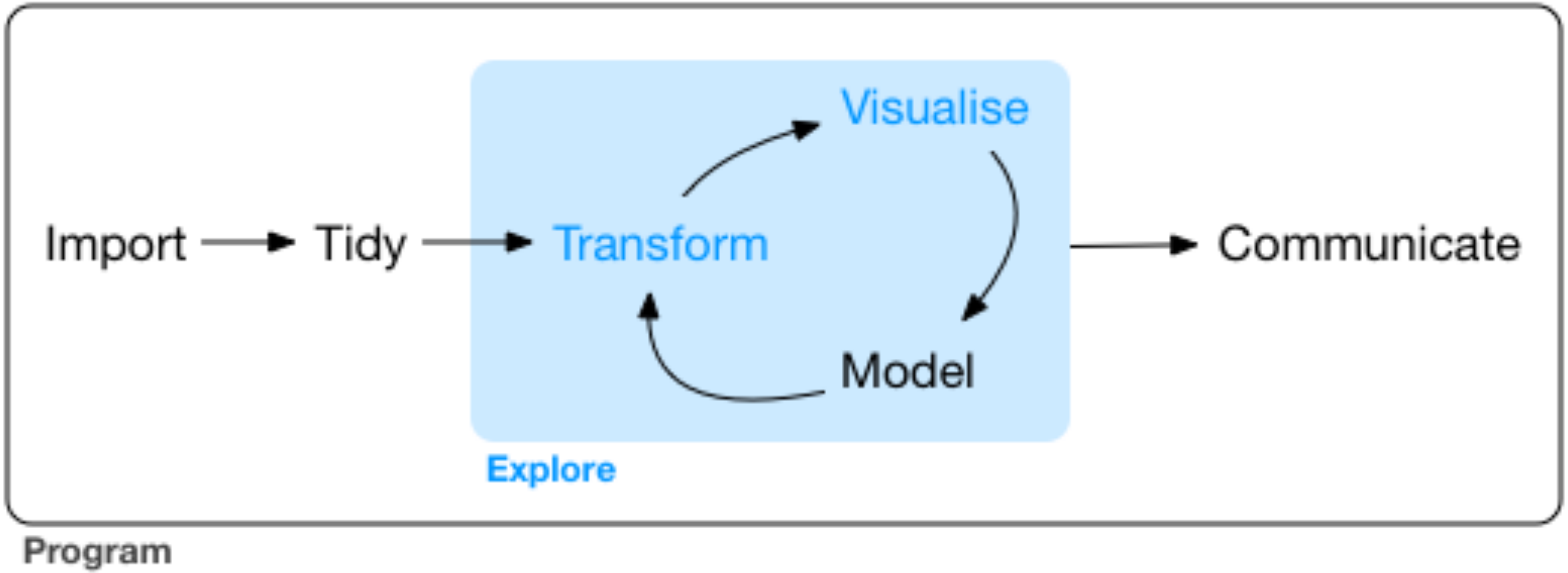
Carpeta con material

https://rstats-courses.github.io/CursoR-AEET-2025/materiales.html

Exploración de datos

La exploración de datos nos permite verificar su calidad, generar y probar hipótesis de forma rápida, identificando pistas prometedoras para analizar más a fondo luego.

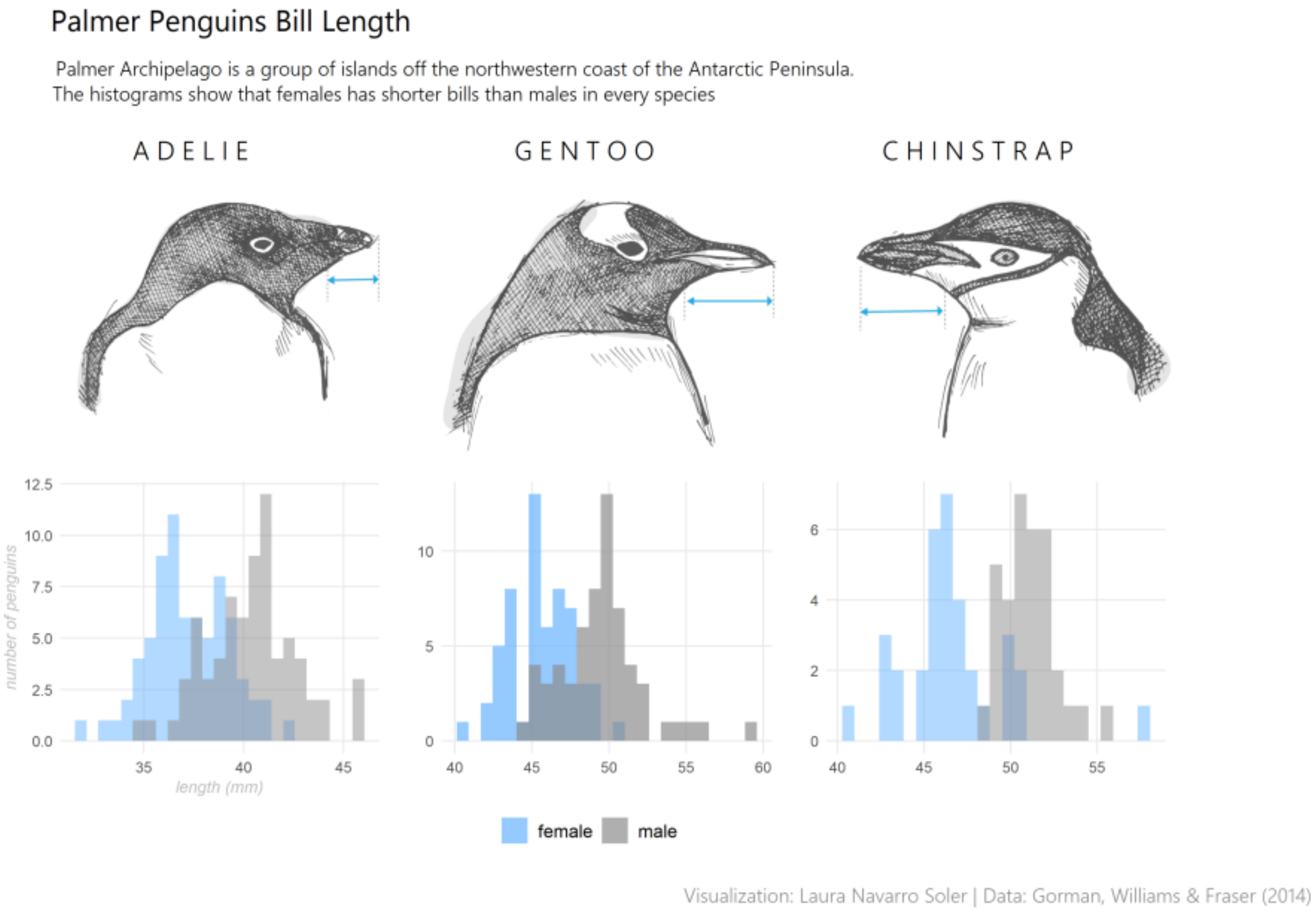
La visualización de los datos es un buen comienzo, pero por sí sola no suele ser suficiente, ya que a menudo requiere transformar los datos previamente.



https://r4ds.had.co.nz/explore-intro.html

Beneficios de usar `ggplot`

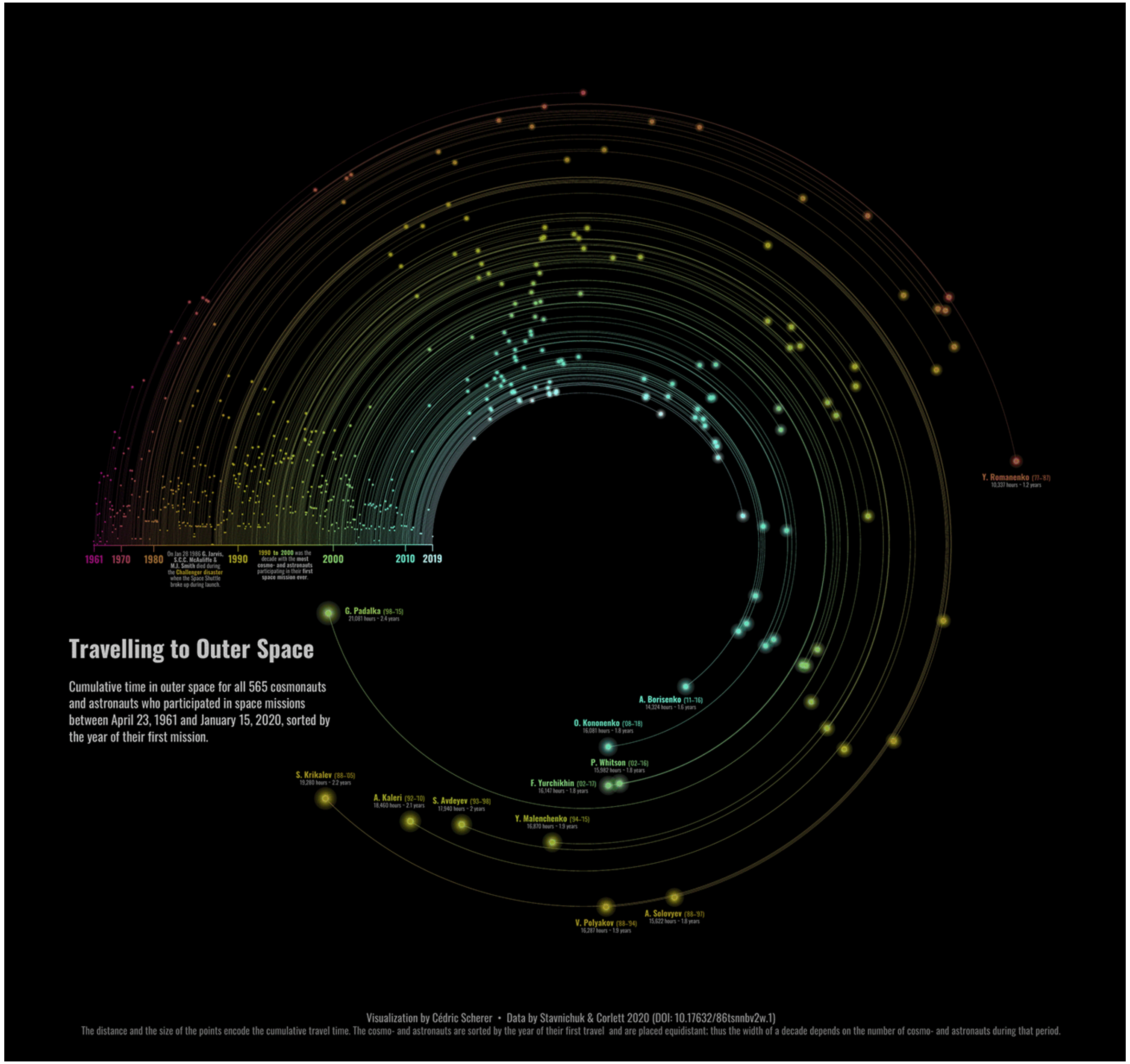
- Reproducible
- Consistencia gramática
- Muy flexible y permite controlar gran cantidad de detalles
- Fácil para uso básico
- Comunidad de usuarios activos


Palmer Penguins Bill Length

Palmer Archipelago is a group of islands off the northwestern coast of the Antarctic Peninsula.
The histograms show that females has shorter bills than males in every species

Visualization: Laura Navarro Soler | Data: Gorman, Williams & Fraser (2014)

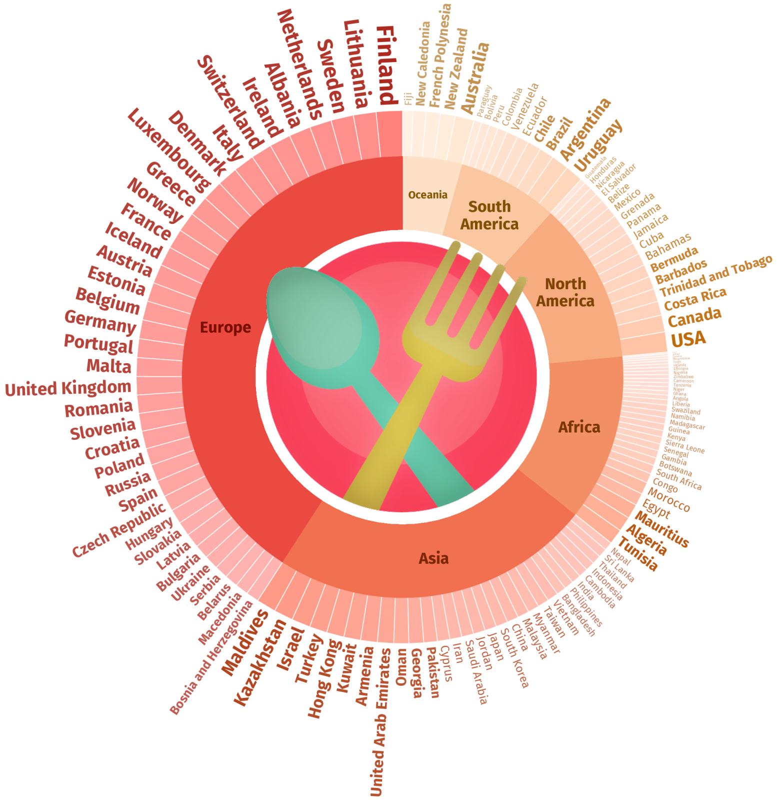
Laura Navarro | code here

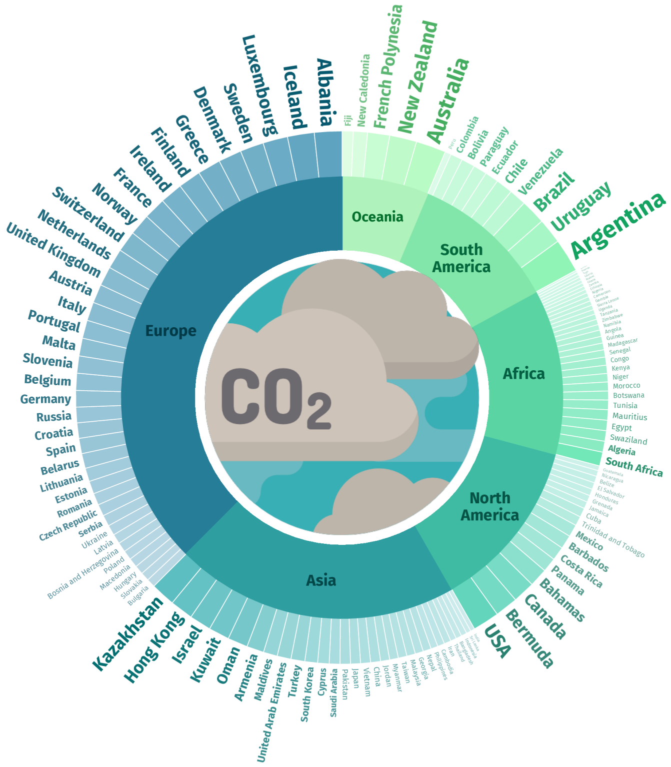


Cédric Scherer | code here

Food Carbon Footprint Index 2018

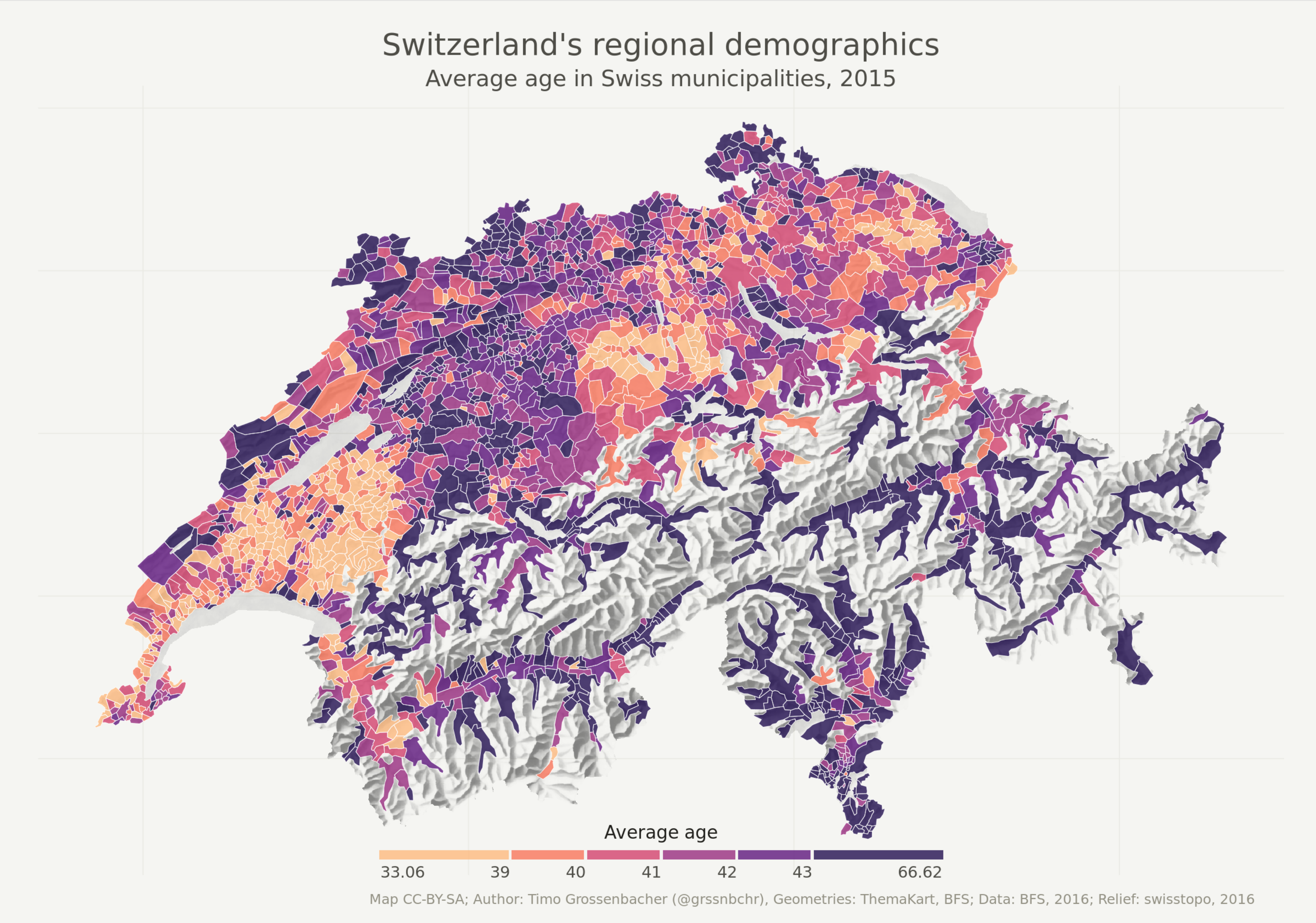
Global comparison of different diets in terms of **Average Consumption** (kg/person/year) of both animal and non-animal products as well as **Carbon Emissions** (kg CO_2/person/year) per continent and country. Font size and color intensity indicate each country's estimate with **countries printed in bold** belonging to the upper 50% of consumers and CO_2 emissioners, respectively.





Visualization by Cédric Scherer • Data by Food and Agriculture Organization of the United Nations (FAO) via nu3 • Icons by FreePik

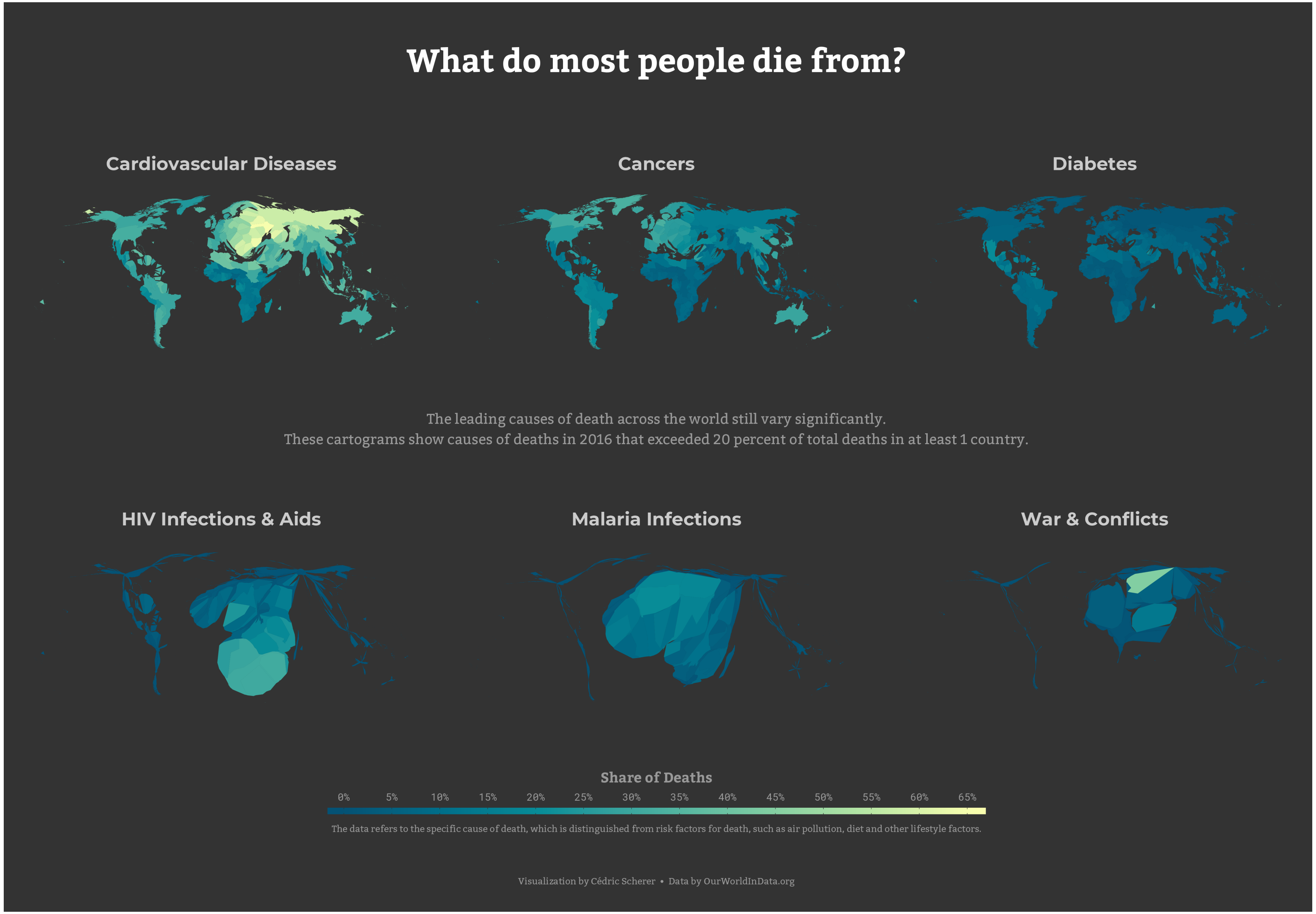
Cédric Scherer


Switzerland's regional demographics

Average age in Swiss municipalities, 2015

Map CC-BY-SA; Author: Timo Grossenbacher (@grssnbchr), Geometries: ThemaKart, BFS; Data: BFS, 2016; Relief: swisstopo, 2016

Timo Grossenbacher | code here



What do most people die from?

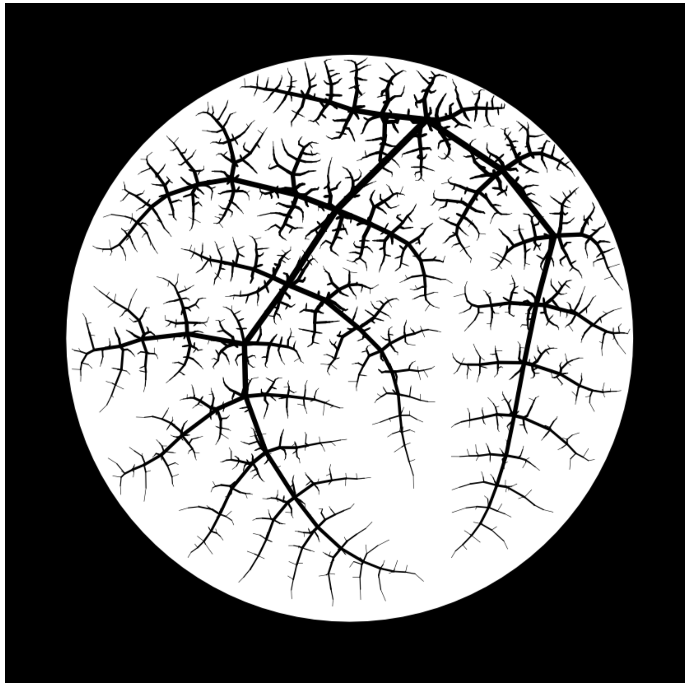
The leading causes of death across the world still vary significantly.
These cartograms show causes of deaths in 2016 that exceeded 20 percent of total deaths in at least 1 country.

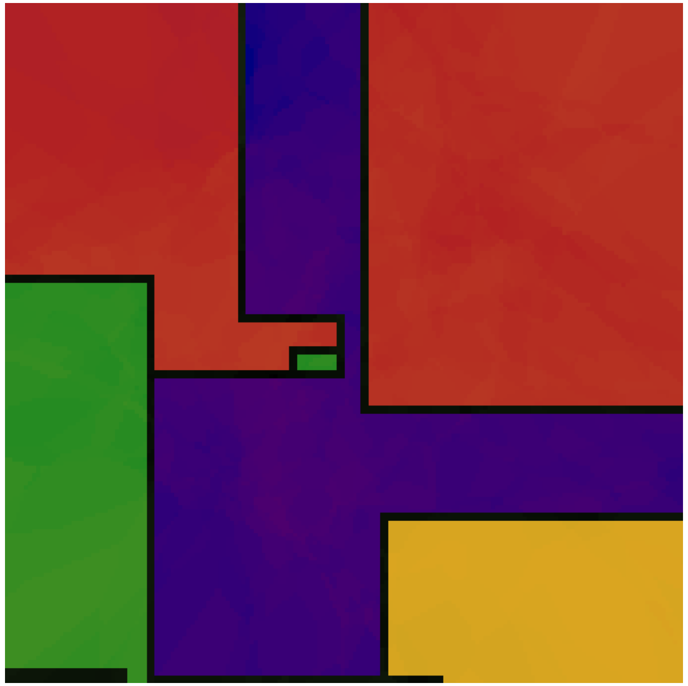
The data refers to the specific cause of death, which is distinguished from risk factors for death, such as air pollution, diet and other lifestyle factors.

