Visualización de datos en R

`ggplot`

Elena Quintero | Curso R AEET | 20 Sept 2022

Paquetes necesarios:

```
install.packages(c("here",
                   "tidyverse",
                   "patchwork",
                   "RColorBrewer"))
```

Paquetes necesarios:

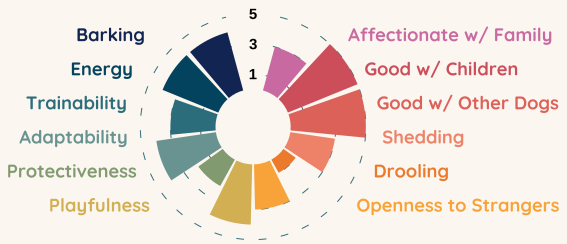
```
library(here)            #refiere la ruta a la carpeta del proyecto
library(tidyverse)       #incluye la librería ggplot2
library(ggplot2)         #VISUALIZACIÓN DE DATOS
library(RColorBrewer)    #paletas de color
library(plotly)          #hacer los gráficos interactivos
library(patchwork)       #combinar gráficos de ggplot
```

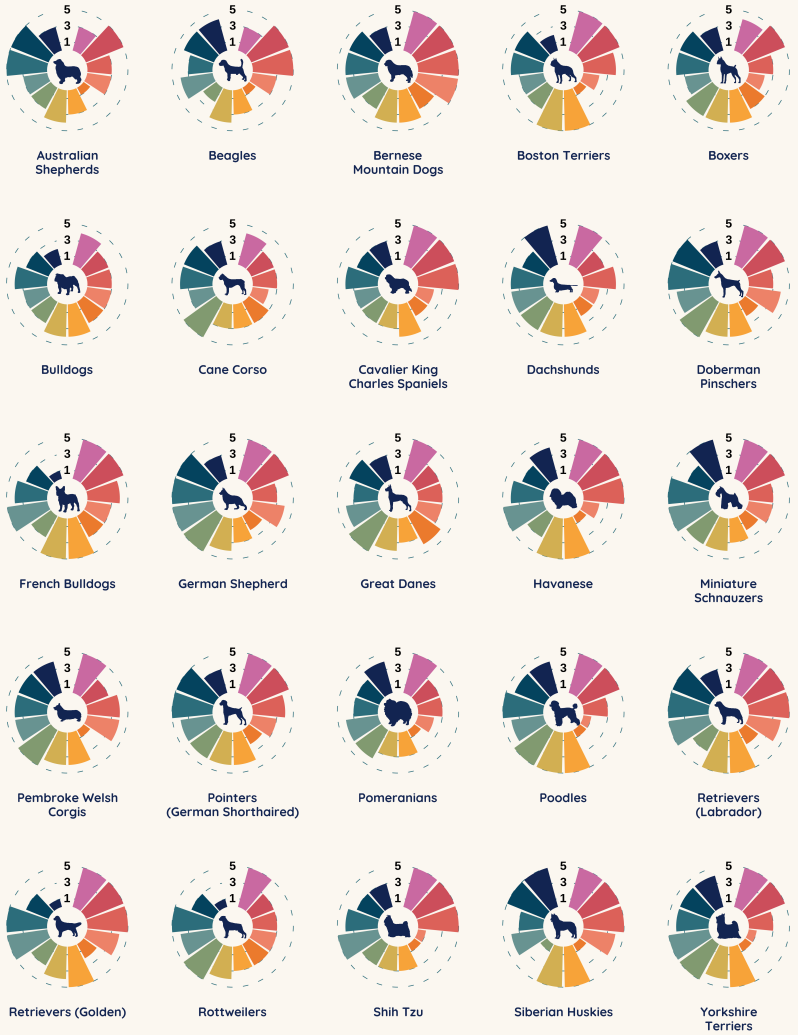
Beneficios de usar `library(ggplot2)`

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

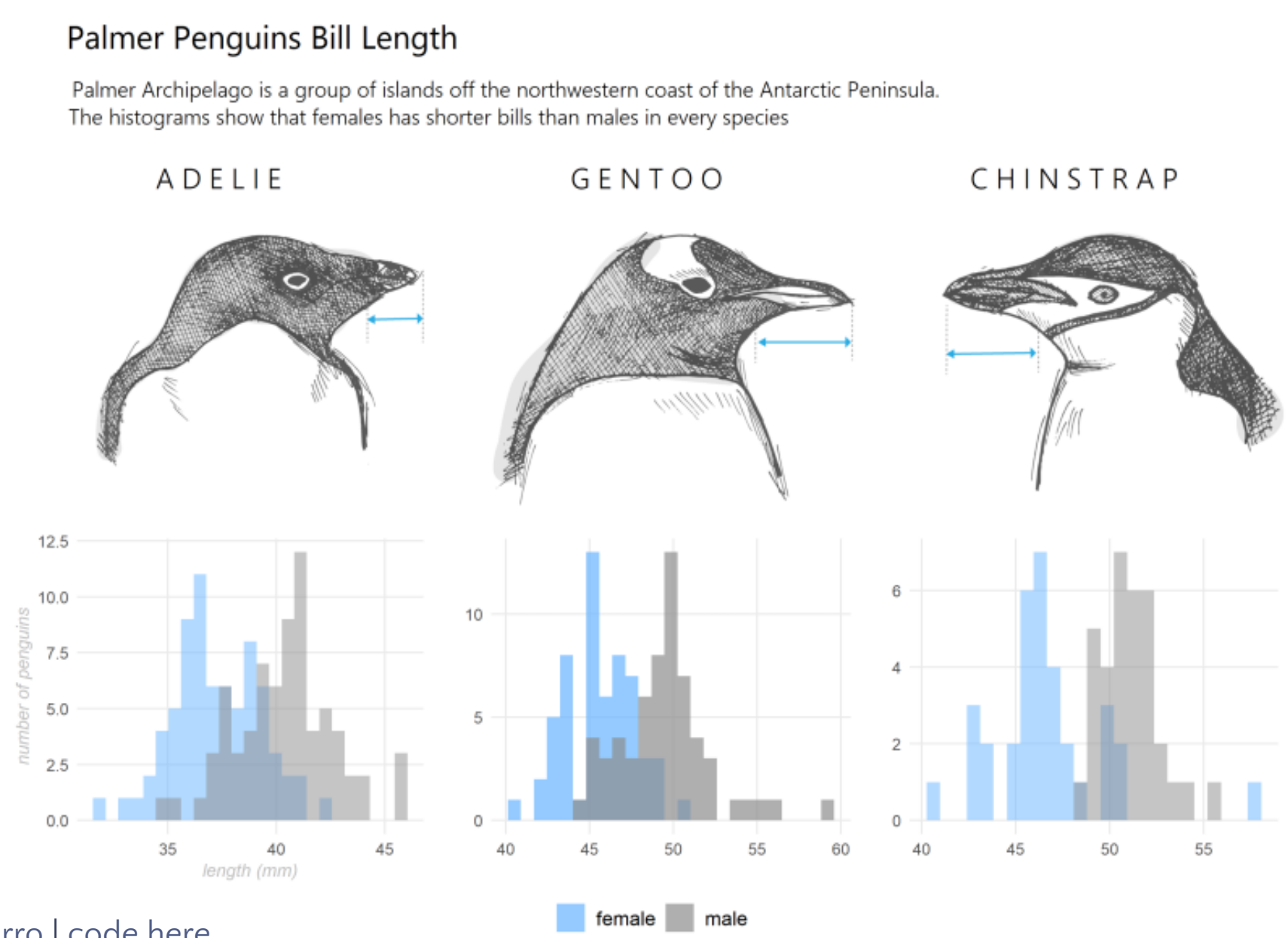
Dog Breeds

Plots show characteristic ratings of the 25 most popular dogs of 2020, according to the American Kennel Club. Ratings are based on a scale of 1 (low) to 5 (high).



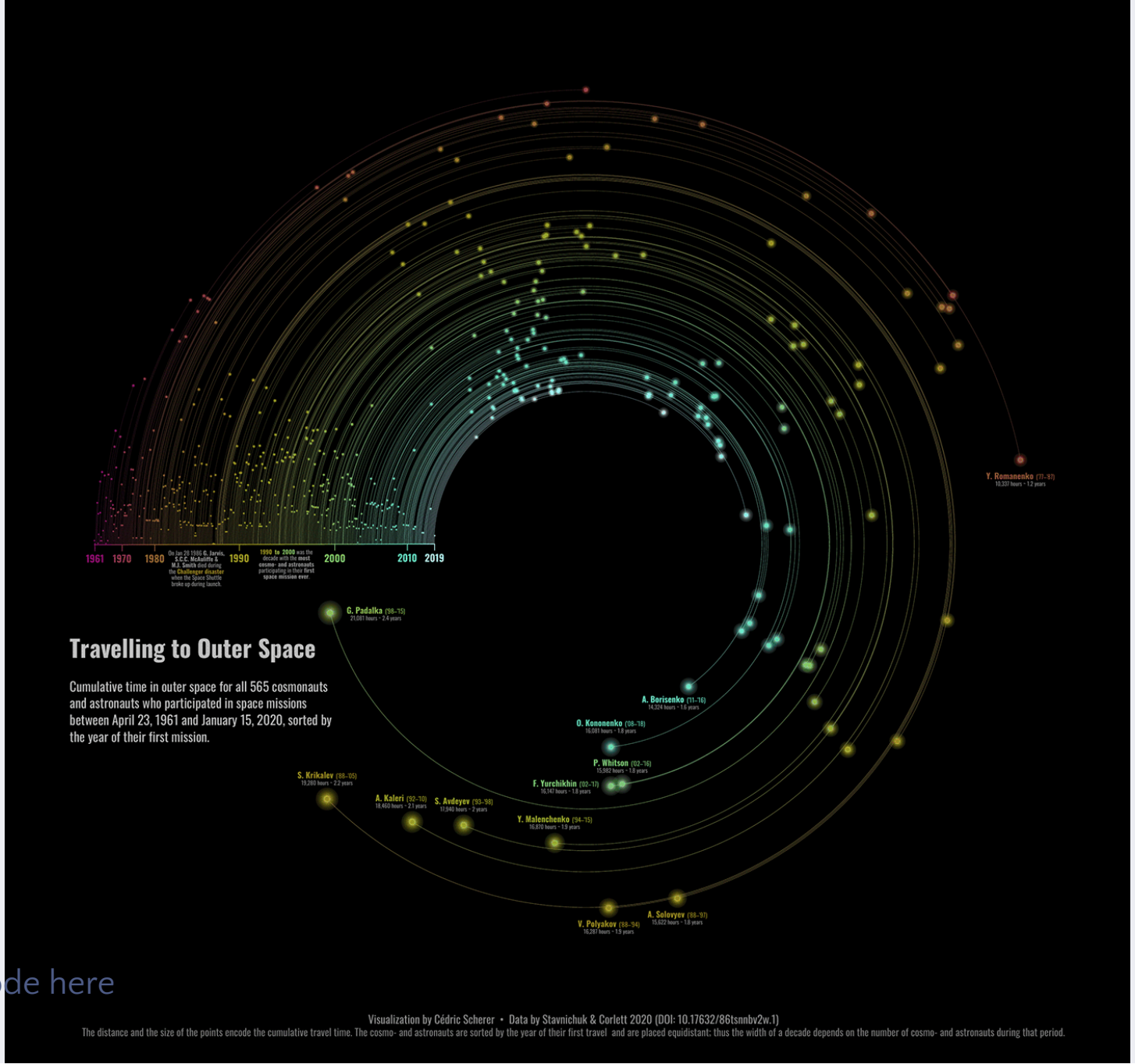


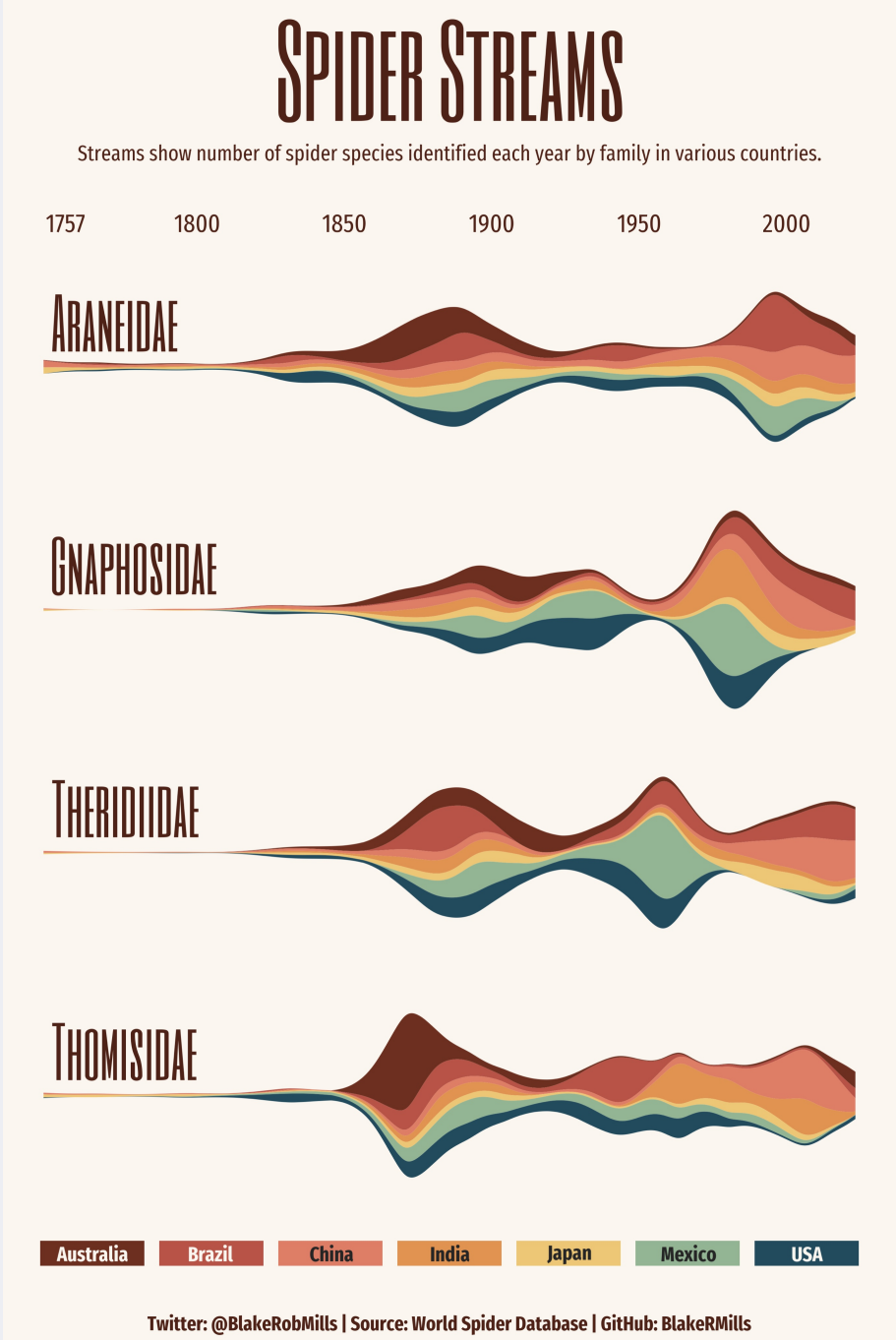
Twitter: @BlakeRobMills | Source: American Kennel Club | GitHub: BlakeRMills

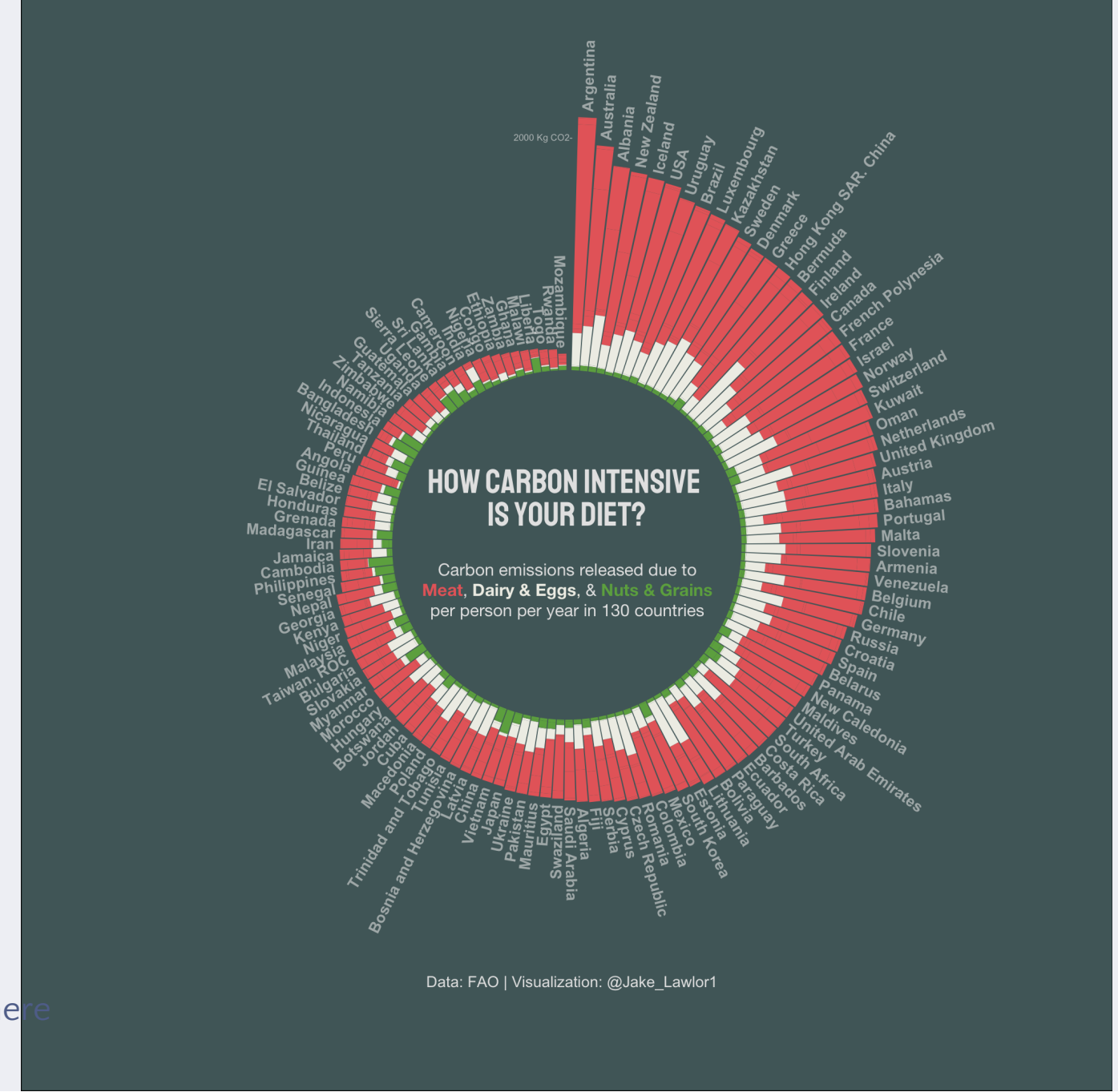


Laura Navarro | code here

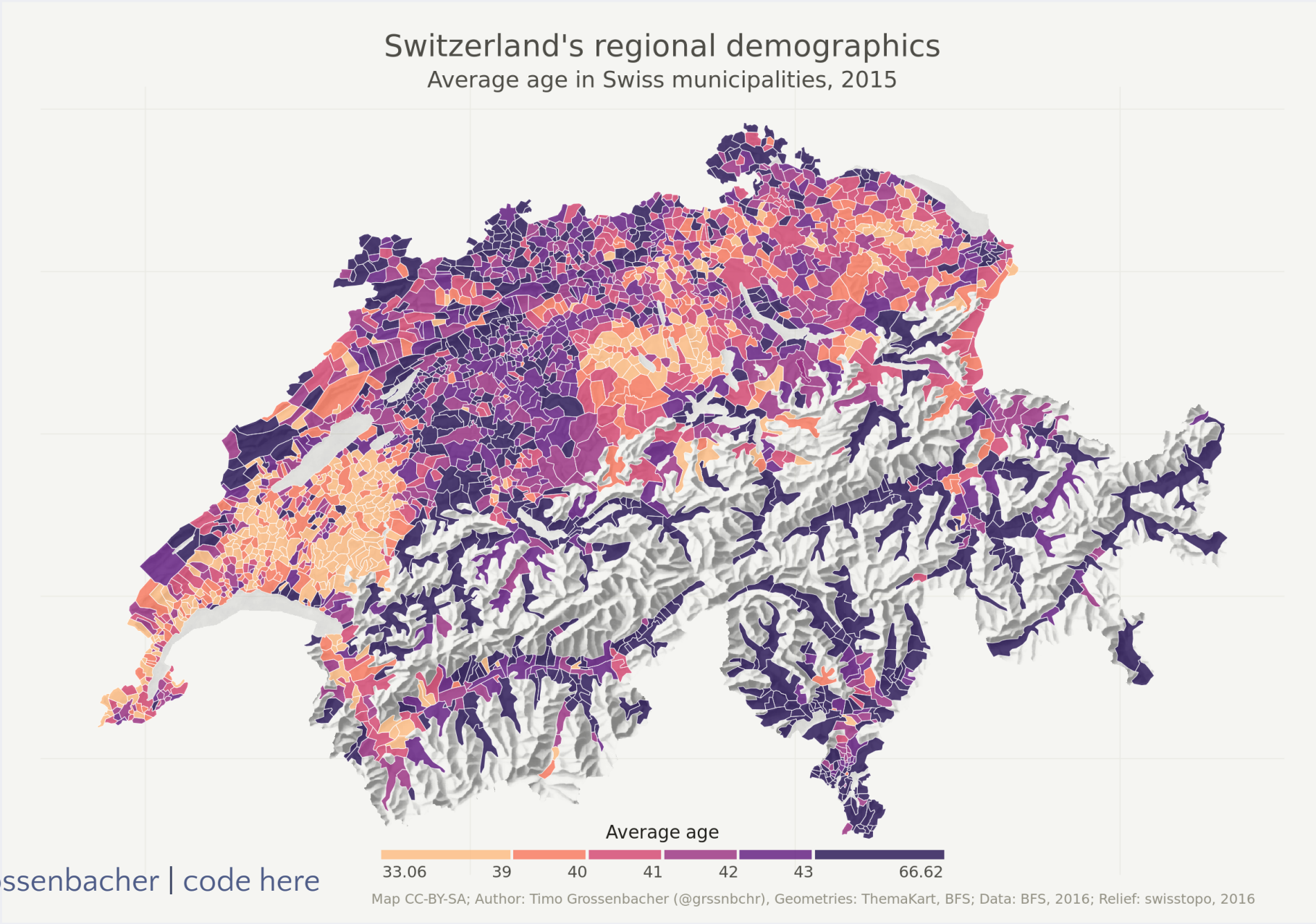
Visualization: Laura Navarro Soler | Data: Gorman, Williams & Fraser (2014)


1961
1970
1980
On Jan 28 1986 G. Jarvis-
S.C.C. McAuliffe &
M.J. Smith died during
the Challenger disaster
when the Space Shuttle
broke up during launch.
1990
1990 to 2000 was the
decade with the most
cosmo- and astronauts
participating in their first
space mission ever.
2000
2010
2019
Travelling to Outer Space
Cumulative time in outer space for all 565 cosmonauts and astronauts who participated in space missions between April 23, 1961 and January 15, 2020, sorted by the year of their first mission.
G. Padalka (98–15)
21,081 hours - 2.4 years
Y. Romanenko (77–87)
10,337 hours - 1.2 years
A. Borisenko (11–16)
14,224 hours - 1.6 years
O. Kononenko (08–18)
16,081 hours - 1.8 years
P. Whitson (02–16)
15,382 hours - 1.8 years
F. Yurchikhin (02–17)
16,147 hours - 1.8 years
S. Krikalev (88–05)
19,280 hours - 2.2 years
A. Kaleri (92–10)
18,483 hours - 2.1 years
S. Avdeyev (93–98)
17,940 hours - 2 years
Y. Malenchenko (94–15)
16,870 hours - 1.9 years
A. Solovyev (88–97)
15,622 hours - 1.8 years
V. Polyakov (88–94)
16,287 hours - 1.9 years
Visualization by Cédric Scherer • Data by Stavnichuk & Corlett 2020 (DOI: 10.17632/86tsnnbv2w.1)
The distance and the size of the points encode the cumulative travel time. The cosmo- and astronauts are sorted by the year of their first travel and are placed equidistant; thus the width of a decade depends on the number of cosmo- and astronauts during that period.


Spider Streams
Streams show number of spider species identified each year by family in various countries.
1757
1800
1850
1900
1950
2000
Araneidae
Gnaphosidae
Theridiidae
Thomisidae
Australia
Brazil
China
India
Japan
Mexico
USA
Twitter: @BlakeRobMills | Source: World Spider Database | GitHub: BlakeRMills



Jake Lawlor | code here



Timo Grossenbacher | code here

What do most people die from?

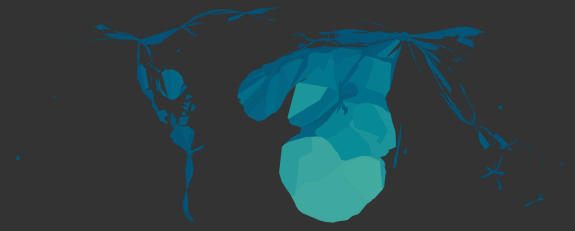
Cardiovascular Diseases

Cancers

Diabetes

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.

HIV Infections & Aids



Malaria Infections

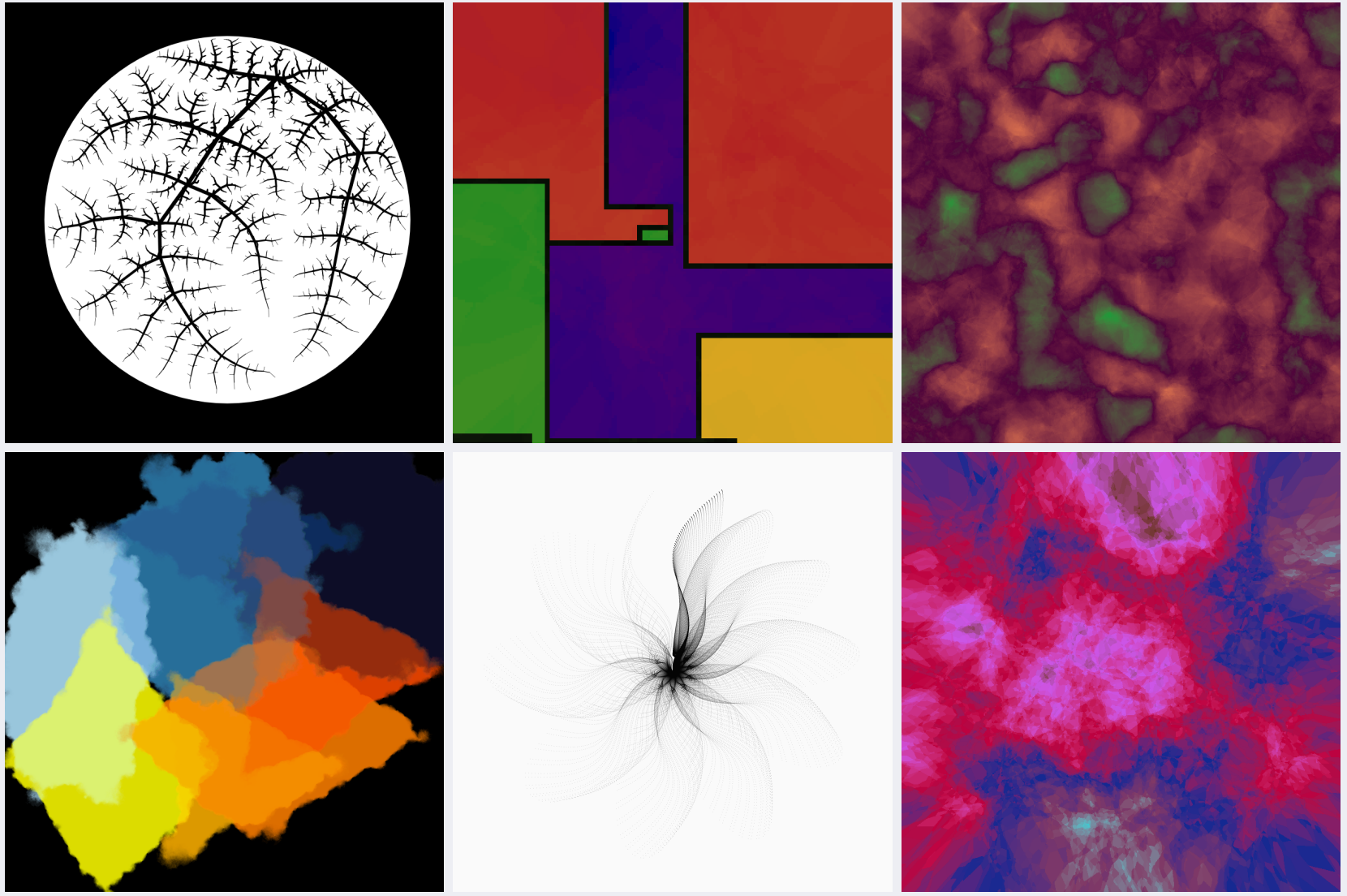
War & Conflicts

Share of Deaths

0% 5% 10% 15% 20% 25% 30% 35% 40% 45% 50% 55% 60% 65%

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



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.

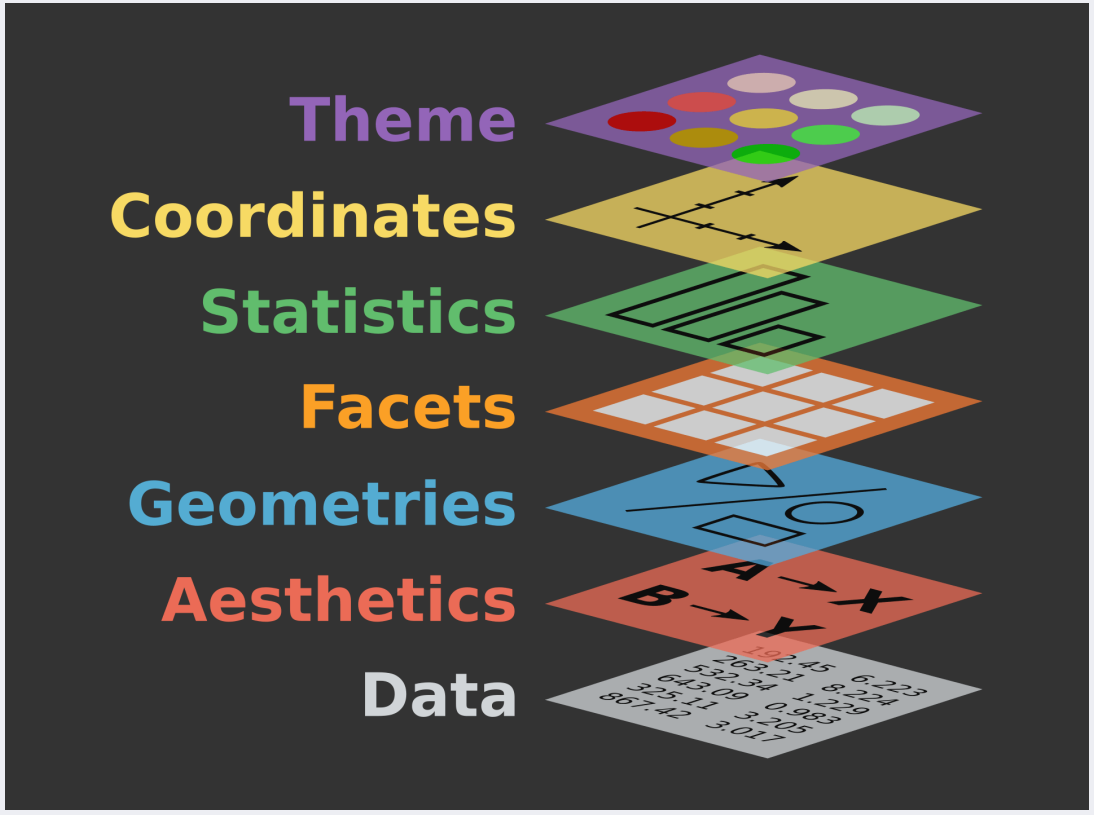


Illustration QCBS R workshop series

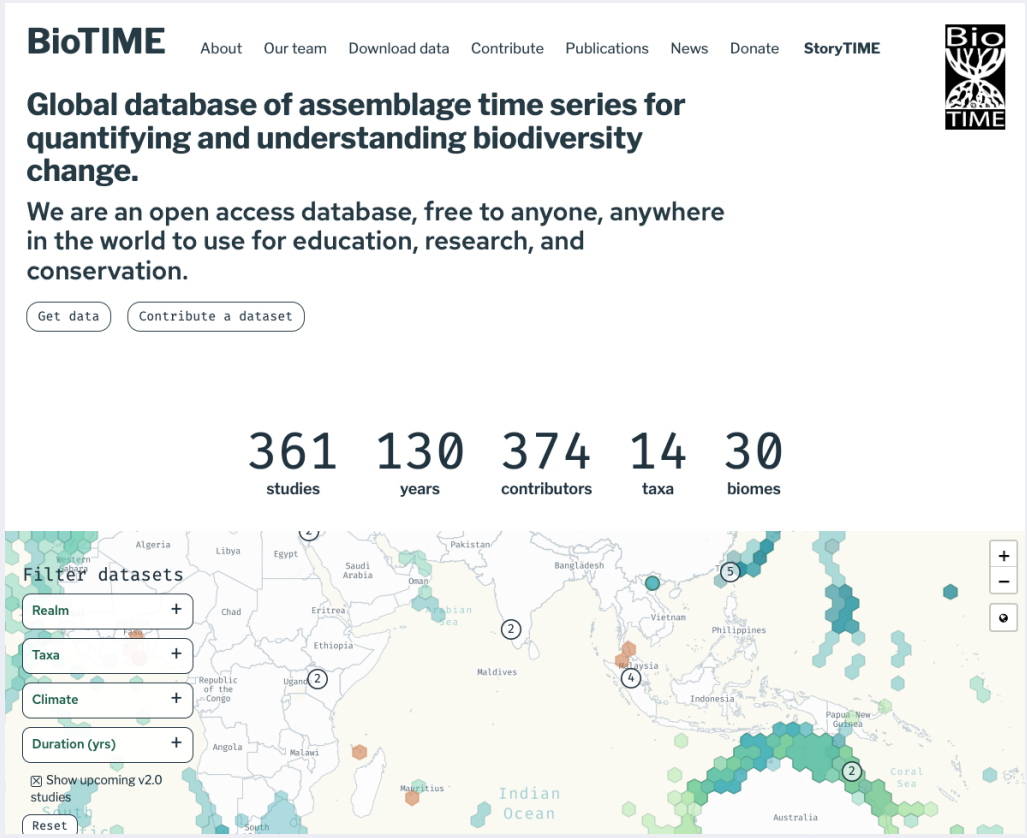
Capas:

- 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. . .)

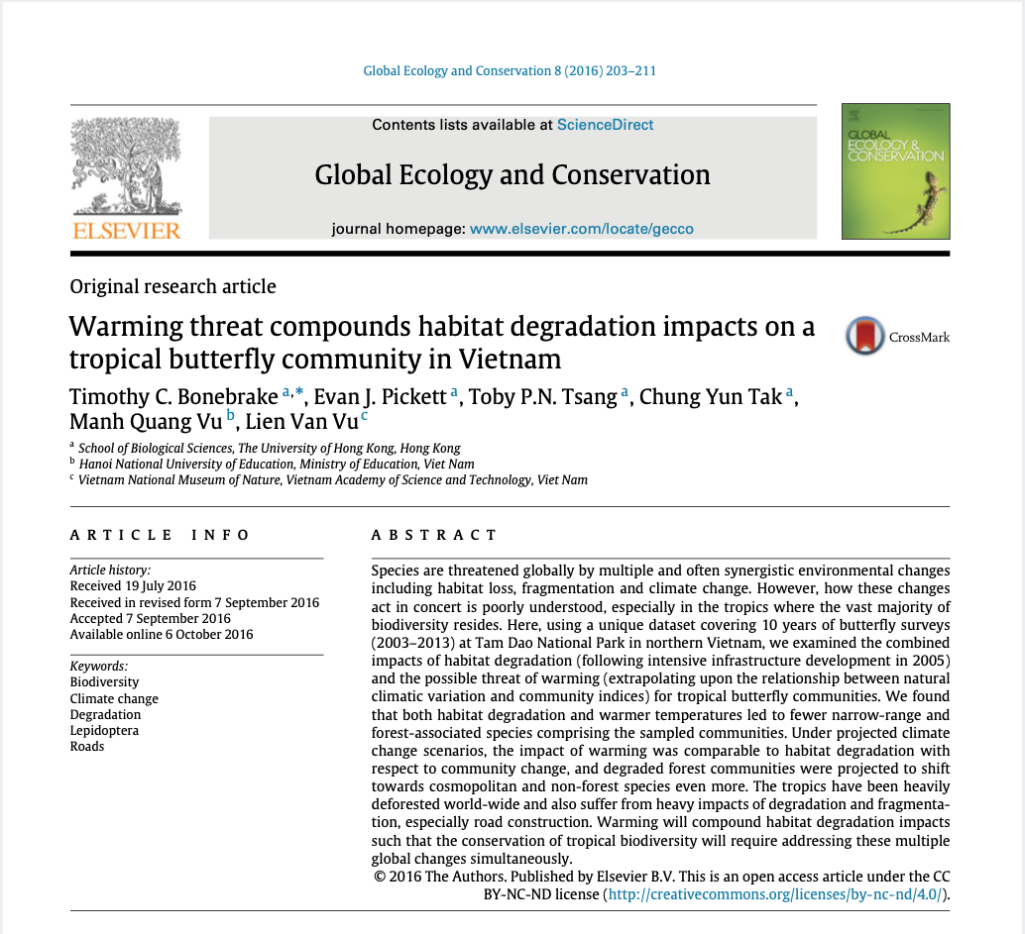
*Capas necesarias

Allison Horst Illustration

https://biotime.st-andrews.ac.uk/



Dornelas M, Antão LH, Moyes F, Bates, AE, Magurran, AE, et al. BioTIME: A database of biodiversity time series for the Anthropocene. Global Ecol Biogeogr. 2018; 27:760 - 786. https://doi.org/10.1111/geb.12729



Bonebrake, T.C., Pickett, E.J., Tsang, T.P., Tak, C.Y., Vu, M.Q. & Van Vu, L. (2016) Warming threat compounds habitat degradation impacts on a tropical butterfly community in Vietnam. Global Ecology and Conservation, 8, 203-211. http://dx.doi.org/10.1016/j.gecco.2016.09.003

Leer dataset

```
library(readr)
but <- read_csv(here("data/butterfly_dataset.csv"))
```

```
## Rows: 8356 Columns: 8
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (4): plot, species, habitat, hab_type
## dbl (4): abundance, day, month, year
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
glimpse(but)
```

```
## Rows: 8,356
## Columns: 8
## $ plot      <chr> "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "
## $ species   <chr> "Appias albina", "Argyreus hyperbius", "Ariadne ariadne", "Astictopterus jama", "Ceth
## $ abundance <dbl> 1, 1, 2, 1, 6, 1, 1, 5, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 3, 3, 5, 2, 3, 2, 6, 1
## $ day       <dbl> 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 1
## $ month     <dbl> 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 1
## $ year      <dbl> 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2
## $ habitat   <chr> "agriculture", "agriculture", "agriculture", "agriculture", "agriculture", "agricultu
## $ hab_type  <chr> "open", "open", "open", "open", "open", "open", "open", "open", "open", "open", "open
```

Convertir variable a formato fecha:

```
but <- but %>%
  mutate(date = str_c(year, "-", month, "-", day)) %>%
  mutate(date = as.Date(date, format = "%Y-%m-%d"))
```

Convertir datos a formato fecha:

```
glimpse(but)
```

```
## Rows: 8,356
## Columns: 9
## $ plot      <chr> "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "T1", "
## $ species   <chr> "Appias albina", "Argyreus hyperbius", "Ariadne ariadne", "Astictopterus jama", "Ceth
## $ abundance <dbl> 1, 1, 2, 1, 6, 1, 1, 5, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 3, 3, 5, 2, 3, 2, 6, 1
## $ day       <dbl> 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 1
## $ month     <dbl> 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 1
## $ year      <dbl> 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2003, 2
## $ habitat   <chr> "agriculture", "agriculture", "agriculture", "agriculture", "agriculture", "agricultu
## $ hab_type  <chr> "open", "open", "open", "open", "open", "open", "open", "open", "open", "open", "open
## $ date      <date> 2003-10-15, 2003-10-15, 2003-10-15, 2003-10-15, 2003-10-15, 2003-10-15, 2003-10-15,
```

Resumir datos:

Calcular número de especies detectadas y abundancia total de mariposas por muestreo.

De forma que cada fila representa un día de muestreo en un sitio concreto.

```
but_sum <- but %>%
  group_by(plot, habitat, hab_type, date) %>%
  summarise(n_species = n(),
            abundance = sum(abundance))
```

```
## `summarise()` has grouped output by 'plot', 'habitat', 'hab_type'. You can override using the `.groups`
```

```
#but_sum <- read_csv(here("data/buttefly_summary.csv"))
```

1. Data

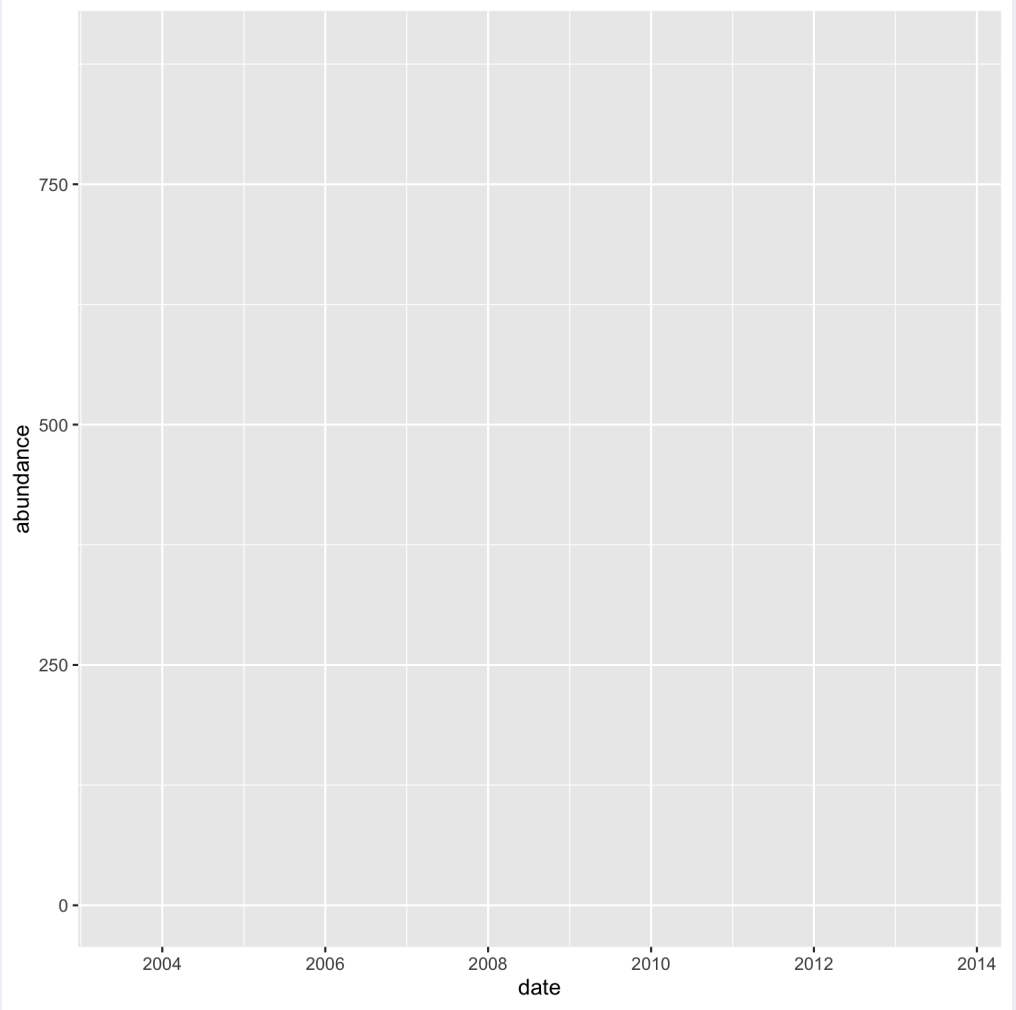
Dar a ggplot el dataset

```
ggplot(data = but_sum)
```

2. Aesthetics

Definir los aesthetics (variables a visualizar)

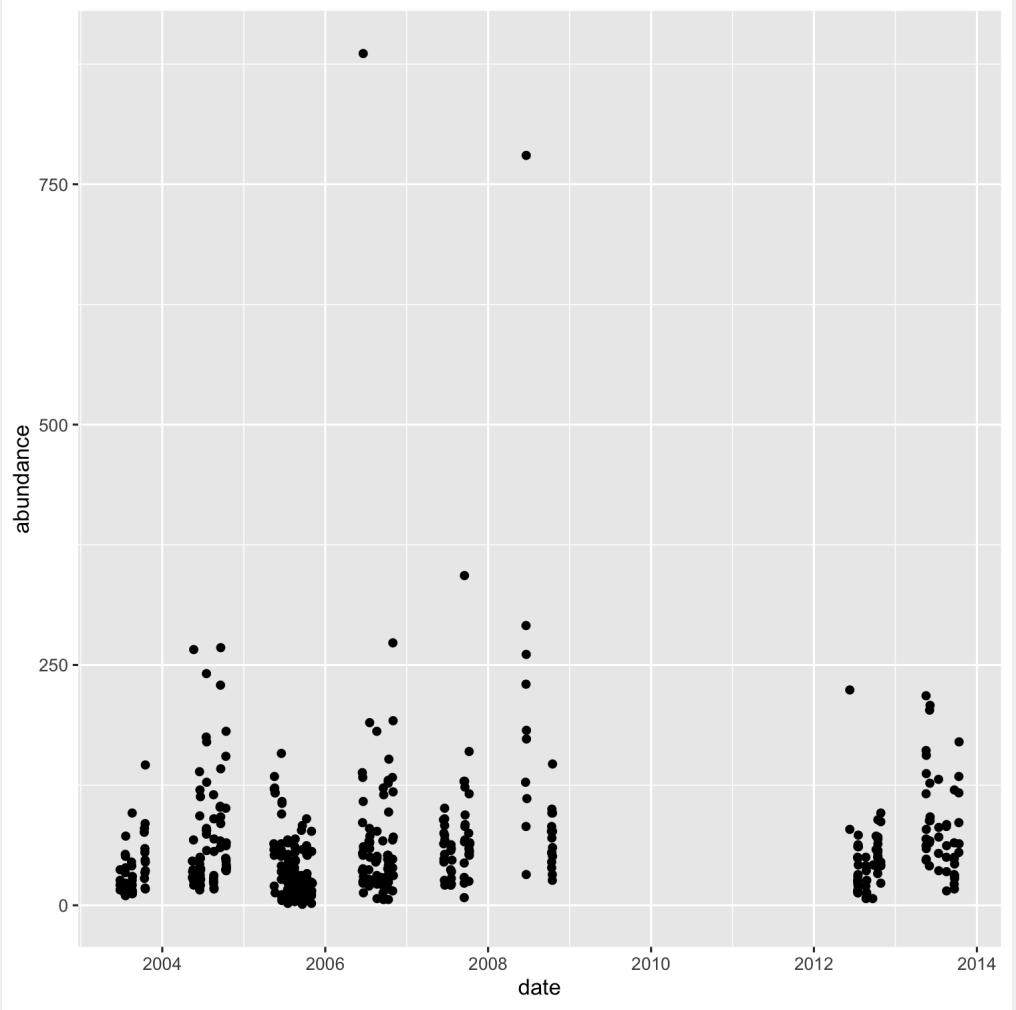
```
ggplot(but_sum, aes(x = date,
                    y = abundance))
```



3. Geoms

Determinar la forma de representar los datos

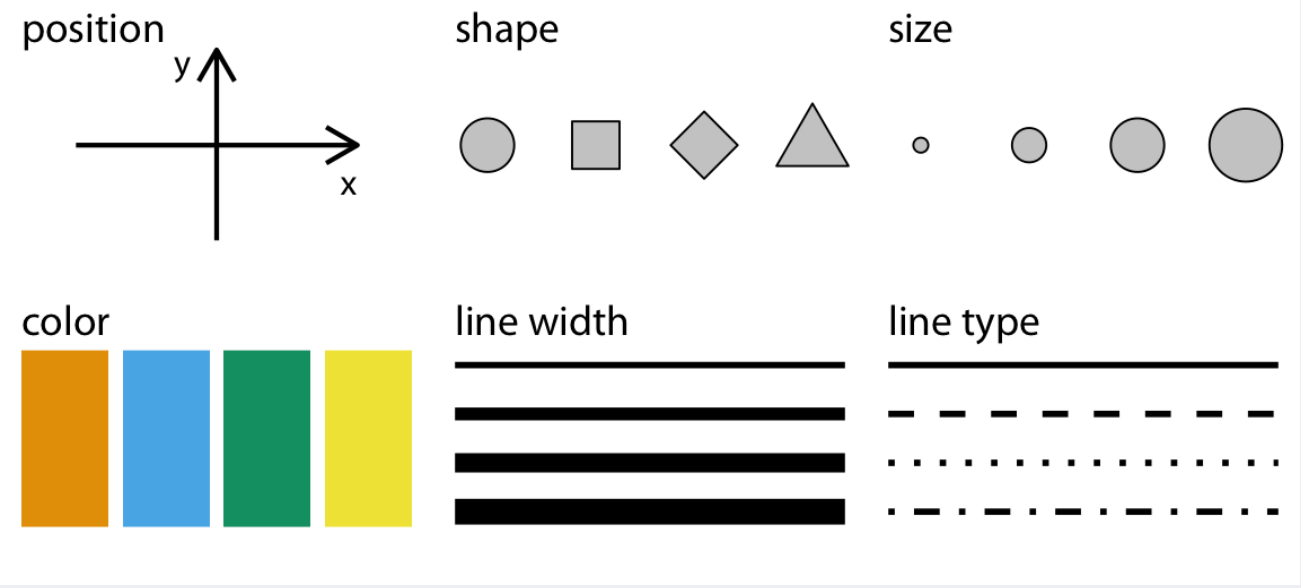
```
ggplot(but_sum, aes(x = date,
                    y = abundance)) +
  geom_point()
```



3. Geoms

Rasgos de los aesthetics:

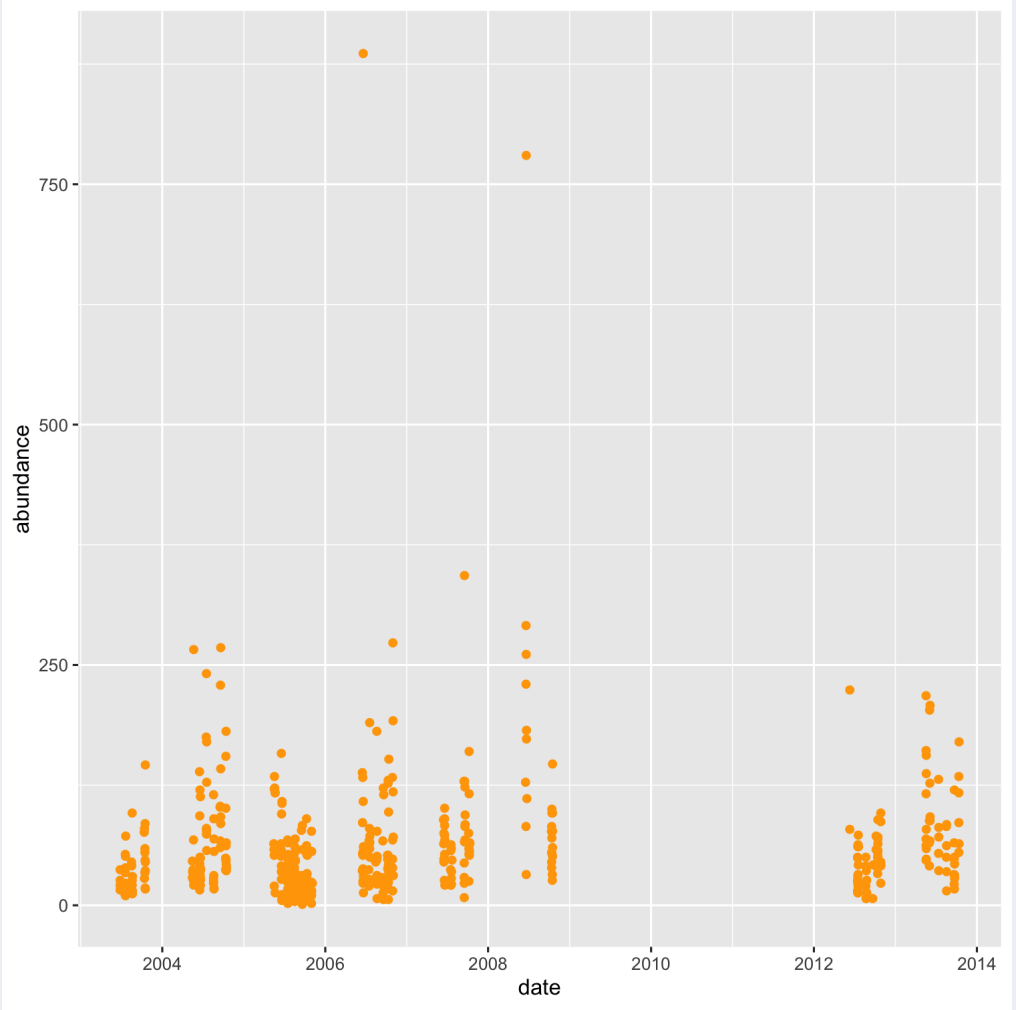
- position - x and y
- shape - integer/shape name or a single character ("a")
- size - integer (line width in mm)
- color and fill - string ("red", formato hexadecimal = #RRGGBB)
- linetype - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "dotdash", 5 = "longdash", 6 = "twodash")
- lineend - string ("round", "butt", or "square")
- linejoin - string ("round", "mitre", or "bevel")



3. Geoms

Añadir color a los puntos

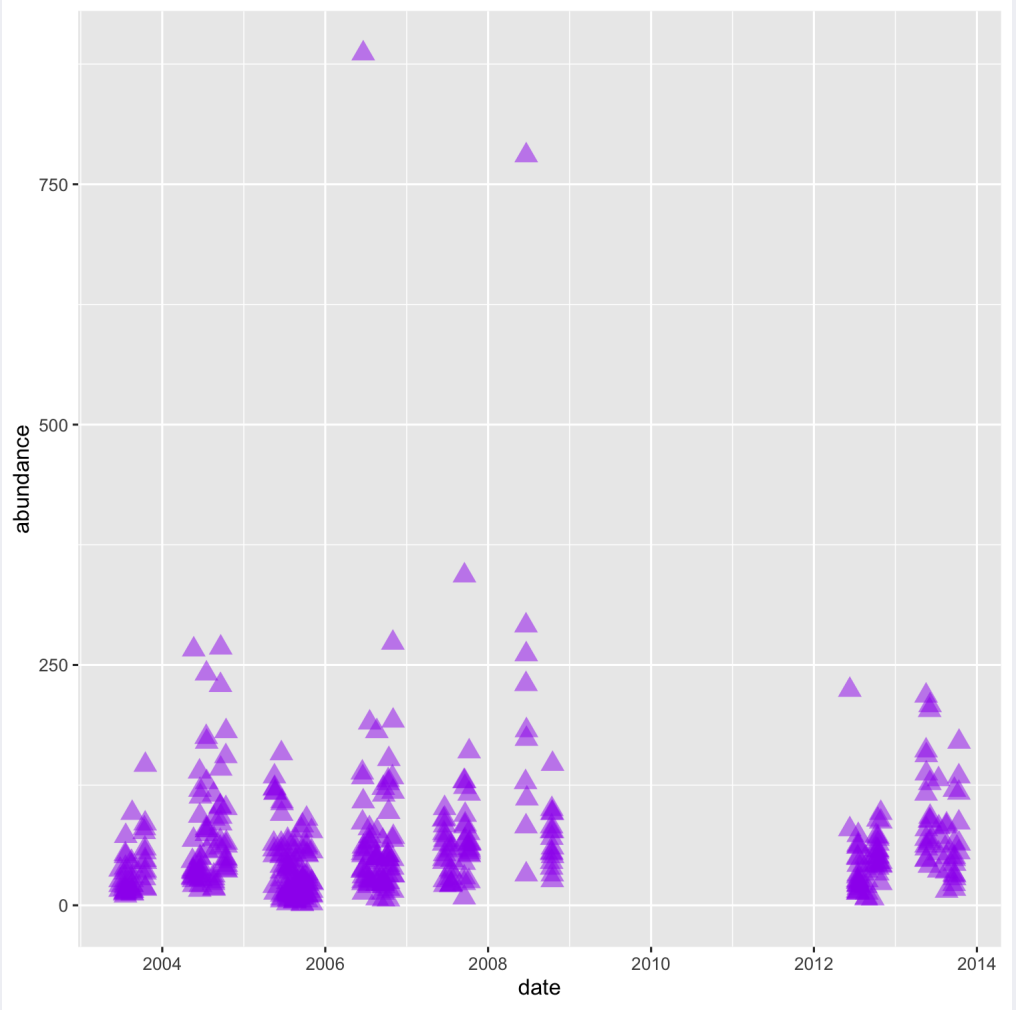
```
ggplot(but_sum, aes(x = date,
                    y = abundance)) +
  geom_point(color = "orange")
```



3. Geoms

Modificar la estética de los puntos

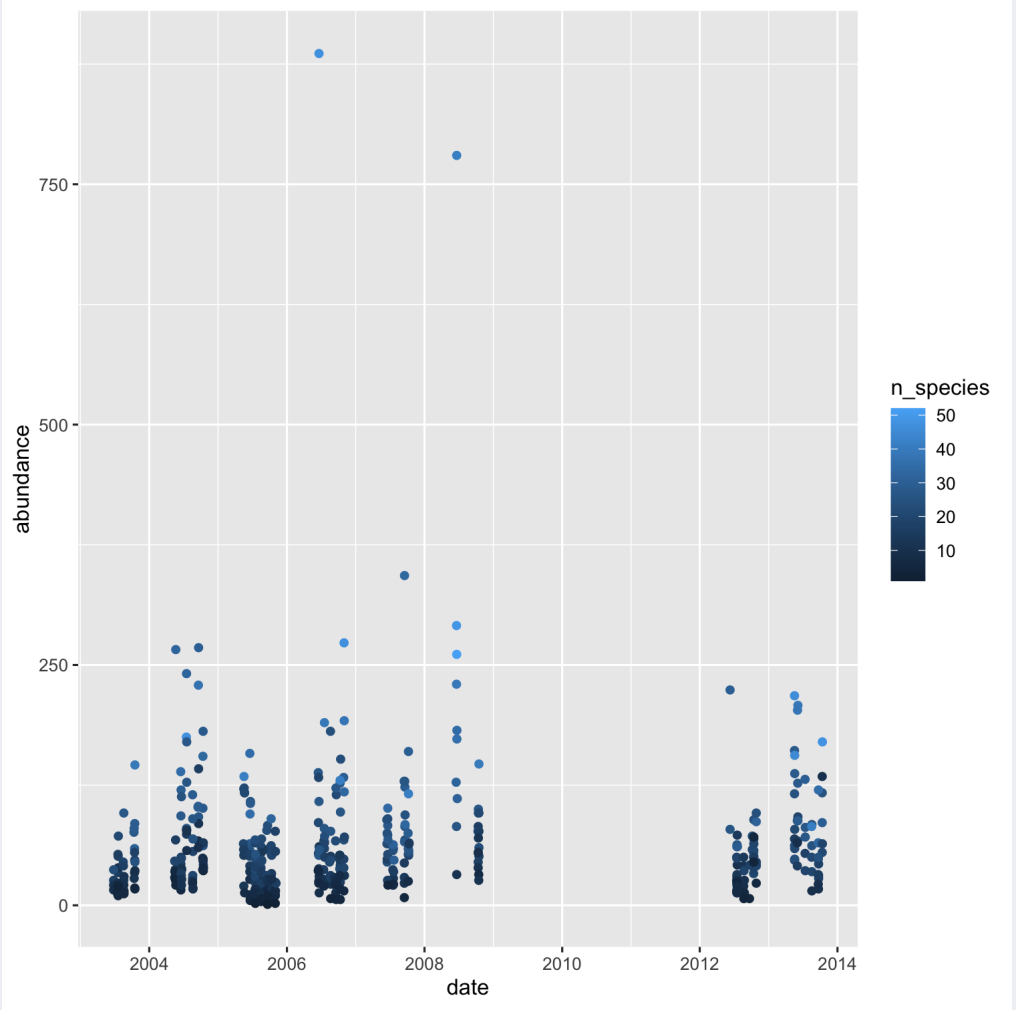
```
ggplot(but_sum, aes(x = date,
                    y = abundance)) +
  geom_point(color = "purple",
             shape = "triangle",
             size = 3.8,
             alpha = 0.5)
```