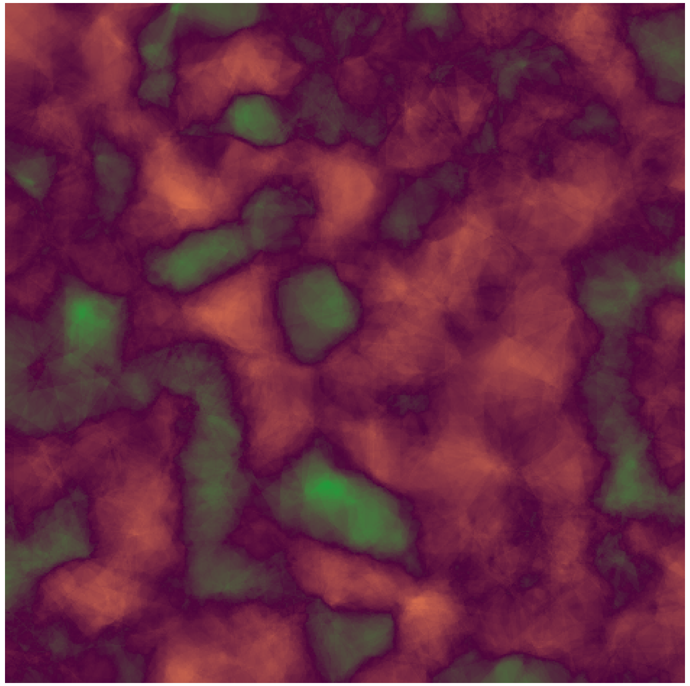
Visualization by Cédric Scherer • Data by OurWorldInData.org

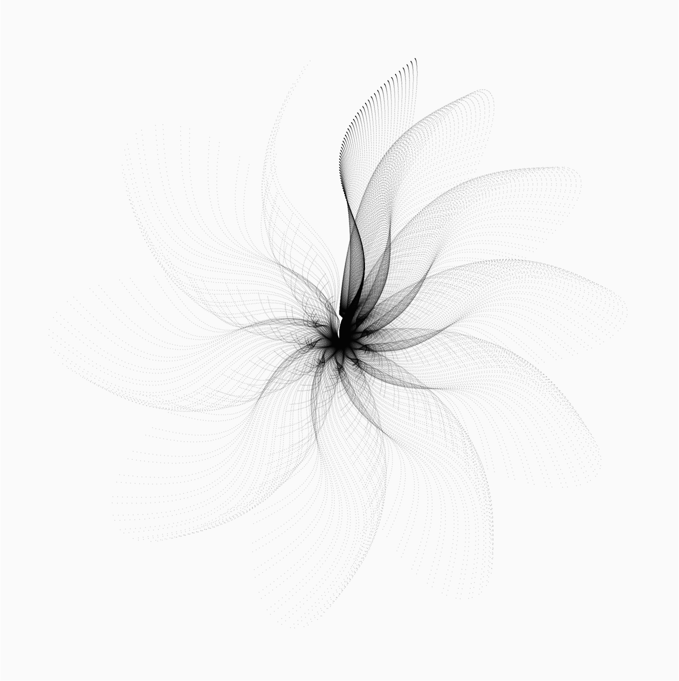
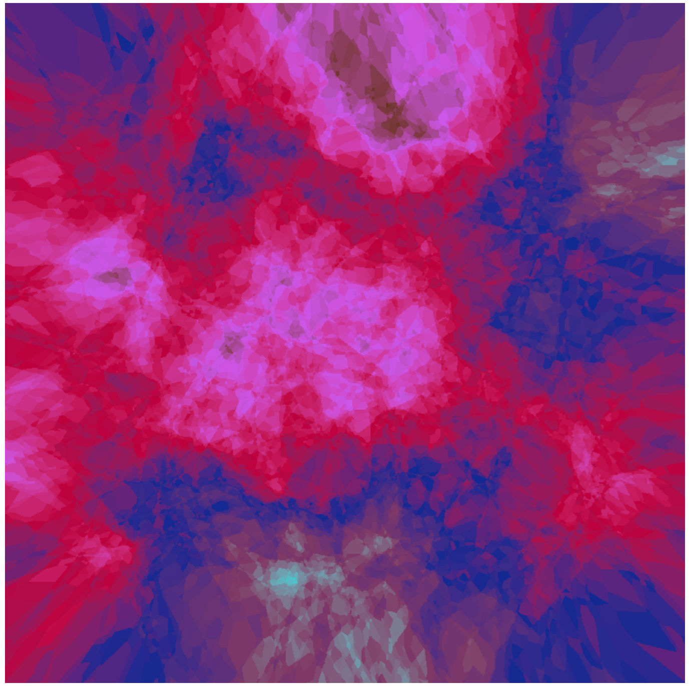
Cédric Scherer | code here

aRtsy: Generative Art with R and ggplot2







Gramática de `ggplot`

Basado en la gramática *layered grammar of graphics* - Bertin 1983, Wilkinson *et al.* 2005, Wickham 2010.

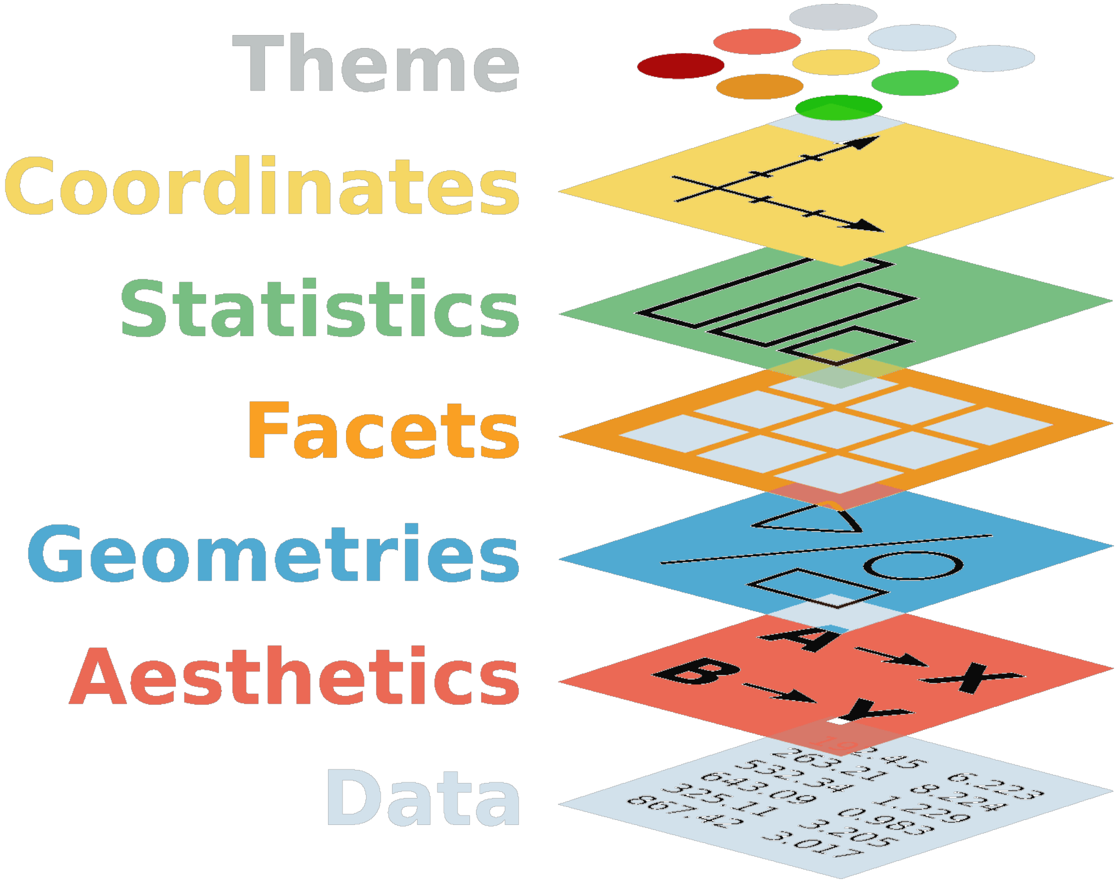


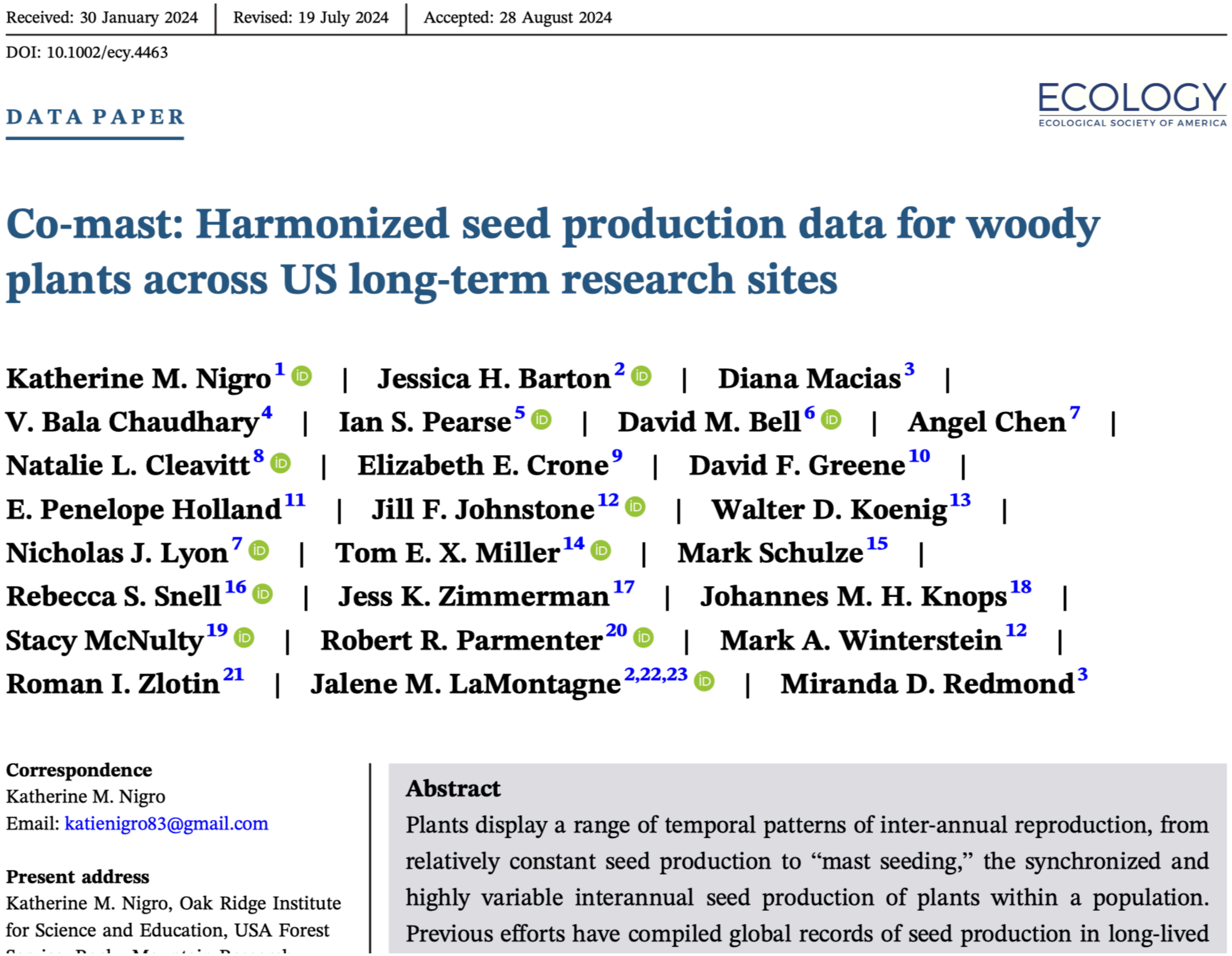
Illustration QCBS R workshop series

Capas

- **Data** - *dataset a graficar; necesita formato tidy data*
- **Aesthetics** - *describe cómo se asignan las variables del dataset (y sus escalas) a las propiedades visuales* (x, y, size, colour, fill, group, sise, shape. . .)
- **Geometries** - *determina la forma de representar los datos* (geoms: points, lines, bars, boxplot. . .)
- Scales - *maneja las escalas de los aesthetics* (x & y format, colors continuous or discrete, sizes, shapes. . .)
- Facets - *crea subplots* (facet_wrap or facet_grid. . .)
- Themes - *apariencia general del gráfico, no ligada a los datos* (title, x.axis.text, legend. . .)
- Statistics - *resume los datos con estadísticos. Muchas veces ya va implícito en el "geom"* (stats: count, density, bins, means, density. . .)
- Coordinate system - *determina el sistema de coordenadas a usar en los ejes* (cartesian, polar, map projections. . .)

*Capas necesarias

Dataset



Received: 30 January 2024 | Revised: 19 July 2024 | Accepted: 28 August 2024

DOI: 10.1002/ecy.4463

DATA PAPER

ECOLOGY
ECOLOGICAL SOCIETY OF AMERICA

Co-mast: Harmonized seed production data for woody plants across US long-term research sites

Katherine M. Nigro[1] | Jessica H. Barton[2] | Diana Macias[3] | V. Bala Chaudhary[4] | Ian S. Pearse[5] | David M. Bell[6] | Angel Chen[7] | Natalie L. Cleavitt[8] | Elizabeth E. Crone[9] | David F. Greene[10] | E. Penelope Holland[11] | Jill F. Johnstone[12] | Walter D. Koenig[13] | Nicholas J. Lyon[7] | Tom E. X. Miller[14] | Mark Schulze[15] | Rebecca S. Snell[16] | Jess K. Zimmerman[17] | Johannes M. H. Knops[18] | Stacy McNulty[19] | Robert R. Parmenter[20] | Mark A. Winterstein[12] | Roman I. Zlotin[21] | Jalene M. LaMontagne[2,22,23] | Miranda D. Redmond[3]

Correspondence
Katherine M. Nigro
Email: katienigro83@gmail.com

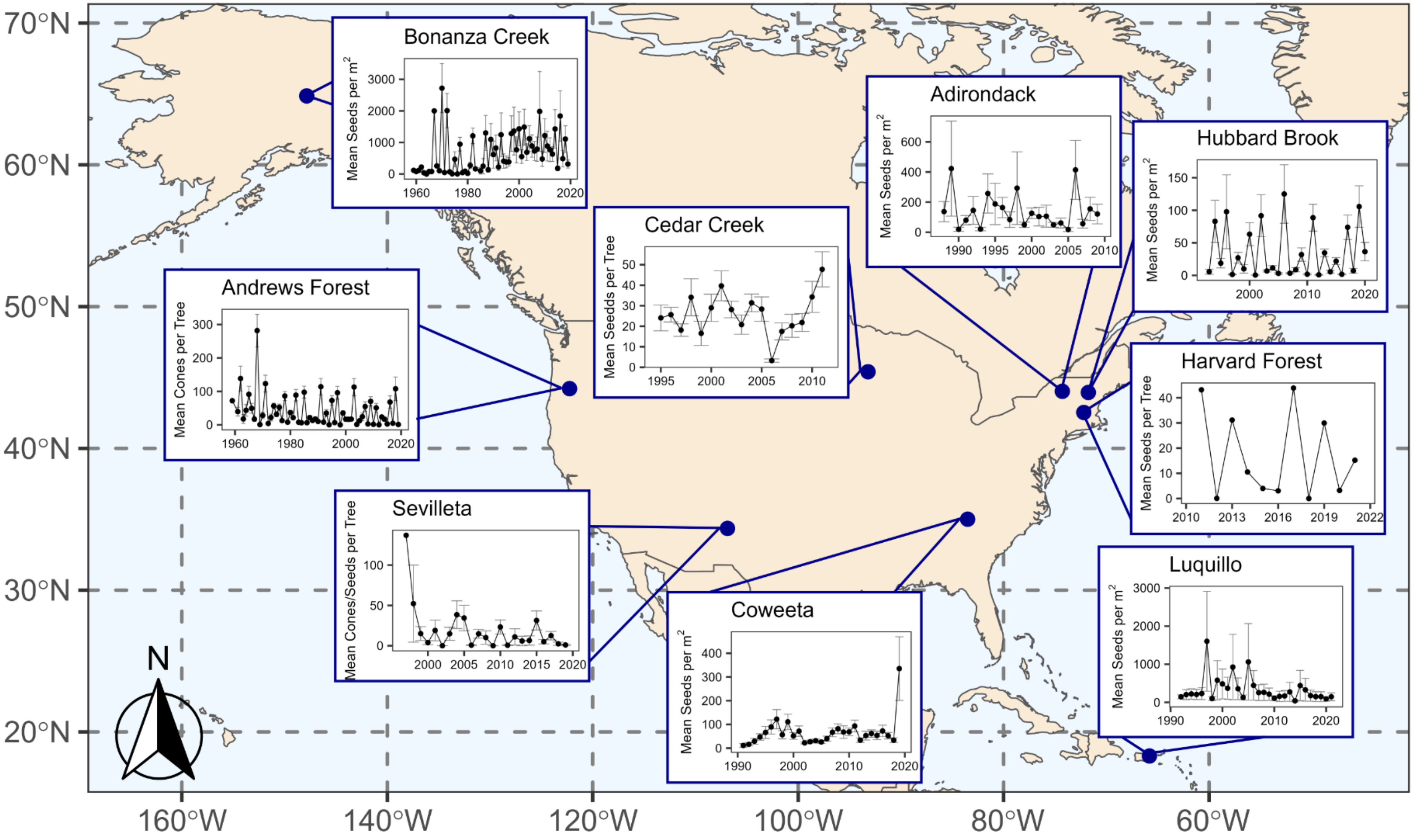
Present address
Katherine M. Nigro, Oak Ridge Institute for Science and Education, USA Forest Service, Rocky Mountain Research

Abstract

Plants display a range of temporal patterns of inter-annual reproduction, from relatively constant seed production to "mast seeding," the synchronized and highly variable interannual seed production of plants within a population. Previous efforts have compiled global records of seed production in long-lived

https://doi.org/10.1002/ecy.4463

Dataset



70°N
60°N
50°N
40°N
30°N
20°N
160°W
140°W
120°W
100°W
80°W
60°W
N
Bonanza Creek
Mean Seeds per m^2
3000
2000
1000
0
1960
1980
2000
2020
Andrews Forest
Mean Cones per Tree
300
200
100
0
1960
1980
2000
2020
Cedar Creek
Mean Seedds per Tree
50
40
30
20
10
0
1995
2000
2005
2010
Adirondack
Mean Seeds per m^2
600
400
200
0
1990
1995
2000
2005
2010
Hubbard Brook
Mean Seeds per m^2
150
100
50
0
2000
2010
2020
Harvard Forest
Mean Seeds per Tree
40
30
20
10
0
2010
2013
2016
2019
2022
Sevilleta
Mean Cones/Seeds per Tree
100
50
0
2000
2005
2010
2015
2020
Coweeta
Mean Seeds per m^2
400
300
200
100
0
1990
2000
2010
2020
Luquillo
Mean Seeds per m^2
3000
2000
1000
0
1990
2000
2010
2020

Cargar paquetes

```r
library(here)
library(tidyverse)
library(tidylog)
```

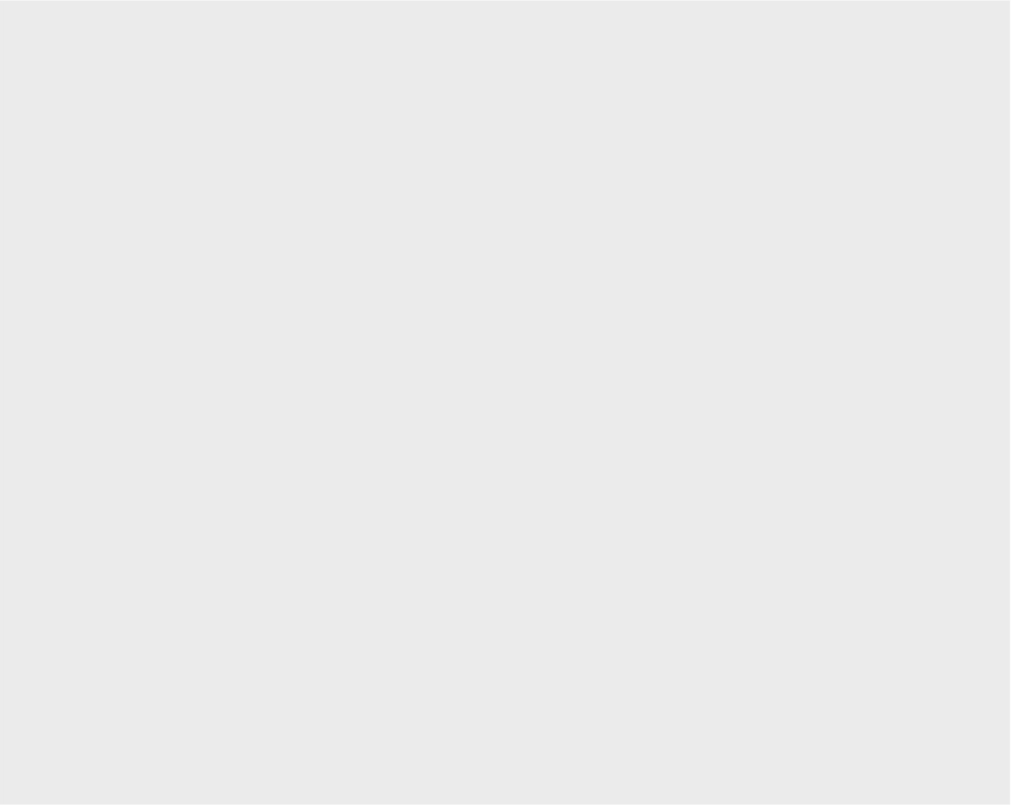
Leer datos

```r
dt <- read_csv(here("data/clean_data.csv"))
```

Graficar

1. Indicar el dataset

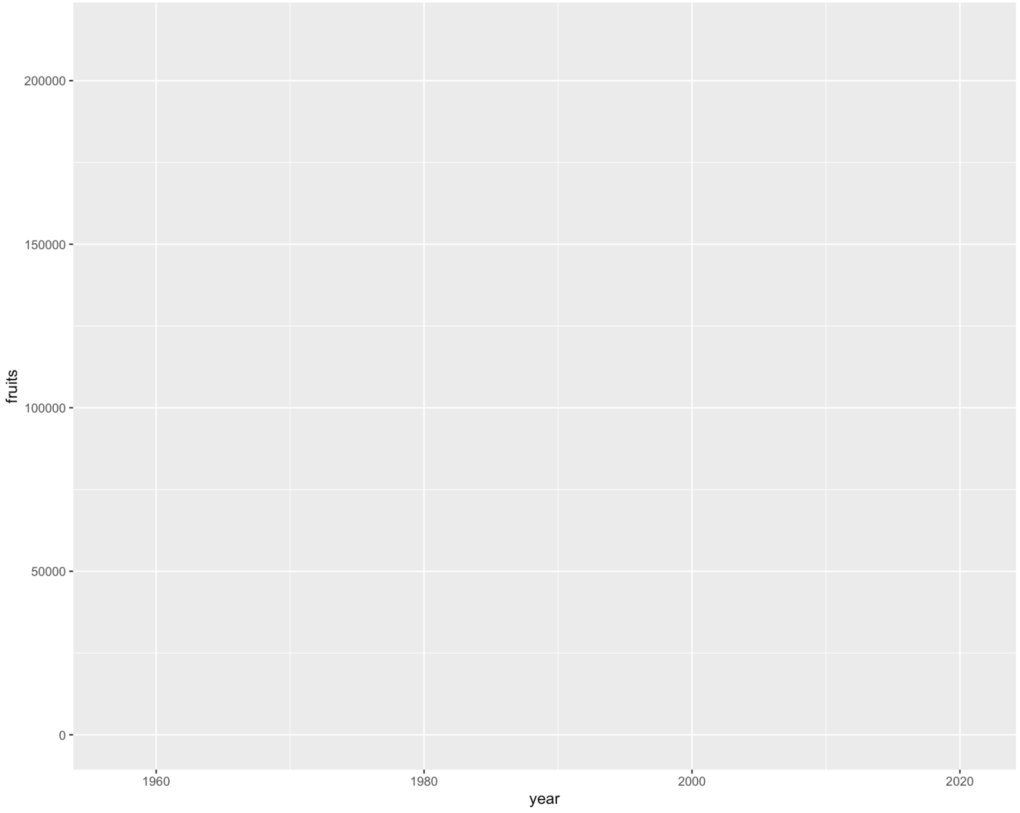
```
ggplot(data = dt)
```



Graficar

2. Definir los aesthetics (variables a visualizar)

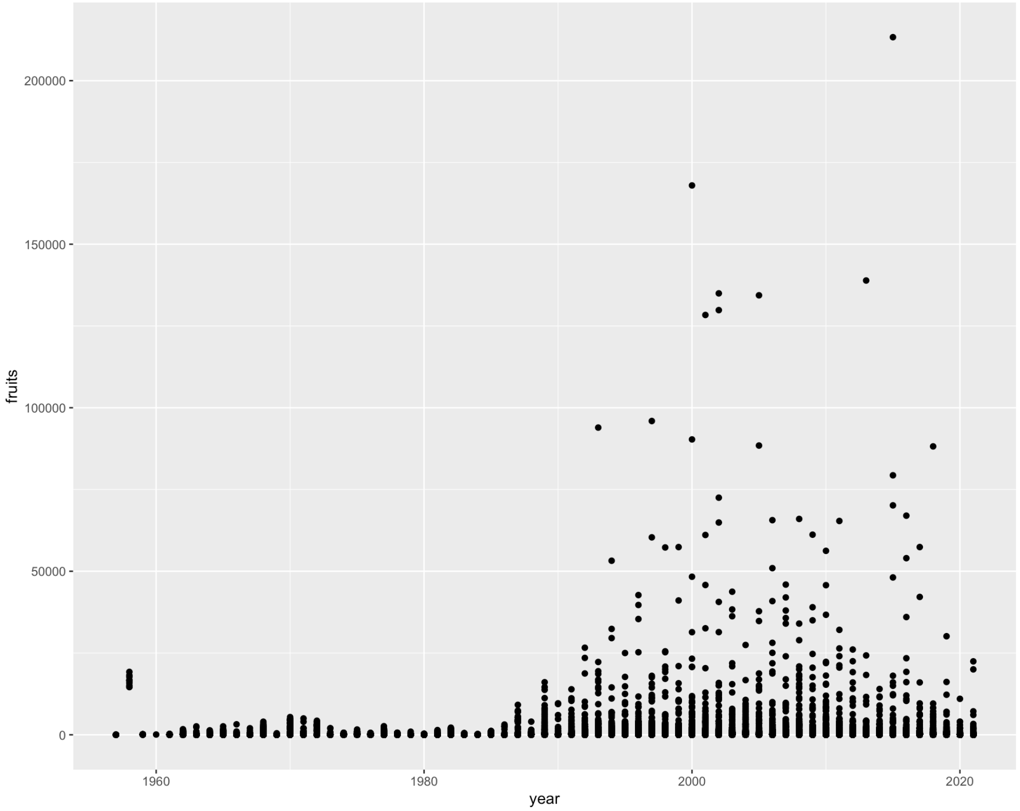
```
ggplot(data = dt,
       aes(x = year, y = fruits))
```