3. Geoms

Definir color basado en una variable

```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = n_species)) +
  geom_point()
```



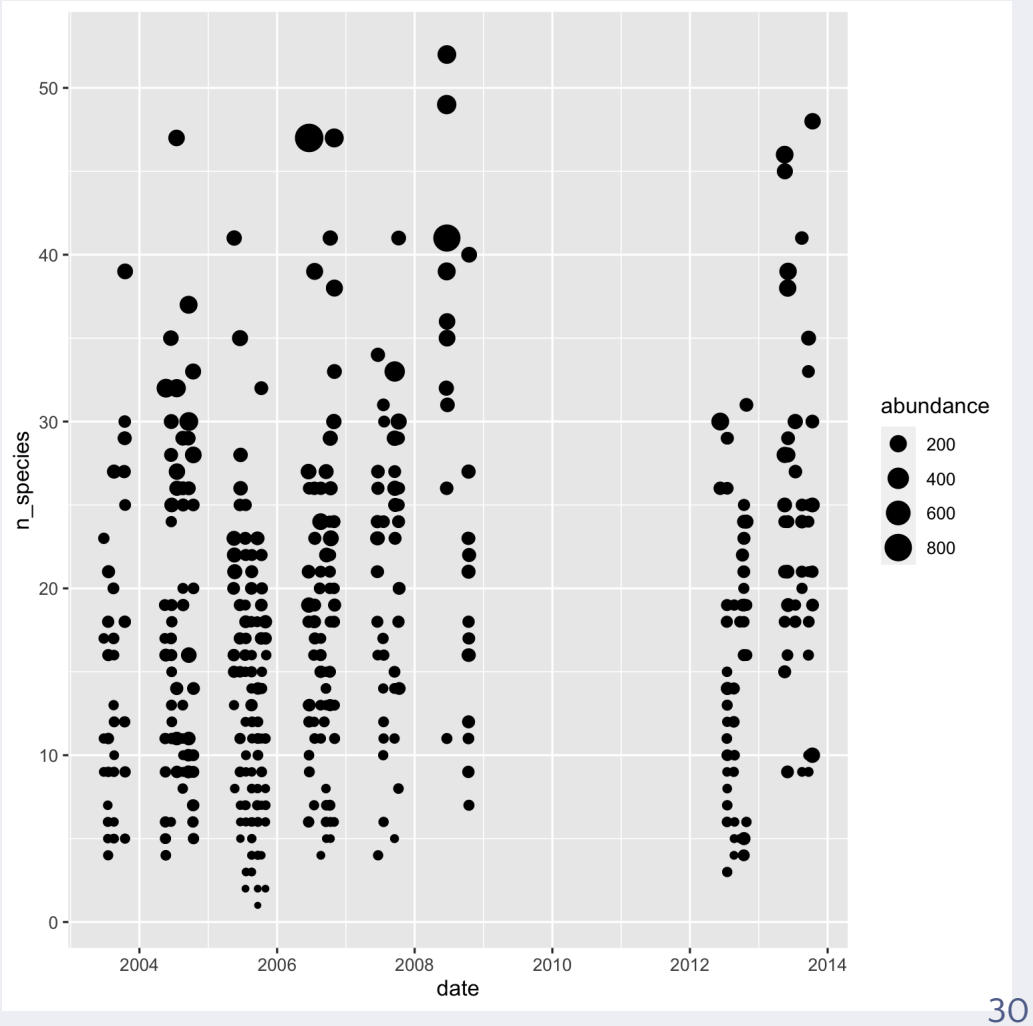
Ejercicio 1:

Graficar número de especies vs. tiempo, donde el tamaño del punto represente la abundancia de mariposas en el muestreo

Ejercicio 1:

Graficar número de especies vs. tiempo, donde el tamaño del punto represente la abundancia de mariposas en el muestreo

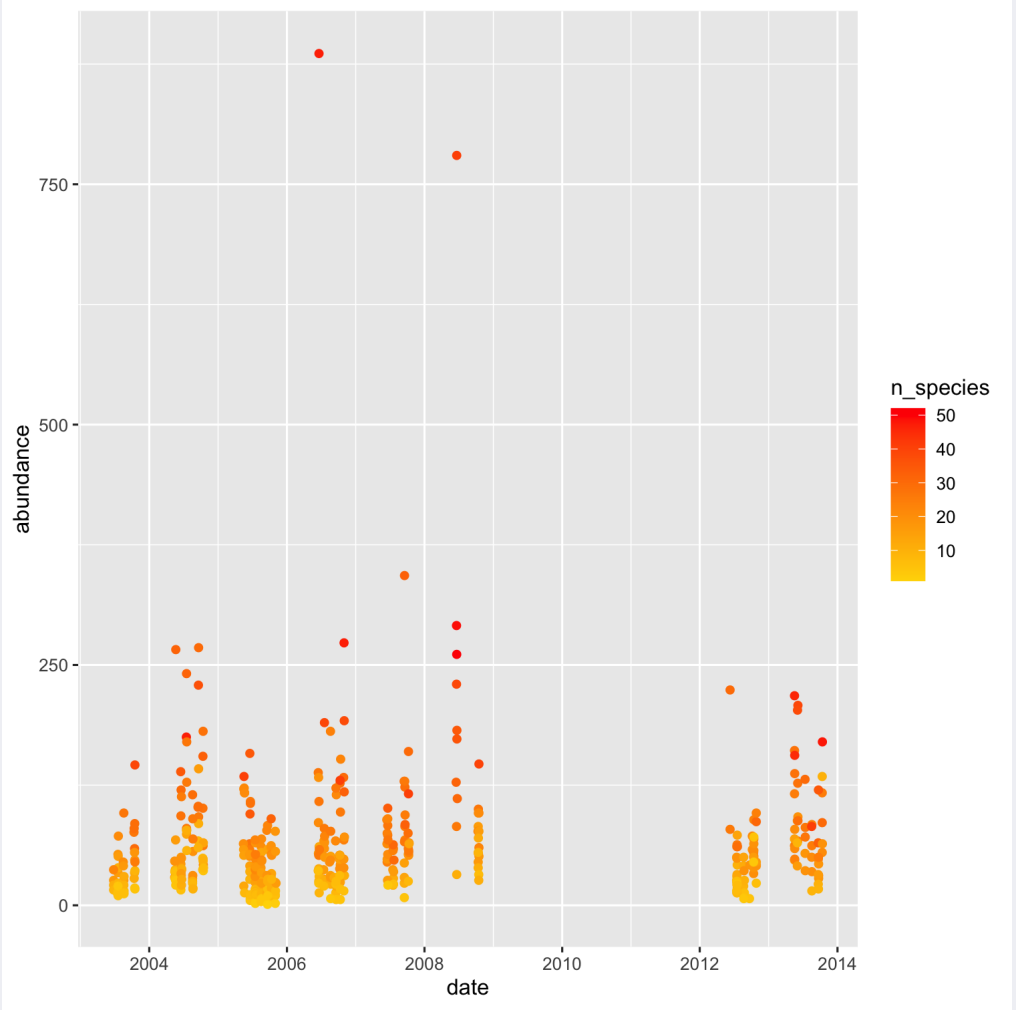
```
ggplot(but_sum, aes(x = date,
                    y = n_species,
                    size = abundance)) +
  geom_point()
```



4. Scales

Escala continua - dar valores de forma manual

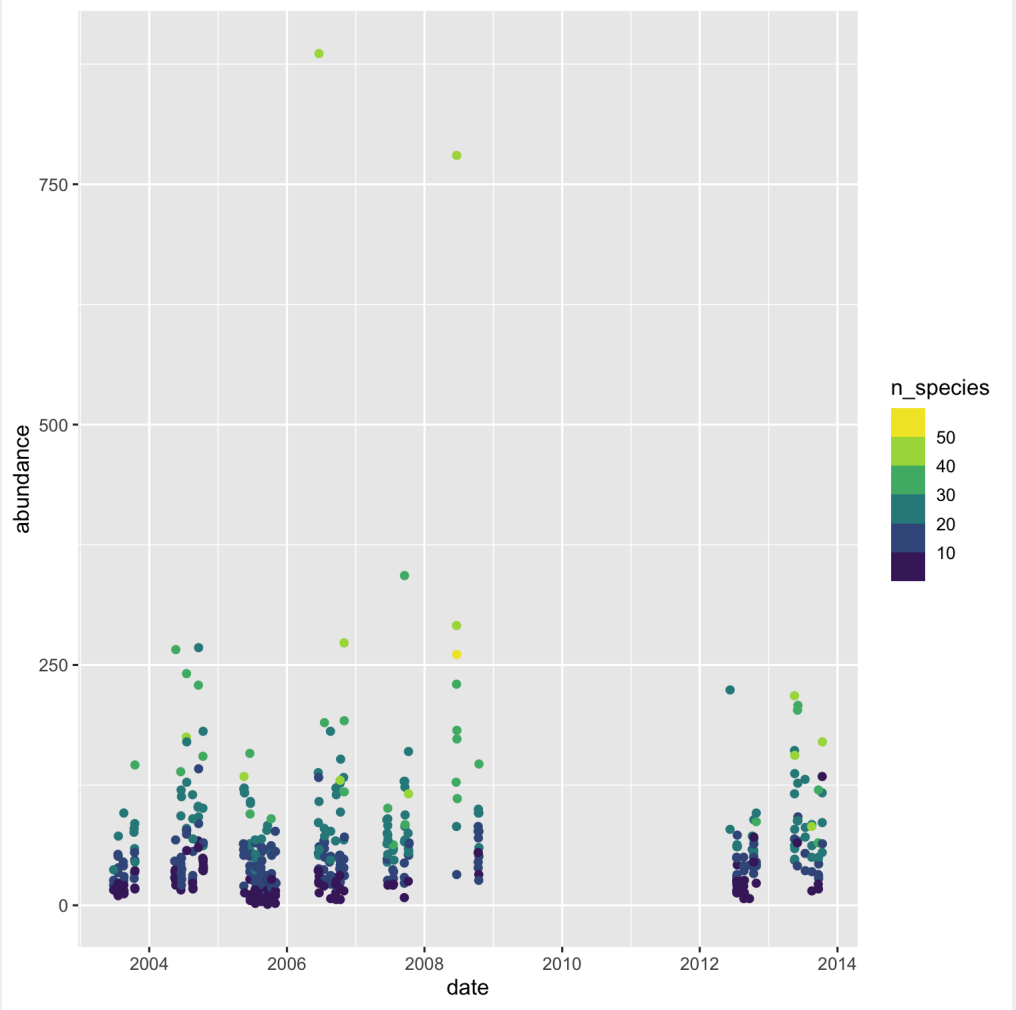
```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = n_species)) +
  geom_point() +
  scale_color_gradient(low = "gold",
                       high = "red")
```



4. Scales

Escala continua - paleta viridis

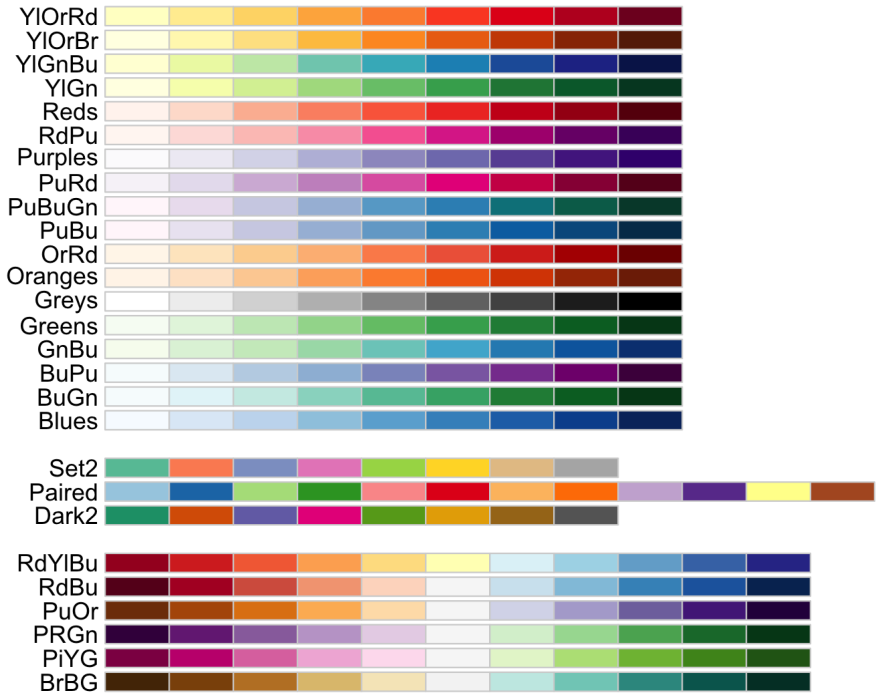
```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = n_species)) +
  geom_point() +
  scale_color_viridis_b()
```



4. Scales

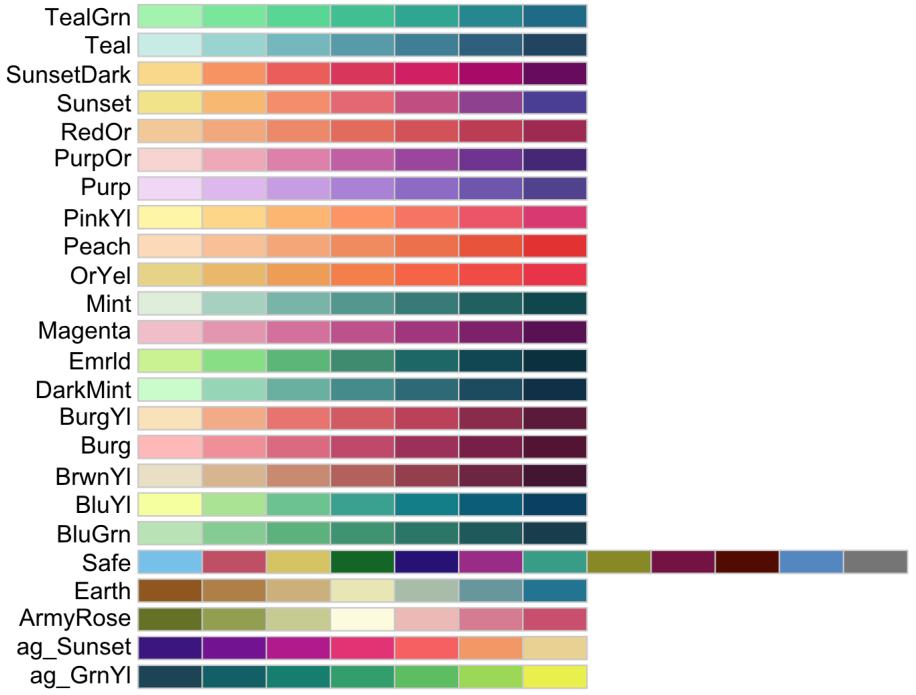
Paletas de color para ggplot

```
library(RColorBrewer)
display.brewer.all(
  colorblindFriendly = TRUE)
```



Paletas de color

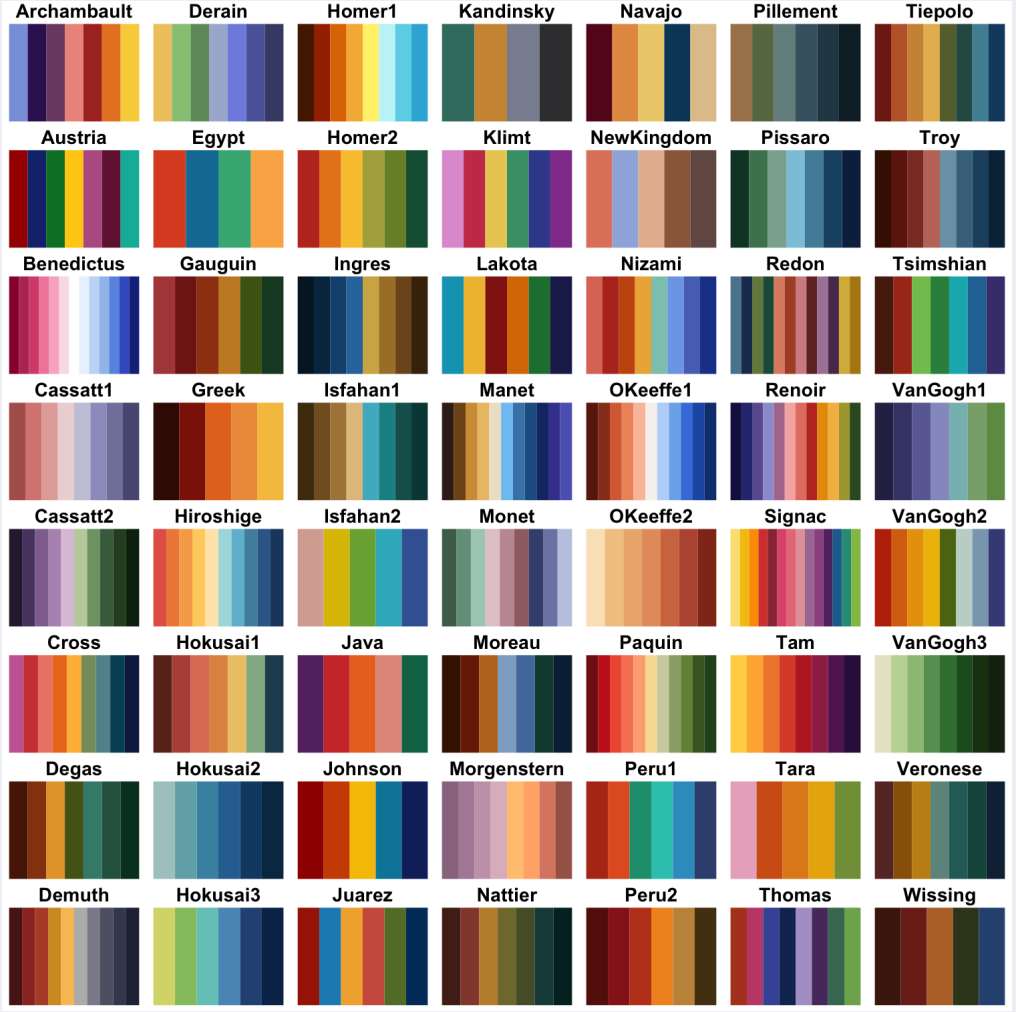
```
library(rcartocolor)

rcartocolor::display_carto_all(
  colorblind_friendly = TRUE)
```



Paletas de color

```
library(MetBrewer)

MetBrewer::display_all()
```

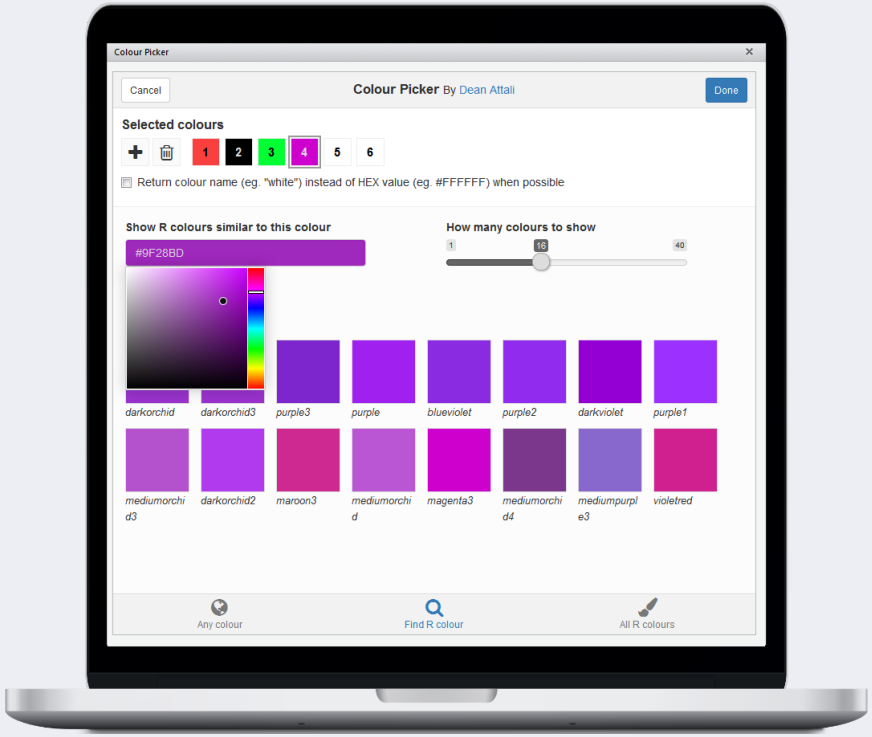
```
#colorblind_only = TRUE
```



Paletas de color

Addin de RStudio para seleccionar colores.

```
library(colourpicker)
```



https://github.com/daattali/colourpicker

Paletas de color

Páginas web para elegir colores:

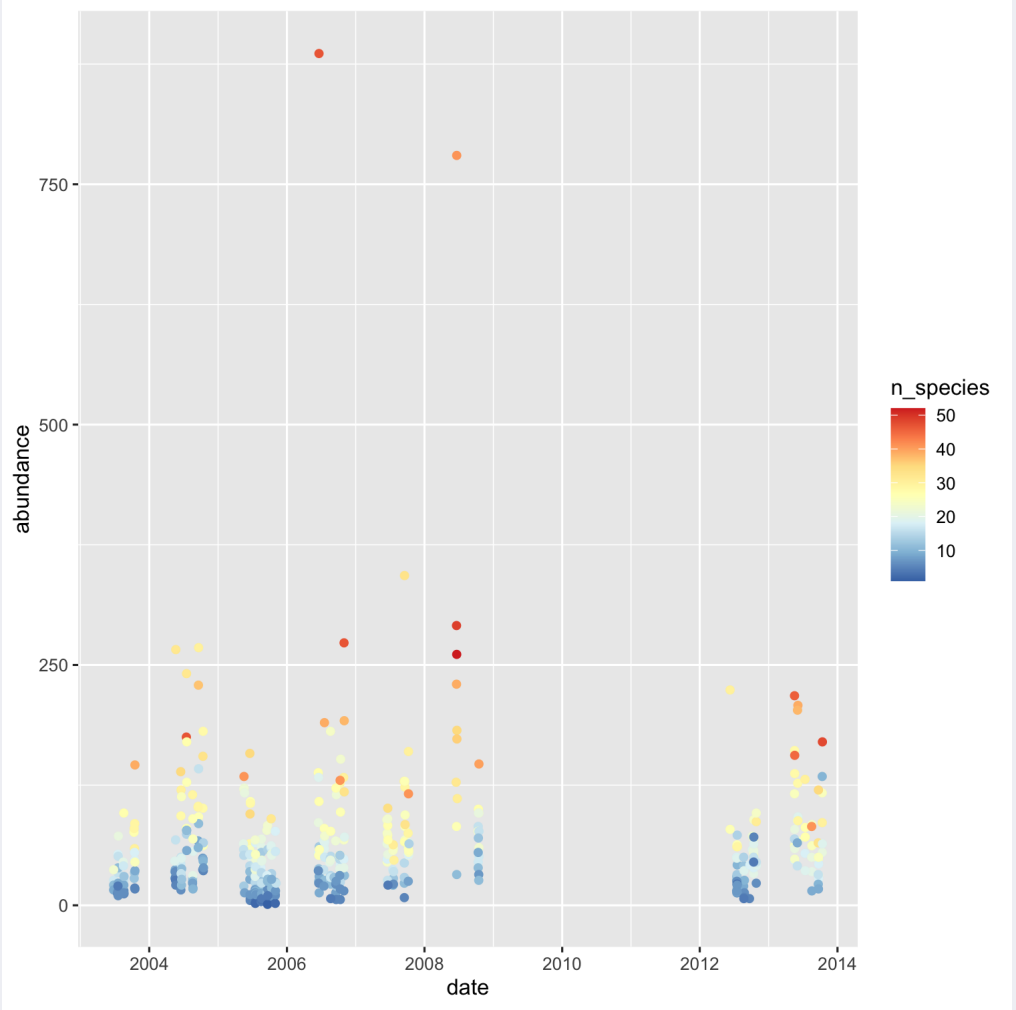
- https://colorbrewer2.org/
- https://projects.susielu.com/viz-palette
- https://medialab.github.io/iwanthue/

Útil para generar figuras adecuadas para daltonismo.

4. Scales

Escala de valores continuos

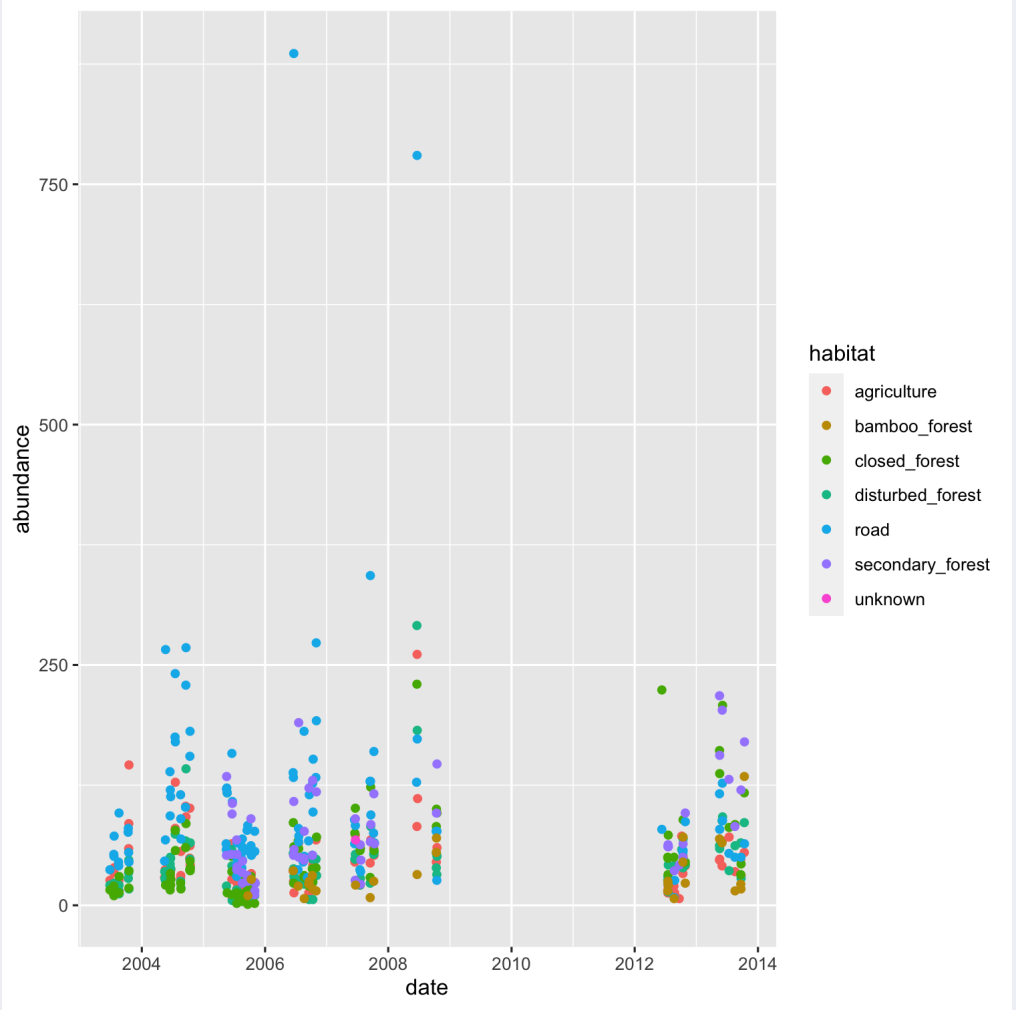
```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = n_species)) +
  geom_point() +
  scale_color_distiller(palette = "RdYlBu")
```



4. Scales

Color basado en una escala discreta (ej. hábitat)