Graficar

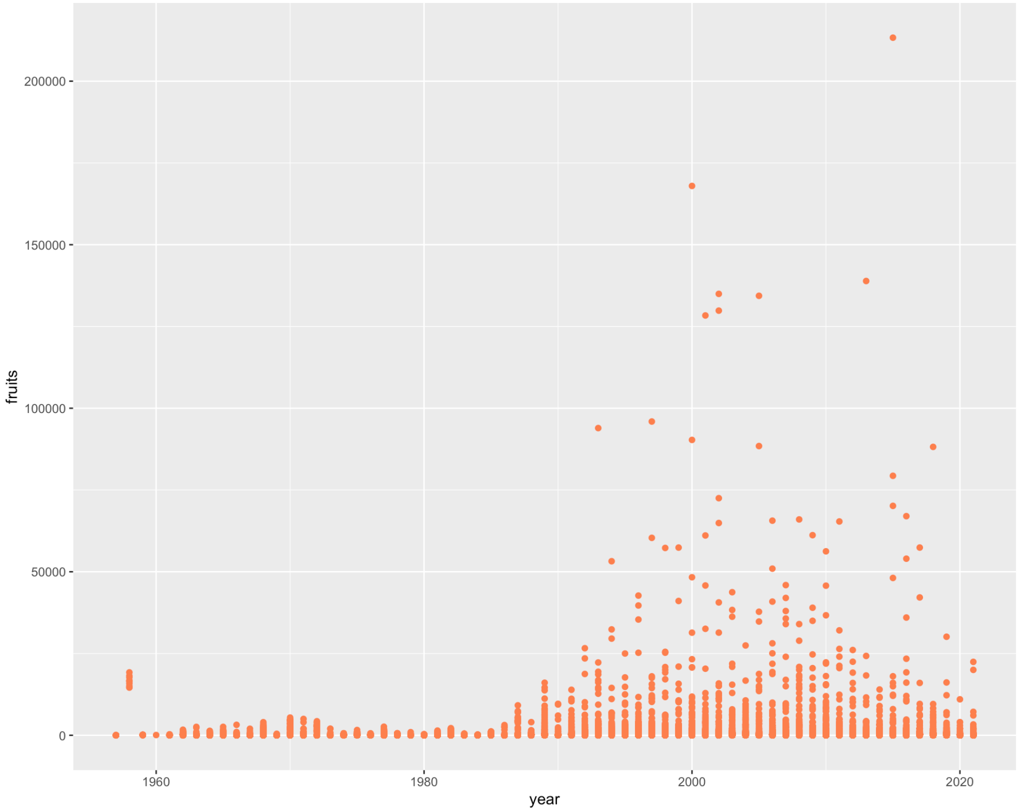
3. Determinar la forma de representar los datos con GEOMs

```r
ggplot(data = dt,
       aes(x = year, y = fruits)) +
  geom_point()
```



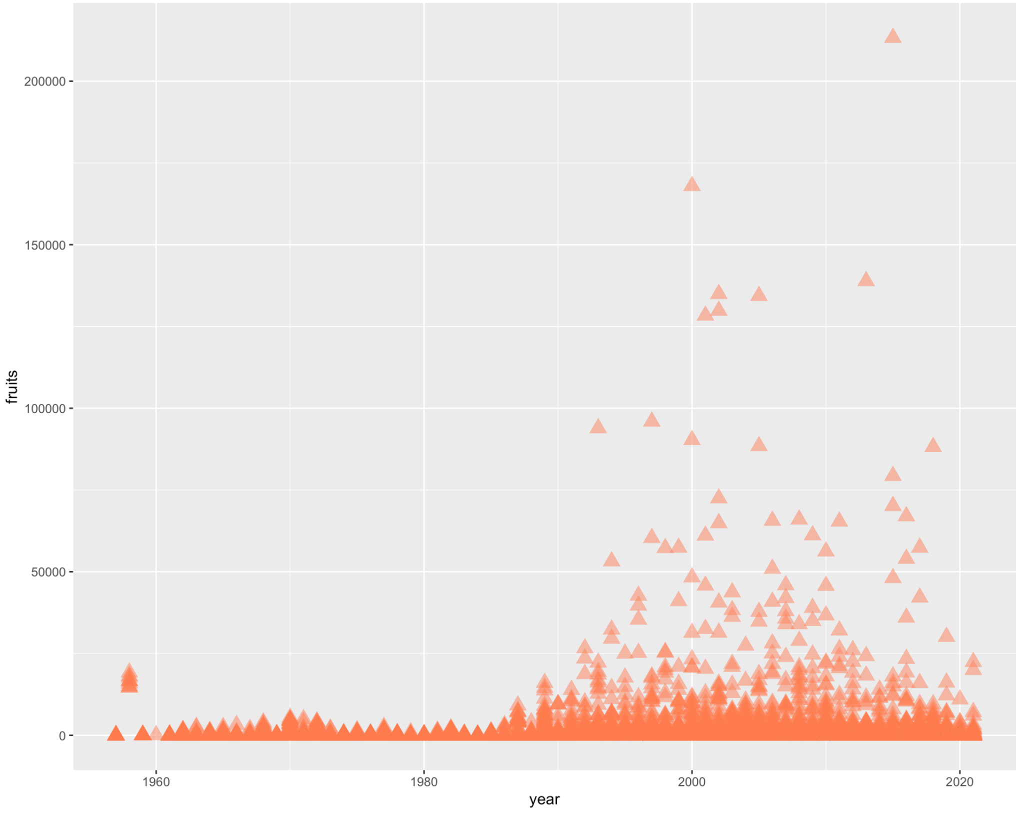
Ajustar detalles de visualizacion

```
ggplot(dt,
       aes(x = year, y = fruits)) +
  geom_point(color = "coral")
```

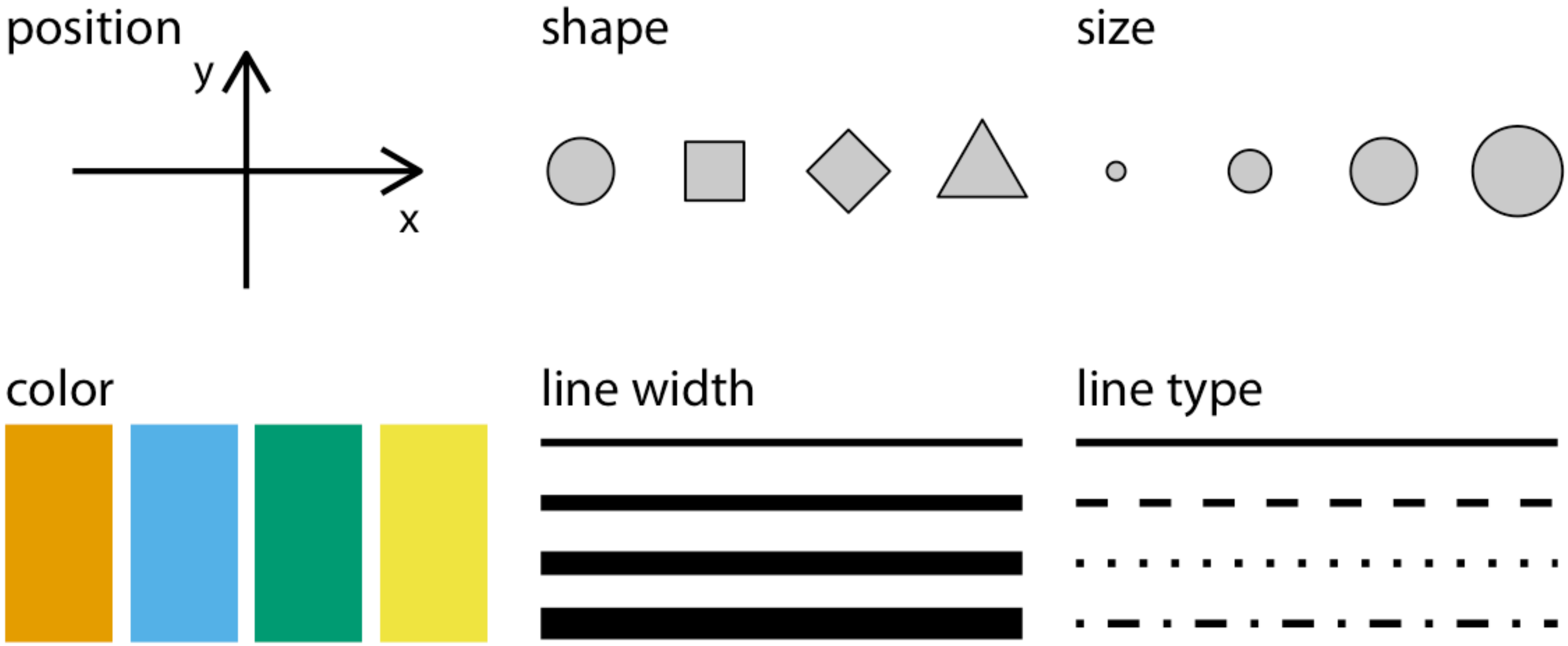


Ajustar detalles de visualizacion

```
ggplot(dt,
       aes(x = year, y = fruits)) +
  geom_point(color = "coral",
             shape = "triangle",
             size = 4,
             alpha = 0.5)
```

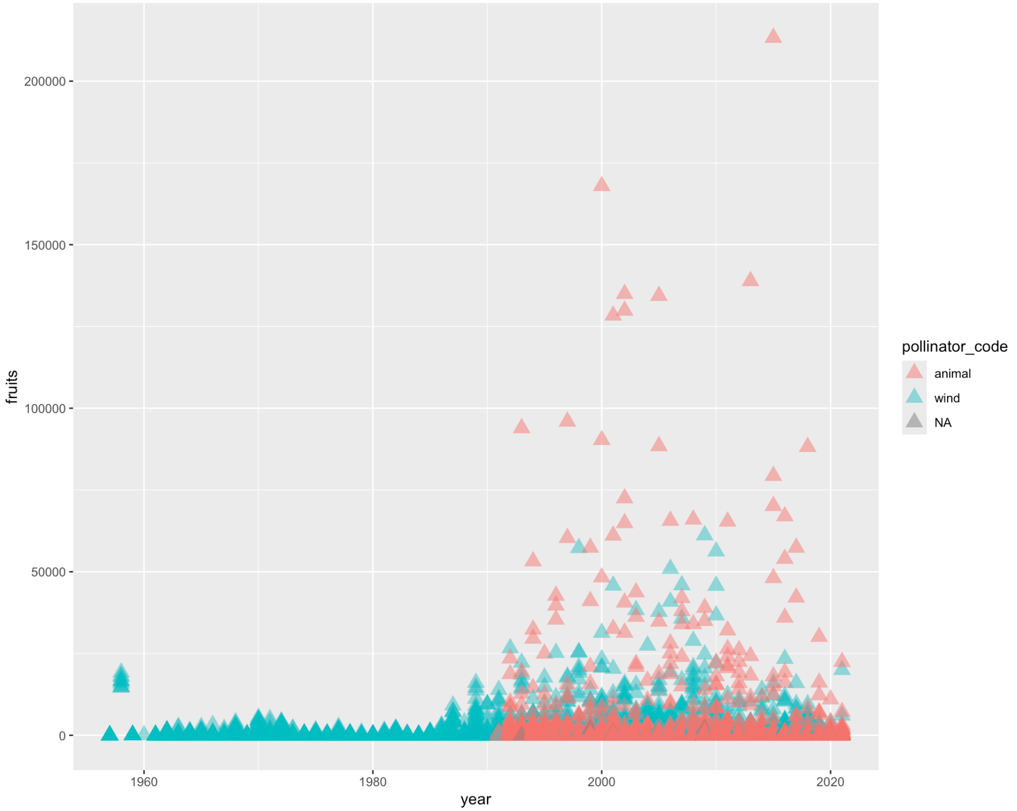


Valores de los aesthetics (aes):



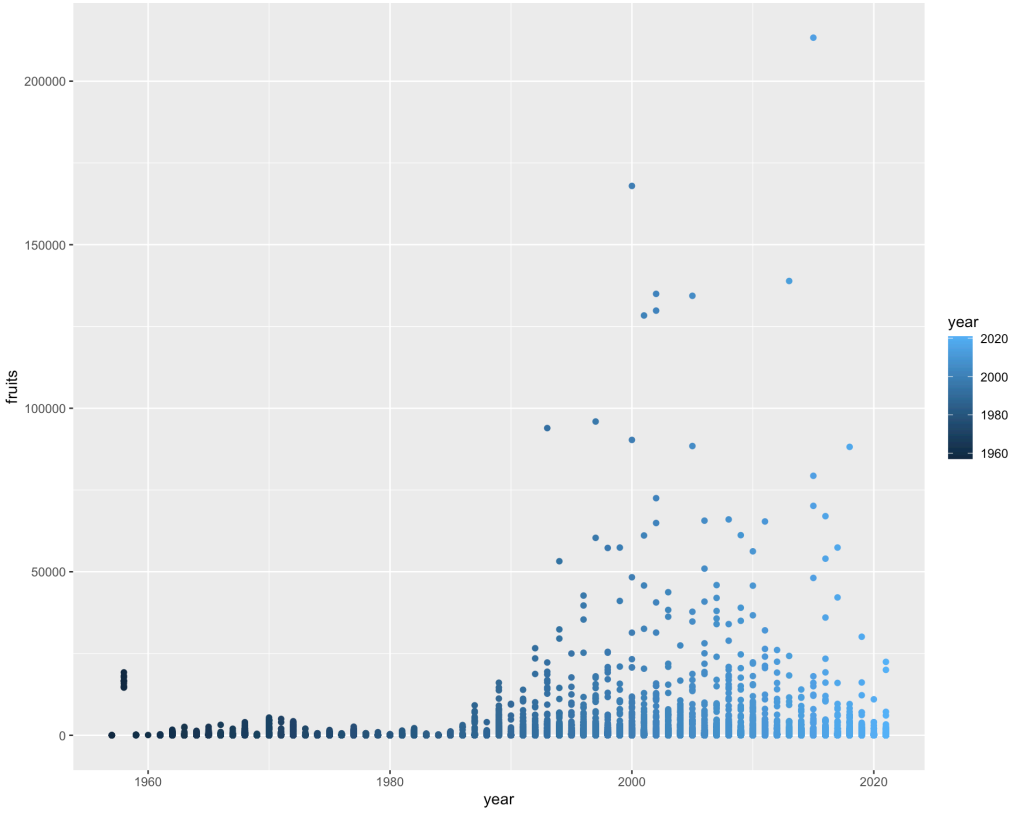
Ajustar detalles de visualizacion

```
ggplot(dt,
       aes(x = year, y = fruits,
           color = pollinator_code)) +
  geom_point(shape = "triangle",
             size = 4,
             alpha = 0.5)
```



Ajustar detalles de visualizacion

```
ggplot(dt,
       aes(x = year, y = fruits,
           color = year)) +
  geom_point()
```



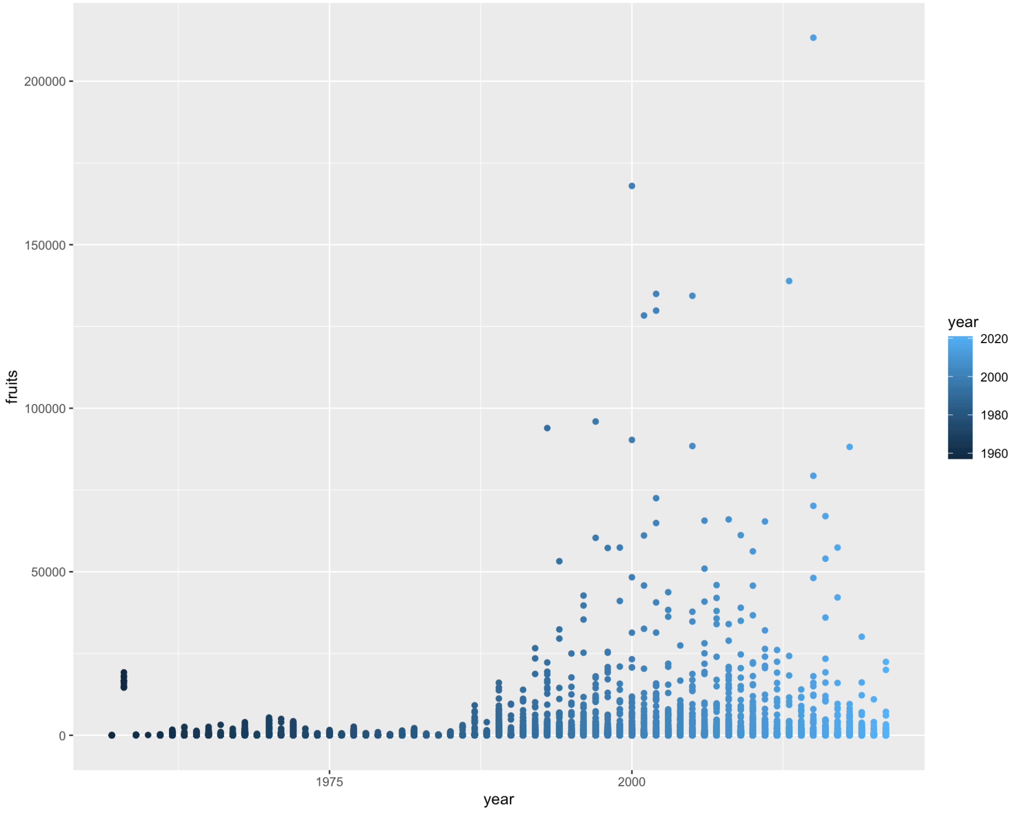
Escalas

Van refereridas a las variables o aethetics (aes):

- X e Y
- Tamaño de punto (size), de linea (linewidth)
- Forma de punto (shape) y de linea (linetype)
- Color y relleno (colour, fill)

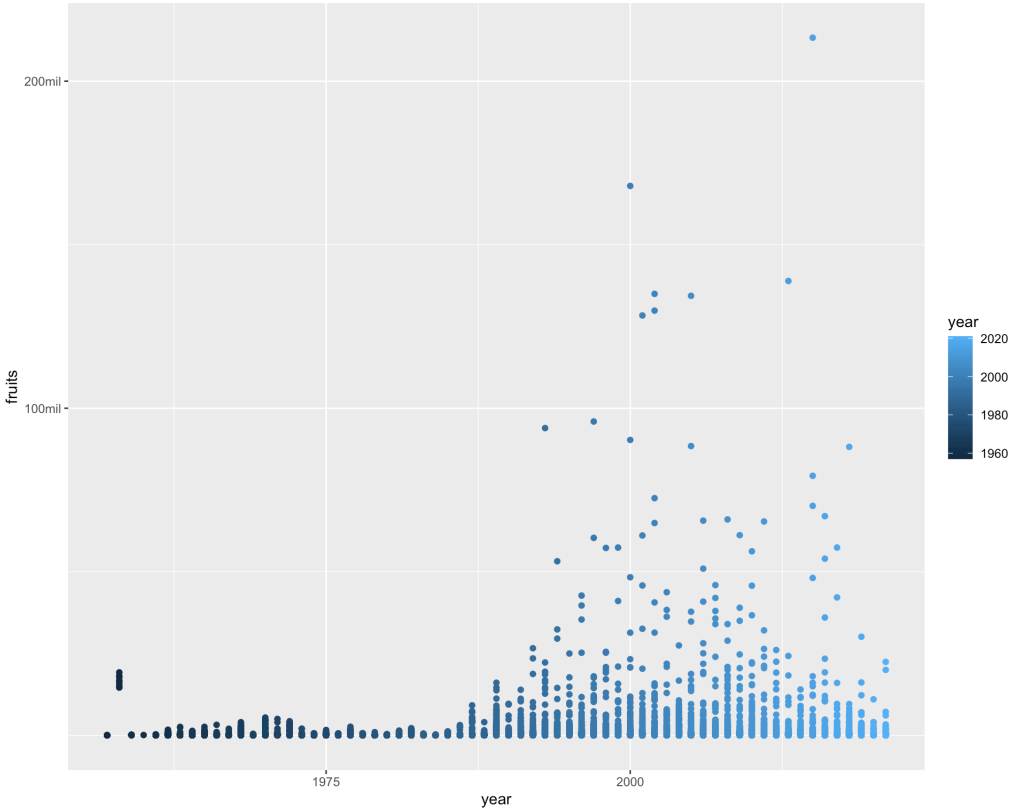
Escalas - X, Y

```
ggplot(dt,
       aes(x = year, y = fruits,
           color = year)) +
  geom_point() +
  scale_x_continuous(n.breaks = 3)
```



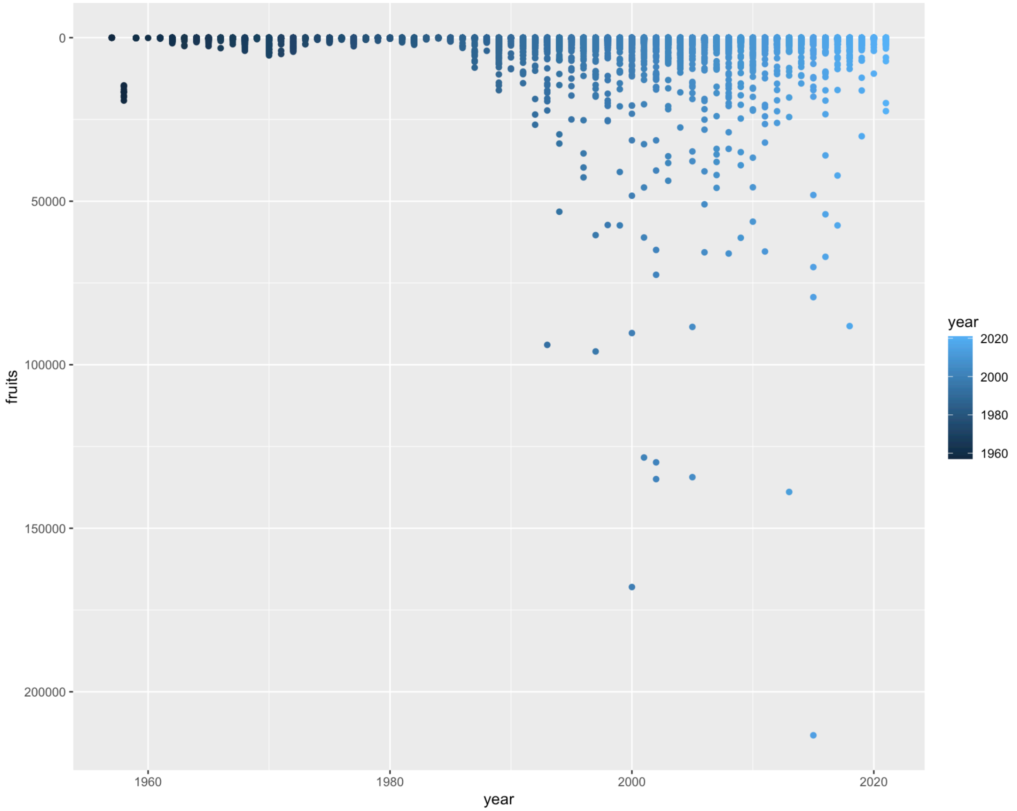
Escalas - X, Y

```
ggplot(dt,
       aes(x = year, y = fruits,
           color = year)) +
  geom_point() +
  scale_x_continuous(n.breaks = 3) +
  scale_y_continuous(
          breaks = c(100000, 200000),
          labels = c("100mil",
                     "200mil"))
```