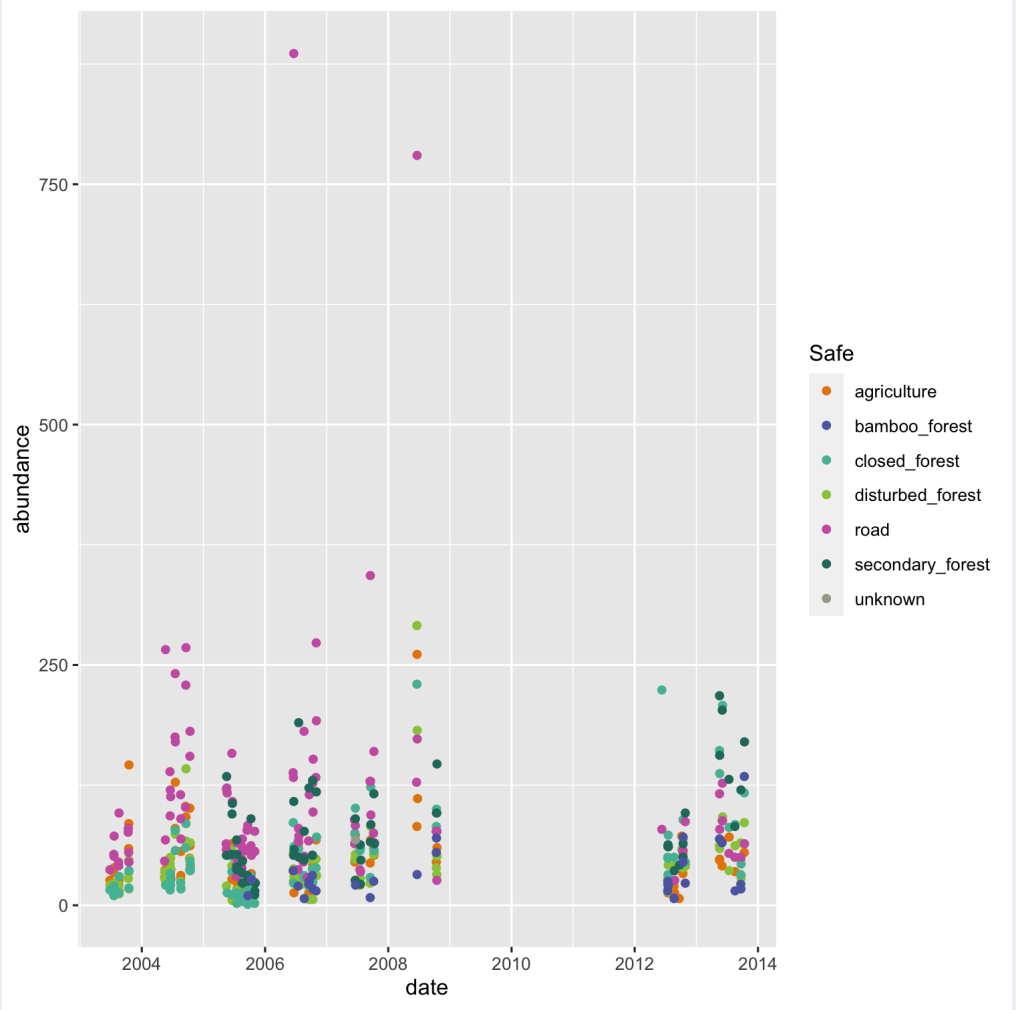
```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = habitat)) +
  geom_point()
```



4. Scales

Color basado en una escala discreta - color brewer

```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = habitat)) +
  geom_point() +
  rcartocolor::scale_color_carto_d("Safe")
```

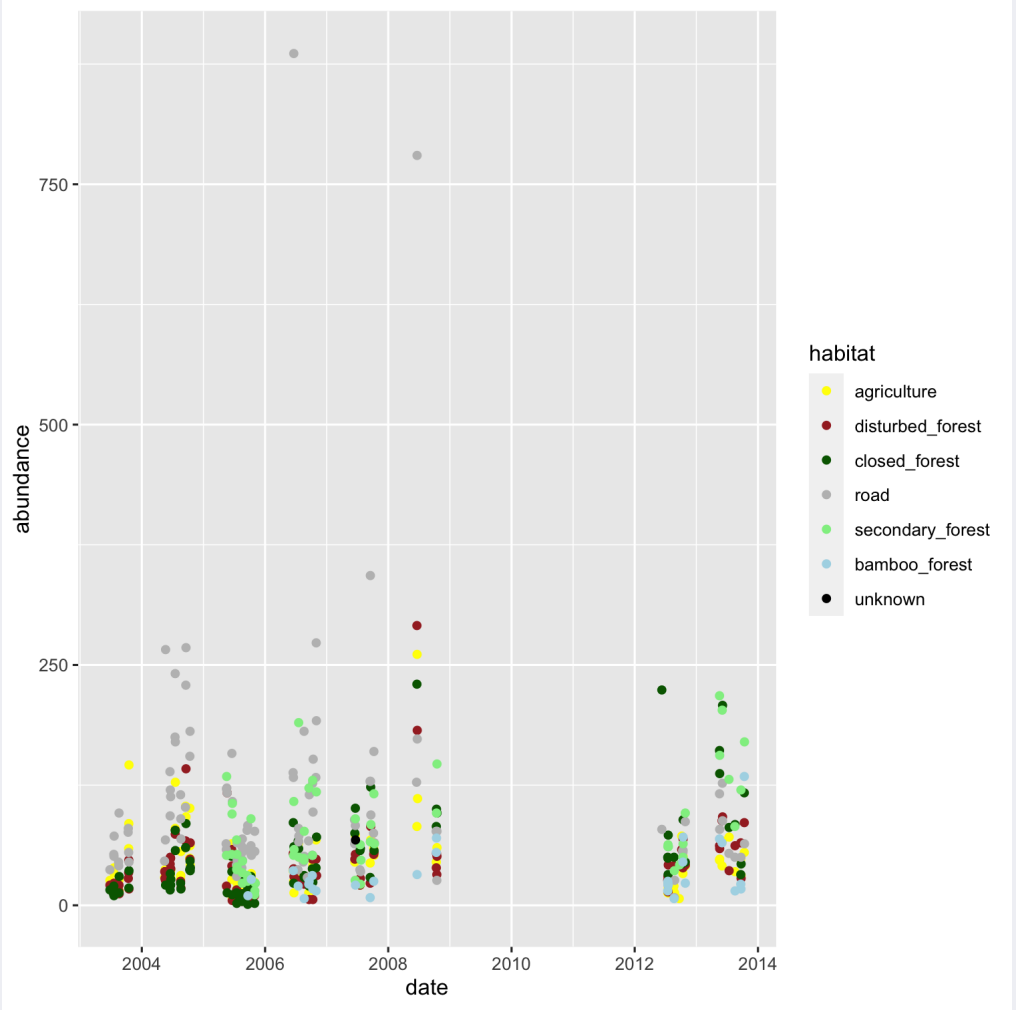


4. Scales

Escala discreta - usar valores pre-establecidos manualmente

```
hab_cols <- c("agriculture" = "yellow",
        "disturbed_forest" = "brown",
        "closed_forest" = "dark green",
        "road" = "grey",
        "secondary_forest" = "light green",
        "bamboo_forest" = "light blue",
        "unknown" = "black")
```

```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = habitat)) +
  geom_point() +
  scale_color_manual(values = hab_cols)
```

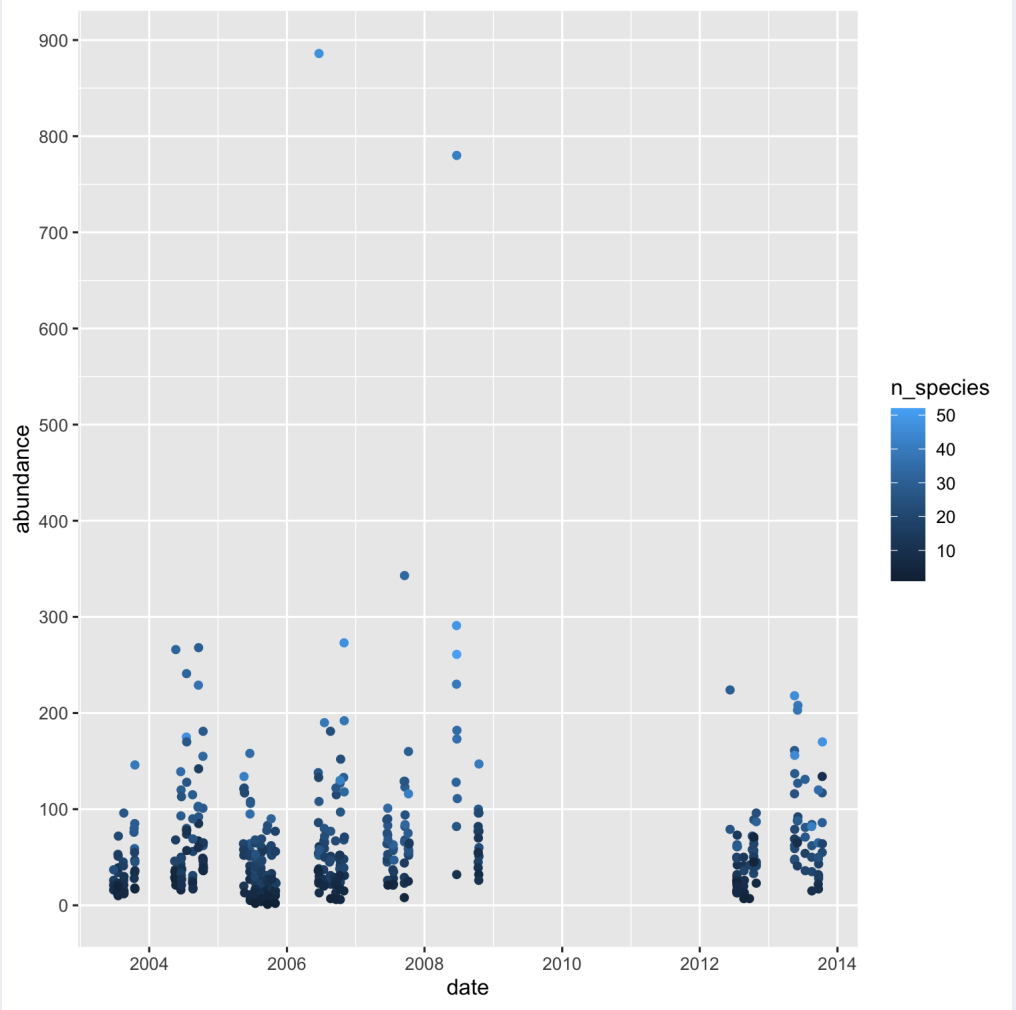


4. Scales

Escala de los ejes

```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = n_species)) +
  geom_point() +
  scale_y_continuous(n.breaks = 10,
    minor_breaks = c(550, 650, 750, 850))
```

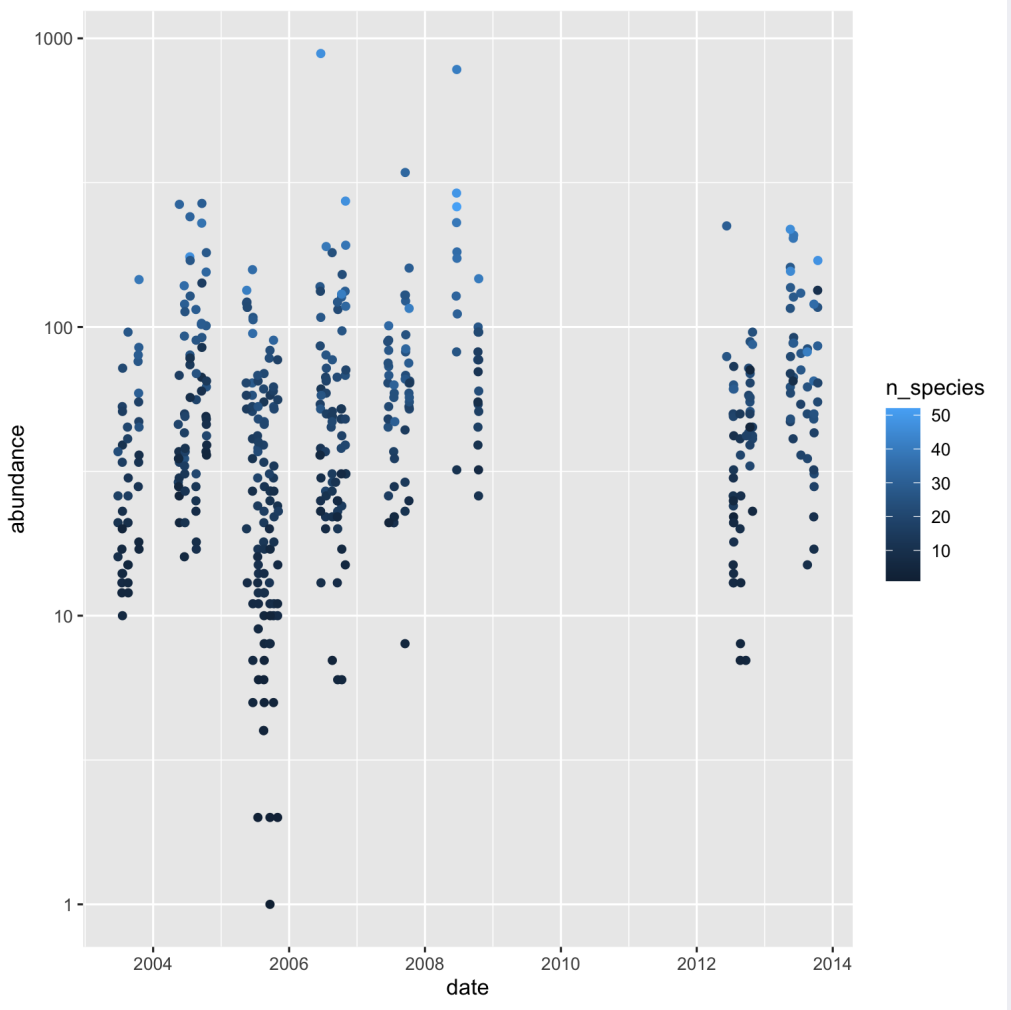
Otros argumentos útiles: breaks, limits, trans (exp, log...)



4. Scales

Escala de los ejes - logarítmica

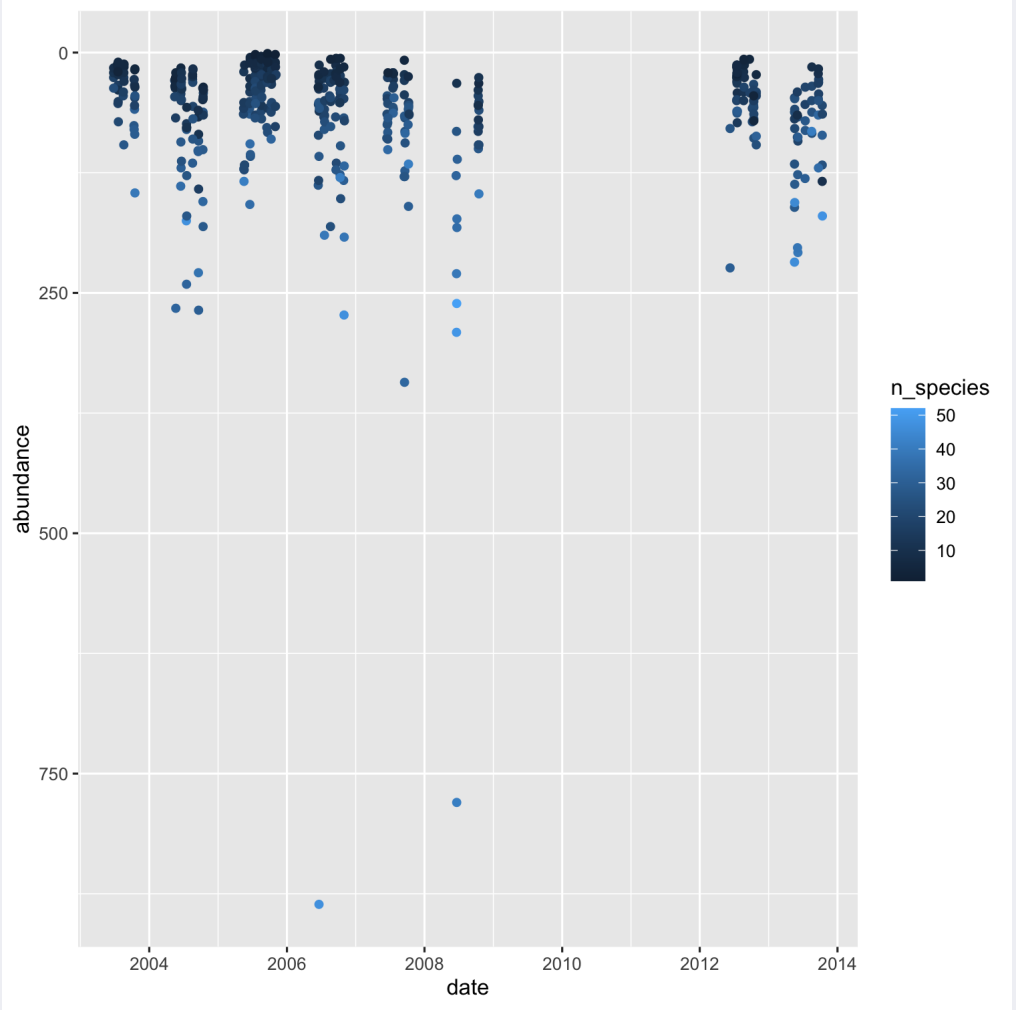
```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = n_species)) +
  geom_point() +
  scale_y_log10()
```



4. Scales

Escala de los ejes - en reverso

```
ggplot(but_sum, aes(x = date,
                    y = abundance,
                    color = n_species)) +
  geom_point() +
  scale_y_reverse()
```



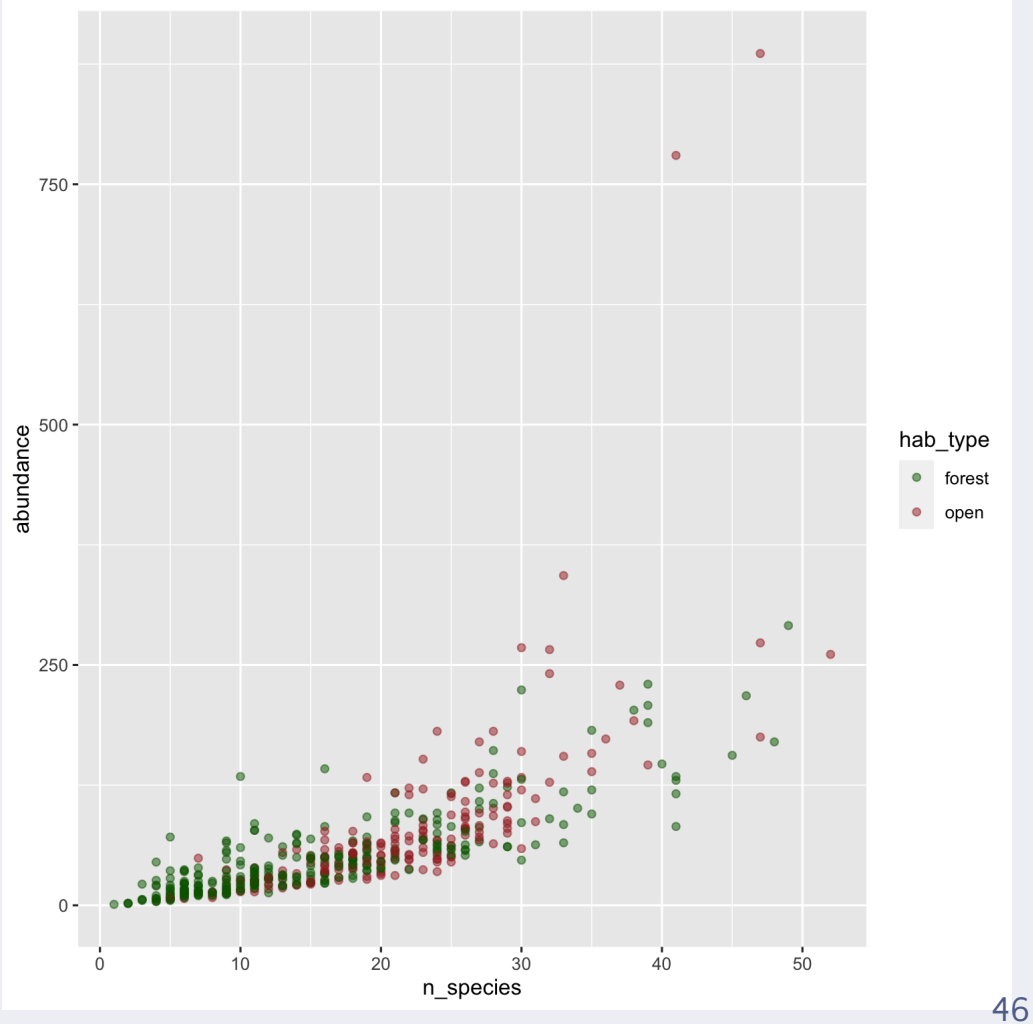
Ejercicio 2:

Graficar numero de especies vs. abundancia, donde el color del punto represente el tipo de hábitat (forest vs. open).

Ejercicio 2:

Graficar numero de especies vs. abundancia, donde el color del punto represente el tipo de hábitat (forest vs. open).

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance,
                    color = hab_type)) +
  geom_point(alpha = 0.5) +
  scale_color_manual(values =
         c("dark green", "brown"))
```

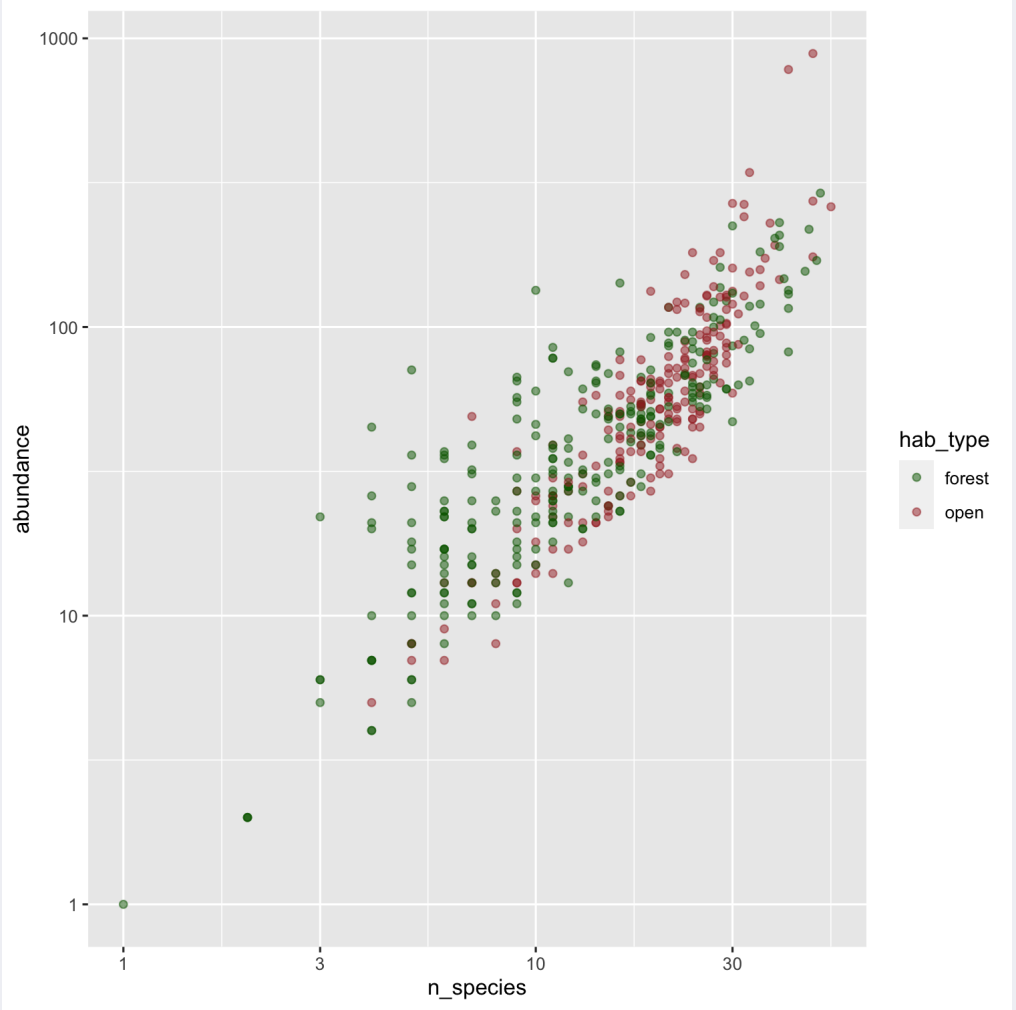


Ejercicio 2:

Graficar numero de especies ~ abundancia, donde el color del punto represente el tipo de hábitat (forest vs. open).

Cambiar a escala log-log.

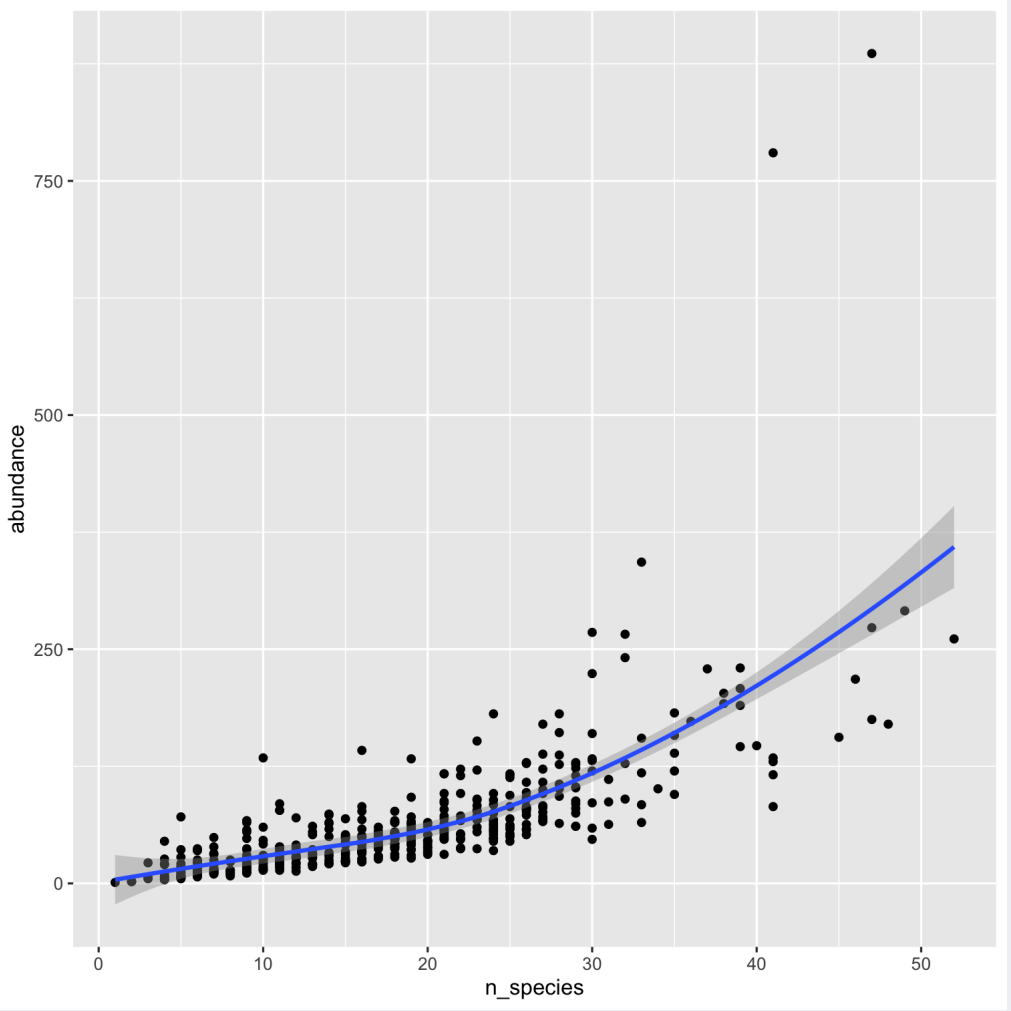
```
ggplot(but_sum, aes(x = n_species,
                    y = abundance,
                    color = hab_type)) +
  geom_point(alpha = 0.5) +
  scale_color_manual(values =
         c("dark green", "brown")) +
  scale_x_log10() +
  scale_y_log10()
```



Geom - línea de tendencia

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance)) +
  geom_point() +
  geom_smooth()
```

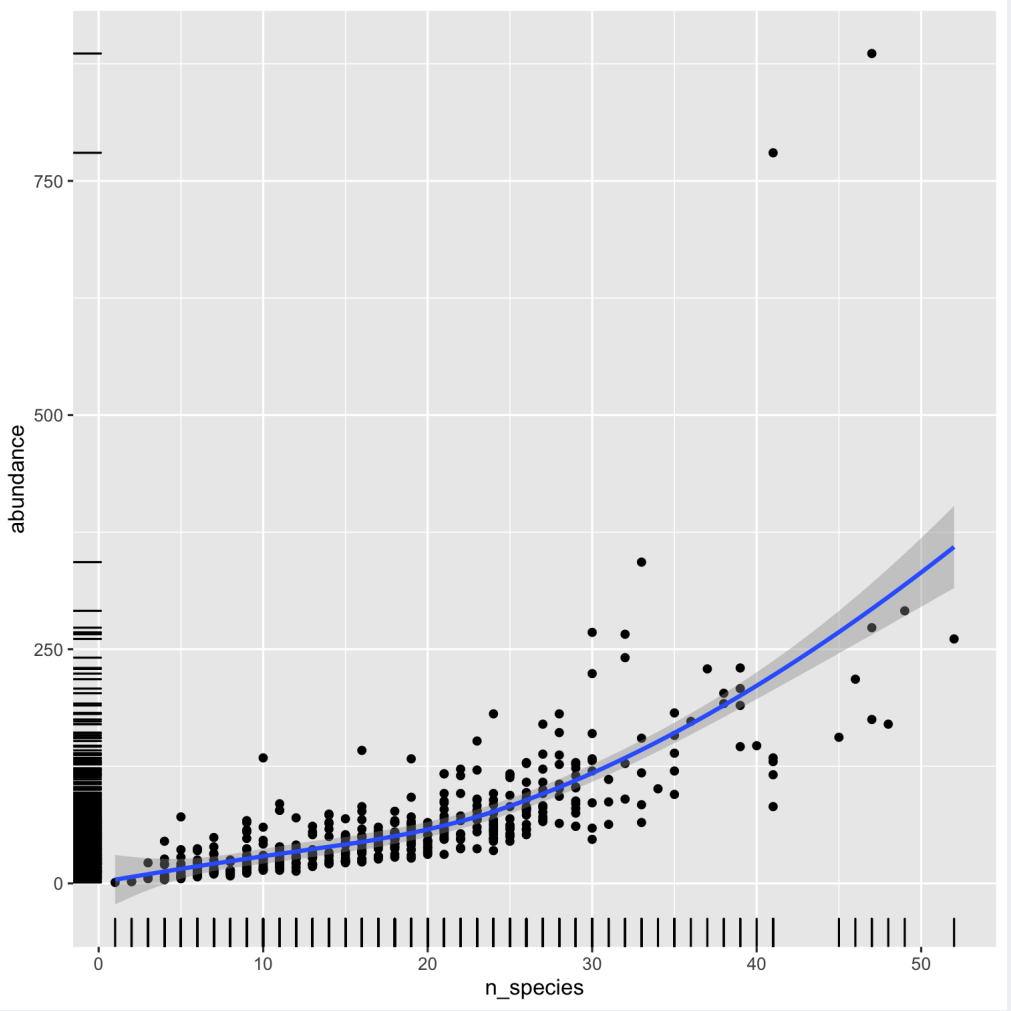
```
## `geom_smooth()` using method = 'loess' and formula
```



Geom - línea de tendencia

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance)) +
  geom_point() +
  geom_smooth() +
  geom_rug()
```

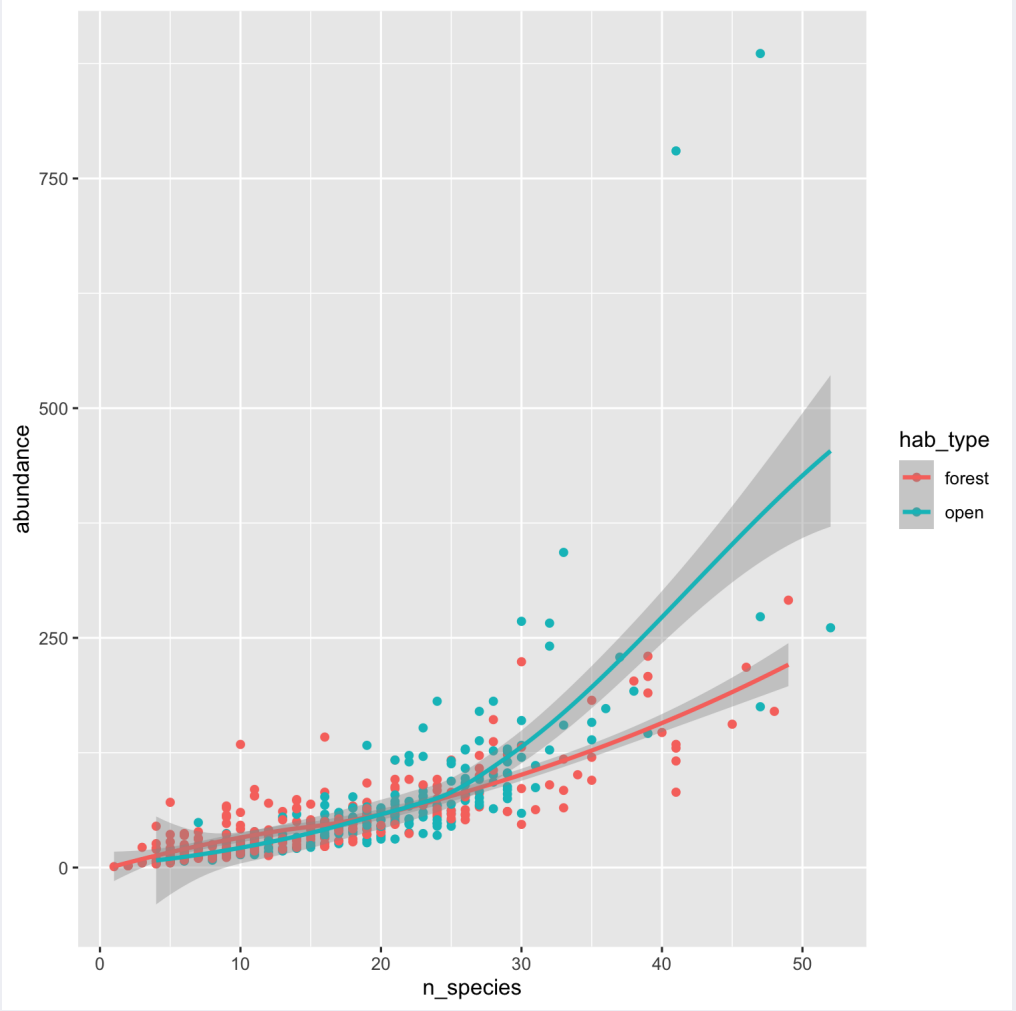
```
## `geom_smooth()` using method = 'loess' and formula
```



Geom - línea de tendencia

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance,
                    color = hab_type)) +
  geom_point() +
  geom_smooth()
```

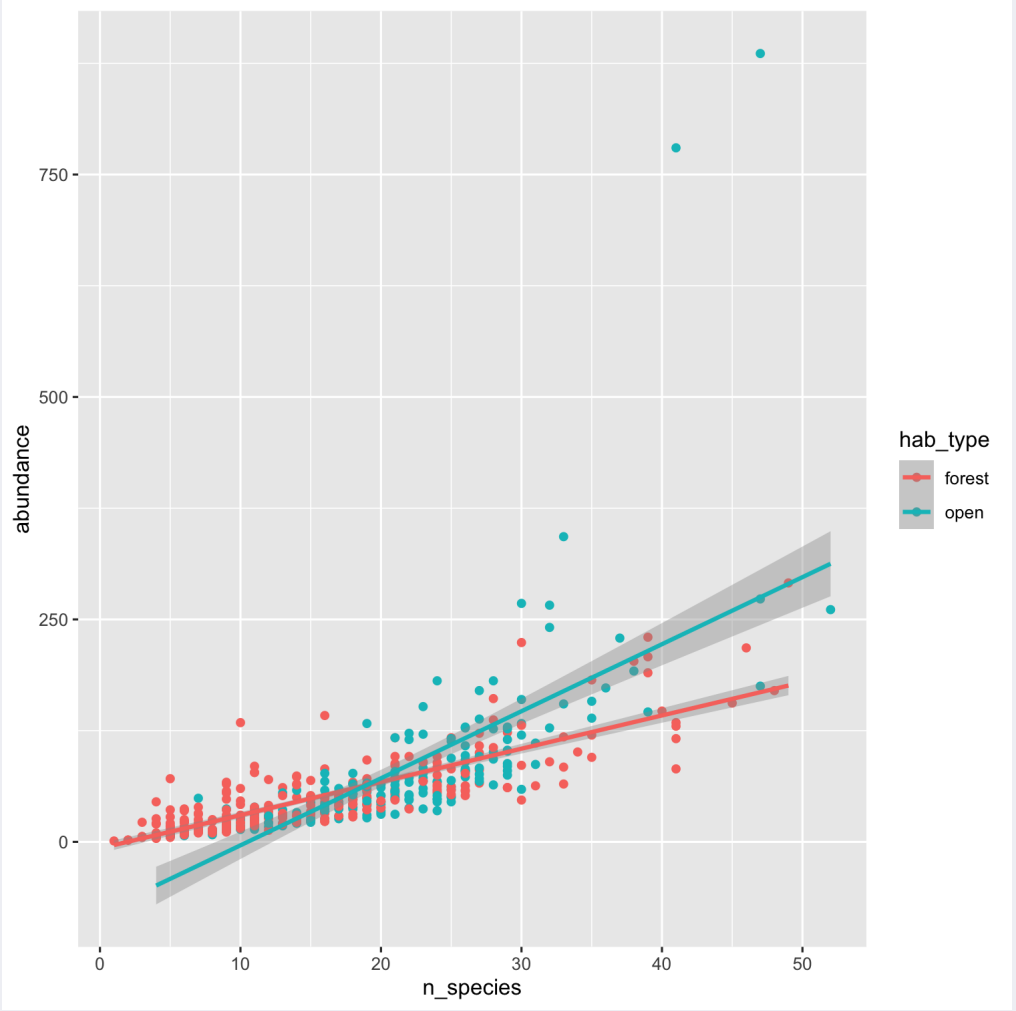
```
## `geom_smooth()` using method = 'loess' and formula
```



Geom - línea de tendencia

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance,
                    color = hab_type)) +
  geom_point() +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



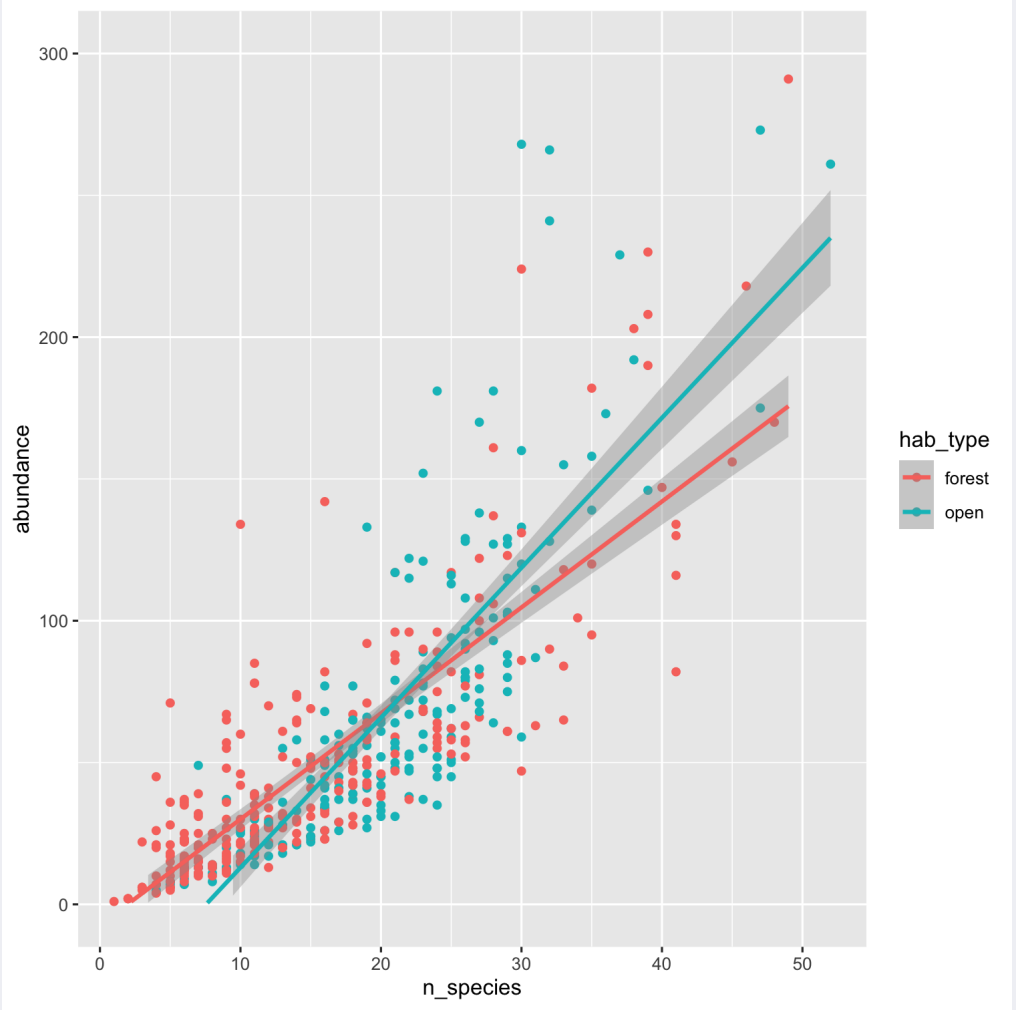
5. Scale vs. Geometry

Limitar la escala del eje vs recortar el eje

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance,
                    color = hab_type)) +
  geom_point() +
  geom_smooth(method="lm") +
  scale_y_continuous(limits = c(0, 300))
```

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 3 rows containing non-finite valu

## Warning: Removed 3 rows containing missing values

## Warning: Removed 8 rows containing missing values
```

Si limitamos la escala, "borramos" los puntos que se quedan fuera del límite, y dejan de ser considerados para estimar las líneas de tendencia.



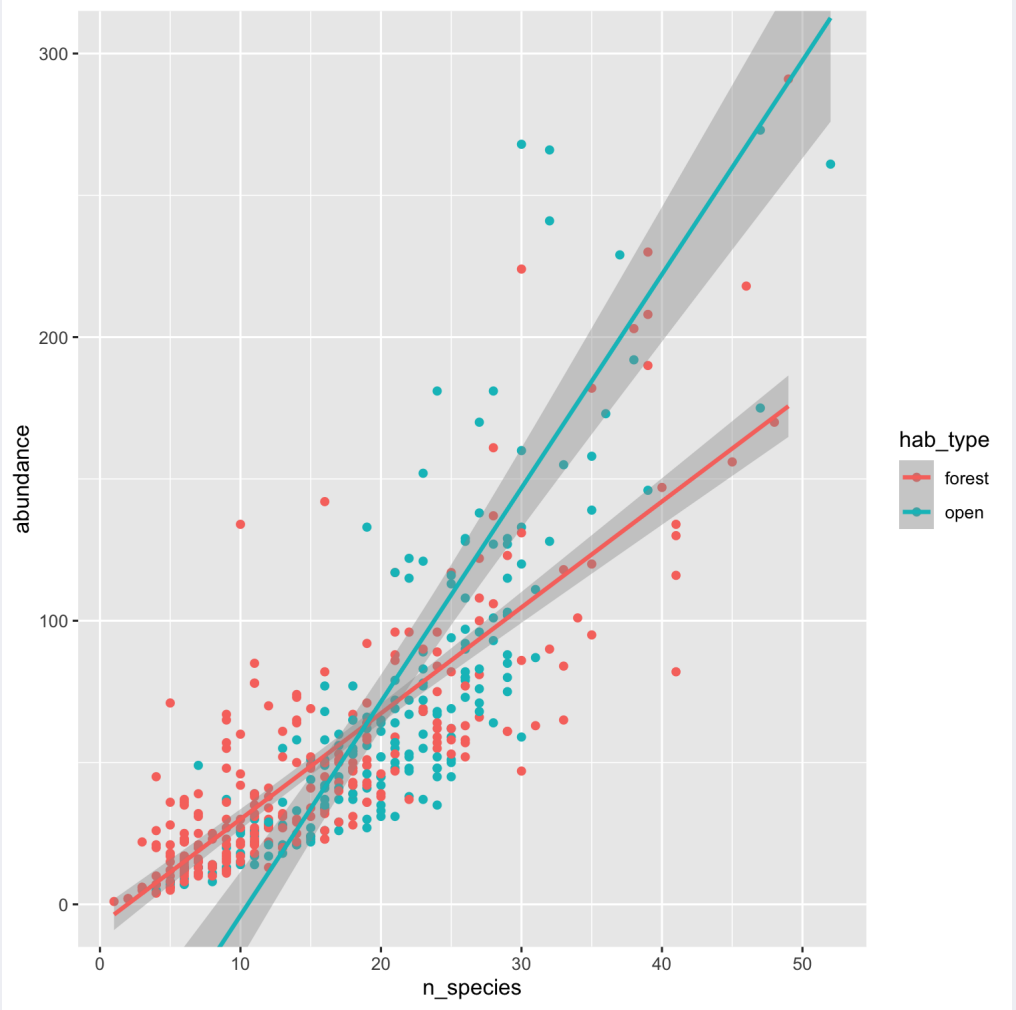
5. Scale vs. Geometry

Limitar la escala del eje vs recortar el eje

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance,
                    color = hab_type)) +
  geom_point() +
  geom_smooth(method ="lm") +
  coord_cartesian(ylim = c(0, 300))
```

```
## `geom_smooth()` using formula 'y ~ x'
```

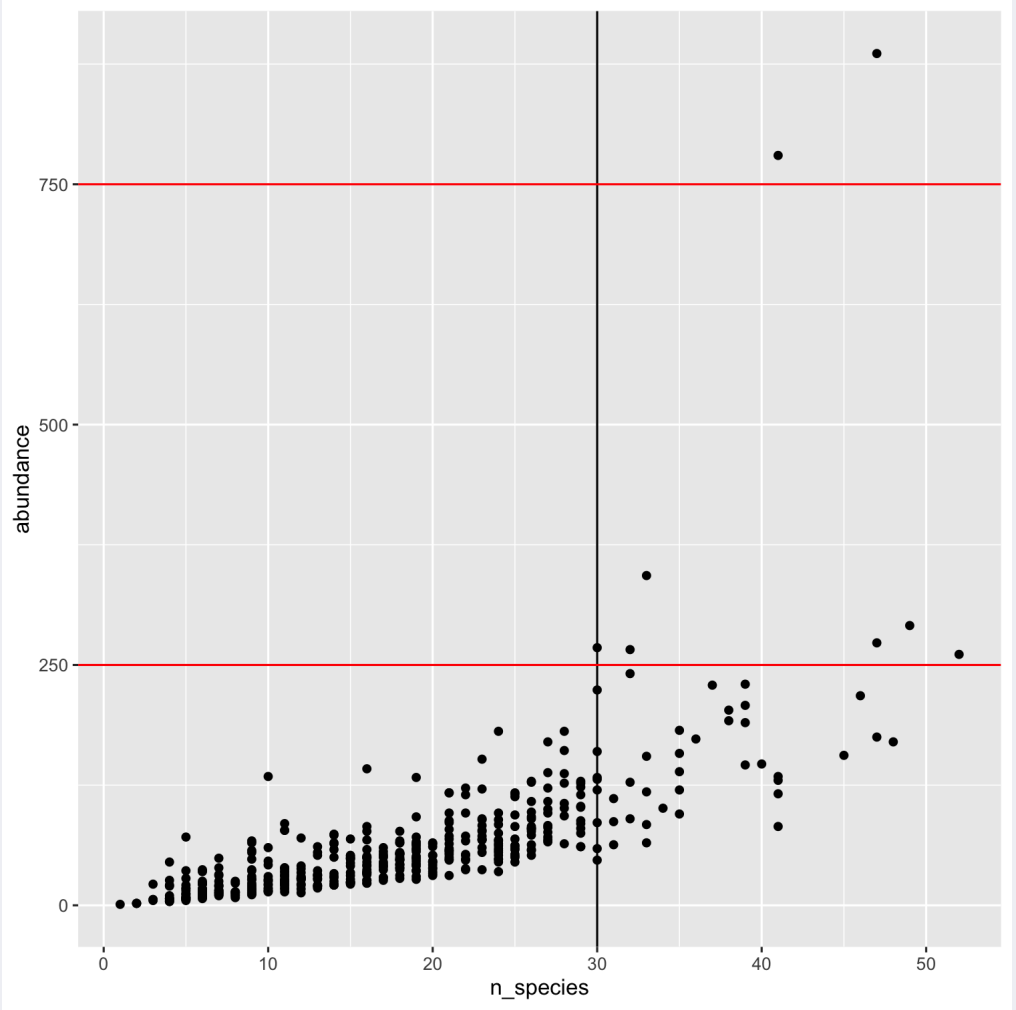
Si limitamos el eje, "recortamos" el gráfico (en este caso hacemos un zoom). Los puntos que quedan fuera del límite siguen siendo considerandos y se mantienen para el calculo de la línea de tendencia.



Más geoms

Lineas horizontales y verticales

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance)) +
  geom_point() +
  geom_vline(xintercept = 30) +
  geom_hline(yintercept = c(250, 750),
             color = "red")
```



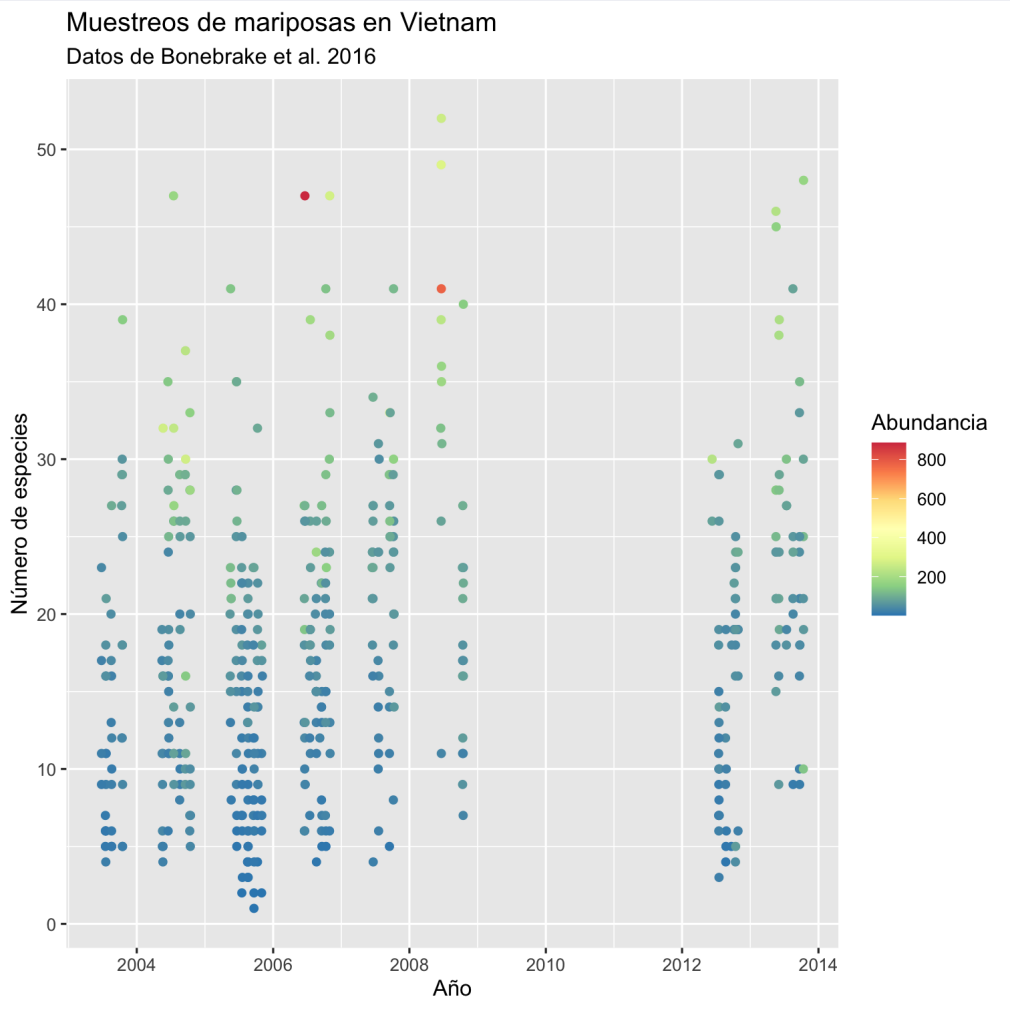
7. Labs

Añadir título, nombre de ejes y leyenda

```
ggplot(but_sum, aes(x = n_species,
                    y = abundance)) +
  geom_point() +
  labs(x = "No. de especies",
       y = "Abundancia",
       title = "Muestreos de mariposas en Viet
       caption = "cada punto representa un mue
```

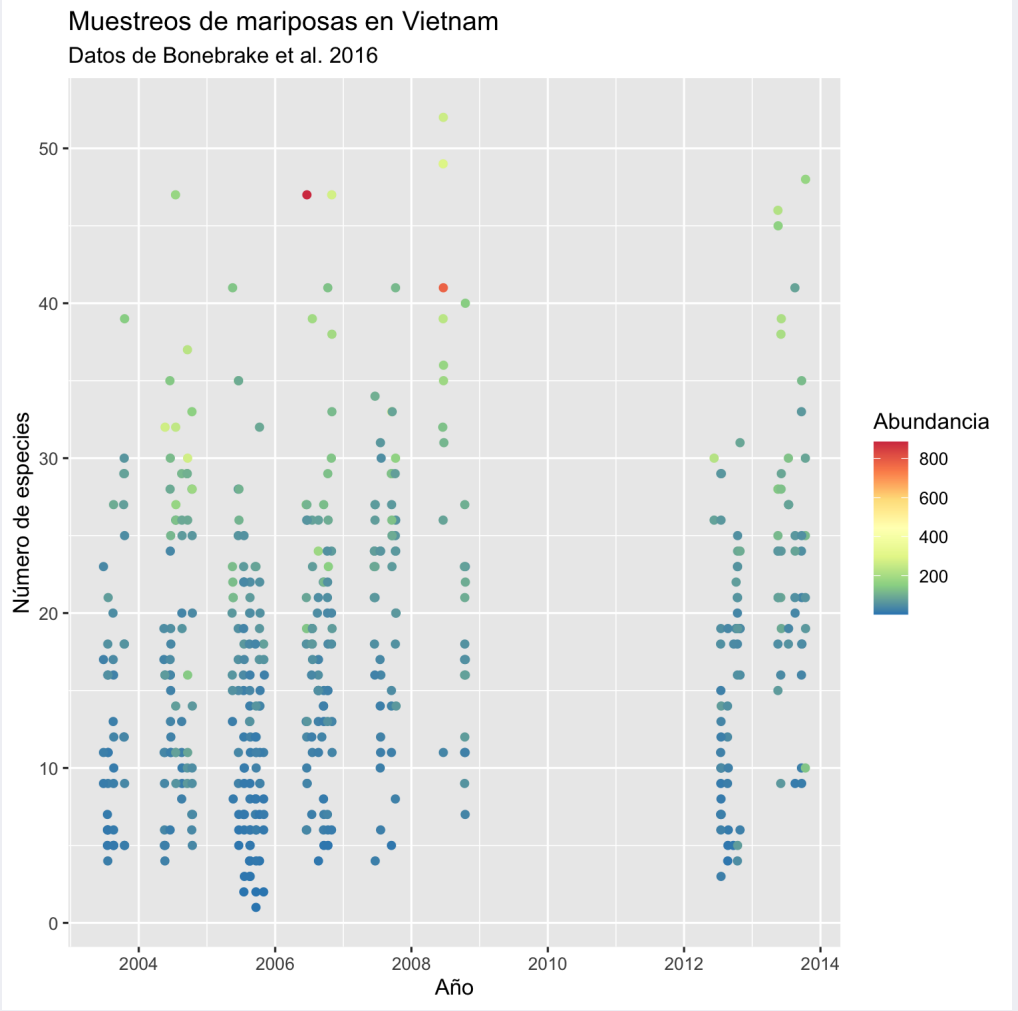
Ejercicio 3

Crear una gráfico lo más similar a este:



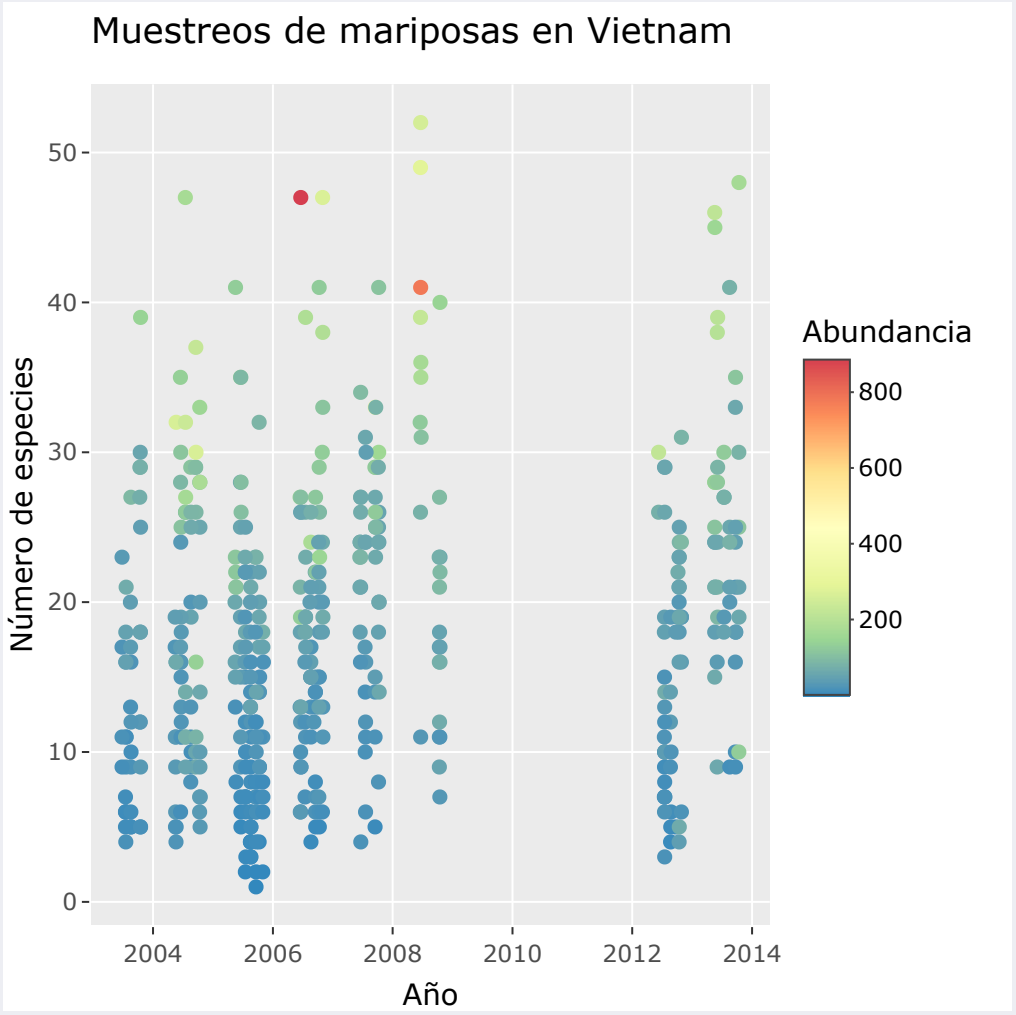
Ejercicio 3

```
ggplot(but_sum, aes(x = date,
                    y = n_species,
                    color = abundance)) +
  geom_point() +
  scale_color_distiller(palette = "Spectral") +
  labs(x = "Año",
       y = "Número de especies",
       title = "Muestreos de mariposas en Viet
       subtitle = "Datos de Bonebrake et al. 2
       color = "Abundancia")
```