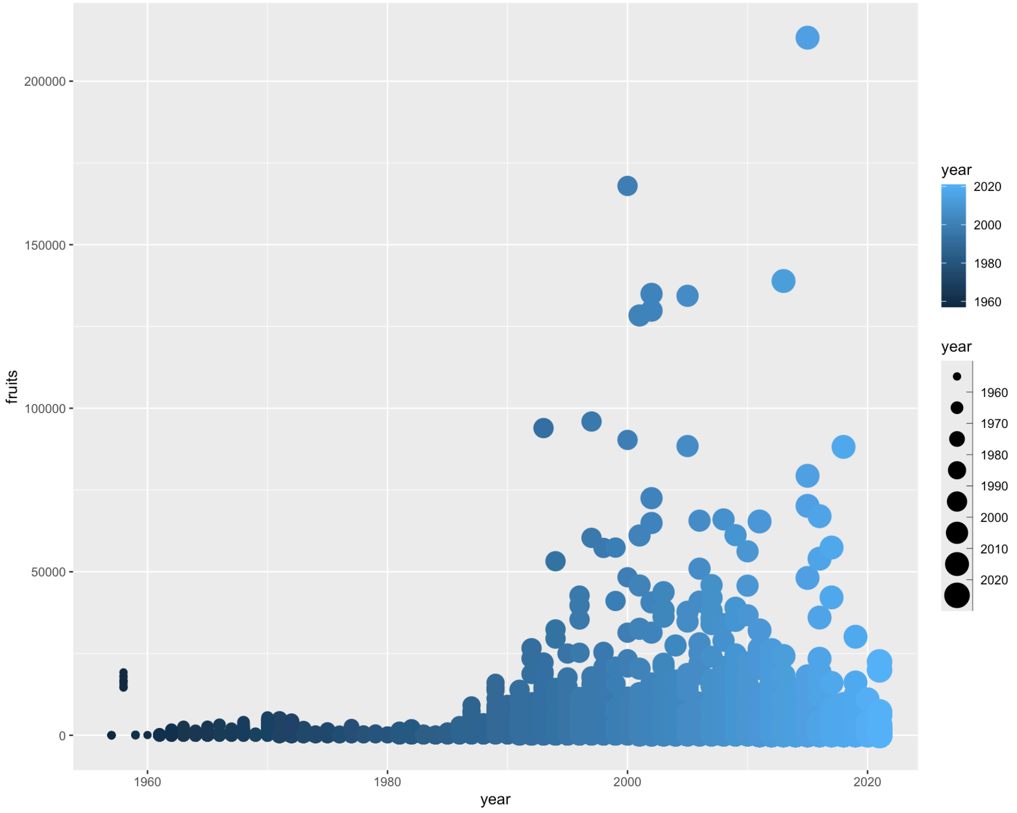
Escalas - X, Y

```
ggplot(dt,
       aes(x = year, y = fruits,
           color = year)) +
  geom_point() +
  scale_y_reverse()
```



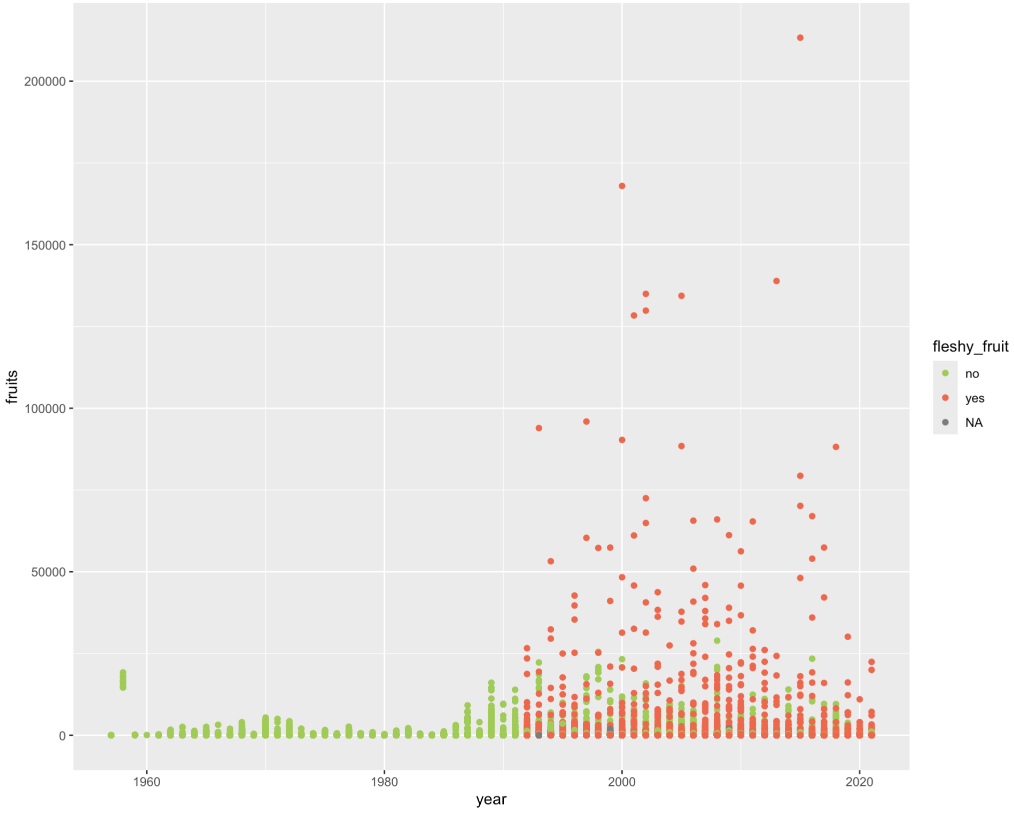
Escalas - Tamaño

```r
ggplot(dt,
       aes(x = year, y = fruits,
           color = year,
           size = year)) +
  geom_point() +
  scale_size_binned(range = c(0.1, 8),
                    n.breaks = 10)
```



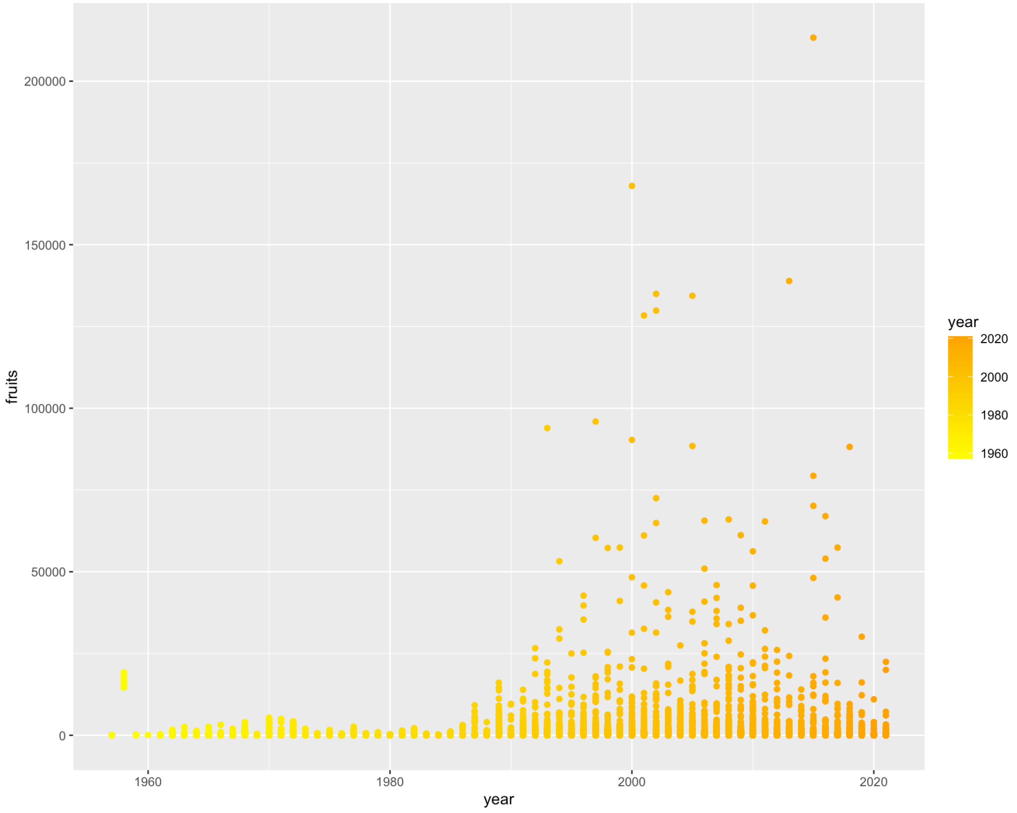
Escalas - Color (discreto)

```
ggplot(dt,
       aes(x = year, y = fruits,
           color = fleshy_fruit)) +
  geom_point() +
  scale_color_manual(
      values = c("darkolivegreen3",
                 "coral2"))
```



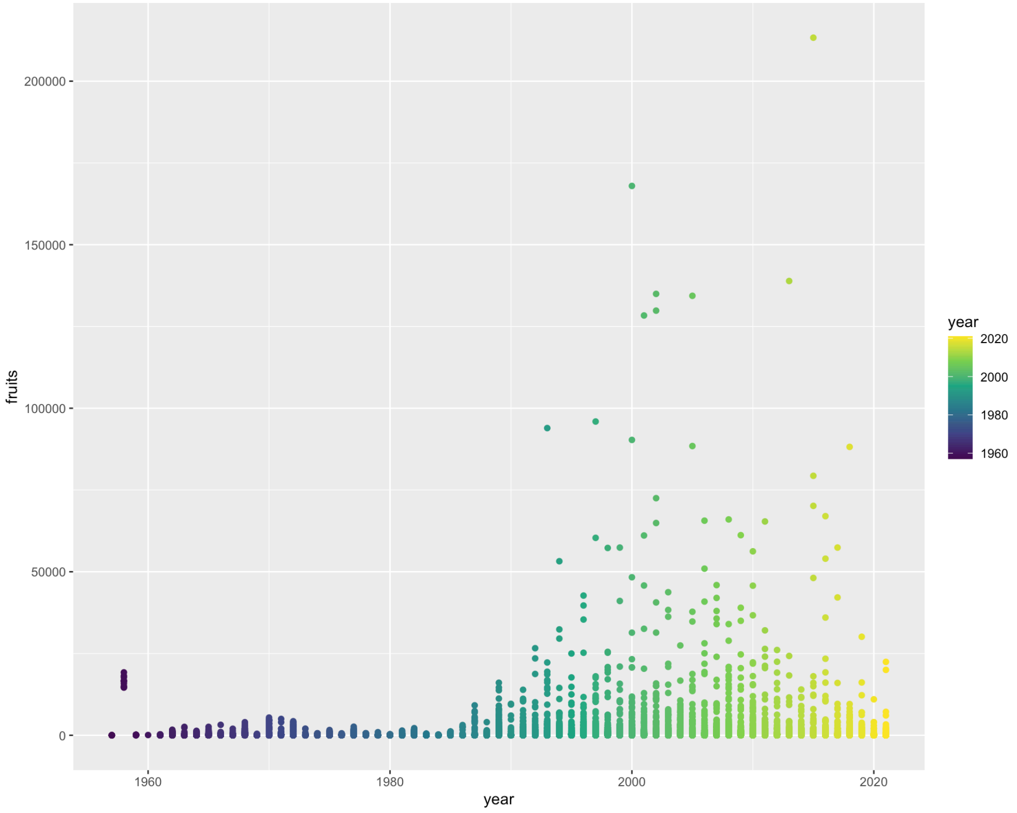
Escalas - Color (continuo)

```r
ggplot(dt,
       aes(x = year, y = fruits,
           color = year)) +
  geom_point() +
  scale_color_gradient(
    low = "yellow", high = "orange")
```



Escalas - Color

```
ggplot(dt,
       aes(x = year, y = fruits,
           color = year)) +
  geom_point() +
  scale_color_viridis_c()
```



Escalas - Color

Paletas de color:

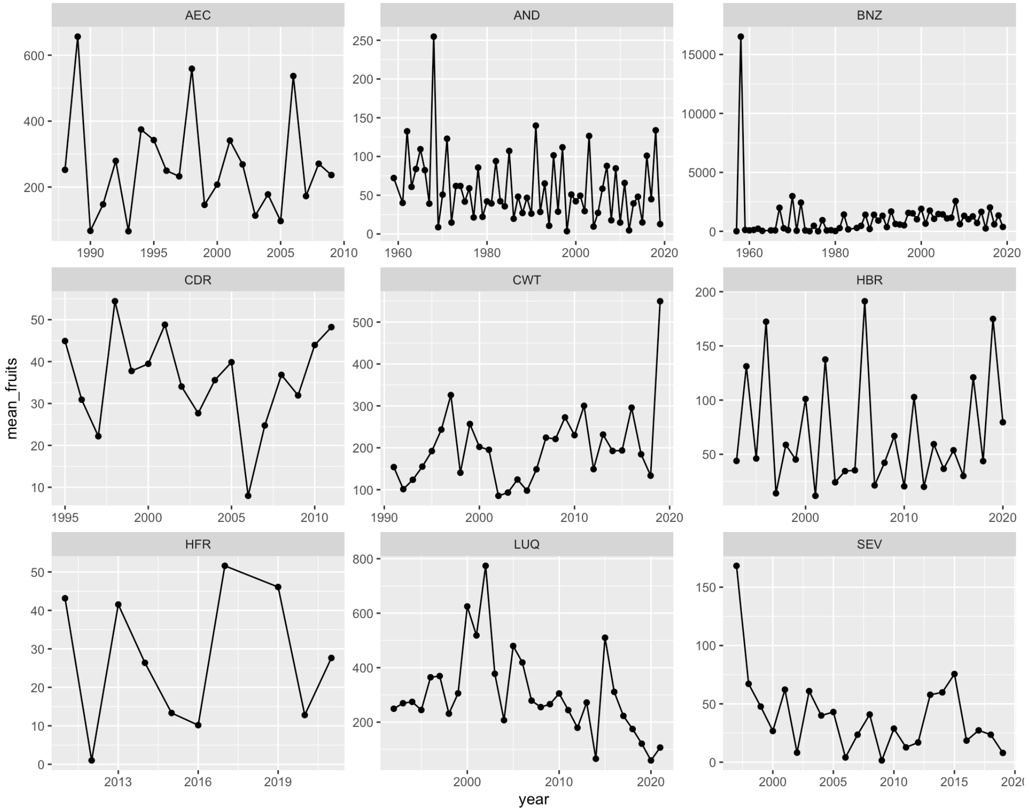
- https://r-charts.com/color-palettes/
- `library(paletter)` - https://github.com/EmilHvitfeldt/paletteer
- https://r-graph-gallery.com/color-palette-finder
- https://medialab.github.io/iwanthue/
- https://projects.susielu.com/viz-palette
- https://colorbrewer2.org/#type=sequential&scheme=YlGnBu&n=3

Facets

- facet_wrap()
- facet_grid()

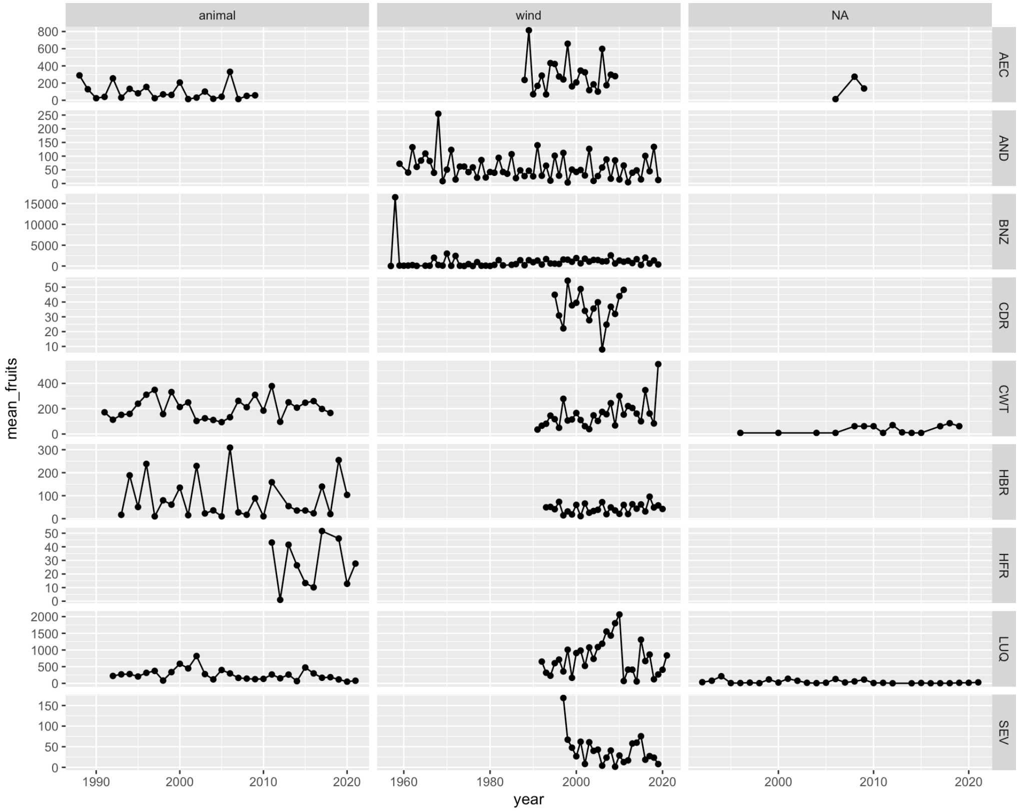
Facets

```
dt |>
  group_by(year, site) |>
  summarise(mean_fruits = mean(fruits,
                               na.rm = TRUE)) |>
  ggplot(aes(x = year,
             y = mean_fruits)) +
  geom_point() +
  geom_line() +
  facet_wrap(~site, scales = "free")
```



Facets

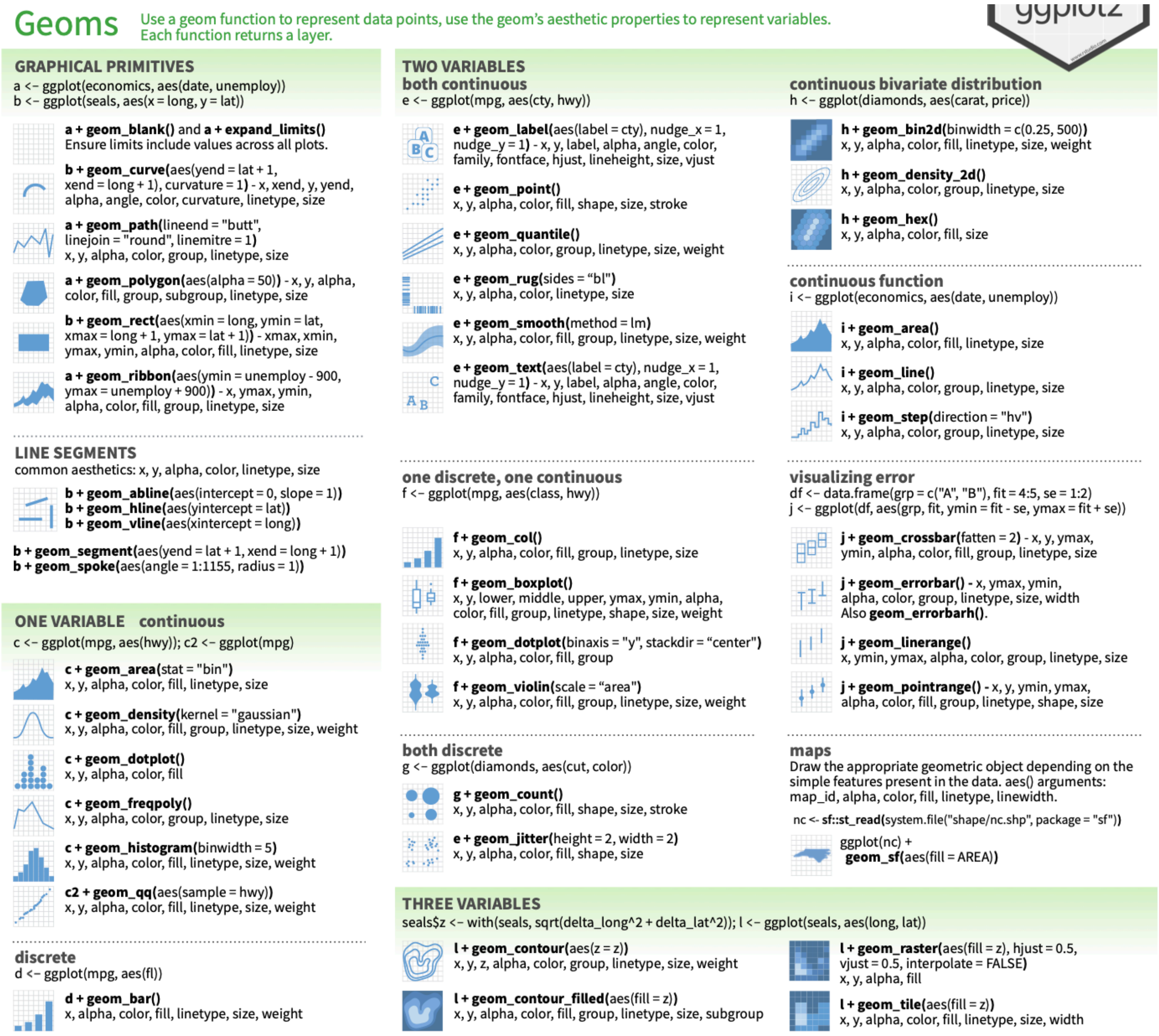
```
dt |>
  group_by(year, site, pollinator_code)
  summarise(mean_fruits = mean(fruits,
                  na.rm = TRUE)) |>
  ggplot(aes(x = year,
             y = mean_fruits)) +
  geom_point() +
  geom_line() +
  facet_grid(site~pollinator_code,
             scales = "free")
```



Tipos de Geoms

```
 [1] "geom_abline"            "geom_area"              "geom_bar"
 [4] "geom_bin_2d"            "geom_bin2d"             "geom_blank"
 [7] "geom_boxplot"           "geom_col"               "geom_contour"
[10] "geom_contour_filled"    "geom_count"             "geom_crossbar"
[13] "geom_curve"             "geom_density"           "geom_density_2d"
[16] "geom_density_2d_filled" "geom_density2d"         "geom_density2d_filled"
[19] "geom_dotplot"           "geom_errorbar"          "geom_errorbarh"
[22] "geom_freqpoly"          "geom_function"          "geom_hex"
[25] "geom_histogram"         "geom_hline"             "geom_jitter"
[28] "geom_label"             "geom_line"              "geom_linerange"
[31] "geom_map"               "geom_path"              "geom_point"
[34] "geom_pointrange"        "geom_polygon"           "geom_qq"
[37] "geom_qq_line"           "geom_quantile"          "geom_raster"
```

Tipos de Geoms



Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

a + geom_blank() and **a + expand_limits()**
Ensure limits include values across all plots.

b + geom_curve(aes(yend = lat + 1, xend = long + 1), curvature = 1**)** - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

a + geom_path(lineend = "butt", linejoin = "round", linemitre = 1**)**
x, y, alpha, color, group, linetype, size

a + geom_polygon(aes(alpha = 50)**)** - x, y, alpha, color, fill, group, subgroup, linetype, size

b + geom_rect(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1)**)** - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

a + geom_ribbon(aes(ymin = unemploy - 900, ymax = unemploy + 900)**)** - x, ymax, ymin, alpha, color, fill, group, linetype, size

LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

b + geom_abline(aes(intercept = 0, slope = 1)**)**
b + geom_hline(aes(yintercept = lat)**)**
b + geom_vline(aes(xintercept = long)**)**

b + geom_segment(aes(yend = lat + 1, xend = long + 1)**)**
b + geom_spoke(aes(angle = 1:1155, radius = 1)**)**

ONE VARIABLE continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + geom_area(stat = "bin"**)**
x, y, alpha, color, fill, linetype, size

c + geom_density(kernel = "gaussian"**)**
x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot()
x, y, alpha, color, fill

c + geom_freqpoly()
x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5**)**
x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy)**)**
x, y, alpha, color, fill, linetype, size, weight

discrete

d <- ggplot(mpg, aes(fl))

d + geom_bar()
x, alpha, color, fill, linetype, size, weight

TWO VARIABLES

both continuous

e <- ggplot(mpg, aes(cty, hwy))

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1**)** - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_point()
x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile()
x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl"**)**
x, y, alpha, color, linetype, size

e + geom_smooth(method = lm**)**
x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1**)** - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

one discrete, one continuous

f <- ggplot(mpg, aes(class, hwy))

f + geom_col()
x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot()
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = "center"**)**
x, y, alpha, color, fill, group

f + geom_violin(scale = "area"**)**
x, y, alpha, color, fill, group, linetype, size, weight

both discrete

g <- ggplot(diamonds, aes(cut, color))

g + geom_count()
x, y, alpha, color, fill, shape, size, stroke

e + geom_jitter(height = 2, width = 2**)**
x, y, alpha, color, fill, shape, size

continuous bivariate distribution

h <- ggplot(diamonds, aes(carat, price))

h + geom_bin2d(binwidth = c(0.25, 500)**)**
x, y, alpha, color, fill, linetype, size, weight

h + geom_density_2d()
x, y, alpha, color, group, linetype, size

h + geom_hex()
x, y, alpha, color, fill, size

continuous function

i <- ggplot(economics, aes(date, unemploy))

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv"**)**
x, y, alpha, color, group, linetype, size

visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))

j + geom_crossbar(fatten = 2**)** - x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j + geom_errorbar() - x, ymax, ymin, alpha, color, group, linetype, size, width
Also **geom_errorbarh()**.

j + geom_linerange()
x, ymin, ymax, alpha, color, group, linetype, size

j + geom_pointrange() - x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

maps

Draw the appropriate geometric object depending on the simple features present in the data. aes() arguments: map_id, alpha, color, fill, linetype, linewidth.