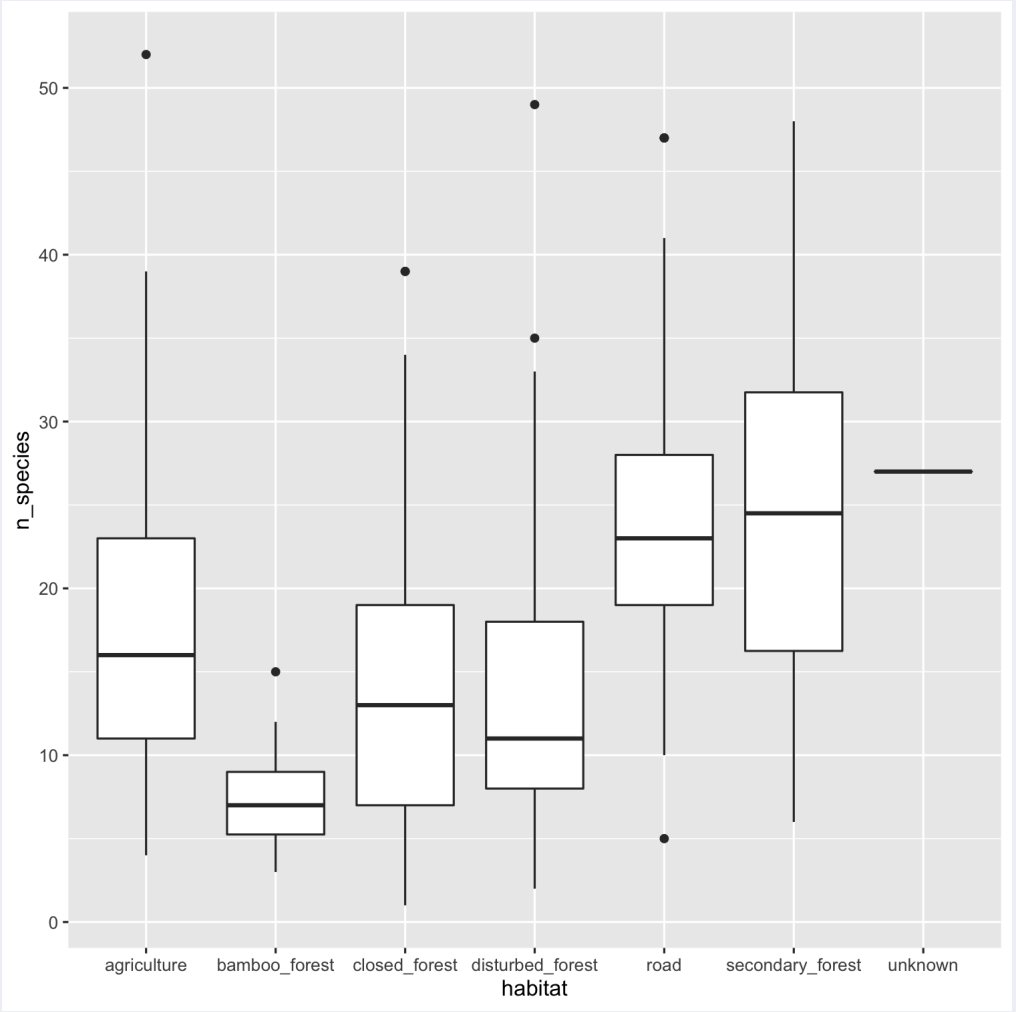
GGplotly

```
library(plotly)
ggplotly()
```



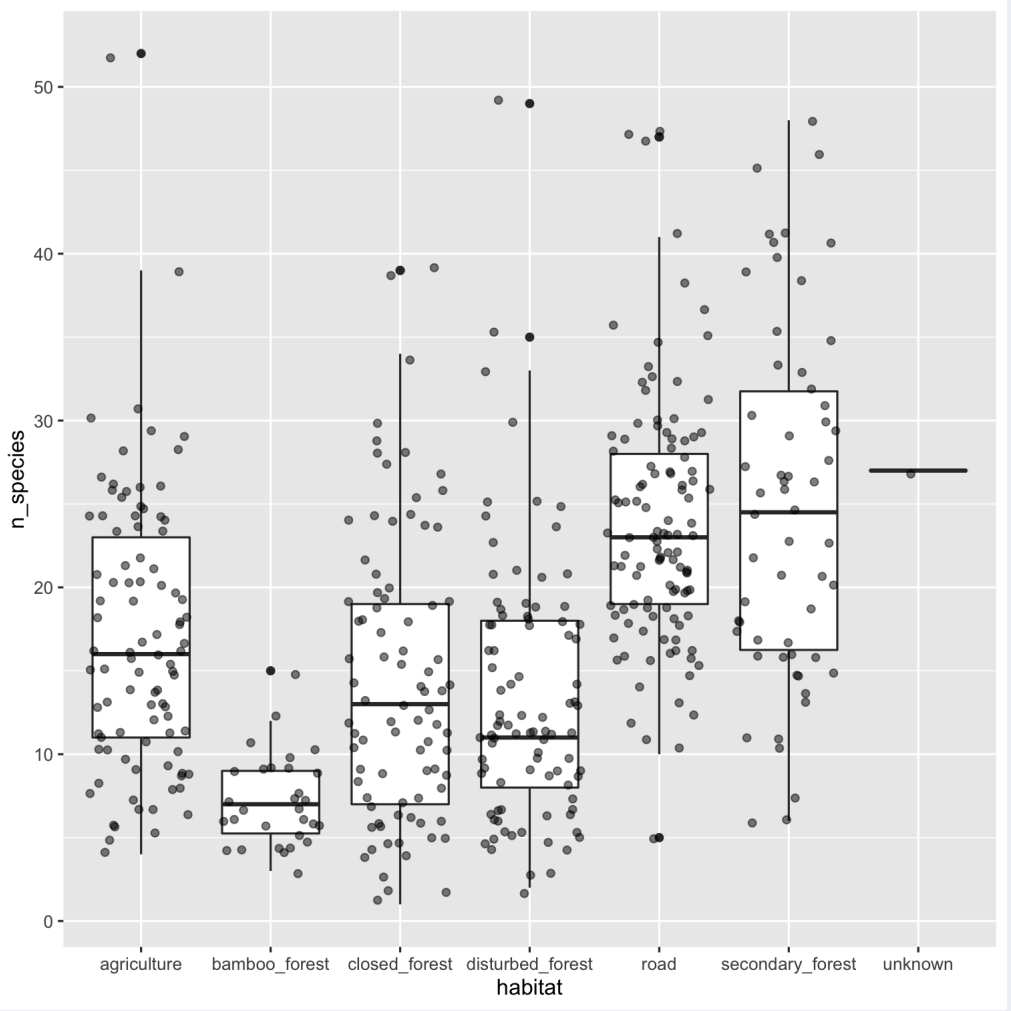
Otros geoms - Box-plot

```
ggplot(but_sum,
       aes(x = habitat,
           y = n_species)) +
  geom_boxplot()
```



Geoms - Box-plot

```
ggplot(but_sum,
       aes(x = habitat,
           y = n_species)) +
  geom_boxplot() +
  geom_jitter(alpha=0.5)
```

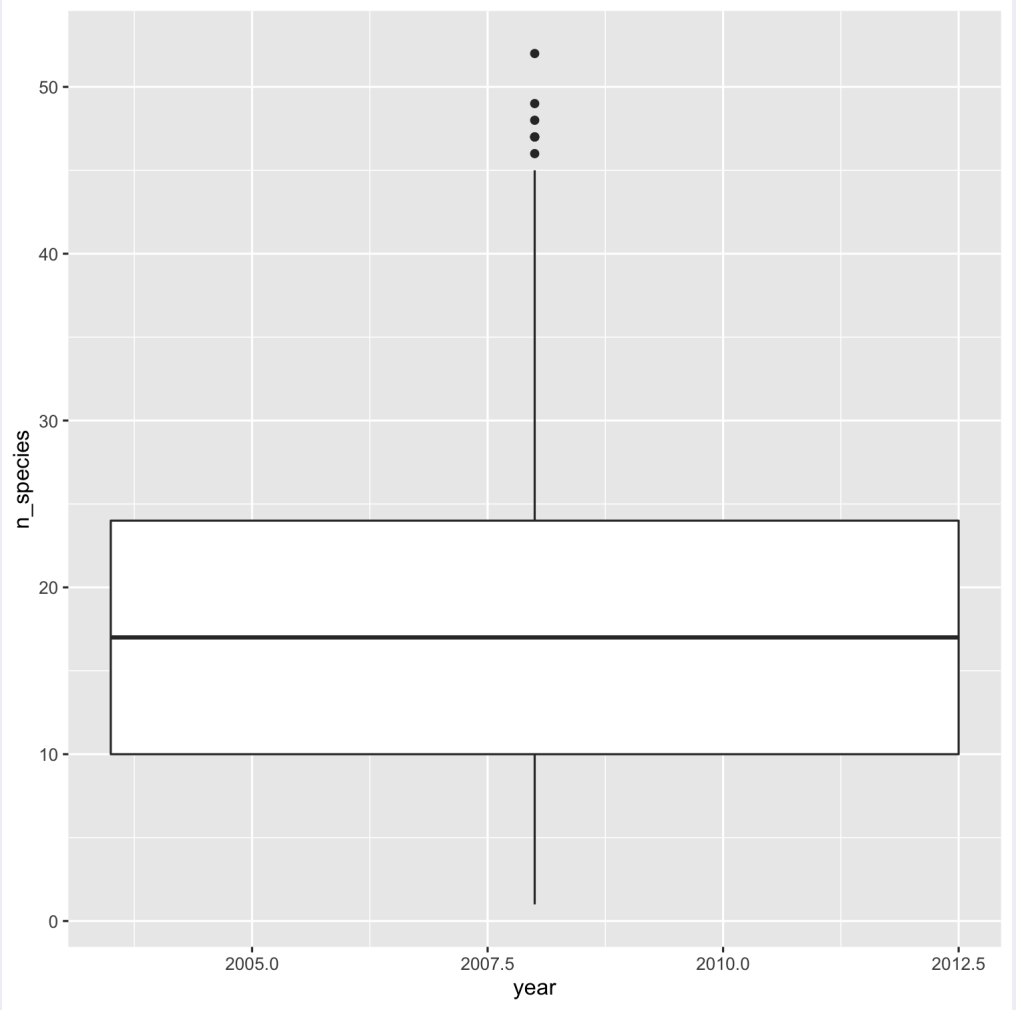


Variable continua vs. factor

Si queremos hacer un boxplot que muestre el numero de especies por año.

```
but_sum %>%
  mutate(year = lubridate::year(date)) %>%
  ggplot(aes(x = year,
             y = n_species)) +
  geom_boxplot()
```

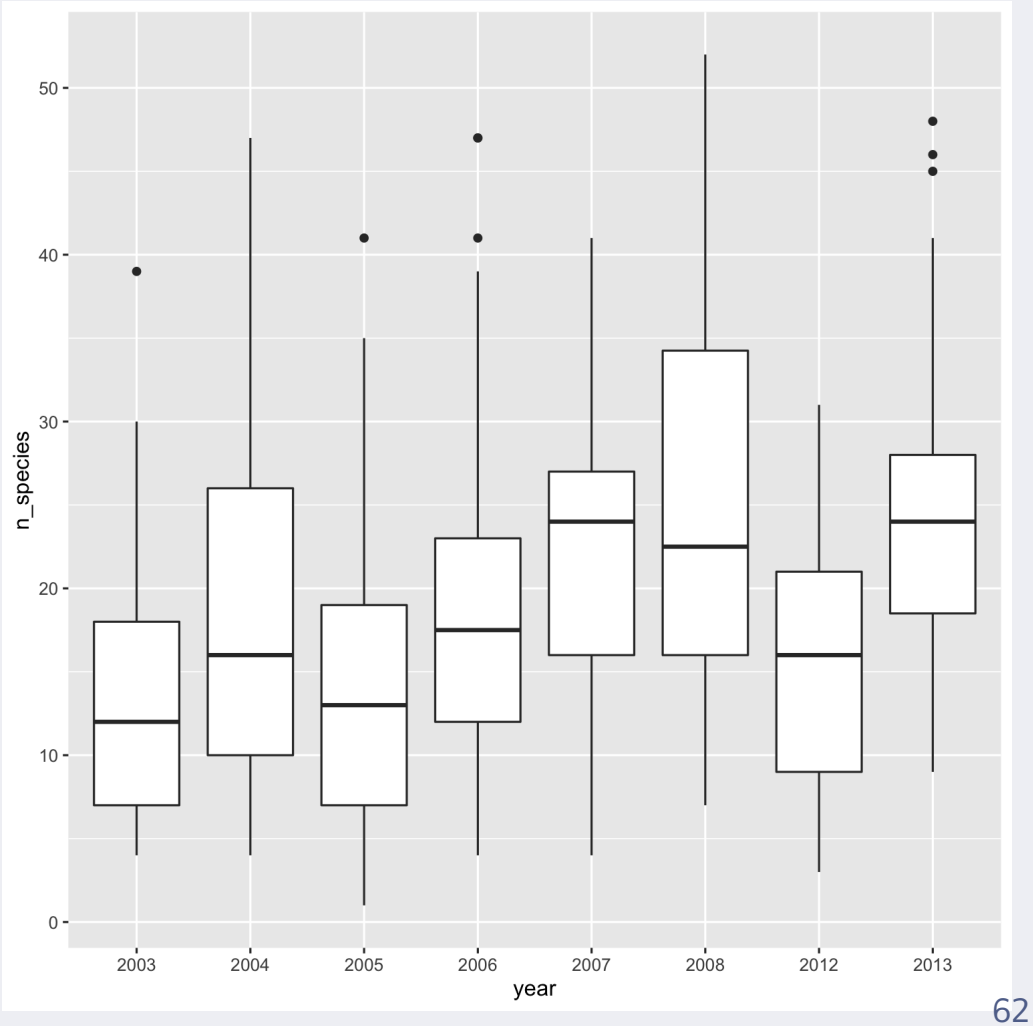
```
## Warning: Continuous x aesthetic -- did you forget
```



Variable continua vs. factor

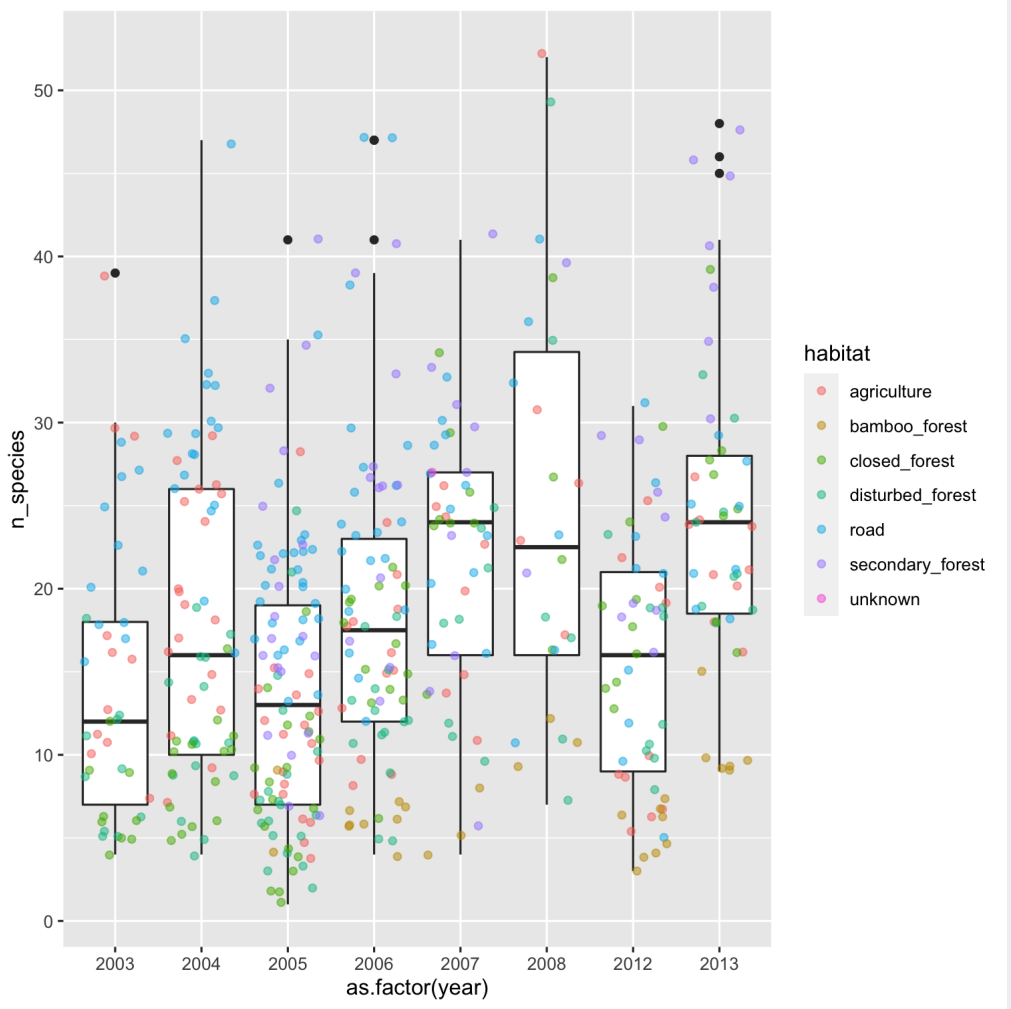
Si queremos hacer un boxplot que muestre el numero de especies por año - asegurar que la variable año es un factor.

```
but_sum %>%
  mutate(year = lubridate::year(date)) %>%
  mutate(year = as.factor(year)) %>%
  ggplot(aes(x = year,
             y = n_species)) +
  geom_boxplot()
```



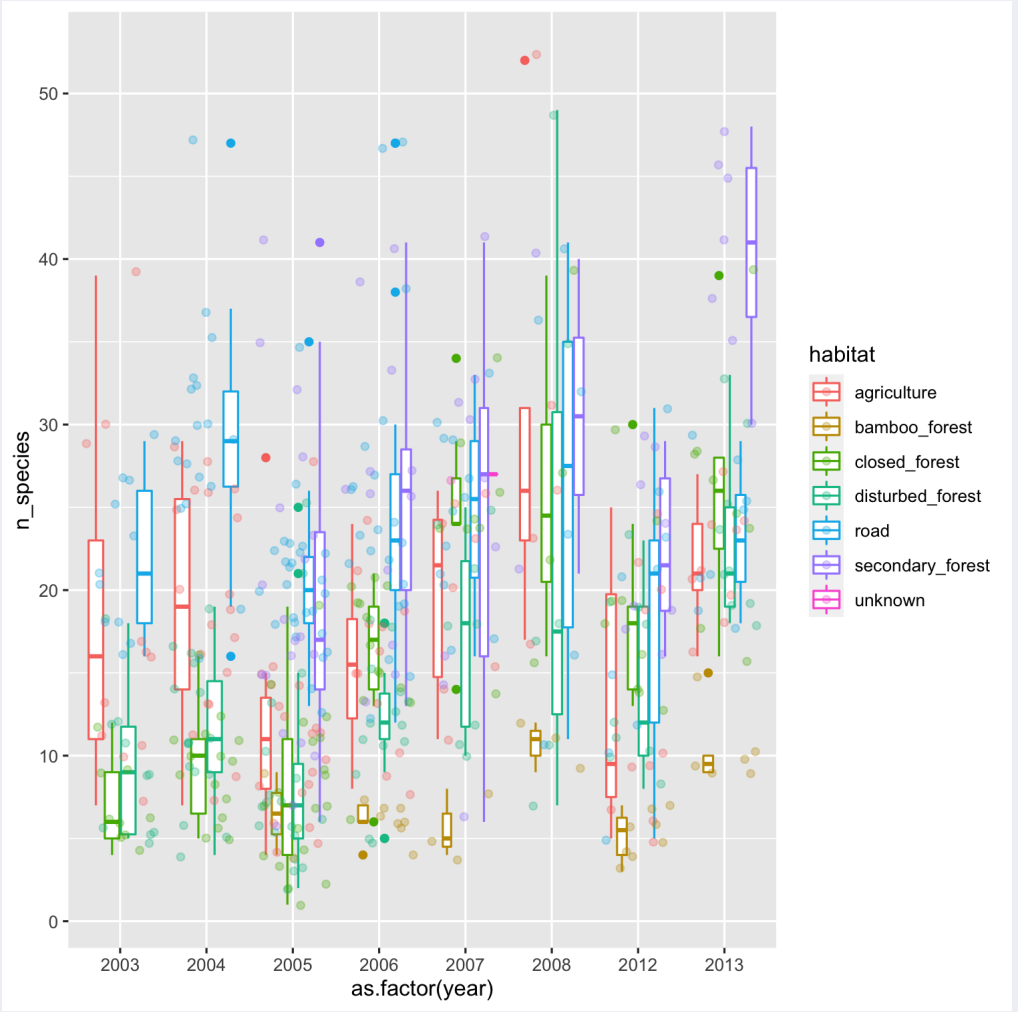
Posición de los aesthetics

```
but_sum %>%
  mutate(year = lubridate::year(date)) %>%
  ggplot(aes(x = as.factor(year),
             y = n_species)) +
  geom_boxplot() +
  geom_jitter(alpha = 0.5,
              aes(color = habitat))
```



Posición de los aesthetics

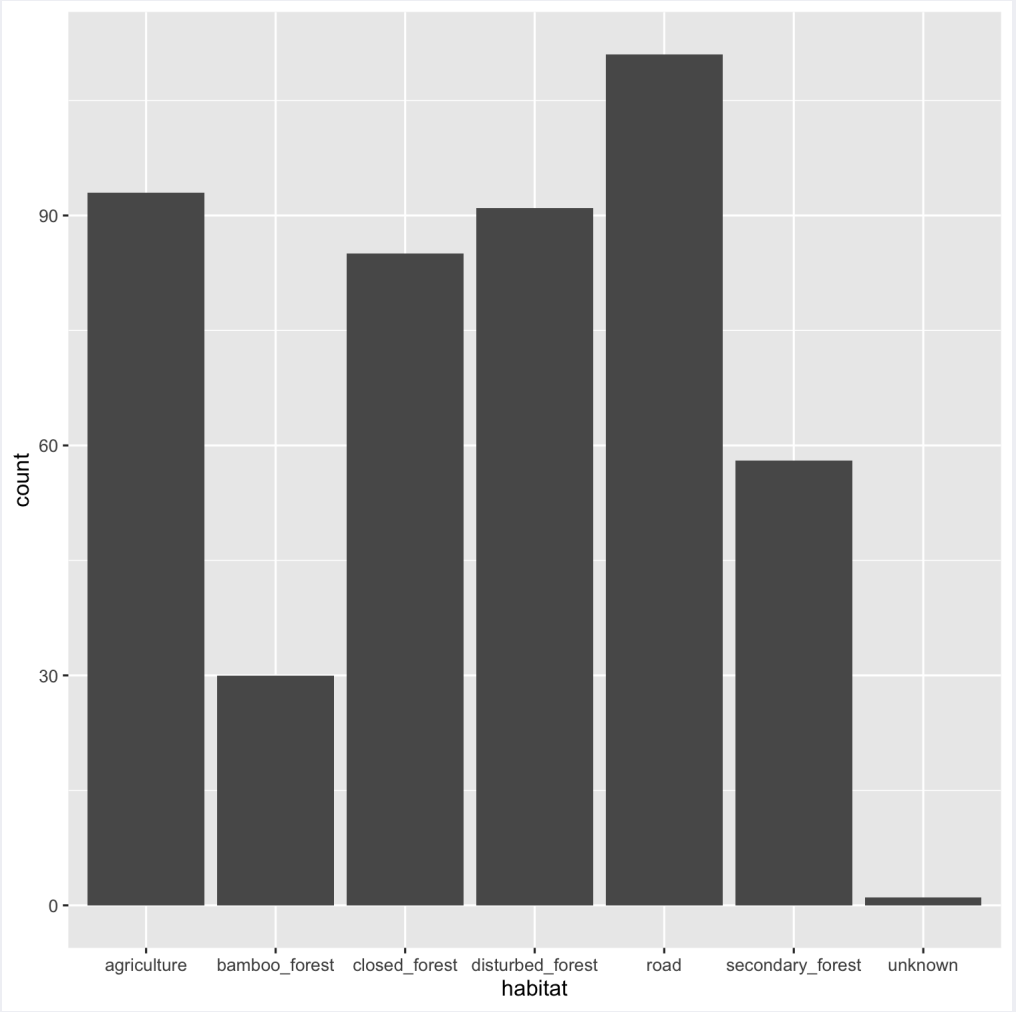
```
but_sum %>%
  mutate(year = lubridate::year(date)) %>%
  ggplot(aes(x = as.factor(year),
             y = n_species,
             color = habitat)) +
  geom_boxplot() +
  geom_jitter(alpha=0.3)
```



Geoms - Gráfico de barras

```
ggplot(but_sum,
       aes(x = habitat)) +
  geom_bar()
```

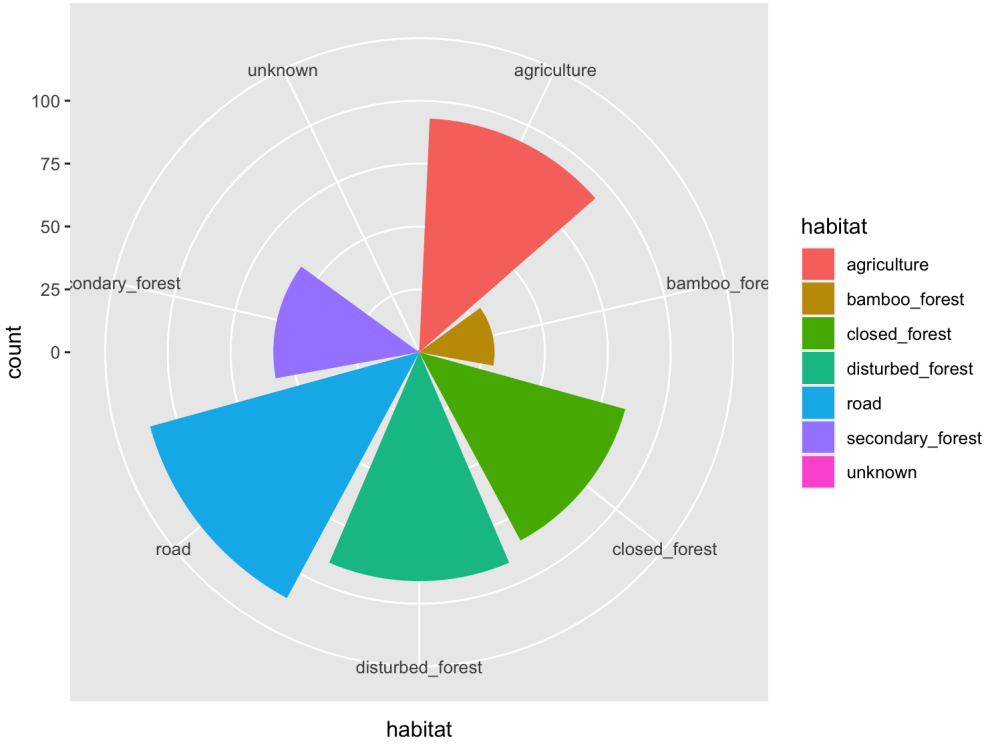
geom_bar usa la funcion `stat_count()` por defecto. Representa el número (cuenta) de observaciones por hábitat, en este caso de muestreos.