nc <- **sf::st_read(**system.file("shape/nc.shp", package = "sf")**)**

ggplot(nc) +
 geom_sf(aes(fill = AREA)**)**

THREE VARIABLES

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2)); l <- ggplot(seals, aes(long, lat))

l + geom_contour(aes(z = z)**)**
x, y, z, alpha, color, group, linetype, size, weight

l + geom_contour_filled(aes(fill = z)**)**
x, y, alpha, color, fill, group, linetype, size, subgroup

l + geom_raster(aes(fill = z), hjust = 0.5, vjust = 0.5, interpolate = FALSE**)**
x, y, alpha, fill

l + geom_tile(aes(fill = z)**)**
x, y, alpha, color, fill, linetype, size, width

Subset de los datos

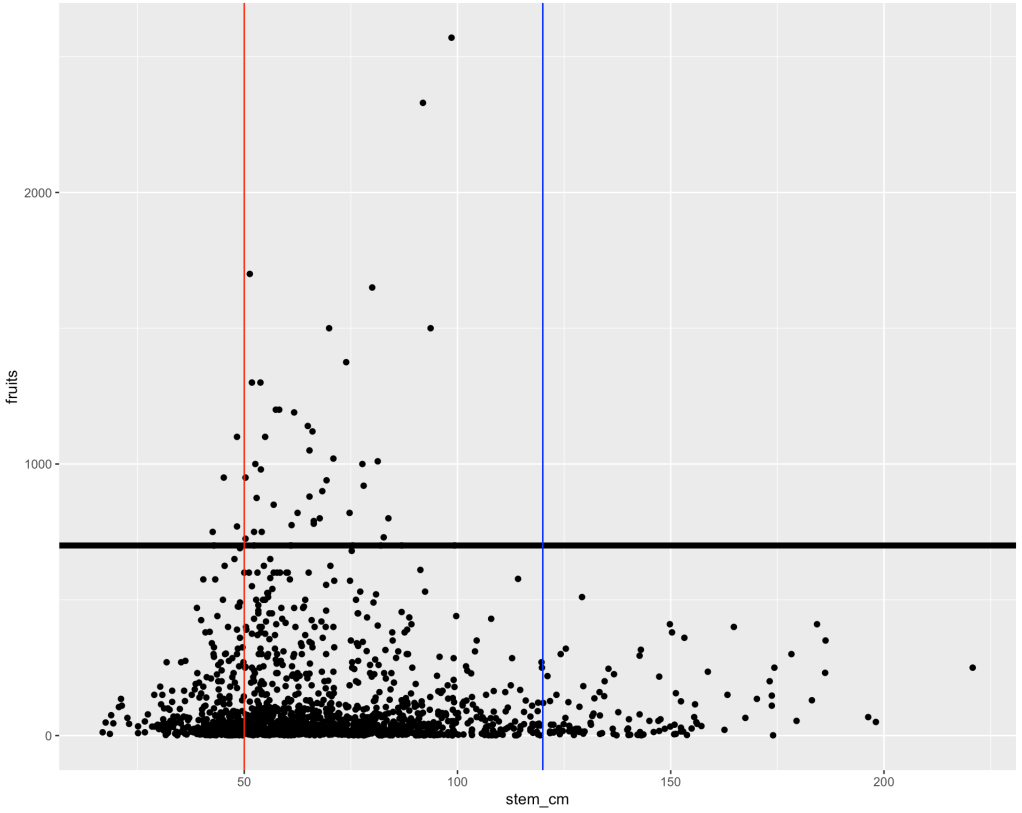
Seleccionar datos con información para diámetro de tronco. Sólo hay informacion para el sitio llamado “AND”.

```r
dt_diam <- dt |> filter(!is.na(stem_cm))
```

```
filter: removed 80,777 rows (98%), 1,503 rows remaining
```

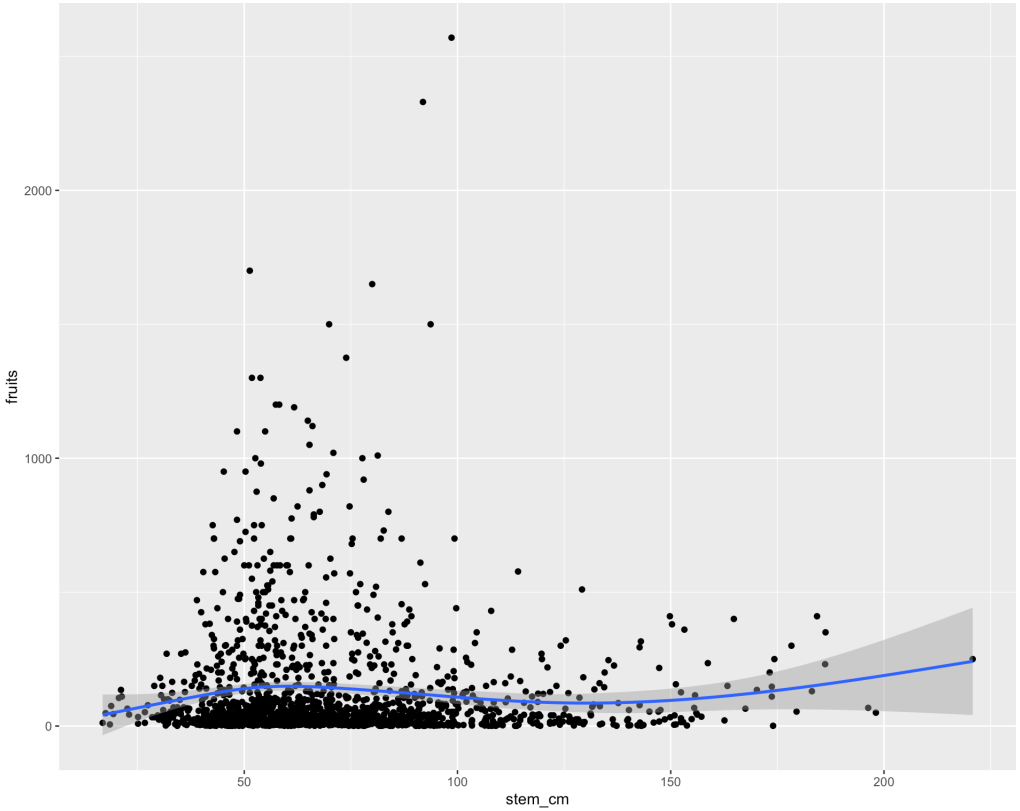
Geoms - Líneas verticales y horizontales

```r
ggplot(dt_diam,
       aes(x = stem_cm, y = fruits)) +
  geom_point() +
  geom_hline(yintercept = 700,
             size = 2) +
  geom_vline(xintercept = c(50, 120),
          color = c("red", "blue"))
```



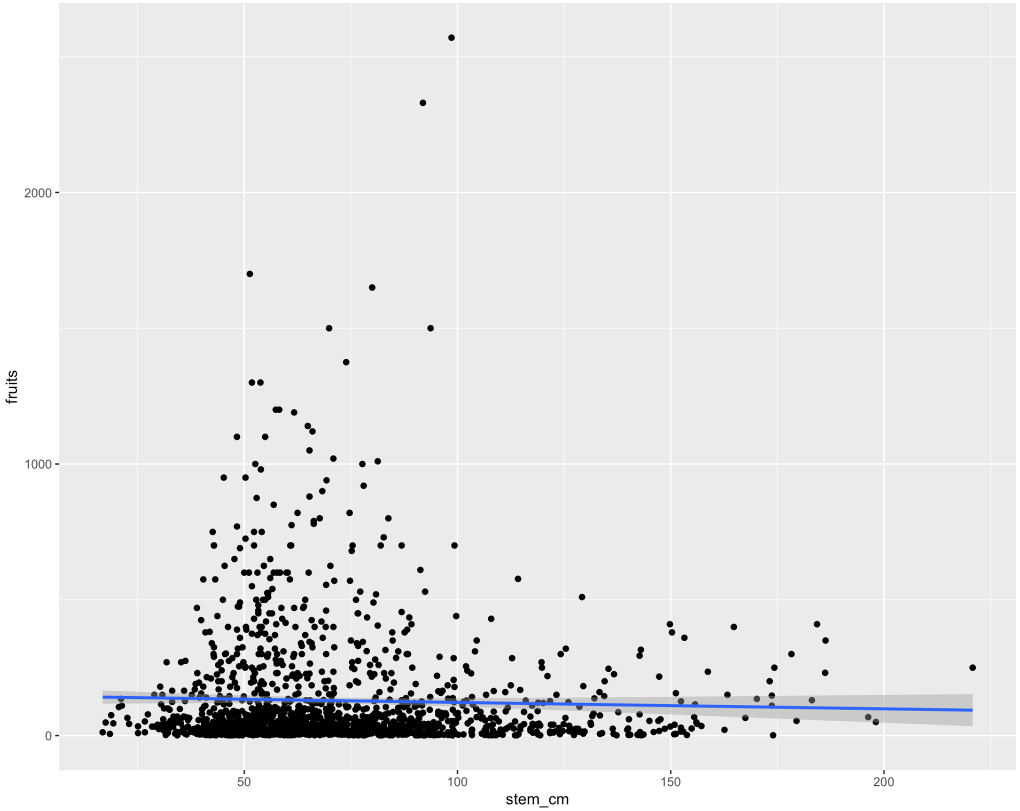
Geoms - Líneas de tendencia

```
ggplot(dt_diam,
       aes(x = stem_cm, y = fruits)) +
  geom_point() +
  geom_smooth()
```



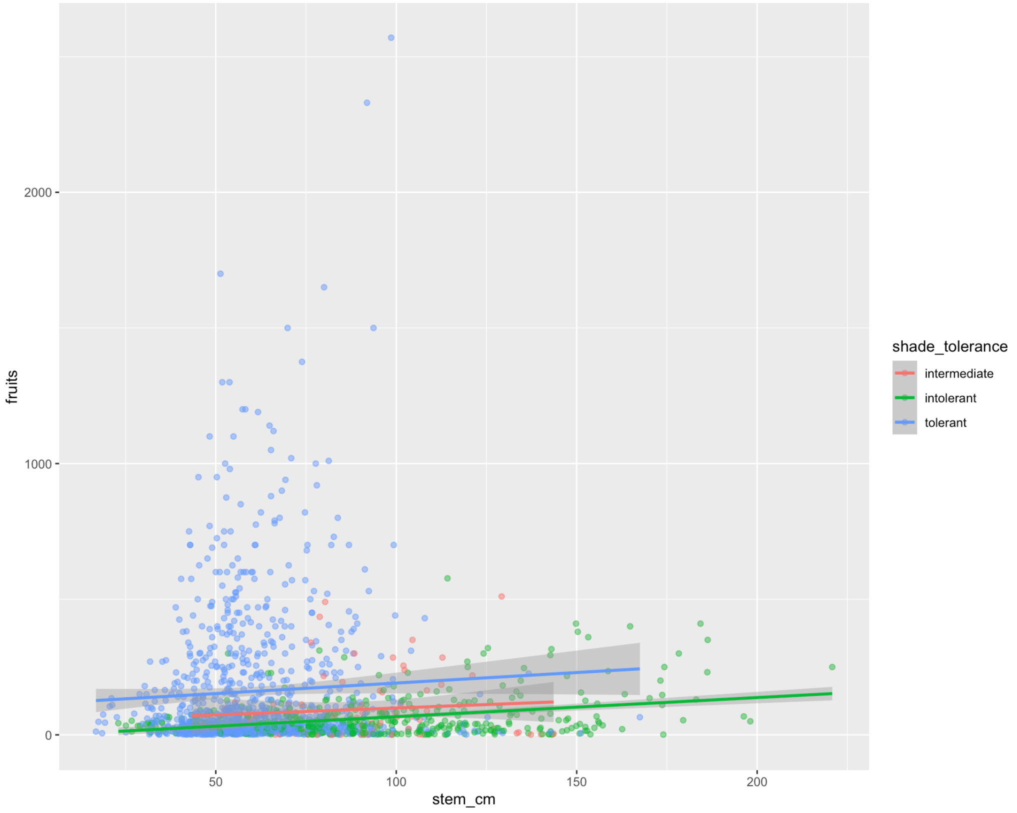
Geoms - Líneas de tendencia

```
ggplot(dt_diam,
       aes(x = stem_cm, y = fruits)) +
  geom_point() +
  geom_smooth(method = "lm")
```



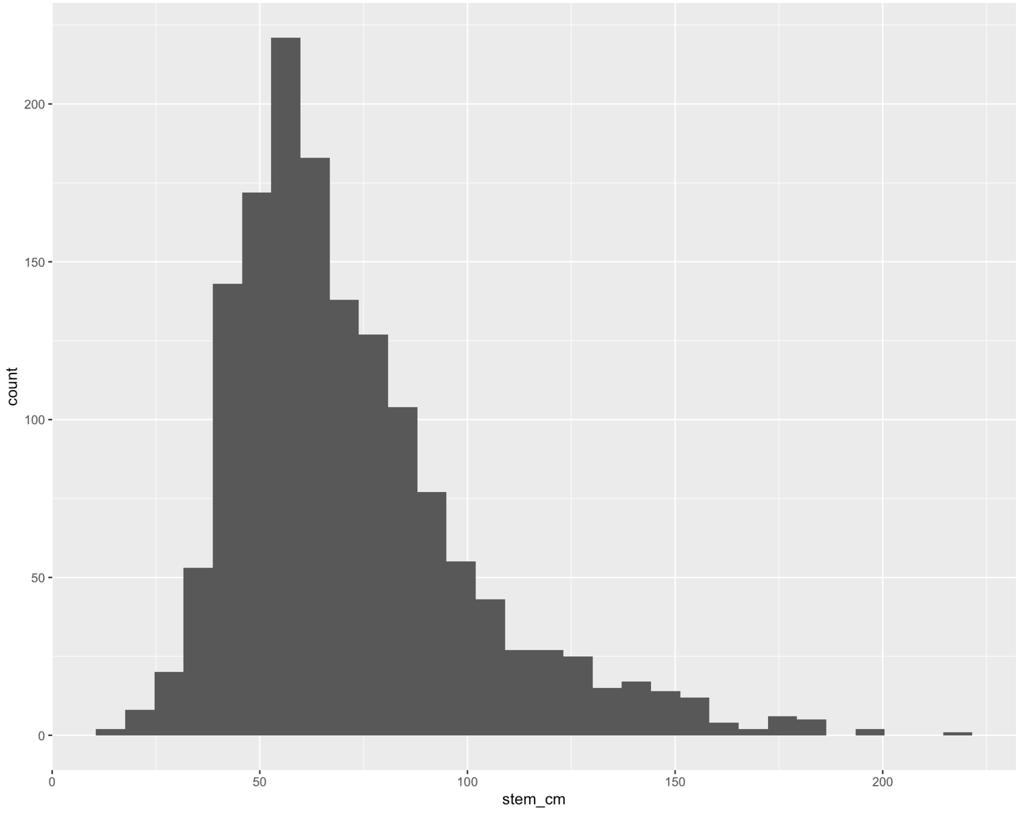
Geoms - Líneas de tendencia

```r
ggplot(dt_diam,
       aes(x = stem_cm, y = fruits,
           color = shade_tolerance)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm")
```



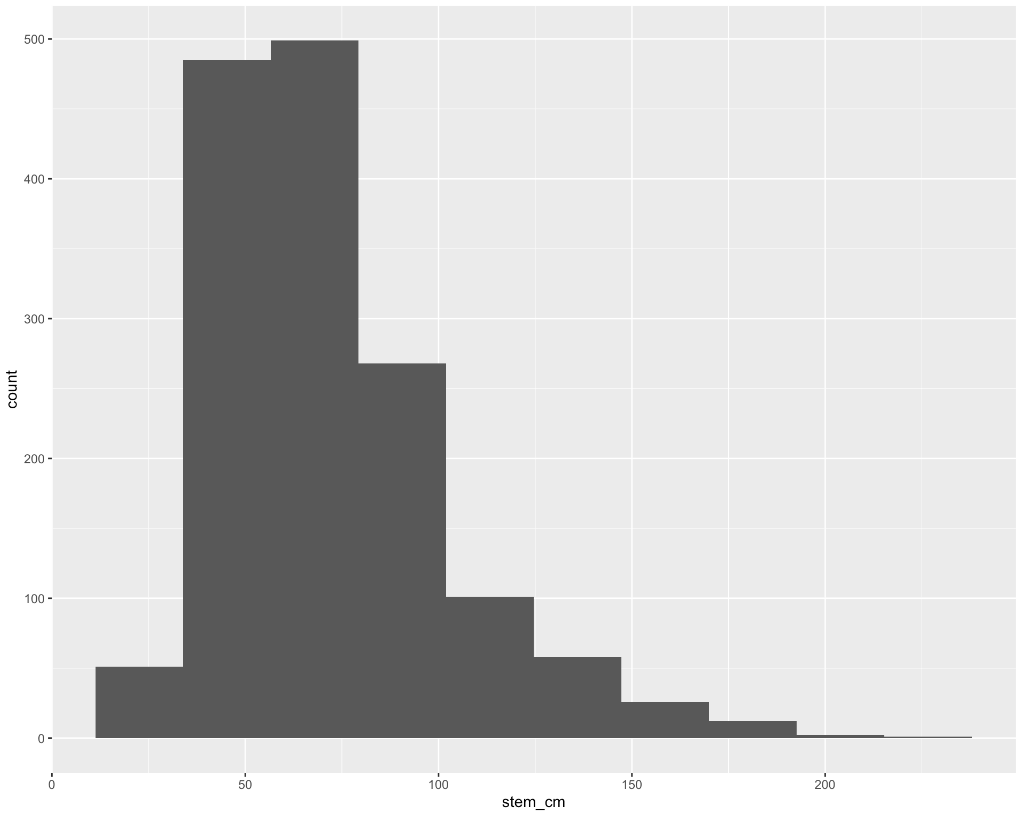
Geoms - Histograma

```r
ggplot(dt_diam,
       aes(x = stem_cm)) +
  geom_histogram()
```



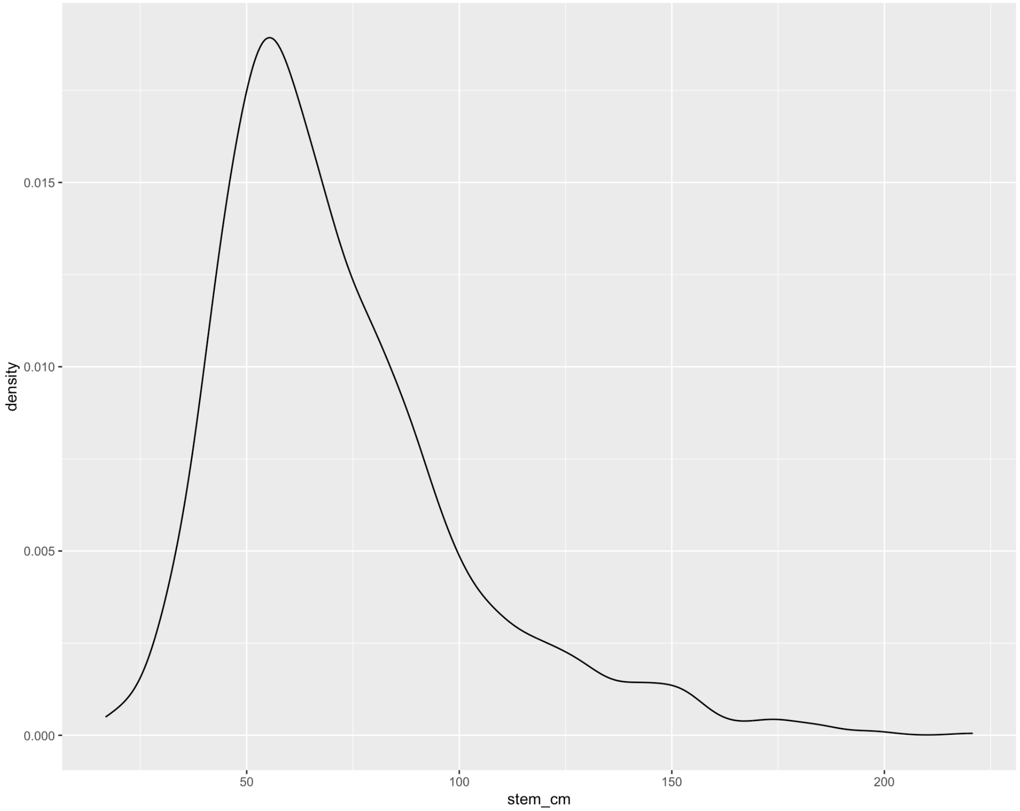
Geoms - Histograma

```
ggplot(dt_diam,
       aes(x = stem_cm)) +
  geom_histogram(bins = 10)
```



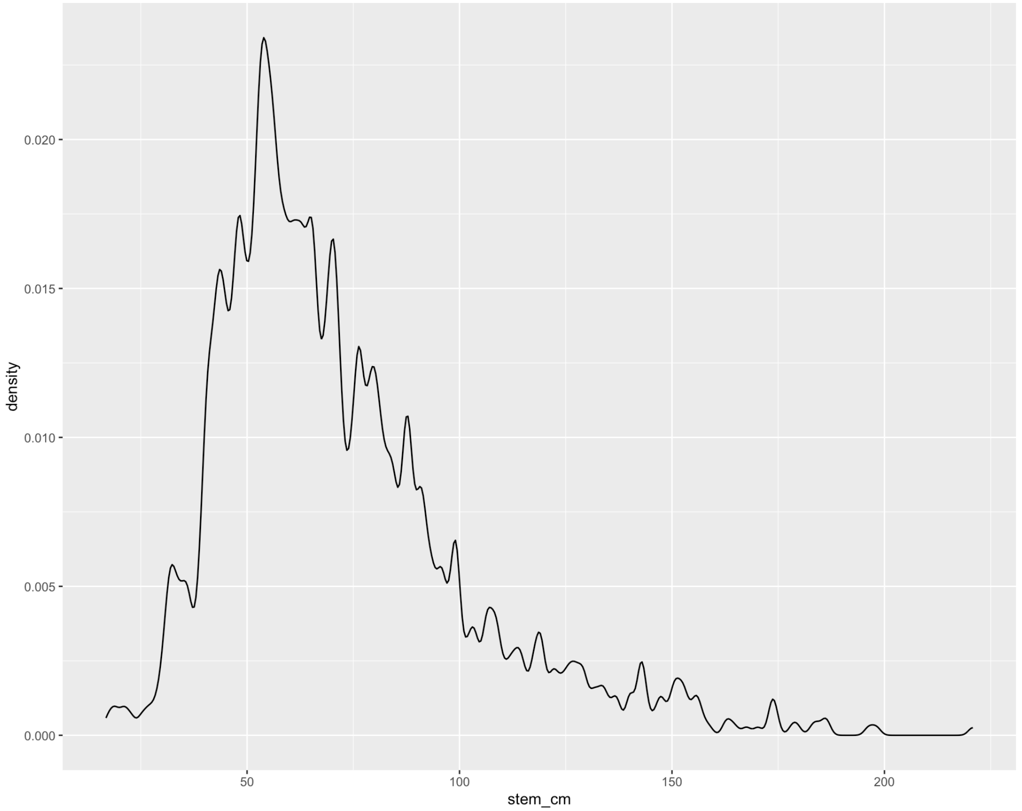
Geoms - Densidad

```
ggplot(dt_diam,
       aes(x = stem_cm)) +
  geom_density()
```



Geoms - Densidad

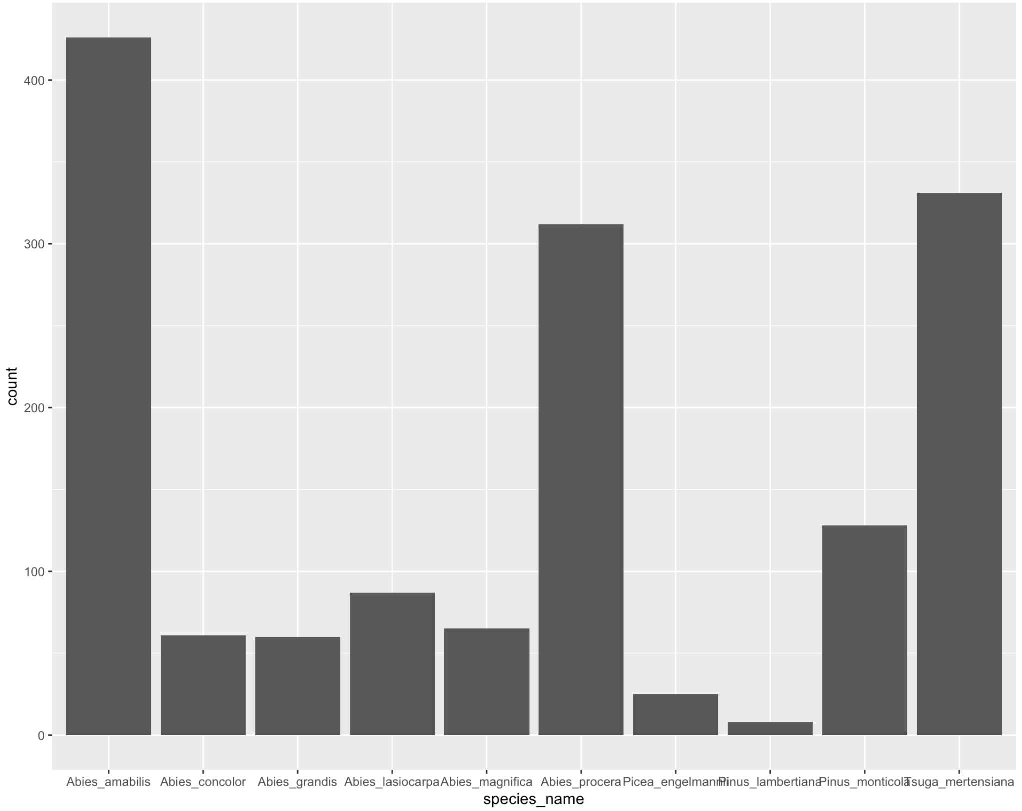
```r
ggplot(dt_diam,
       aes(x = stem_cm)) +
  geom_density(bw = 1)
```



Geoms - Barras

Contar número de casos: `stat = "count"`

```
ggplot(dt_diam,
         aes(x = species_name)) +
  geom_bar(stat = "count")
```

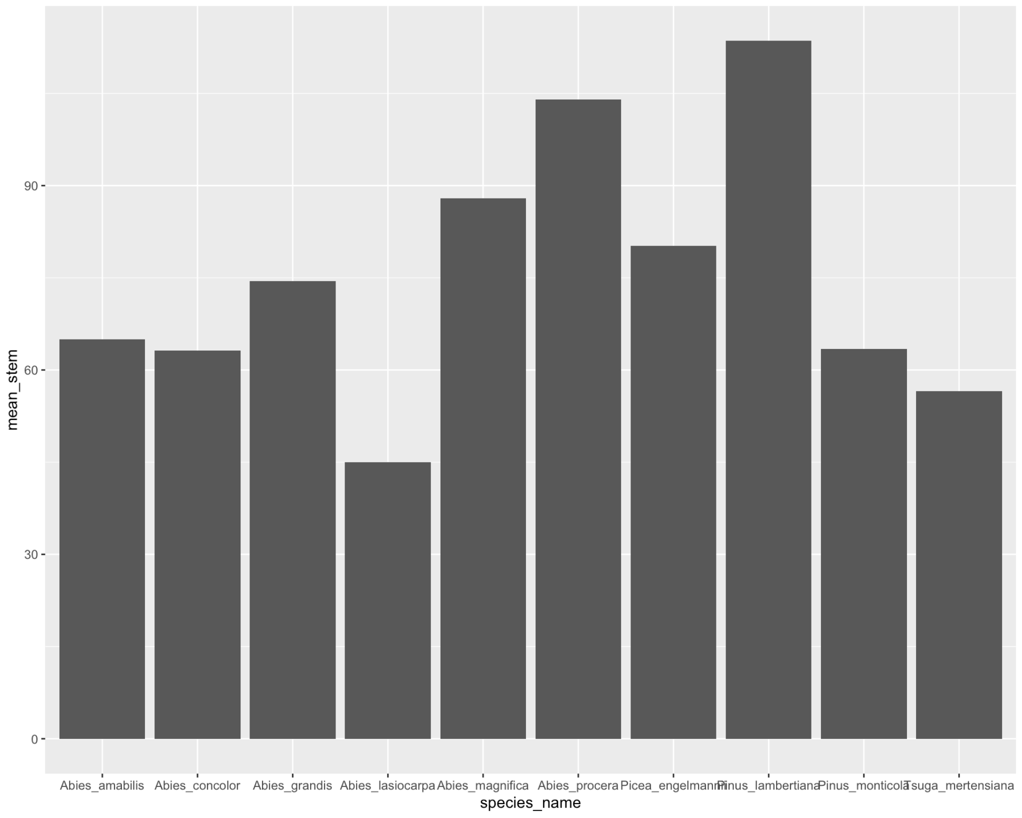


Geoms - Barras

Para datos resumidos - *alternativa 1*:

Usar valores ya calculados: `stat = "identity"`

```r
dt_diam |>
  group_by(species_name) |>
  summarise(mean_stem = mean(stem_cm)) |>
  ggplot(aes(x = species_name,
             y = mean_stem)) +
  geom_bar(stat = "identity")
```

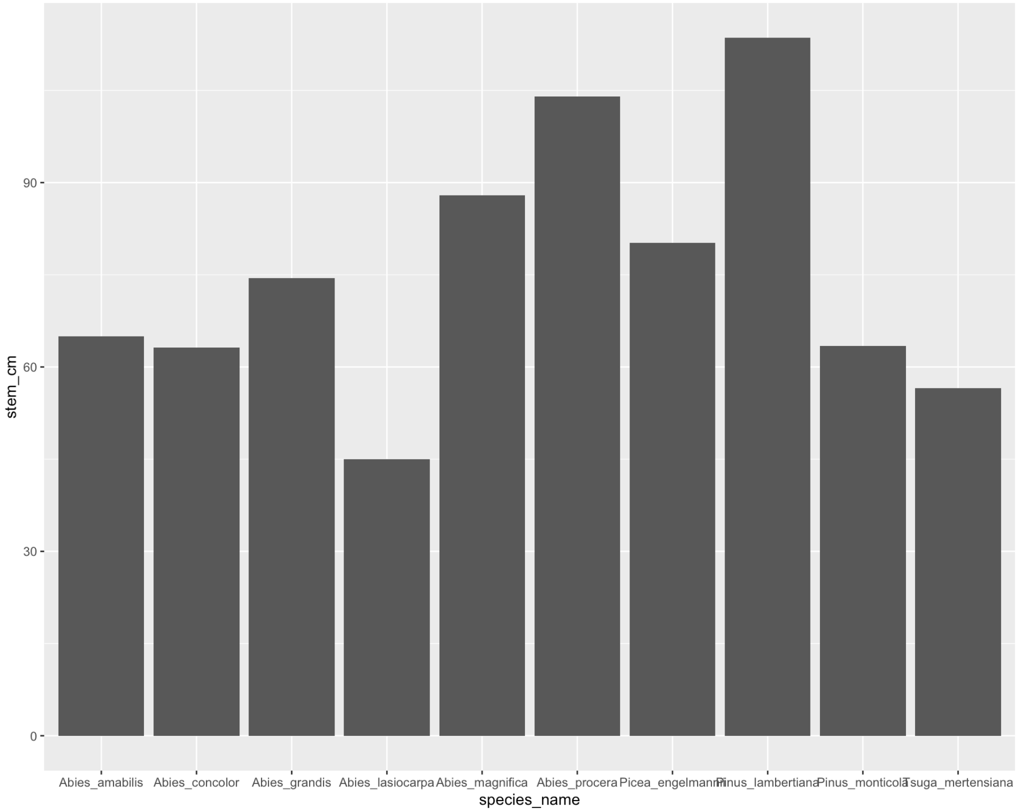


Geoms - Barras

Para datos resumidos - *alternativa 2*:

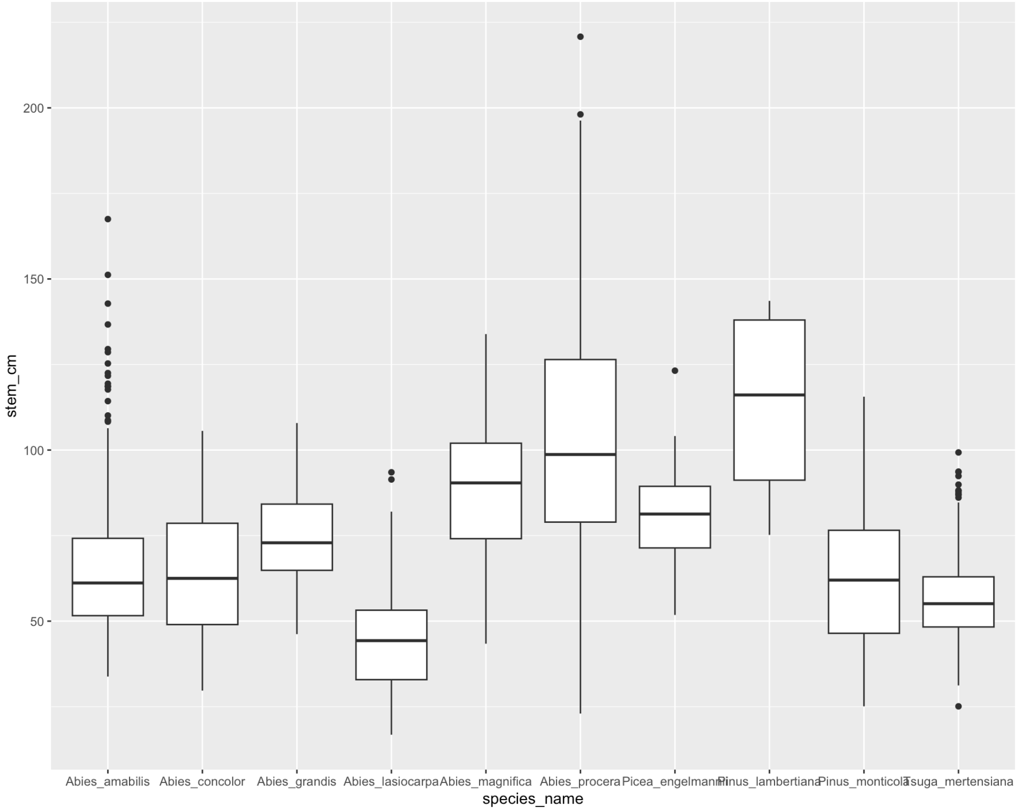
Calcular valores dentro de ggplot: `stat = "summary"`

```
ggplot(dt_diam,
       aes(x = species_name,
           y = stem_cm)) +
  geom_bar(stat = "summary",
           fun = "mean")
```



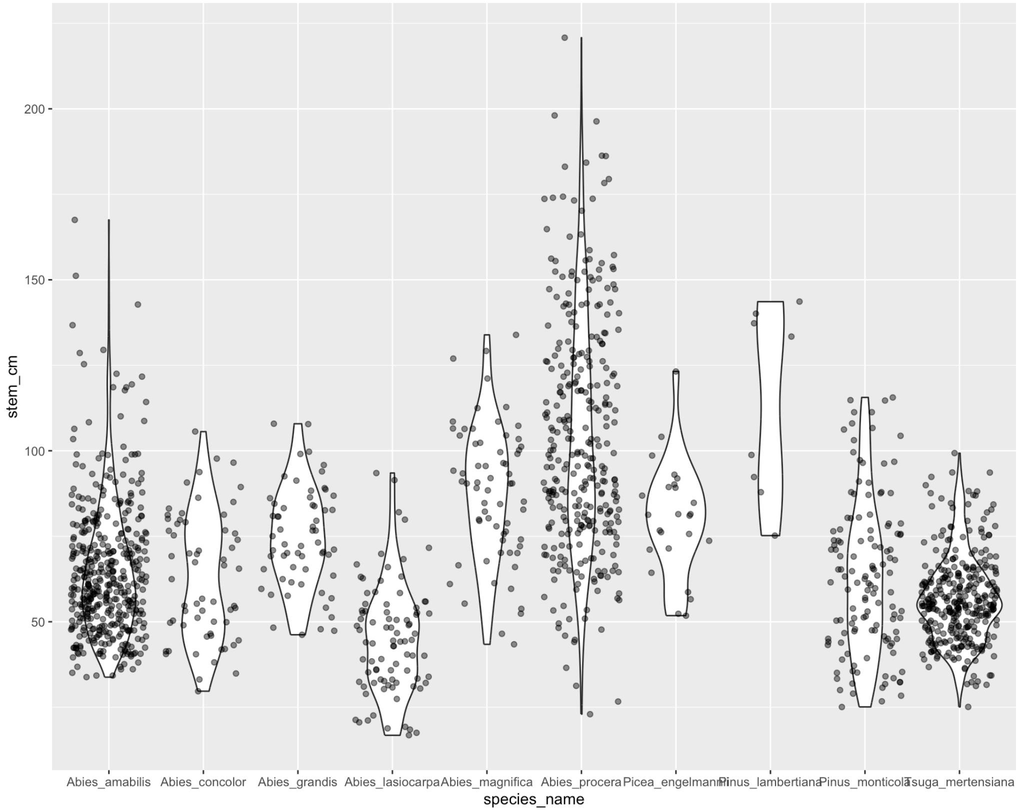
Geoms - Boxplot

```
ggplot(dt_diam,
       aes(x = species_name,
           y = stem_cm)) +
  geom_boxplot()
```



Geoms - Violin y puntos

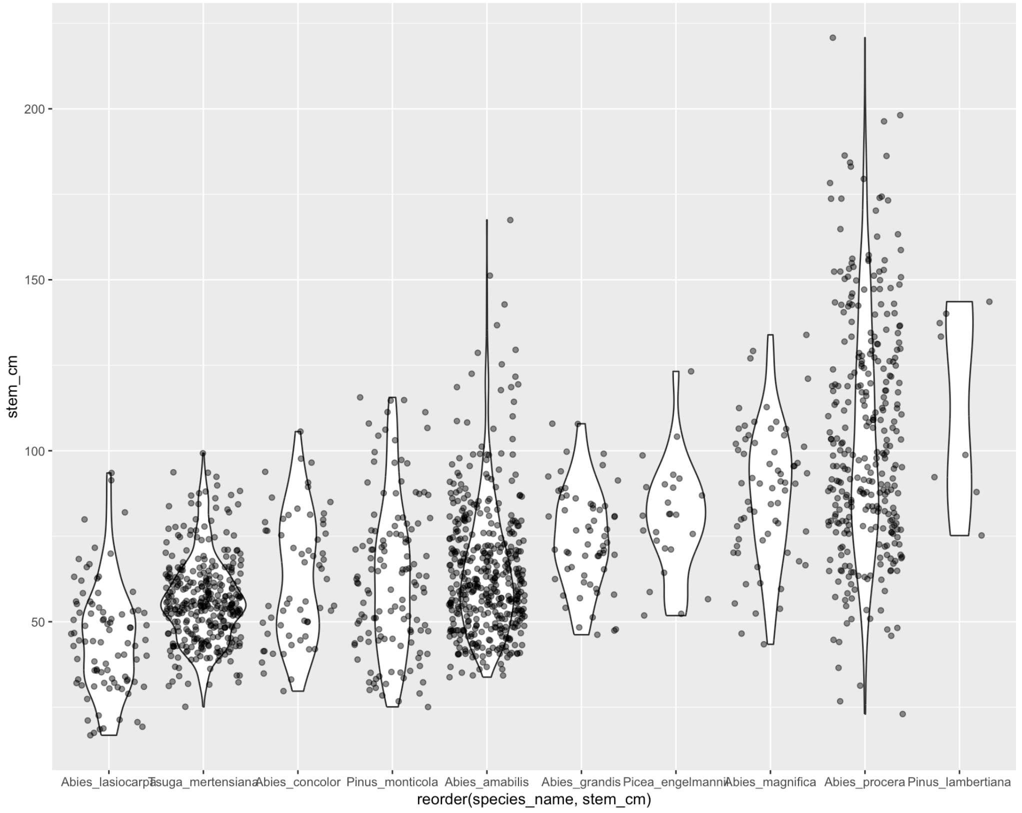
```
ggplot(dt_diam,
       aes(x = species_name,
           y = stem_cm)) +
  geom_violin() +
  geom_jitter(alpha = 0.5)
```



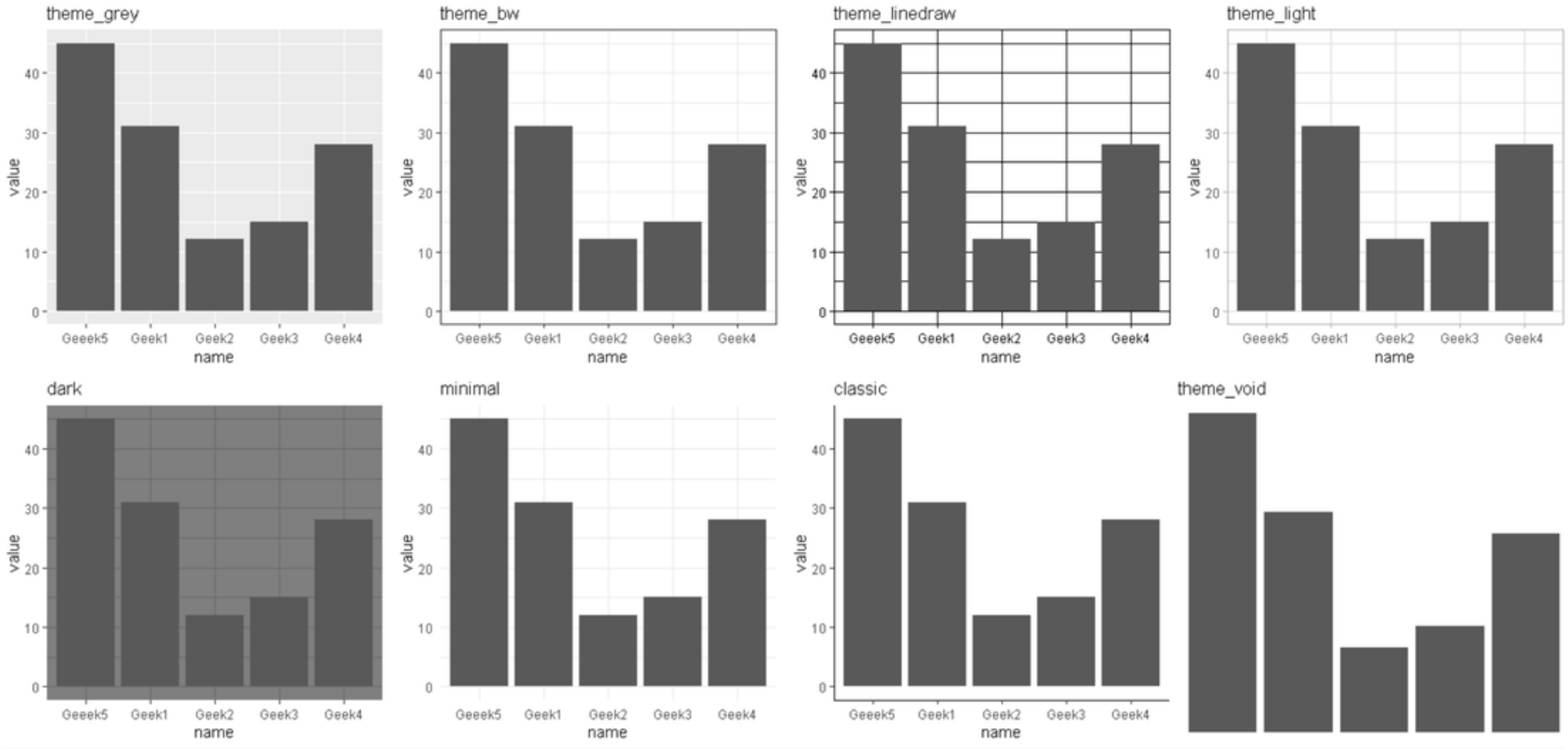
Geoms - Violin y puntos

Ordernar eje de la X en base a valores del eje de la Y

```
ggplot(dt_diam,
 aes(x = reorder(species_name, stem_cm),
     y = stem_cm)) +
  geom_violin() +
  geom_jitter(alpha = 0.5)
```

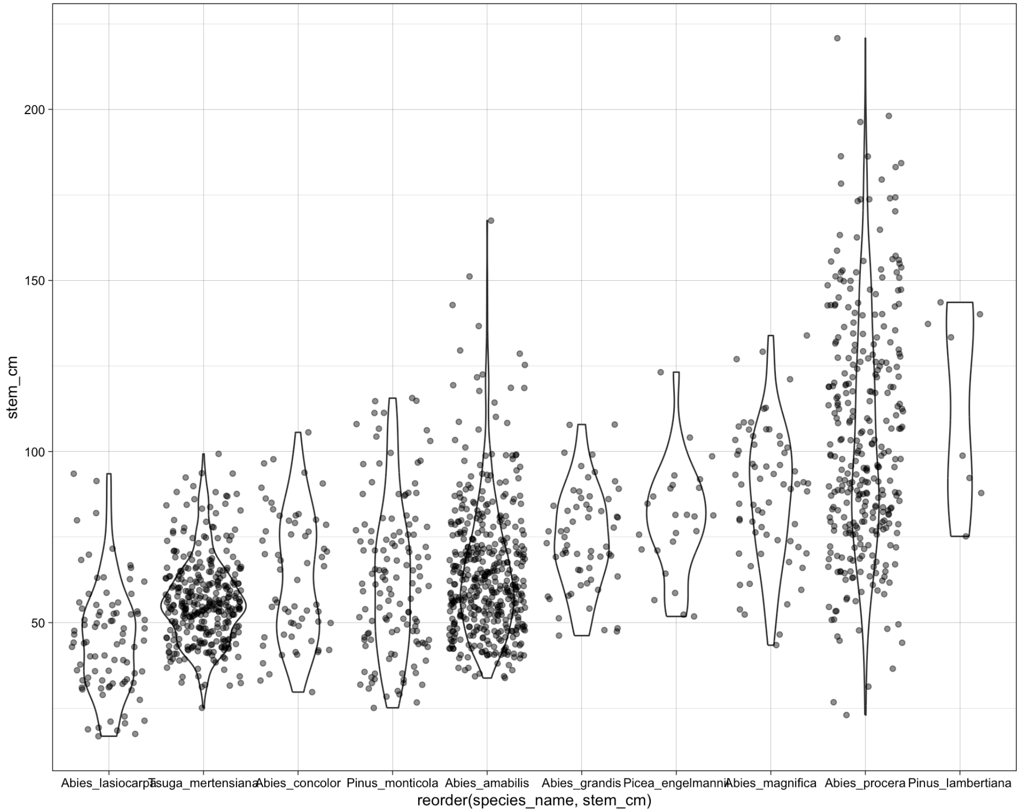


Themes


theme_grey
value
40
30
20
10
0
Geeek5
Geek1
Geek2
Geek3
Geek4
name
theme_bw
value
40
30
20
10
0
Geeek5
Geek1
Geek2
Geek3
Geek4
name
theme_linedraw
value
40
30
20
10
0
Geeek5
Geek1
Geek2
Geek3
Geek4
name
theme_light
value
40
30
20
10
0
Geeek5
Geek1
Geek2
Geek3
Geek4
name
dark
value
40
30
20
10
0
Geeek5
Geek1
Geek2
Geek3
Geek4
name
minimal
value
40
30
20
10
0
Geeek5
Geek1
Geek2
Geek3
Geek4
name
classic
value
40
30
20
10
0
Geeek5
Geek1
Geek2
Geek3
Geek4
name
theme_void

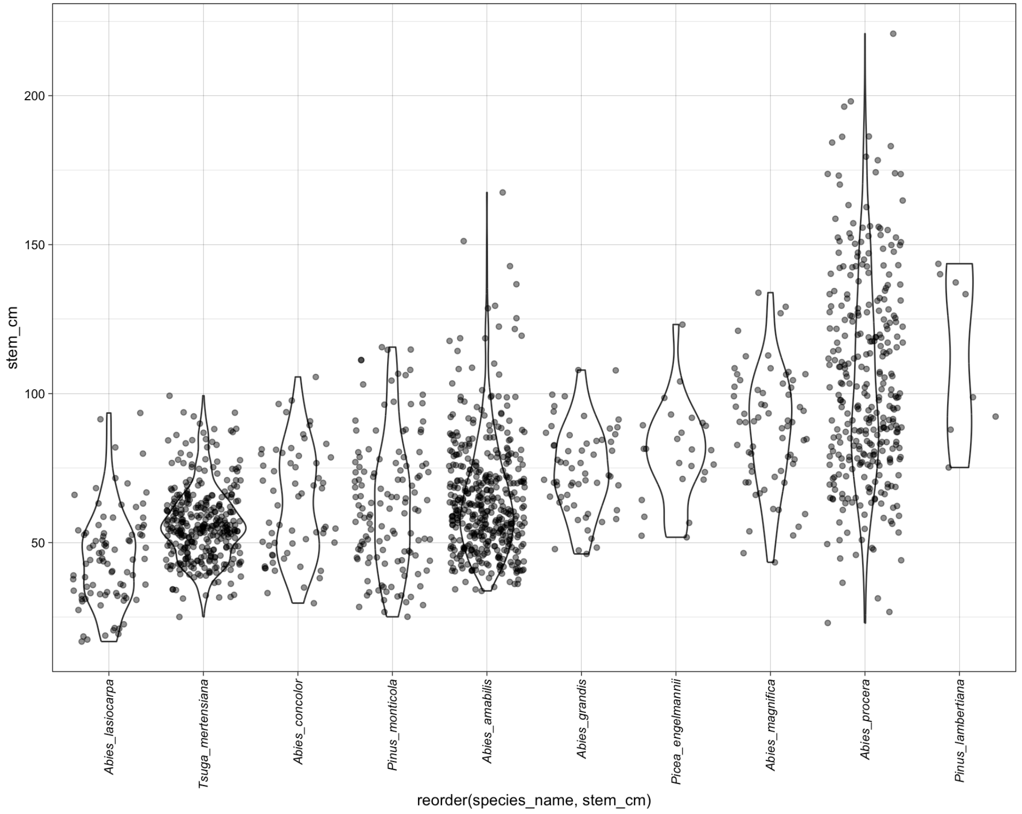
Themes

```
ggplot(dt_diam,
       aes(x = reorder(species_name, ster
           y = stem_cm)) +
  geom_violin() +
  geom_jitter(alpha = 0.5) +
  theme_linedraw()
```



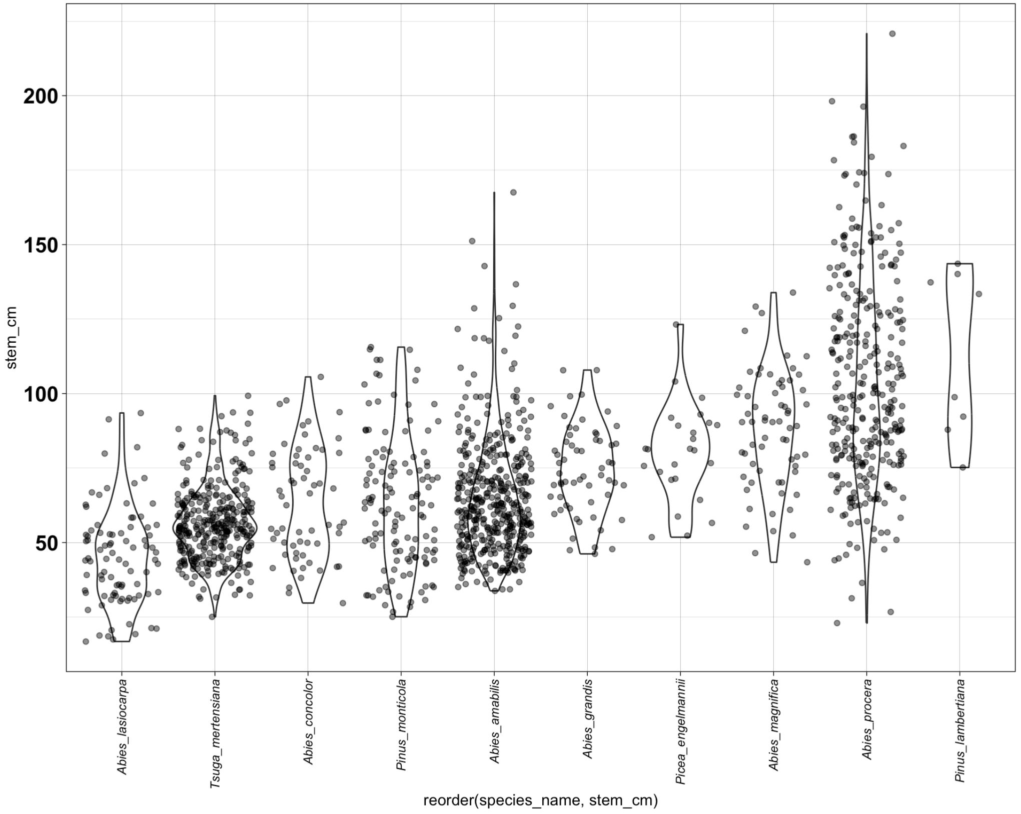
Themes

```
ggplot(dt_diam,
       aes(x = reorder(species_name, ster
           y = stem_cm)) +
  geom_violin() +
  geom_jitter(alpha = 0.5) +
  theme_linedraw() +
  theme(axis.text.x = element_text(
          angle = 90,
          hjust = 1, vjust = 0.5,
          face = "italic"))
```



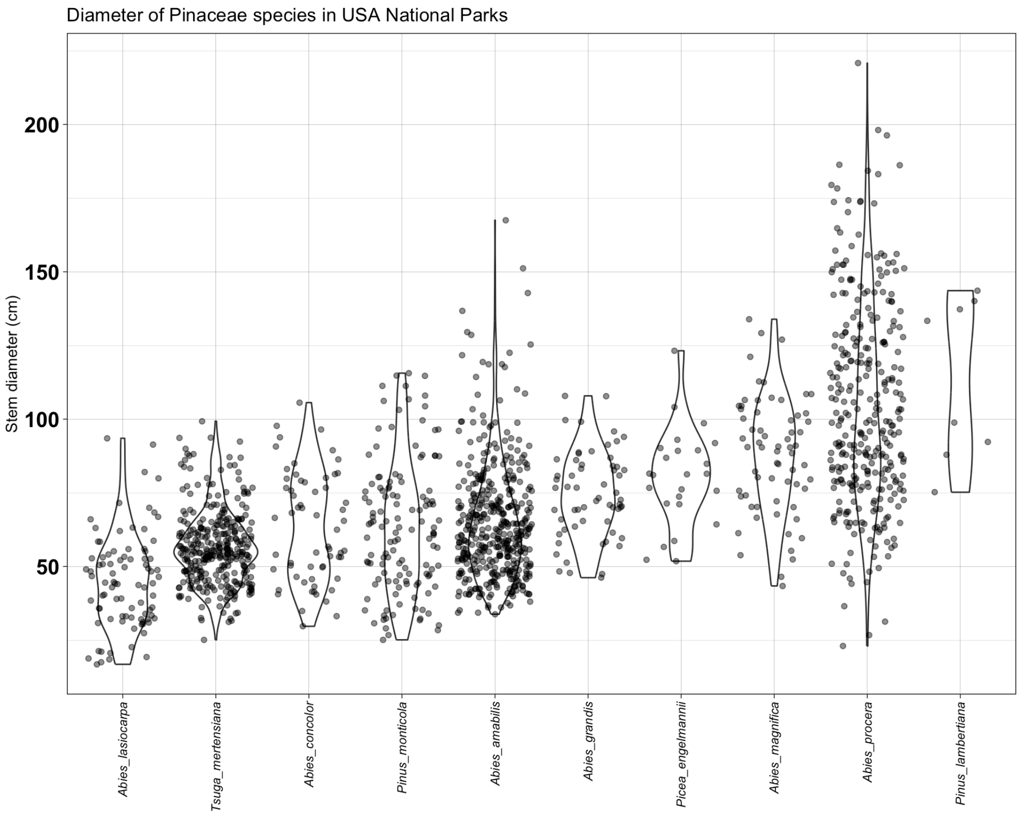
Themes

```
ggplot(dt_diam,
       aes(x = reorder(species_name, ster
           y = stem_cm)) +
  geom_violin() +
  geom_jitter(alpha = 0.5) +
  theme_linedraw() +
  theme(axis.text.x = element_text(
          angle = 90,
          hjust = 1, vjust = 0.5,
          face = "italic"),
        axis.text.y = element_text(
          size = 15, face = "bold"))
```



Themes

```
ggplot(dt_diam,
       aes(x = reorder(species_name, stem
           y = stem_cm)) +
  geom_violin() +
  geom_jitter(alpha = 0.5) +
  theme_linedraw() +
  theme(axis.text.x = element_text(
          angle = 90,
          hjust = 1, vjust = 0.5,
          face = "italic"),
        axis.text.y = element_text(
          size = 15, face = "bold")) +
  labs(x = NULL,
       y = "Stem diameter (cm)",
       title = "Diameter of Pinaceae spe
```



Themes

Para fijar un tema que se aplique a todos los gráficos:

```r
theme_set(theme_minimal())
```

Existen muchos paquetes con temas predeterminados. Muchos también vienen con especificaciones para las escalas de los aesthetics (scales). Ejemplos:

```r
library(hrbrthemes)
library(ggthemes)
library(ggpomological)
library(tvthemes)
library(ggtech)
library(ggthemr)
library(ggsci)
```

`library(ggThemeAssist)` es una addin de RStudio que ayuda a cambiar la apariencia de los temas. https://github.com/calligross/ggthemeassist

Composición de figuras

```
library(patchwork)
```

Composición de figuras

Nombrar los plots como objetos

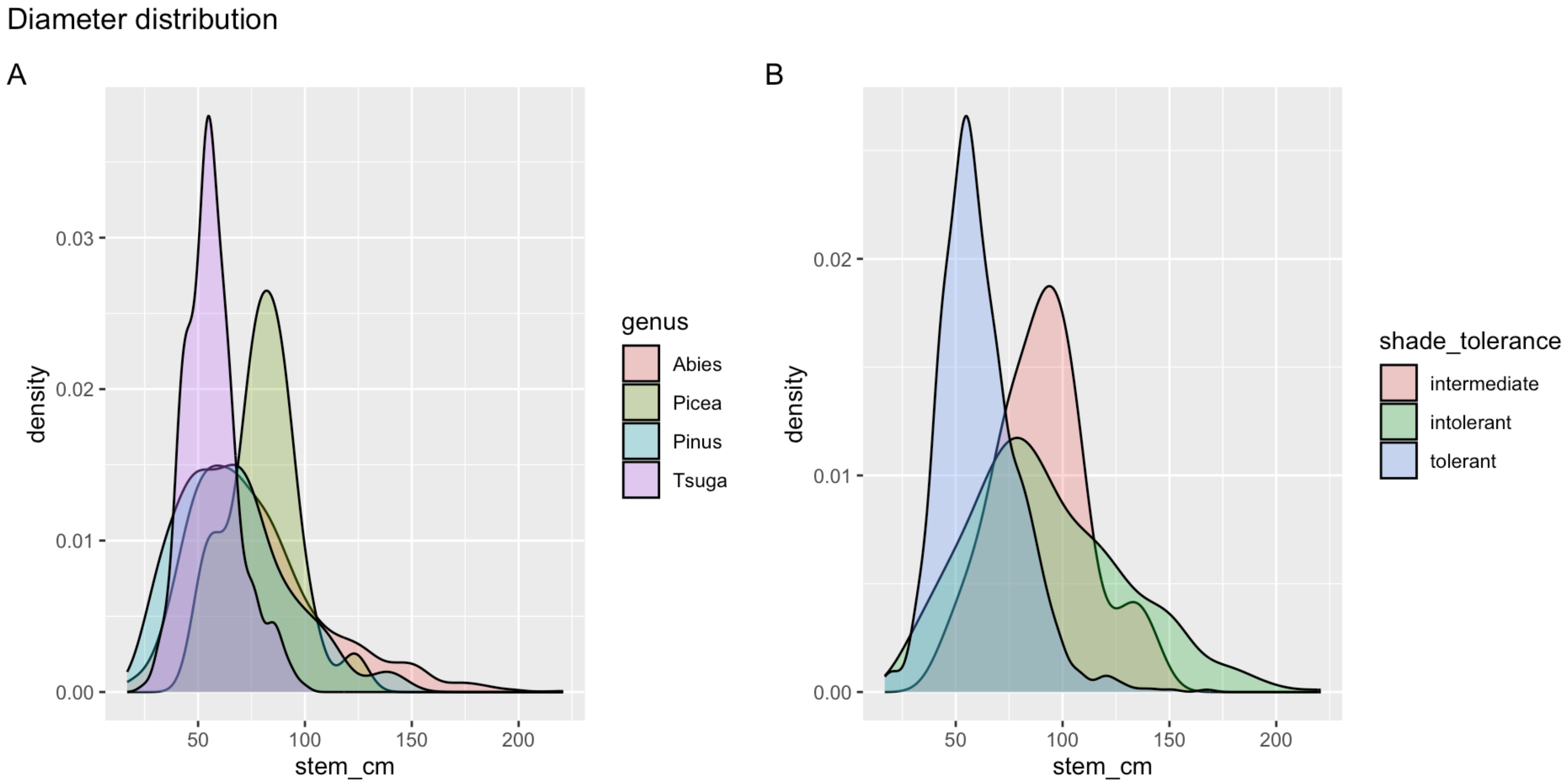
```r
library(patchwork)

p1 <- ggplot(dt_diam,
       aes(x = stem_cm,
           fill = genus)) +
  geom_density(alpha = 0.3)

p2 <- ggplot(dt,
       aes(x = stem_cm,
           fill = shade_tolerance)) +
  geom_density(alpha = 0.3)

p1 + p2 +
  plot_annotation(
  title = "Diameter distribution",
  tag_levels = 'A')
```

Composición de figuras



Guardar gráficos

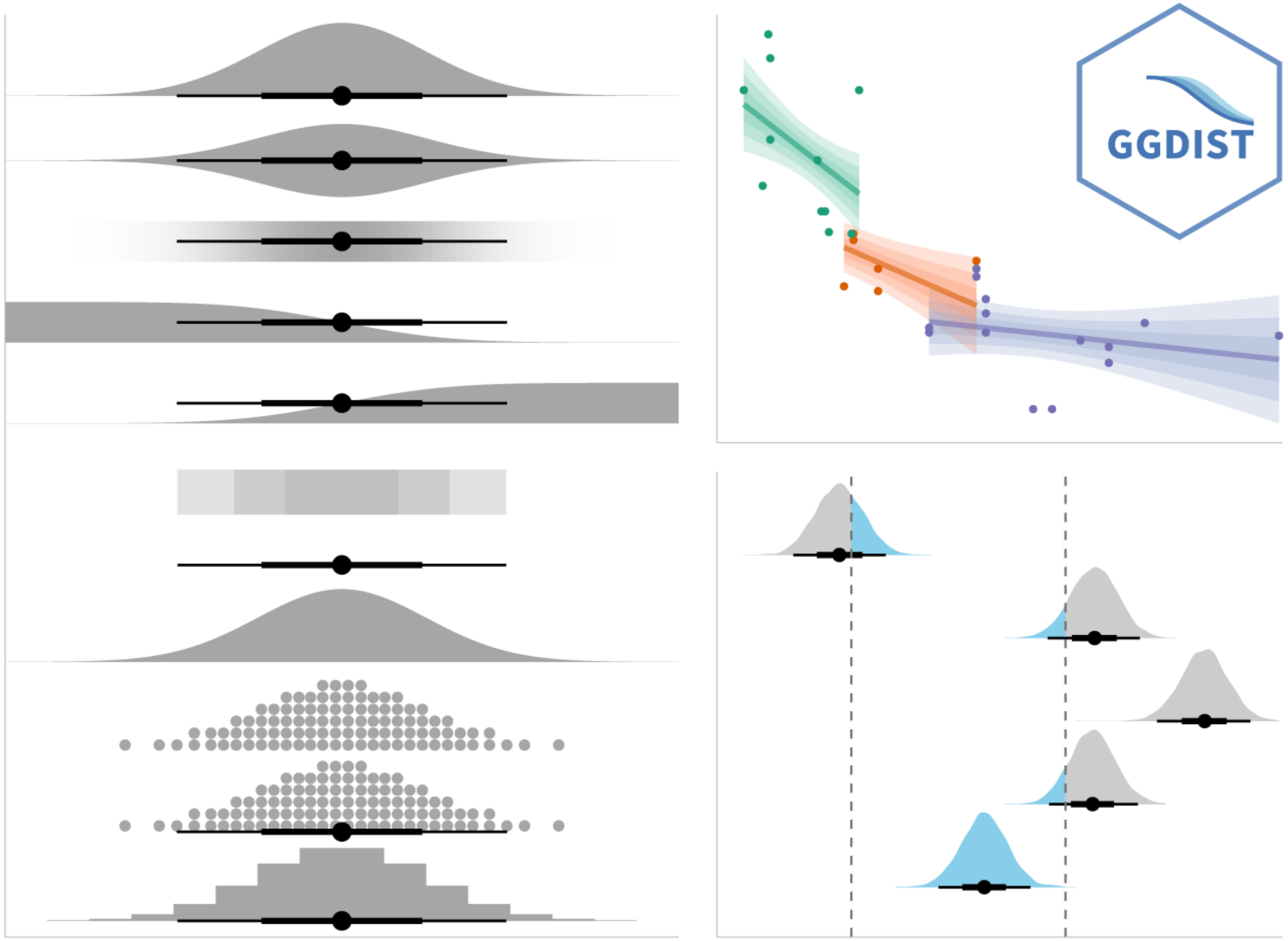
```
ggsave(p1, width = 20, units = "cm",
  filename = here("img/figure_1.pdf"))
```

Argumentos:

- width
- height
- units = (“in”, “cm”, “mm”, “px”)
- device = (“png”, “pdf,”jpeg”, “tiff”, “svg”…)

Otros paquetes útiles

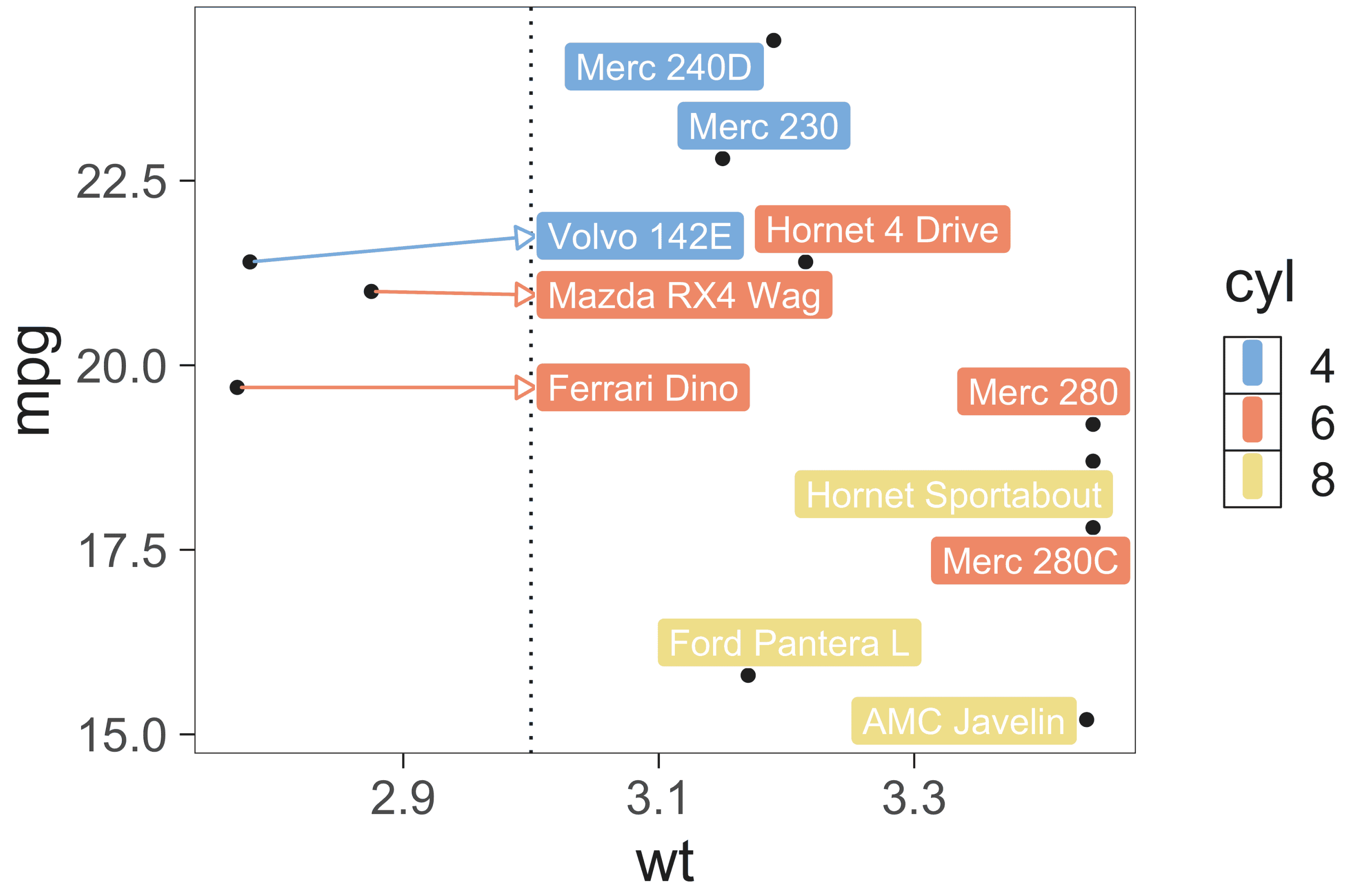
`library(ggdist)` - graficar datos con distribuciones amplias (ej. Bayes posteriors).



https://mjskay.github.io/ggdist/

Otros paquetes útiles

`library(ggrepel)` - etiquetar datos

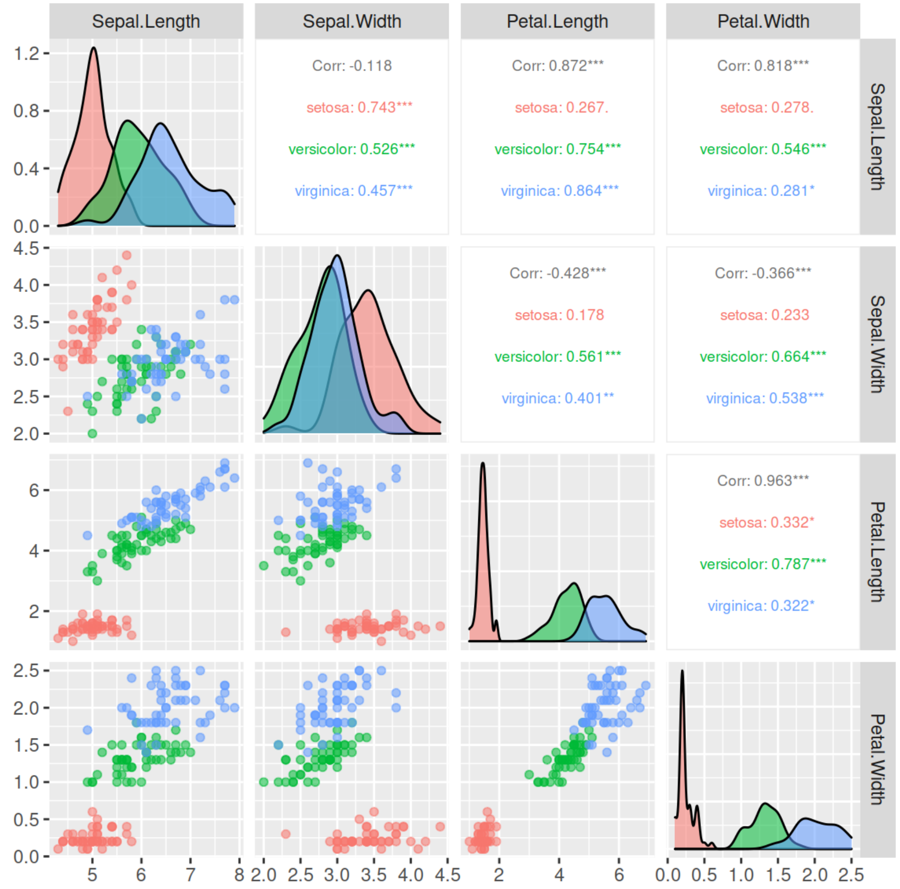


https://cran.r-project.org/web/packages/ggrepel/vignettes/ggrepel.html

Otros paquetes útiles

`library(GGally)` - exploración de datos y relación entre variables.