Geoms - Gráfico de 'quesos'

Pie-chart

```
ggplot(but_sum,
       aes(x = habitat,
           fill = habitat)) +
  geom_bar() +
  coord_polar()
```



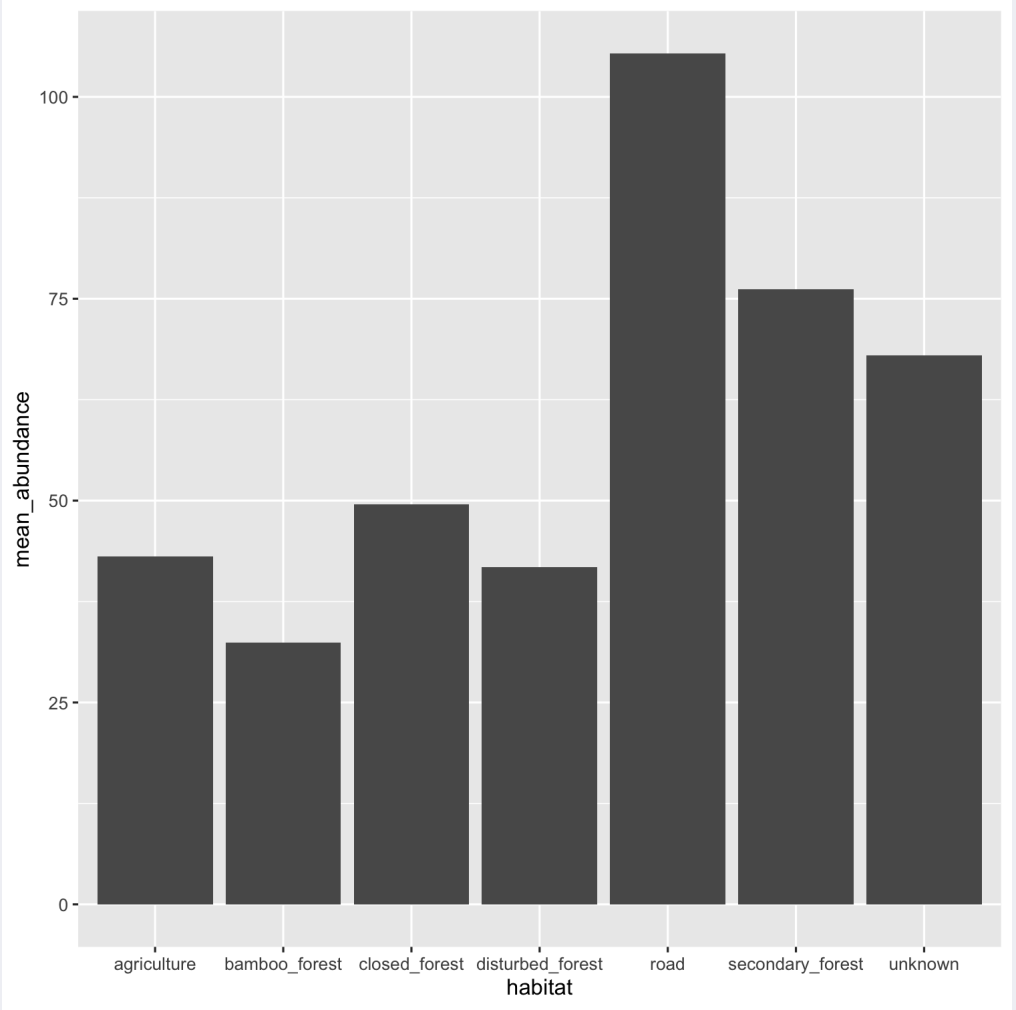
Geoms - Gráfico de barras

Gráfico de barras para datos sumarizados

```
but_sum %>%
  group_by(habitat) %>%
  summarise(mean_abundance = mean(abundance)) %
  ggplot(aes(x = habitat,
             y = mean_abundance)) +
  geom_col()
```

Alternativa (mismo resultado):

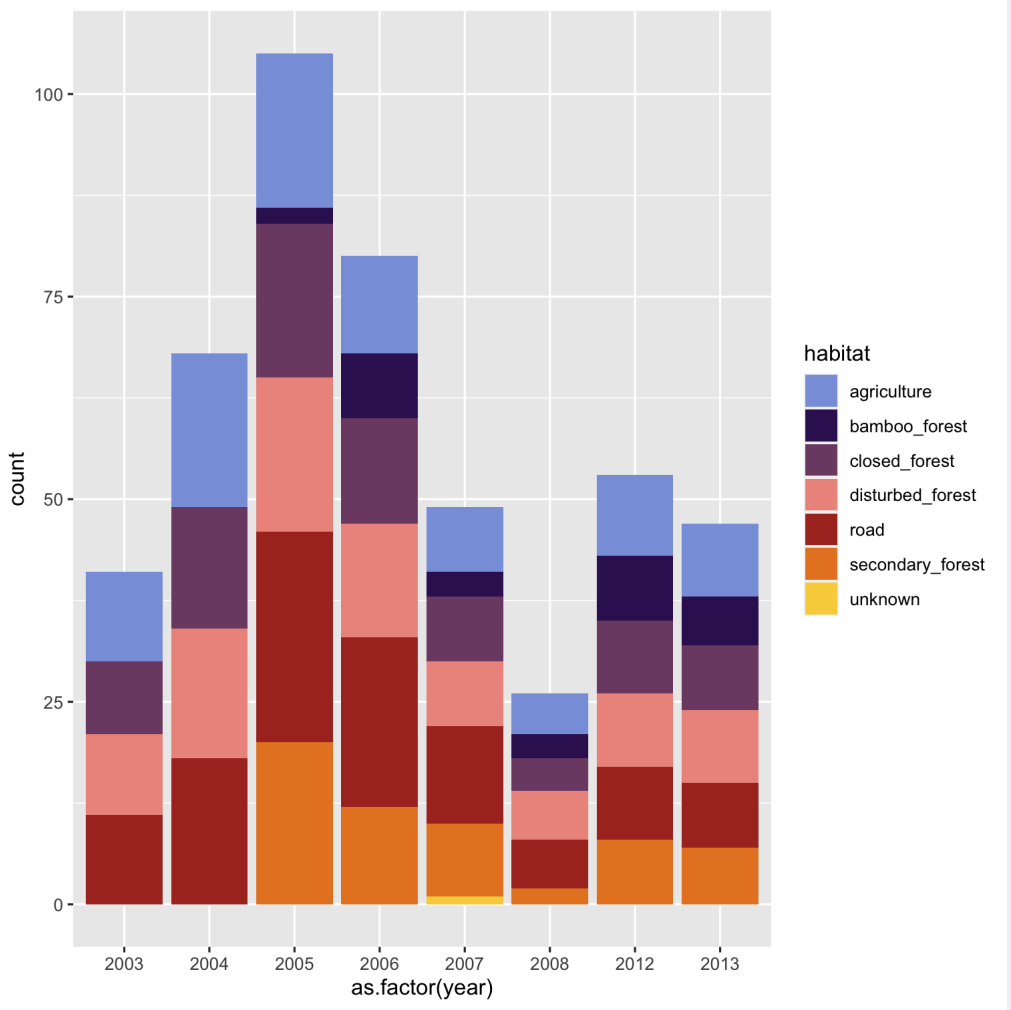
```
but_sum %>%
  group_by(habitat) %>%
  summarise(mean_abundance = mean(abundance)) %
  ggplot(aes(x = habitat,
             y = mean_abundance)) +
  geom_bar(stat = "identity")
```



Geoms - Gráfico de barras

Gráfico de barras apilados

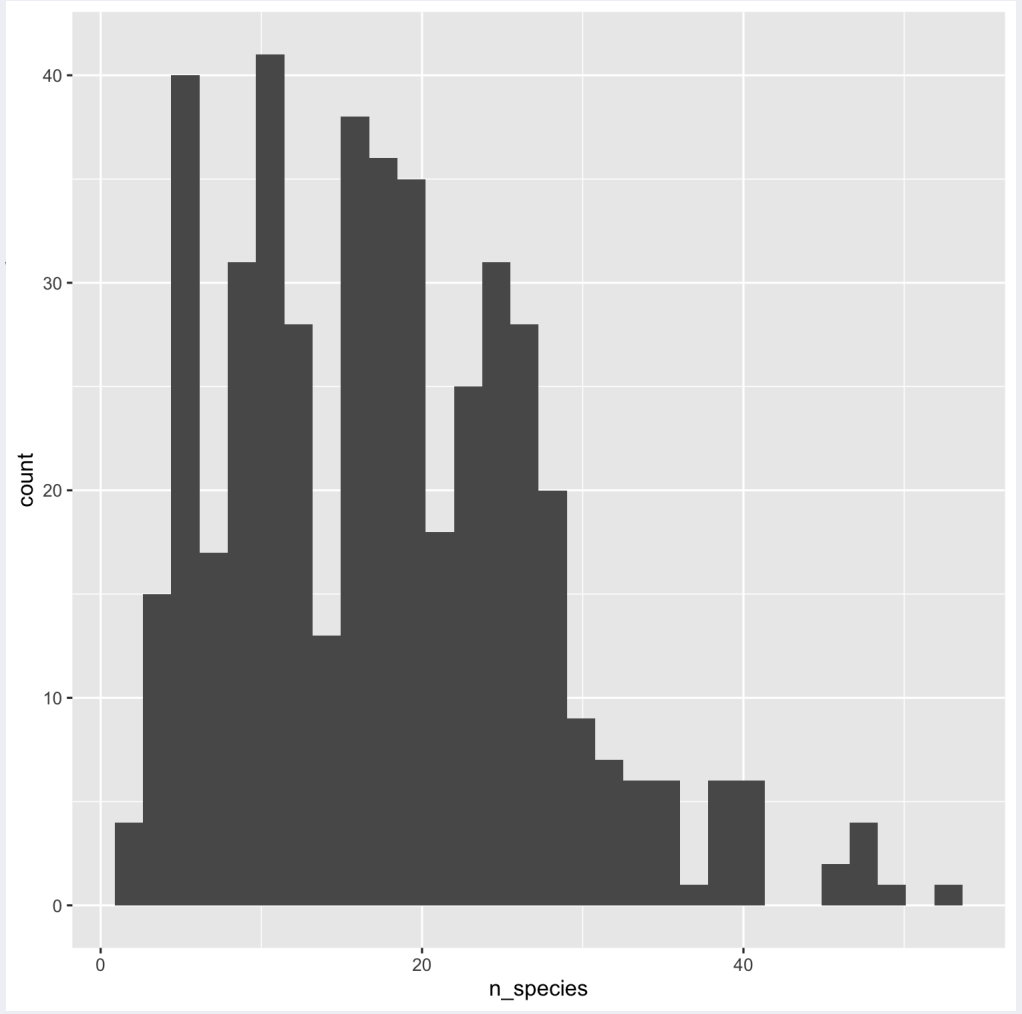
```
but_sum %>%
  mutate(year = lubridate::year(date)) %>%
  ggplot(aes(x = as.factor(year),
             fill = habitat)) +
  geom_bar() +
  MetBrewer::scale_fill_met_d("Archambault")
```



Geoms - Histograma

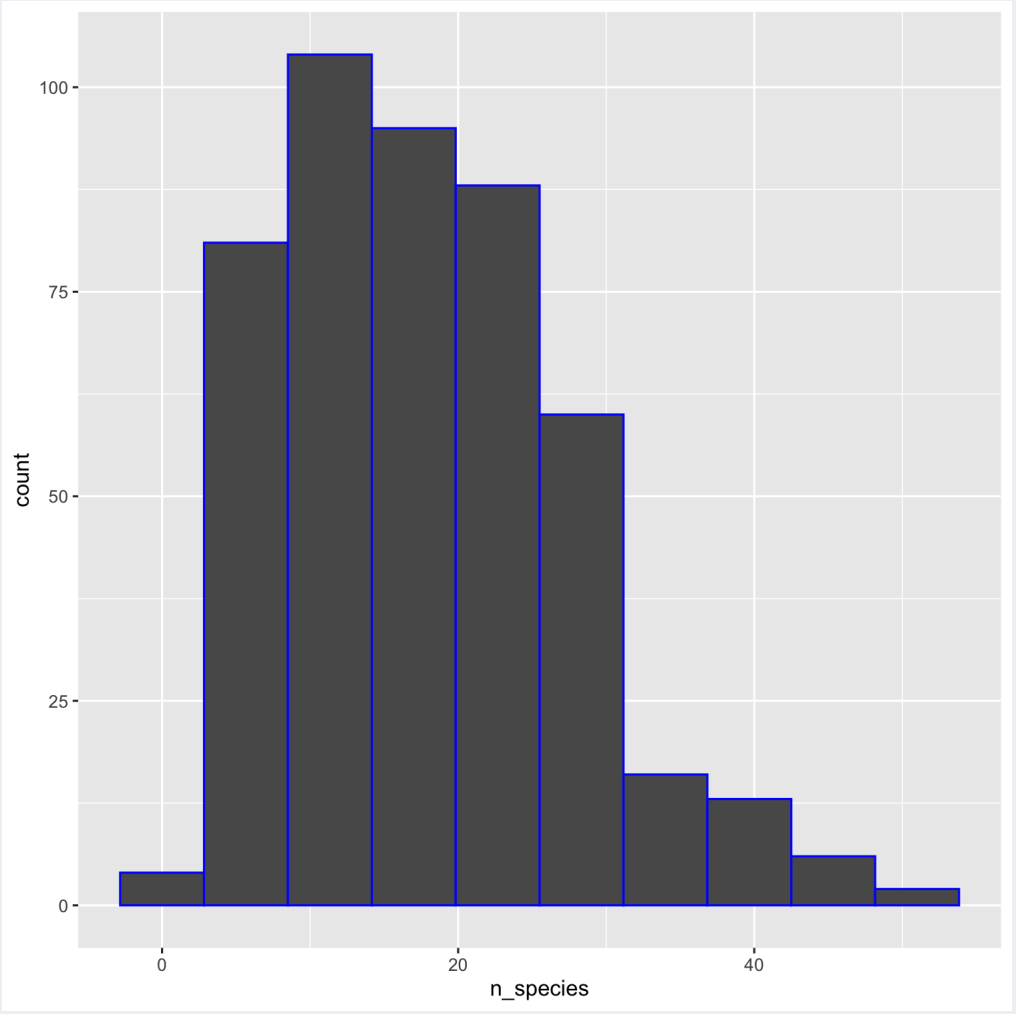
```
ggplot(but_sum,
       aes(x = n_species)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value
```



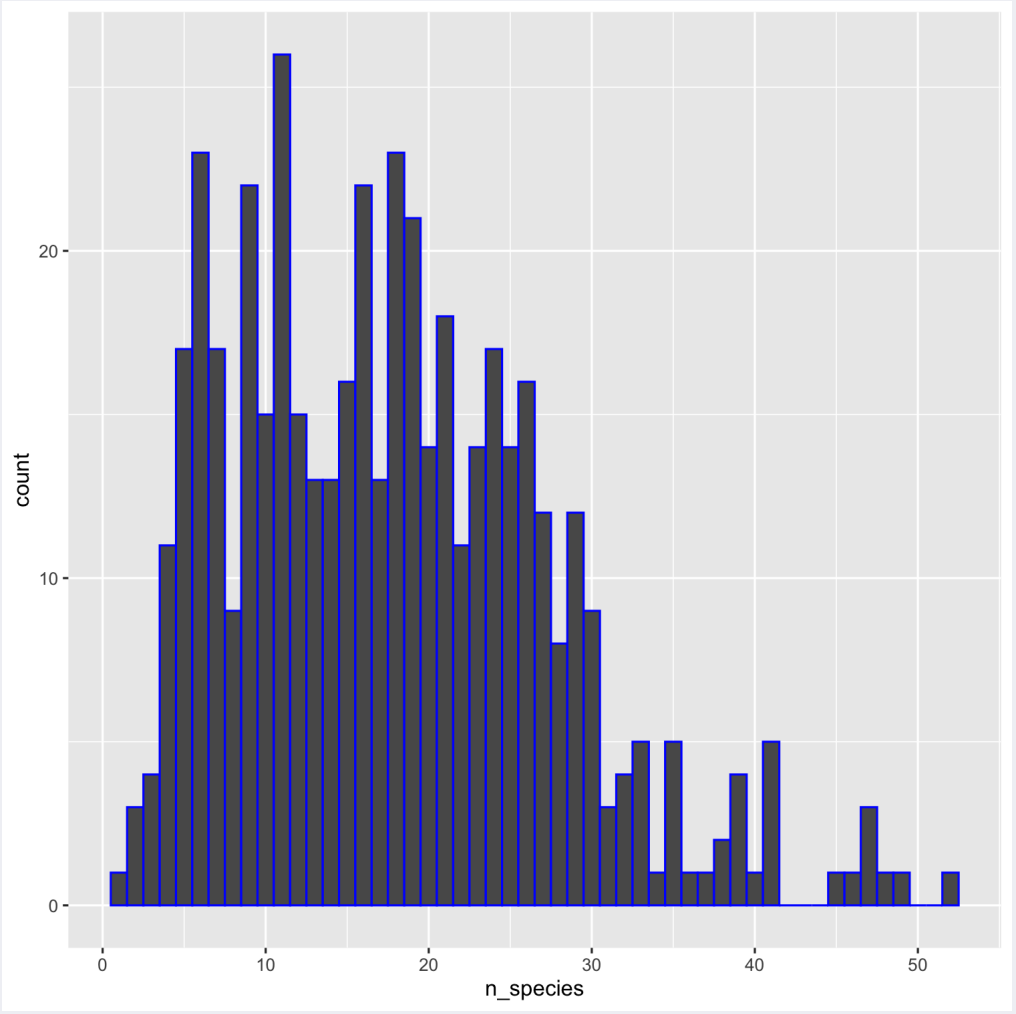
Geoms - Histograma

```
ggplot(but_sum,
       aes(x = n_species)) +
  geom_histogram(color = "blue",
                 bins = 10)
```



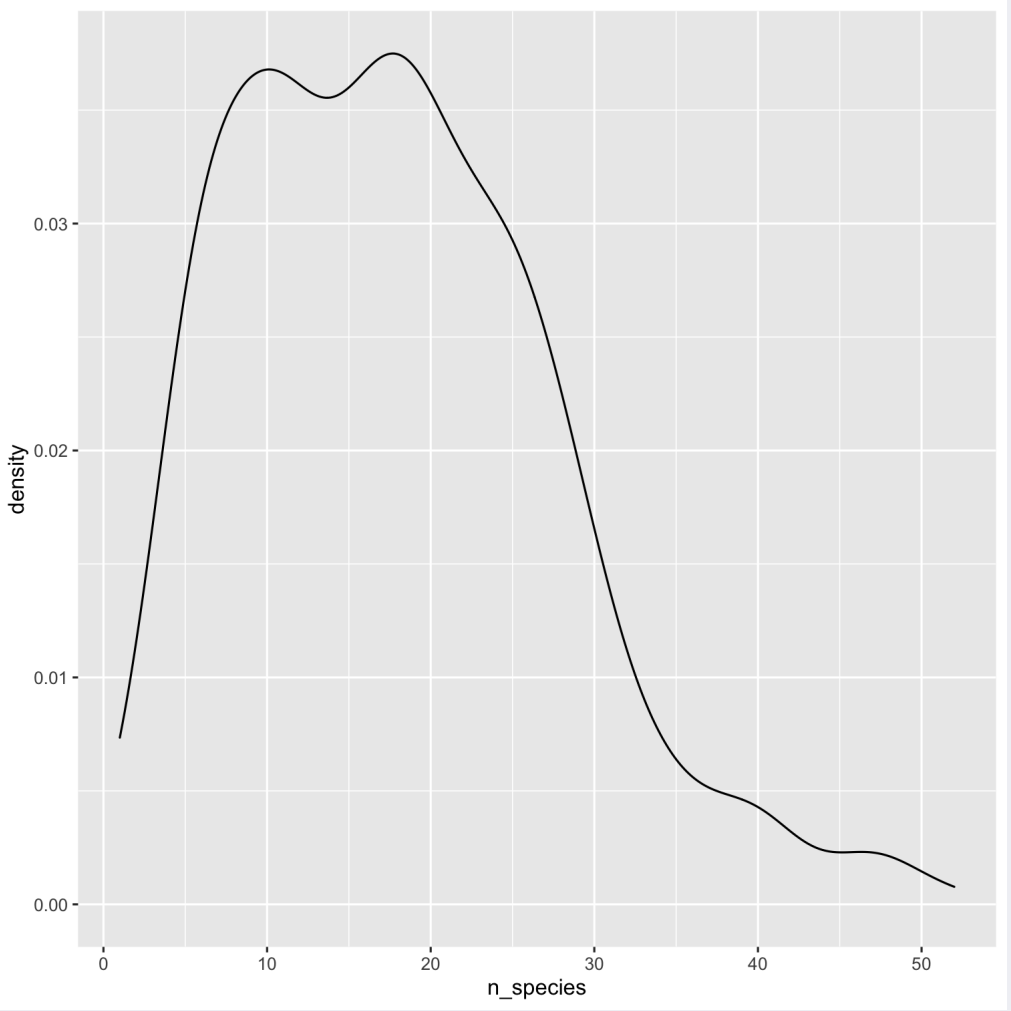
Geoms - Histograma

```
ggplot(but_sum,
       aes(x = n_species)) +
  geom_histogram(color = "blue",
                 binwidth = 1)
```



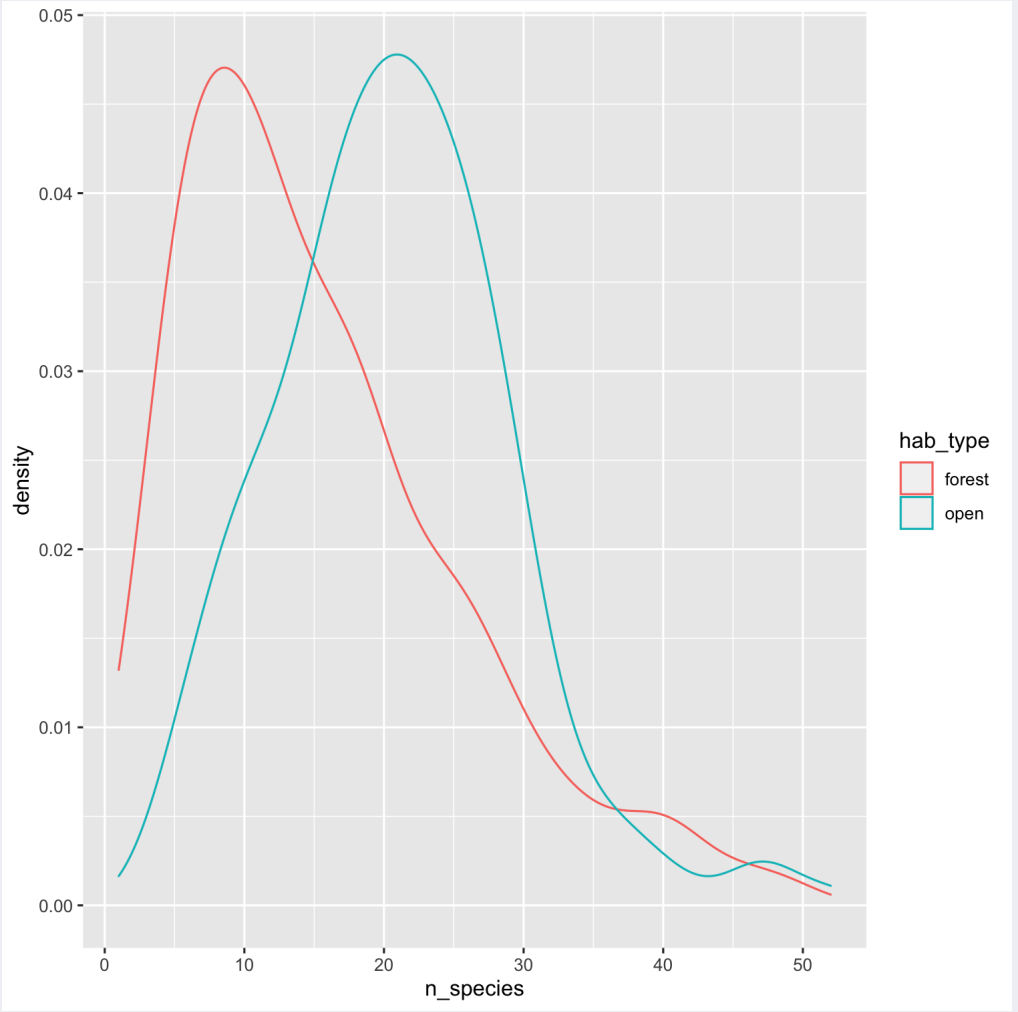
Geoms - Densidad

```
ggplot(but_sum,
       aes(x = n_species)) +
  geom_density()
```



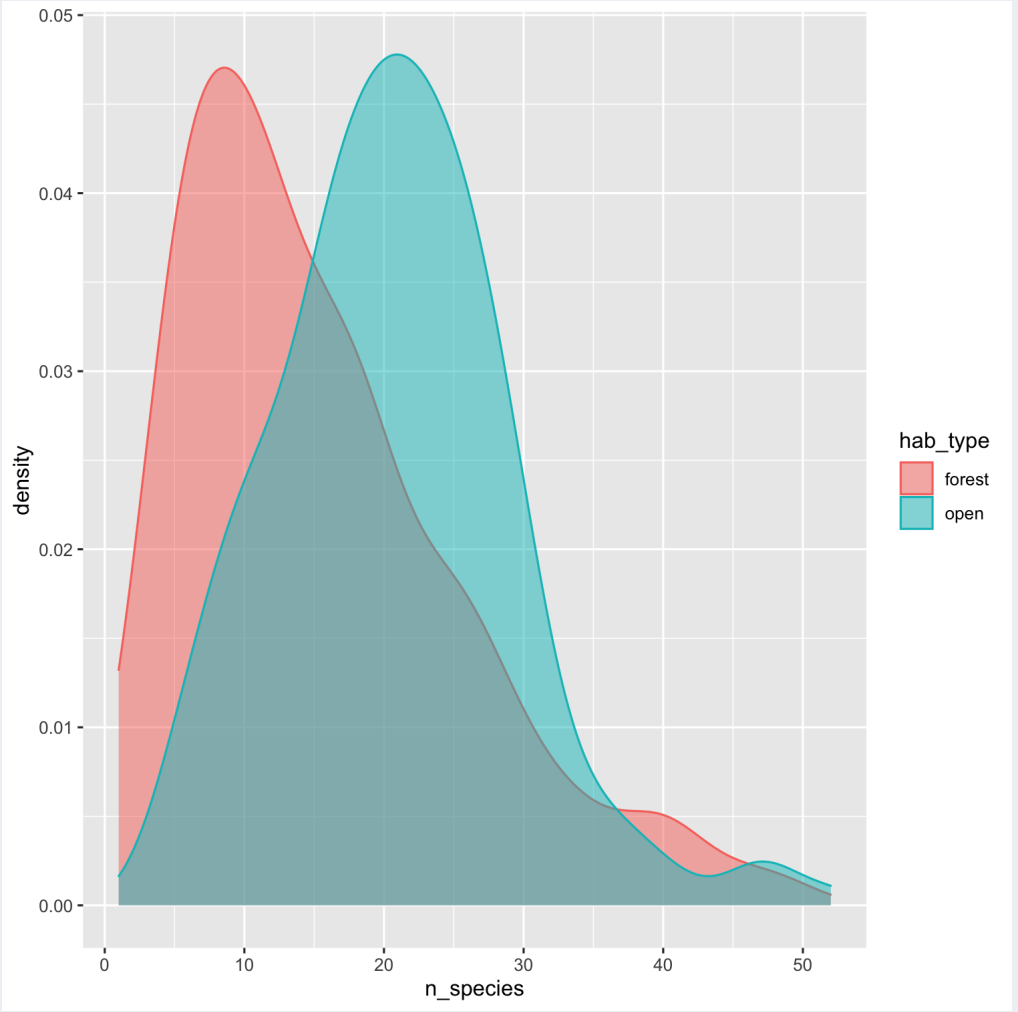
Geoms - Densidad

```
ggplot(but_sum,
       aes(x = n_species,
           color = hab_type)) +
  geom_density()
```



Geoms - Densidad

```
ggplot(but_sum,
       aes(x = n_species,
           color = hab_type,
           fill = hab_type)) +
  geom_density(alpha = 0.5)
```



Ejercicio 4:

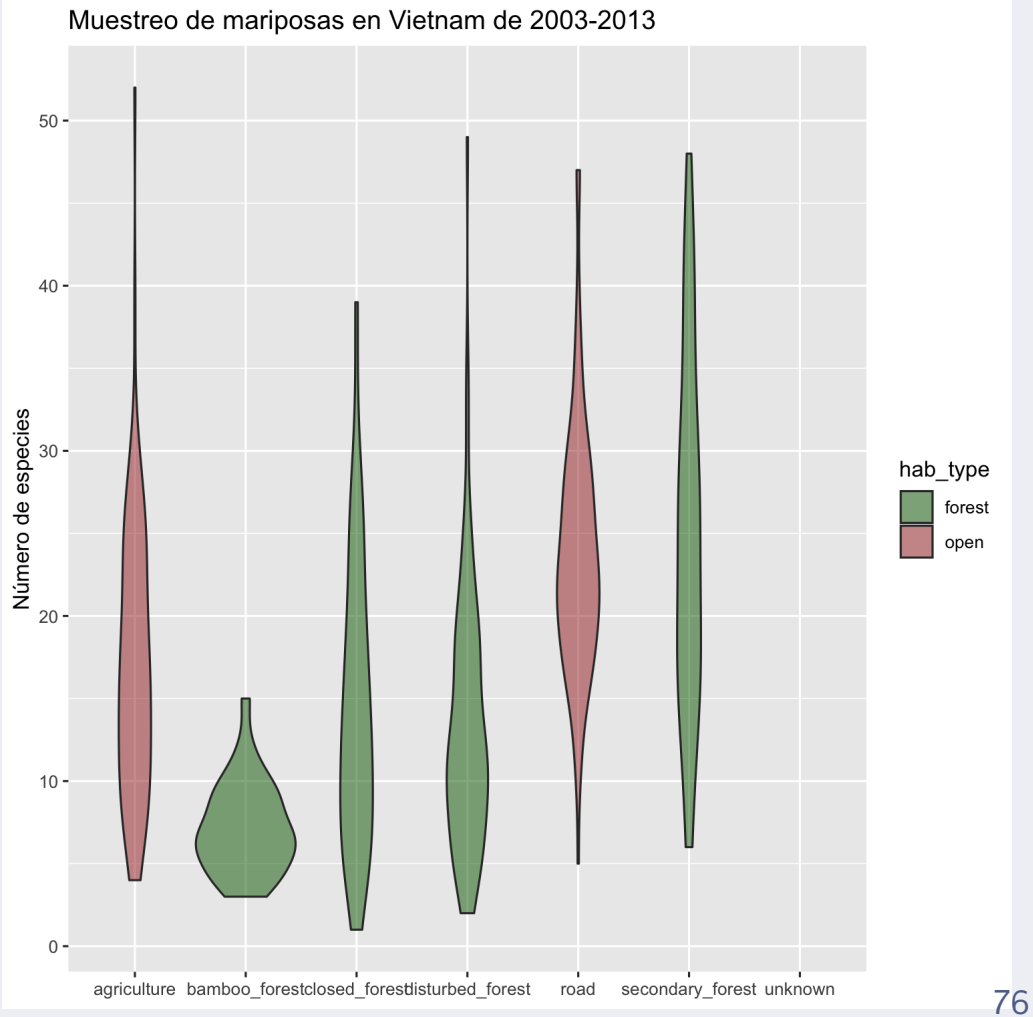
Hacer un violin plot (`geom_violin`) del número de especies de mariposas por hábitat donde el color represente el tipo de hábitat (forest vs. open)

Ejercicio 4:

Hacer un violin plot (`geom_violin`) del número de especies de mariposas por hábitat donde el color represente el tipo de hábitat (forest vs. open)

```
ggplot(but_sum,
       aes(x = habitat,
           y = n_species,
           fill = hab_type)) +
  geom_violin(alpha = 0.5) +
  scale_fill_manual(values =
         c("dark green", "brown")) +
  labs(title = "Muestreo de mariposas en Vietna
       y = "Número de especies",
       x = NULL)
```

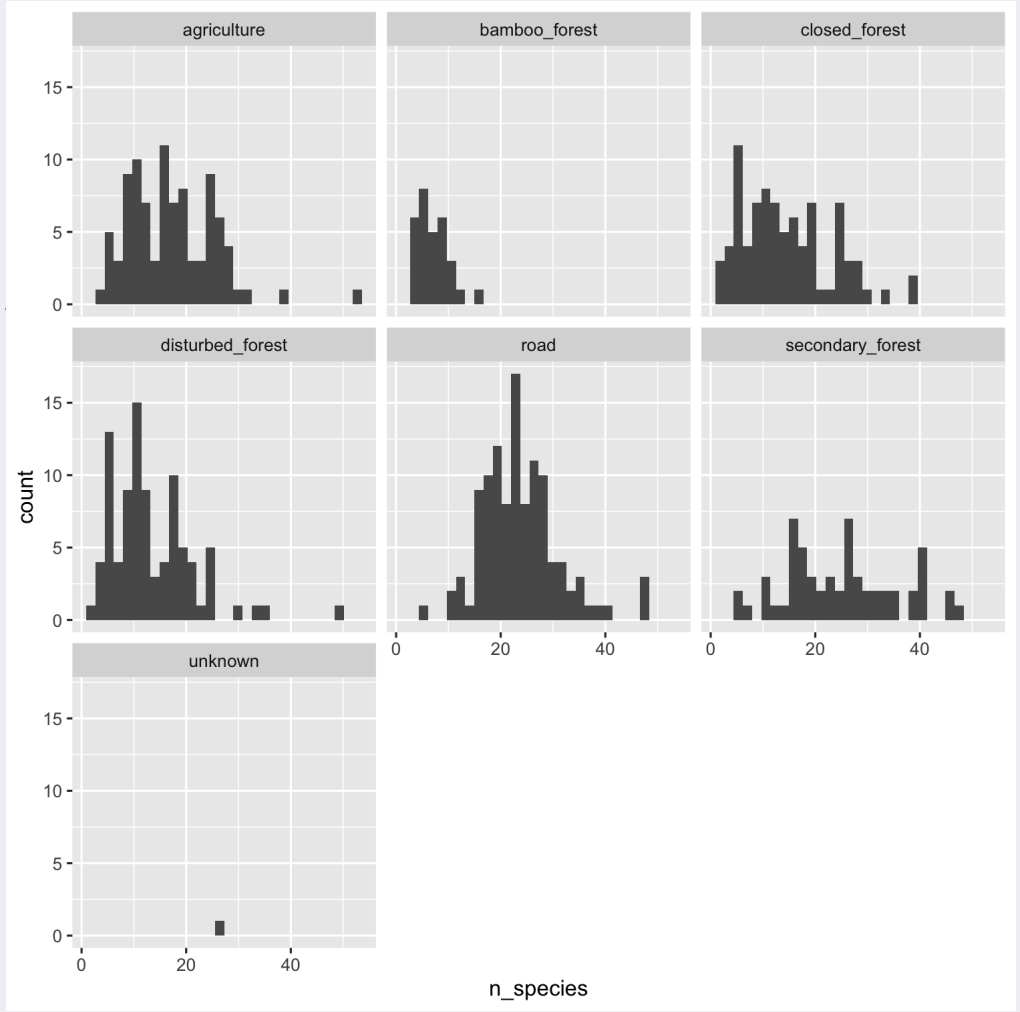
```
## Warning: Groups with fewer than two data points ha
```