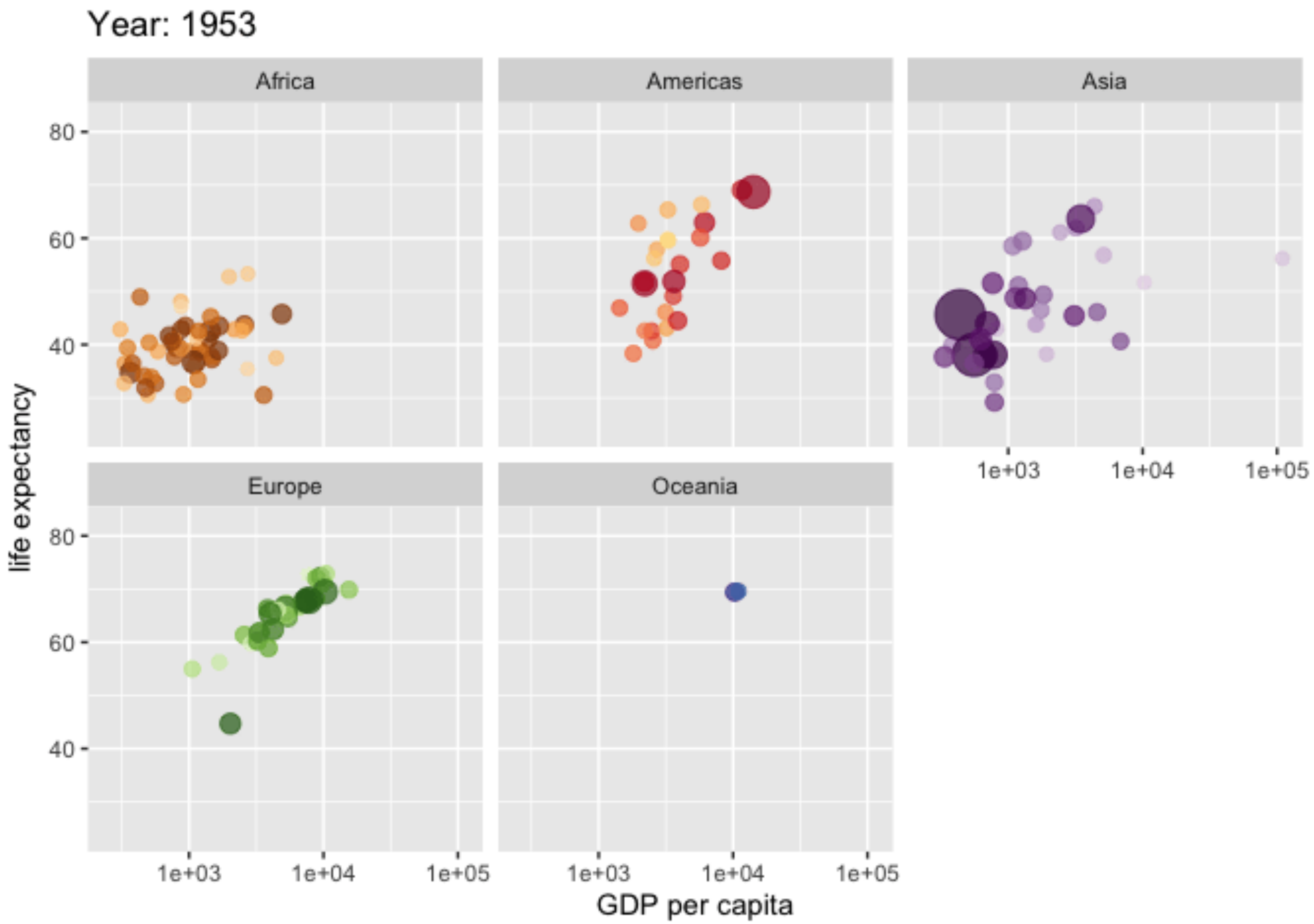
```
GGally::ggpairs(dataset)
```



https://github.com/ggobi/ggally

Otros paquetes útiles

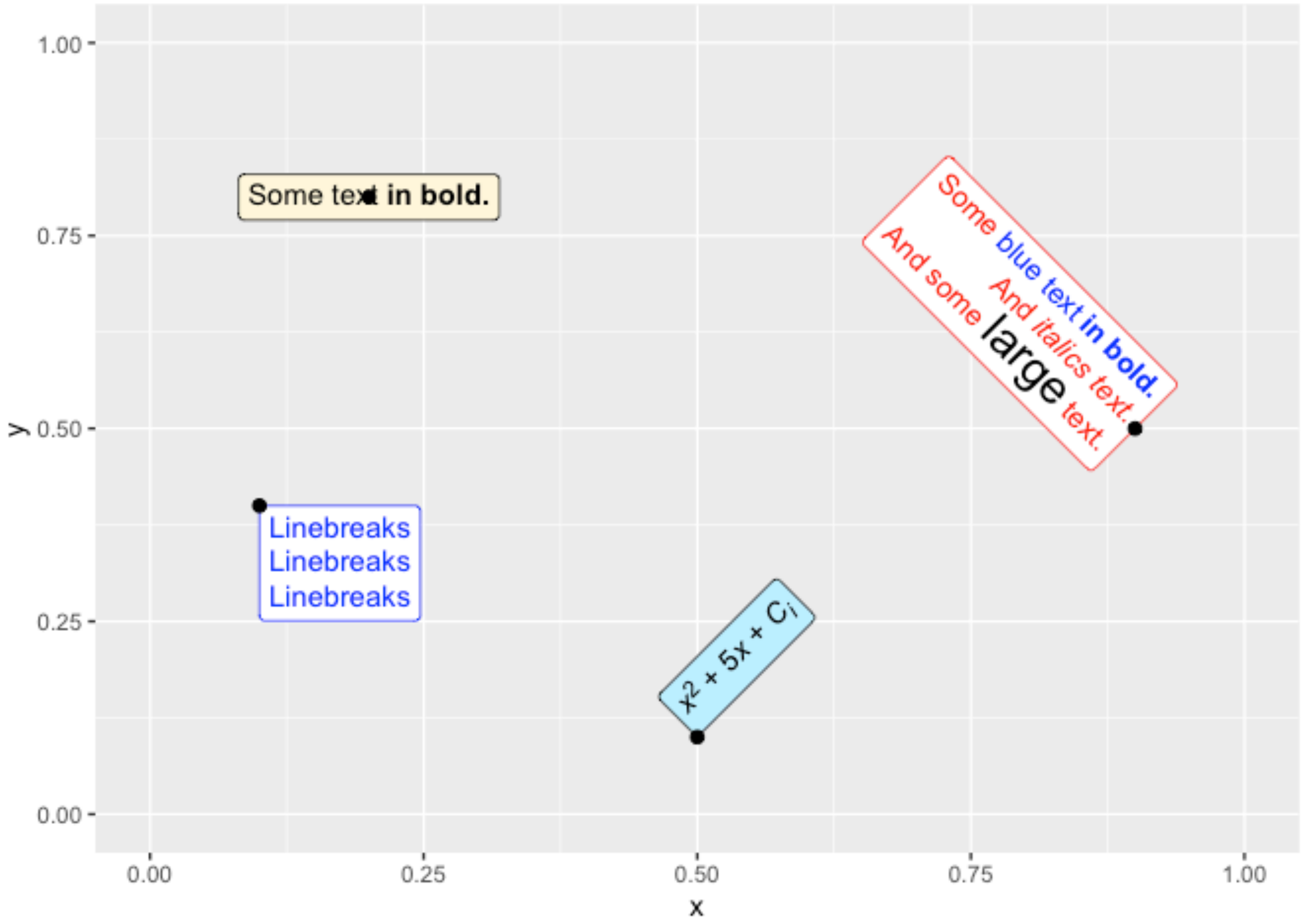
`library(gganimate)` - animar gráficos



https://gganimate.com/

Otros paquetes útiles

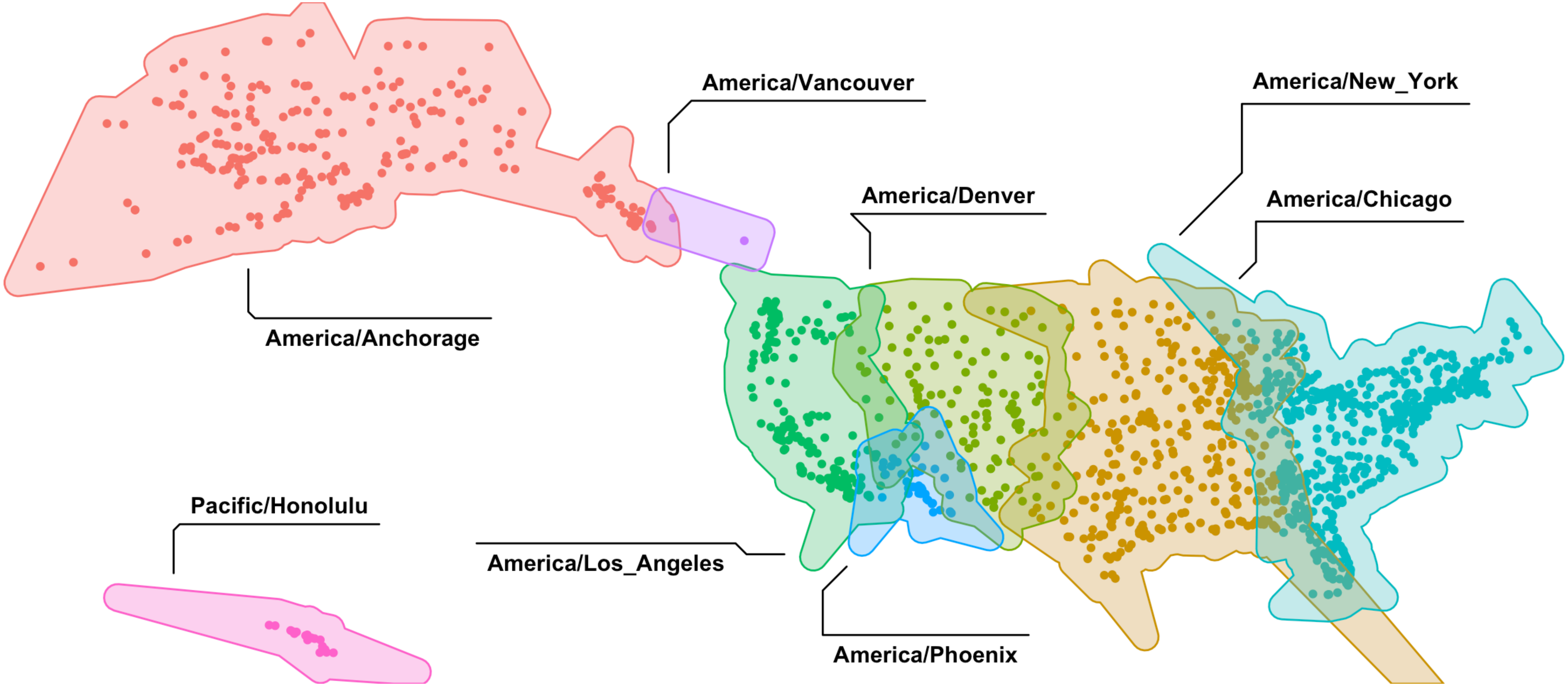
`library(ggtext)` -



https://github.com/wilkelab/ggtext

Otros paquetes útiles

`library(ggforce)` - resaltar características de los datos.



https://github.com/thomasp85/ggforce

Más extensiones de GGplot

https://github.com/erikgahner/awesome-ggplot2
https://exts.ggplot2.tidyverse.org/gallery/

Recursos

- Referencia base a ggplot2
- ggplot2 book - Hadley Wickham
- R for Data Science Book - 3. Data visualization
- R Graphics Cookbook, 2nd edition - Winston Chang
- Fundamentals of Data Visualization - Claus O. Wilke
- RStudio CheatSheets - *“Data visualization with ggplot2”*

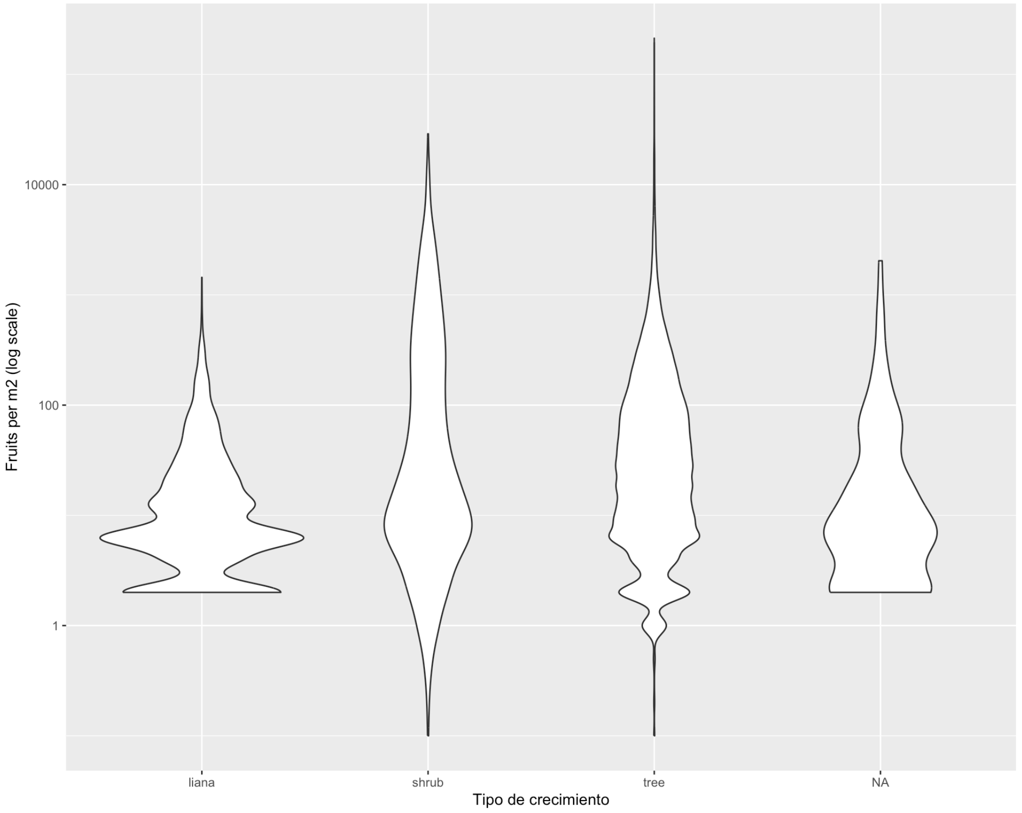
Ejercicio 1

Con el dataset completo (dt) genera un violin plot del número de frutos por m2 (fruits) en escala logarítmica para los distintos tipos de crecimiento (growth_form)

Ejercicio 1

Con el dataset completo (dt) genera un violin plot del número de frutos por m2 (fruits) en escala logarítmica para los distintos tipos de crecimiento (growth_form)

```r
ggplot(dt,
       aes(x = growth_form,
           y = fruits)) +
  geom_violin() +
  labs(x = "Tipo de crecimiento",
       y = "Fruits per m2 (log scale)") +
  scale_y_log10()
```



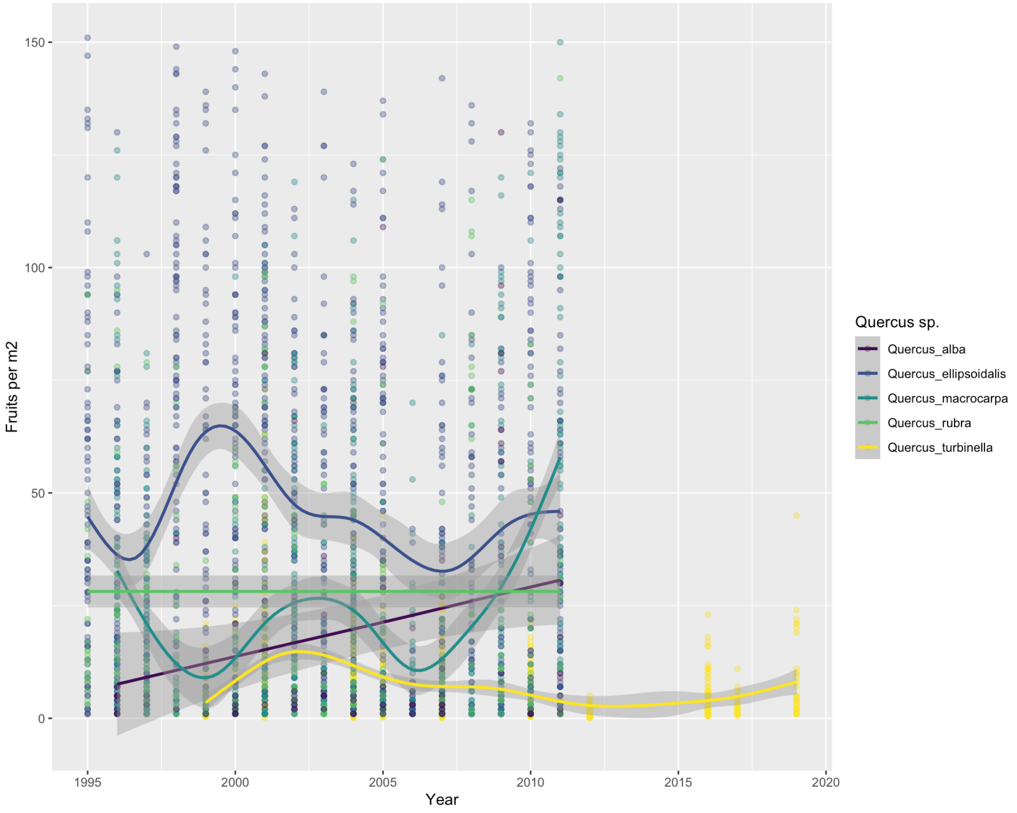
Ejercicio 2

Para las distintas especies del género *Quercus*, sacar una tendencia del número de frutos por m2 a lo largo de los años.

Ejercicio 2

Para las distintas especies del género *Quercus*, sacar una tendencia del número de frutos por m2 a lo largo de los años.

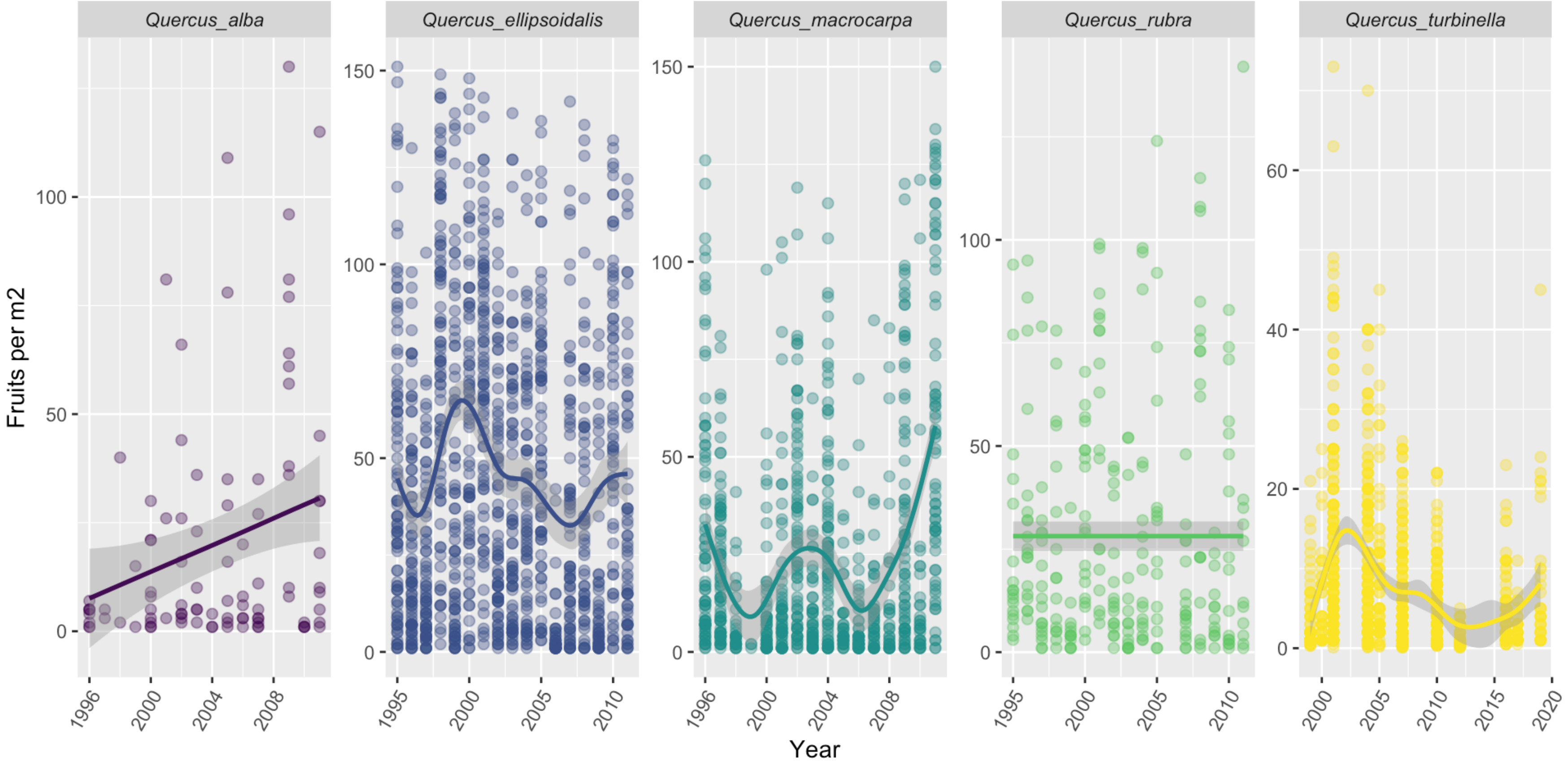
```
dt |> filter(genus == "Quercus") |>
  ggplot(aes(x = year,
             y = fruits,
             color = species_name)) +
  geom_point(alpha = 0.4)  +
  geom_smooth() +
  scale_color_viridis_d() +
  labs(x = "Year",
       y = "Fruits per m2",
       color = "Quercus sp.")
```



Ejercicio 2

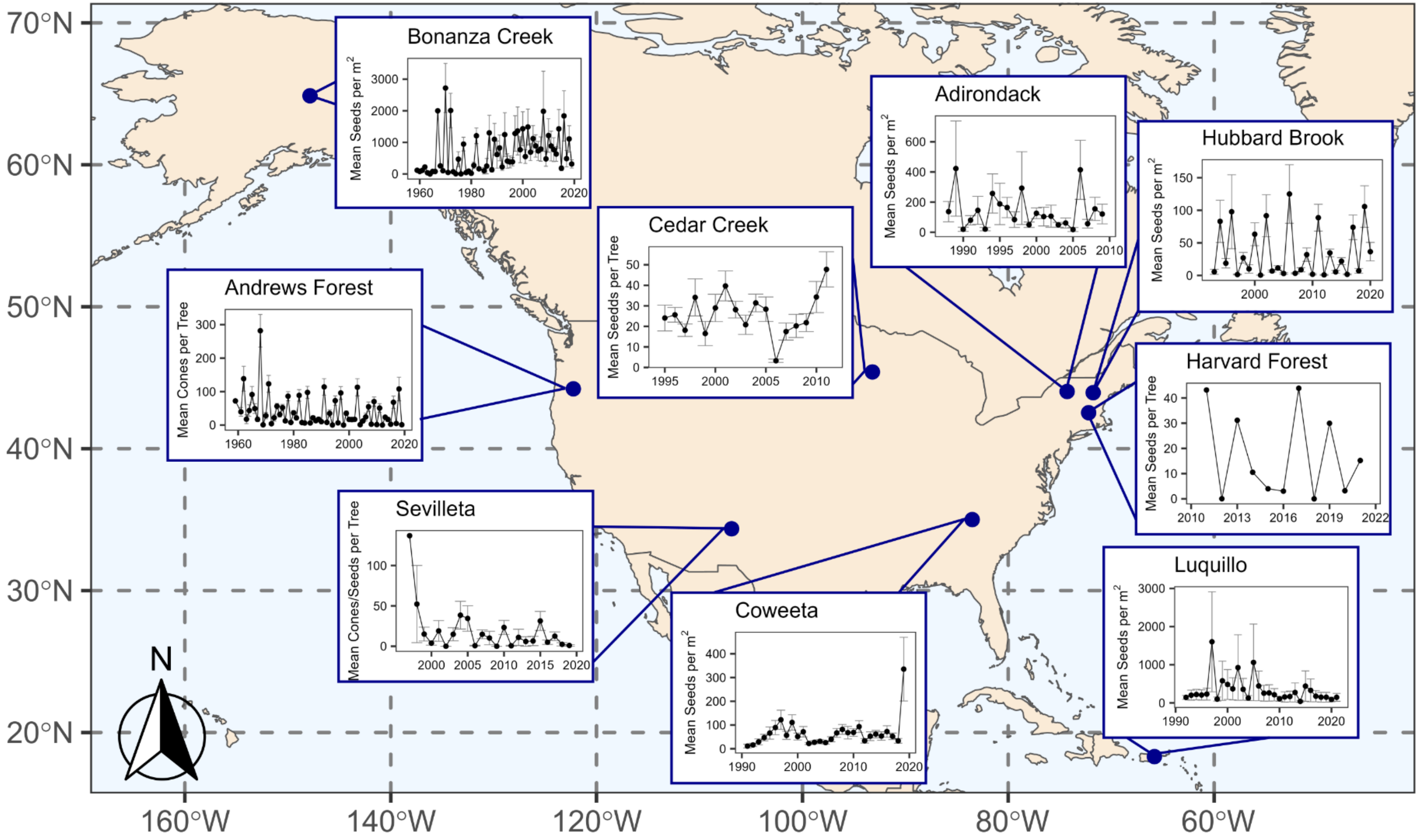
```r
dt |> filter(genus == "Quercus") |>
  ggplot(aes(x = year,
             y = fruits,
             color = species_name)) +
  geom_point(alpha = 0.4, size = 2) +
  geom_smooth() +
  scale_color_viridis_d() +
  facet_wrap(~species_name, scales = "free", nrow = 1) +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 60, hjust = 1),
        strip.text = element_text(face = "italic")) +
  labs(x = "Year",
       y = "Fruits per m2")
```

Ejercicio 2


Quercus_alba
Quercus_ellipsoidalis
Quercus_macrocarpa
Quercus_rubra
Quercus_turbinella
Fruits per m2
Year
0
50
100
150
1995
1996
2000
2004
2005
2008
2010
2015
2020
20
40
60

Ejercicio 3

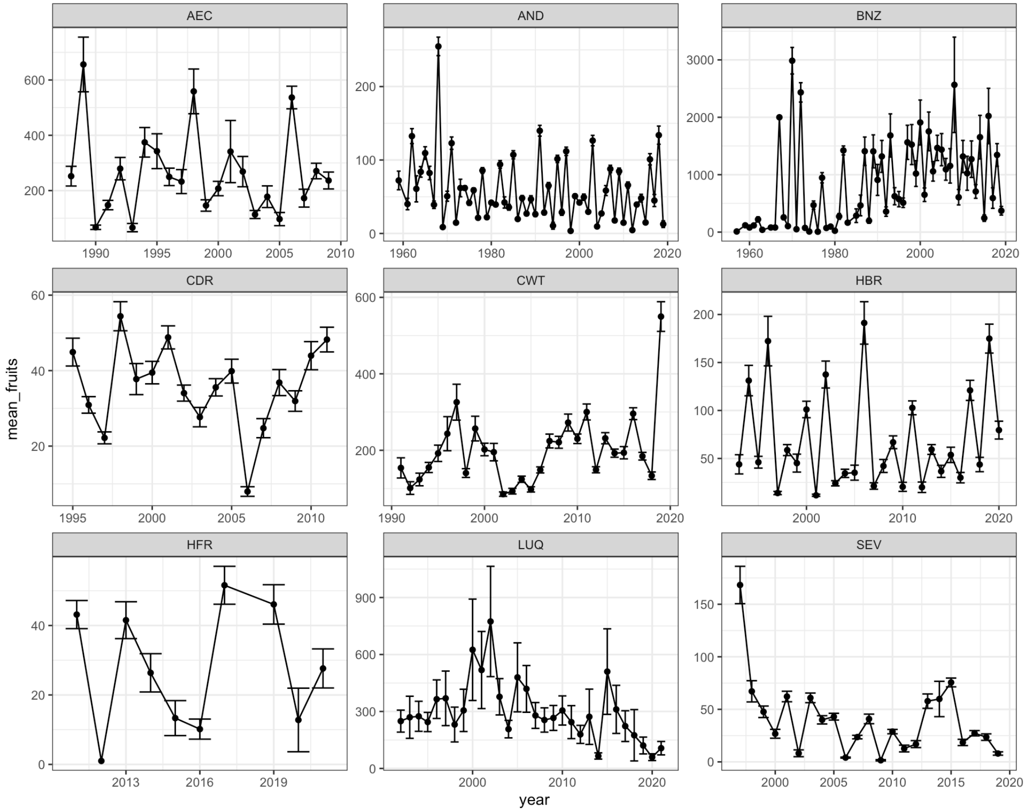
Replica las gráficas de esta figura (lo más similar posible):



Ejercicio 3

Alternativa 1 - Sacar primero la media y la SE del número de frutos y luego graficar

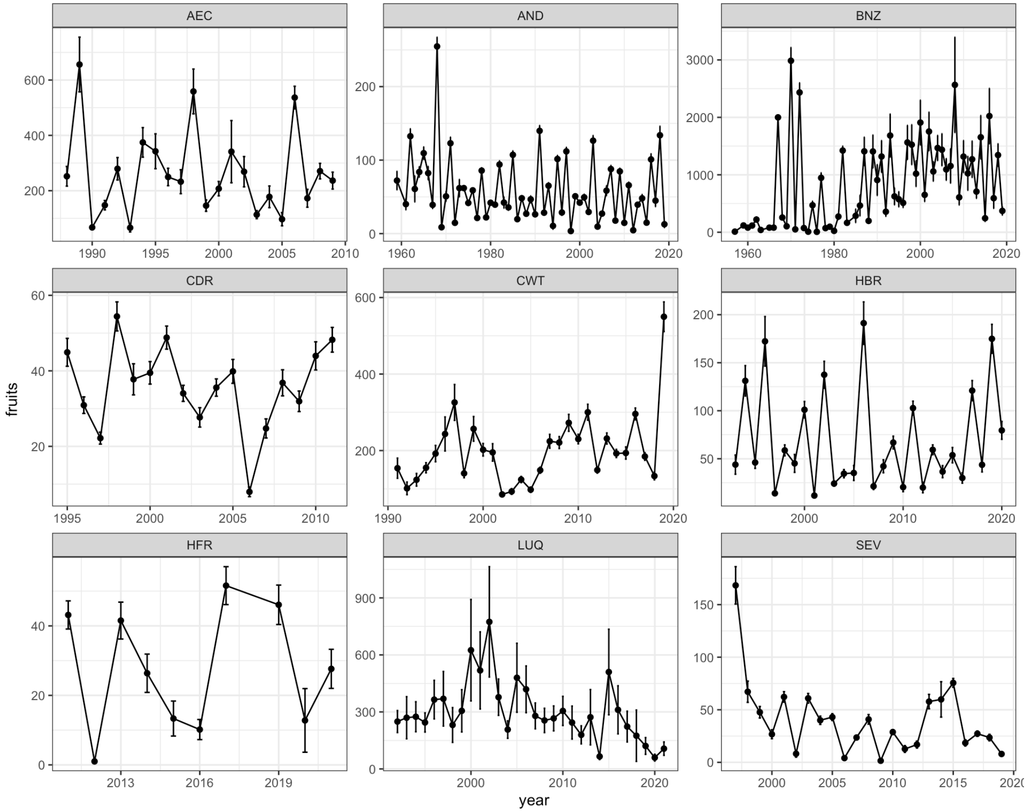
```
dt |>
  #remove outlier year
  filter(!(site == "BNZ" & year == 1958)
  group_by(site, year) |>
  summarise(mean_fruits = mean(fruits, na
            se_fruits = sd(fruits, na.rm
  ggplot(aes(x = year, y = mean_fruits))
  geom_point() +
  geom_errorbar(aes(ymin = mean_fruits -
                    ymax = mean_fruits +
  geom_line() +
  facet_wrap(~site, scales = "free") +
  theme_bw()
```



Ejercicio 3

Alternativa 2 - Sacar la media y la SE de los frutos dentro de ggplot

```
dt |>
  #remove outlier year
  filter(!(site == "BNZ" & year == 1958)
  ggplot(aes(x = year, y = fruits)) +
  geom_point(stat = "summary", fun = "mea
  geom_line(stat = "summary", fun = "mean
  stat_summary(
    fun.data = "mean_se",
    geom = "errorbar",
    width = 0.2) +
  facet_wrap(~site, scales = "free") +
  theme_bw()
```


AEC
AND
BNZ
CDR
CWT
HBR
HFR
LUQ
SEV
fruits
year