8. Facets

```
ggplot(but_sum ,
       aes(x = n_species)) +
  geom_histogram() +
  facet_wrap(~ habitat)
```

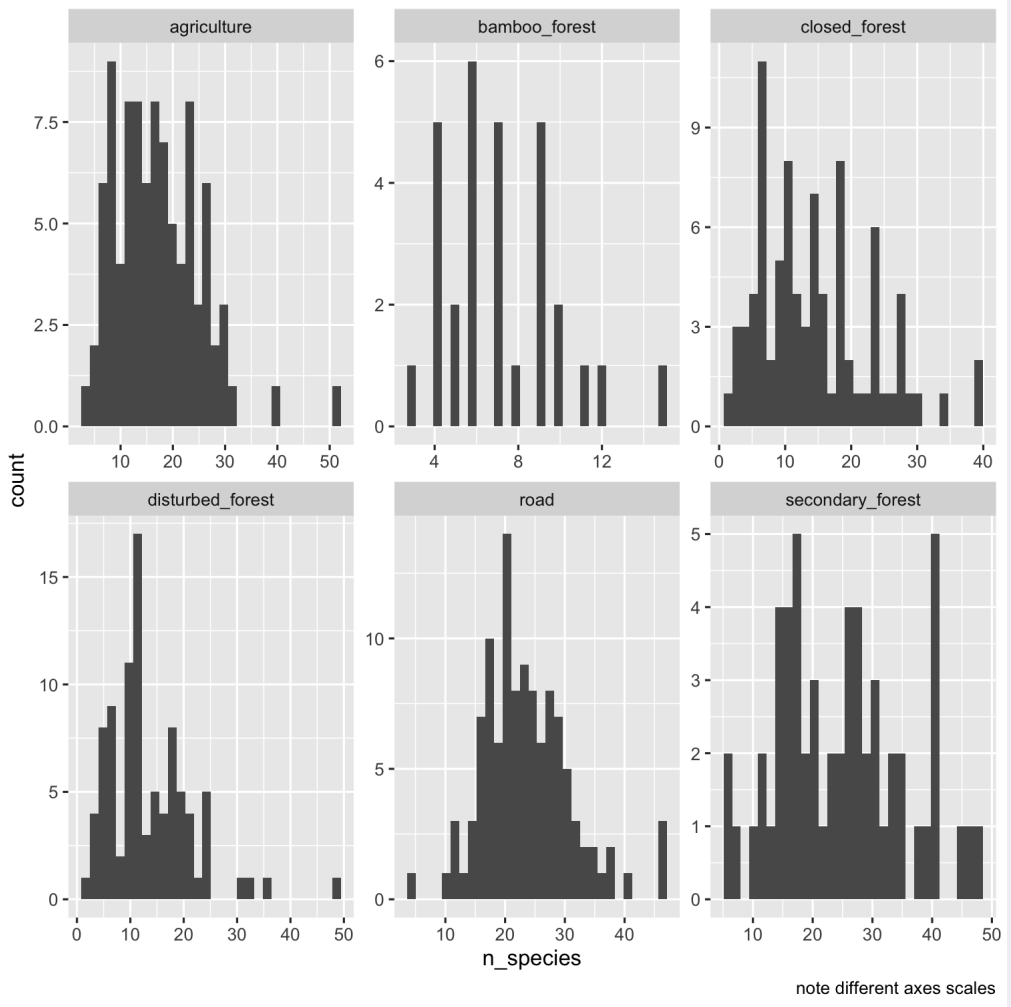
```
## `stat_bin()` using `bins = 30`. Pick better value
```



8. Facets

```
but_sum %>%
  filter(habitat != "unknown") %>%
  ggplot(aes(x = n_species)) +
  geom_histogram() +
  facet_wrap(~ habitat,
             scales = "free") +
  labs(caption =
         "note different axes scales")
```

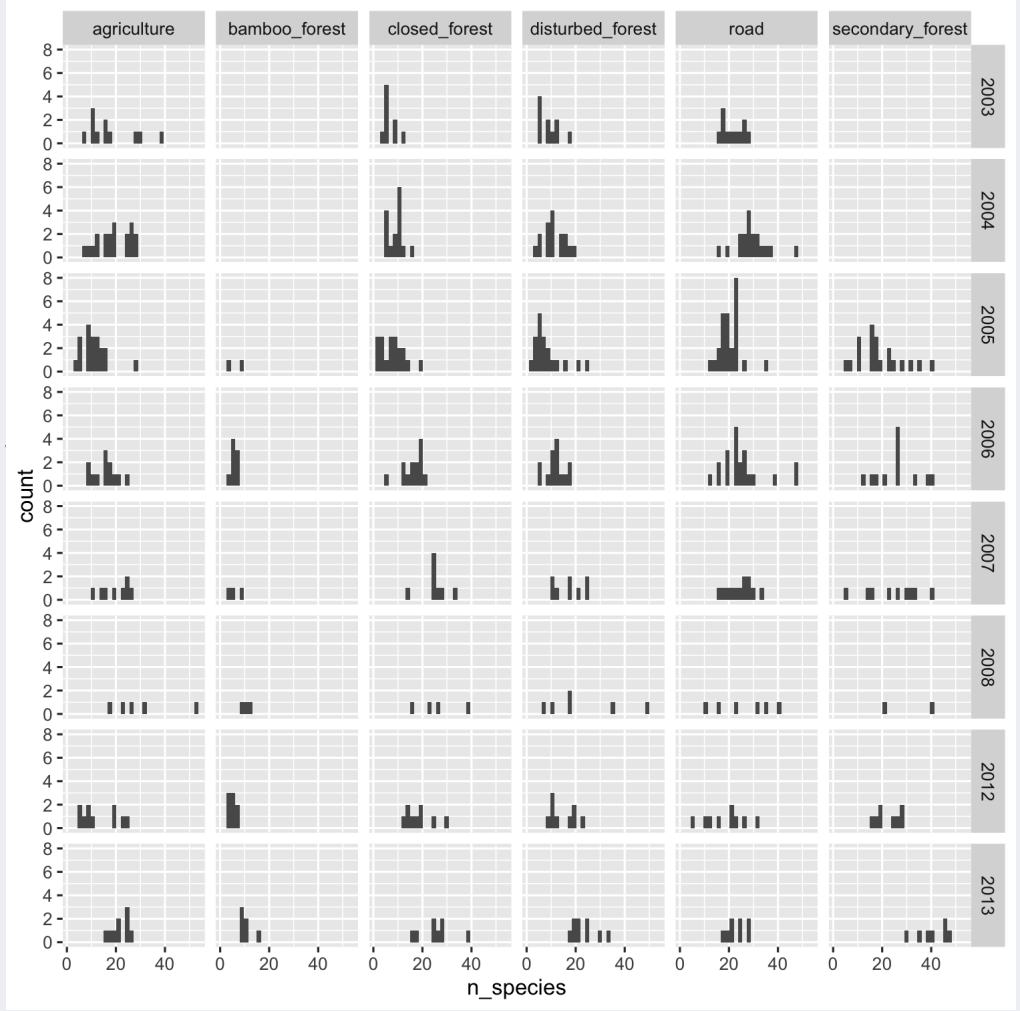
```
## `stat_bin()` using `bins = 30`. Pick better value
```



8. Facets

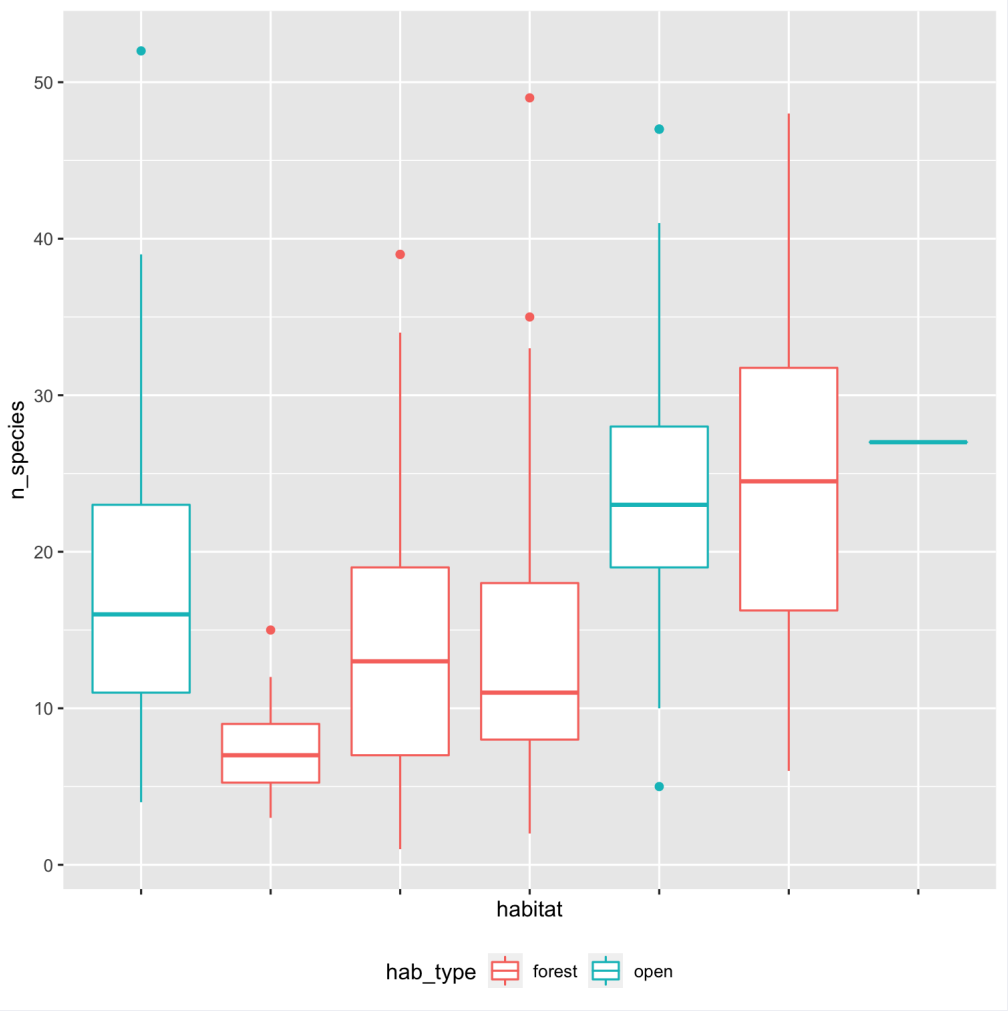
```
but_sum %>%
  filter(habitat != "unknown") %>%
  mutate(year = as.factor(
    lubridate::year(date))) %>%
  ggplot(aes(x = n_species)) +
  geom_histogram() +
  facet_grid(year ~ habitat)
```

```
## `stat_bin()` using `bins = 30`. Pick better value
```



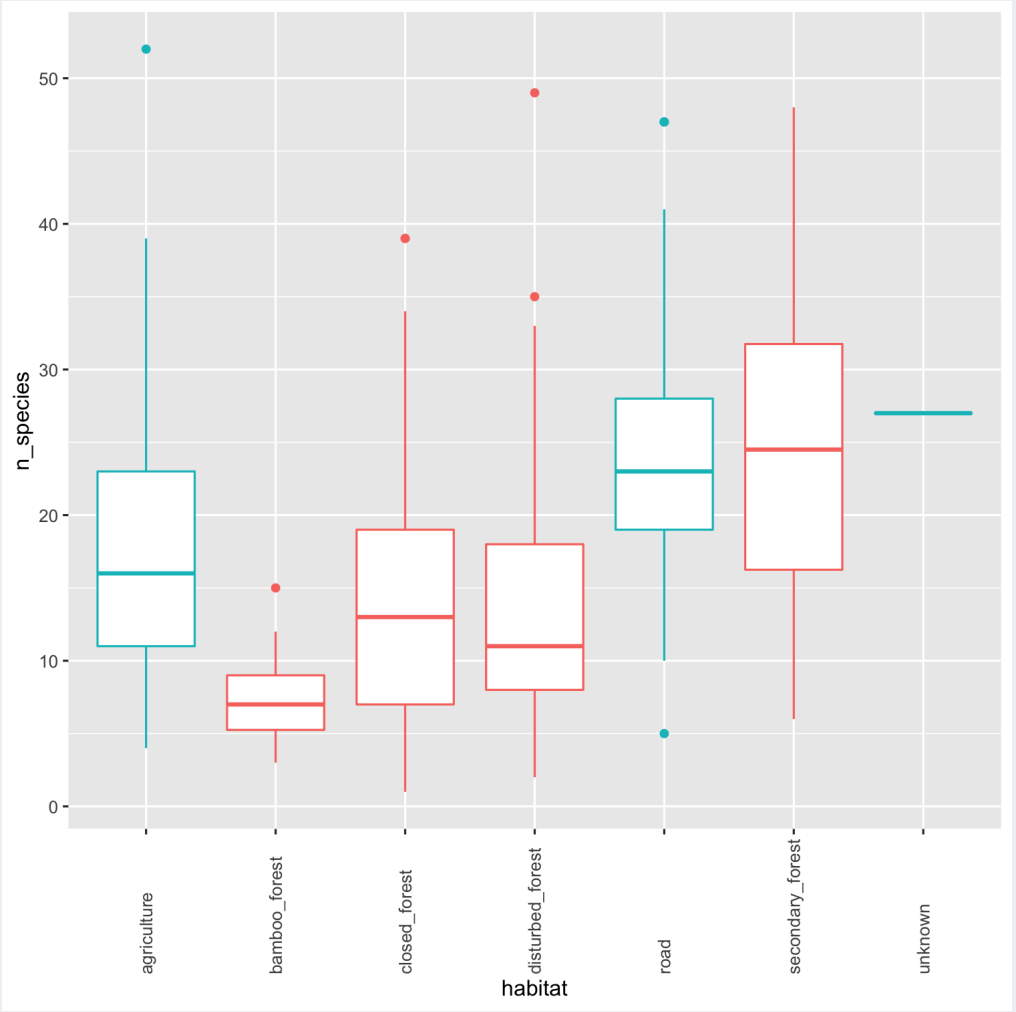
9. Themes

```
ggplot(but_sum,
       aes(x = habitat,
           y = n_species,
           color = hab_type)) +
  geom_boxplot() +
  theme(legend.position = "bottom",
        axis.text.x = element_blank())
```



9. Themes

```
ggplot(but_sum,
       aes(x = habitat,
           y = n_species,
           color = hab_type)) +
  geom_boxplot() +
  theme(legend.position = "none",
        axis.text.x = element_text(
          angle = 90, hjust = 0, vjust = 0.5))
```



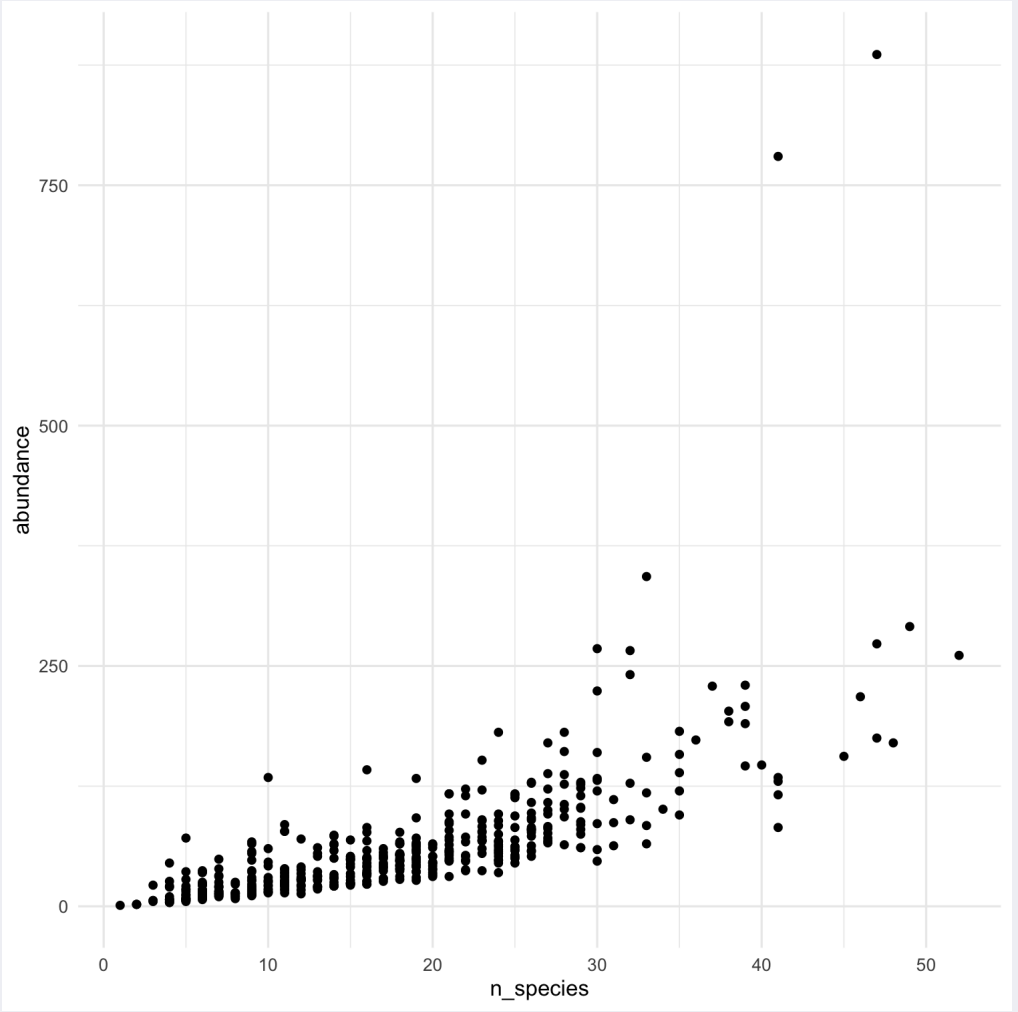
9. Themes

```
?theme
```

- element_blank()
- element_text()
- element_line()
- element_rect() (borders & backgrounds)

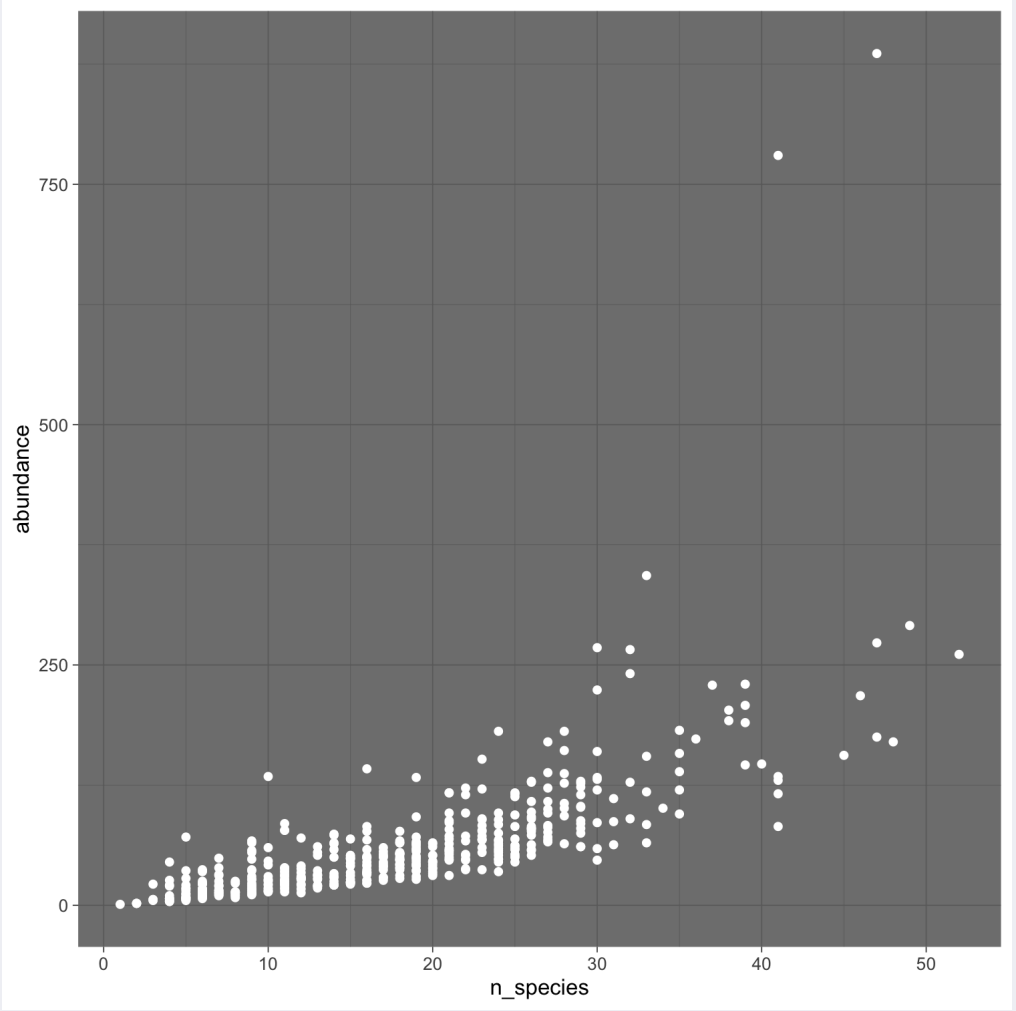
9. Themes

```
ggplot(but_sum,
       aes(x = n_species,
           y = abundance)) +
  geom_point() +
  theme_minimal()
```



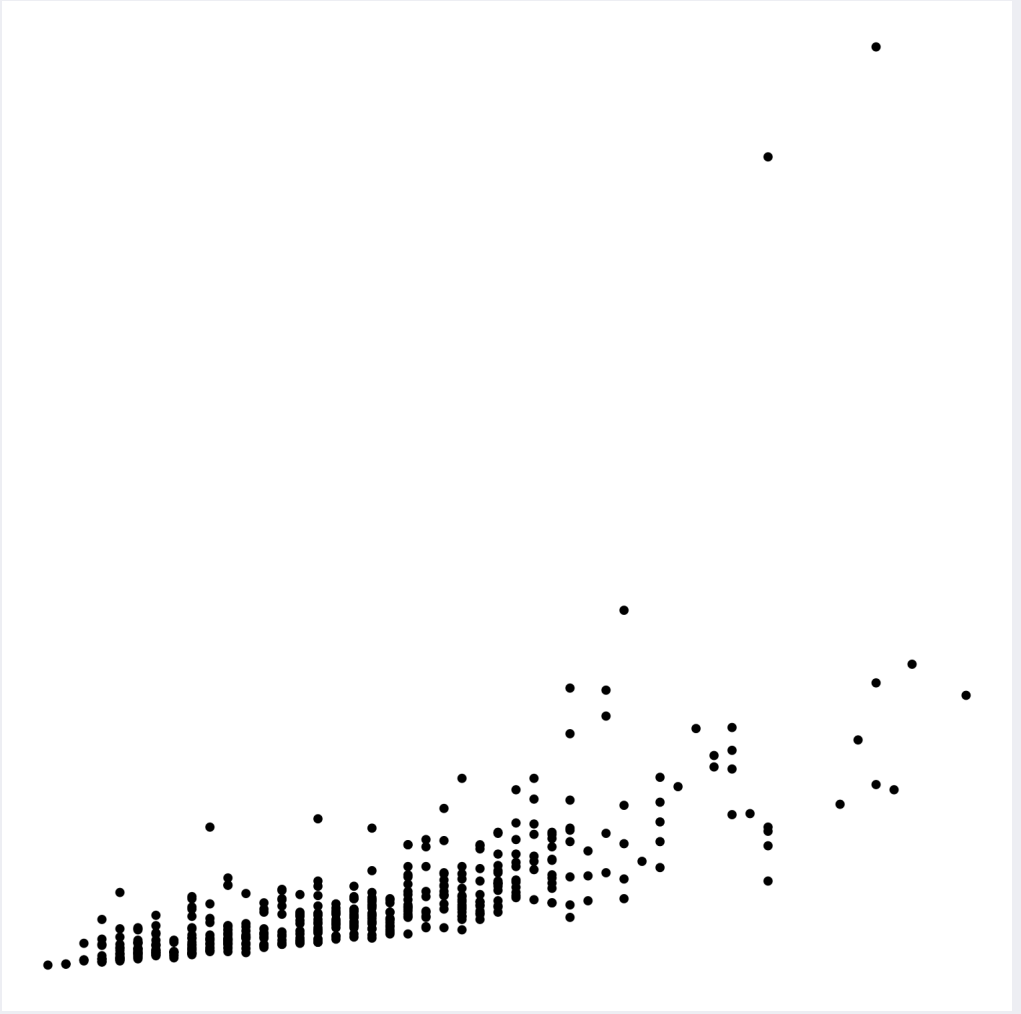
9. Themes

```
ggplot(but_sum,
       aes(x = n_species,
           y = abundance)) +
  geom_point(color= "white") +
  theme_dark()
```



9. Themes

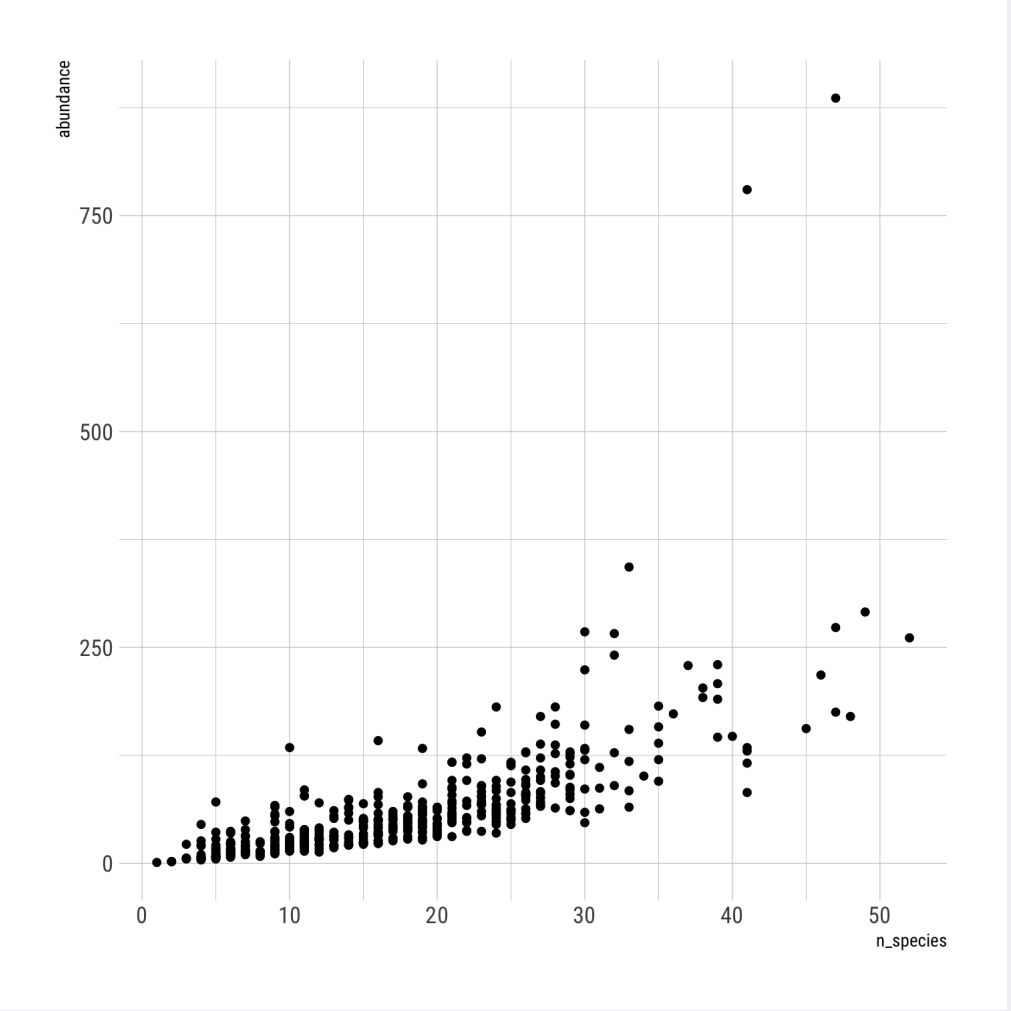
```
ggplot(but_sum,
       aes(x = n_species,
           y = abundance)) +
  geom_point() +
  theme_void()
```



9. Themes

```
library(hrbrthemes)
```

```
ggplot(but_sum,
       aes(x = n_species,
           y = abundance)) +
  geom_point() +
  hrbrthemes::theme_ipsum_rc()
```



9. Themes

Addin de RStudio que ayuda a cambiar la apariencia de los temas.

```
library(ggThemeAssist)
```

https://github.com/calligross/ggthemeassist

9. Themes

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

```
library(hrbrthemes)
library(ggthemes)
library(ggpomological)
#devtools::install_github("gadenbuie/ggpomological")
library(tvthemes)
library(ggtech)
library(ggthemr)
library(ggsci)
```

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

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

10. Composicion de figuras - Patchwork

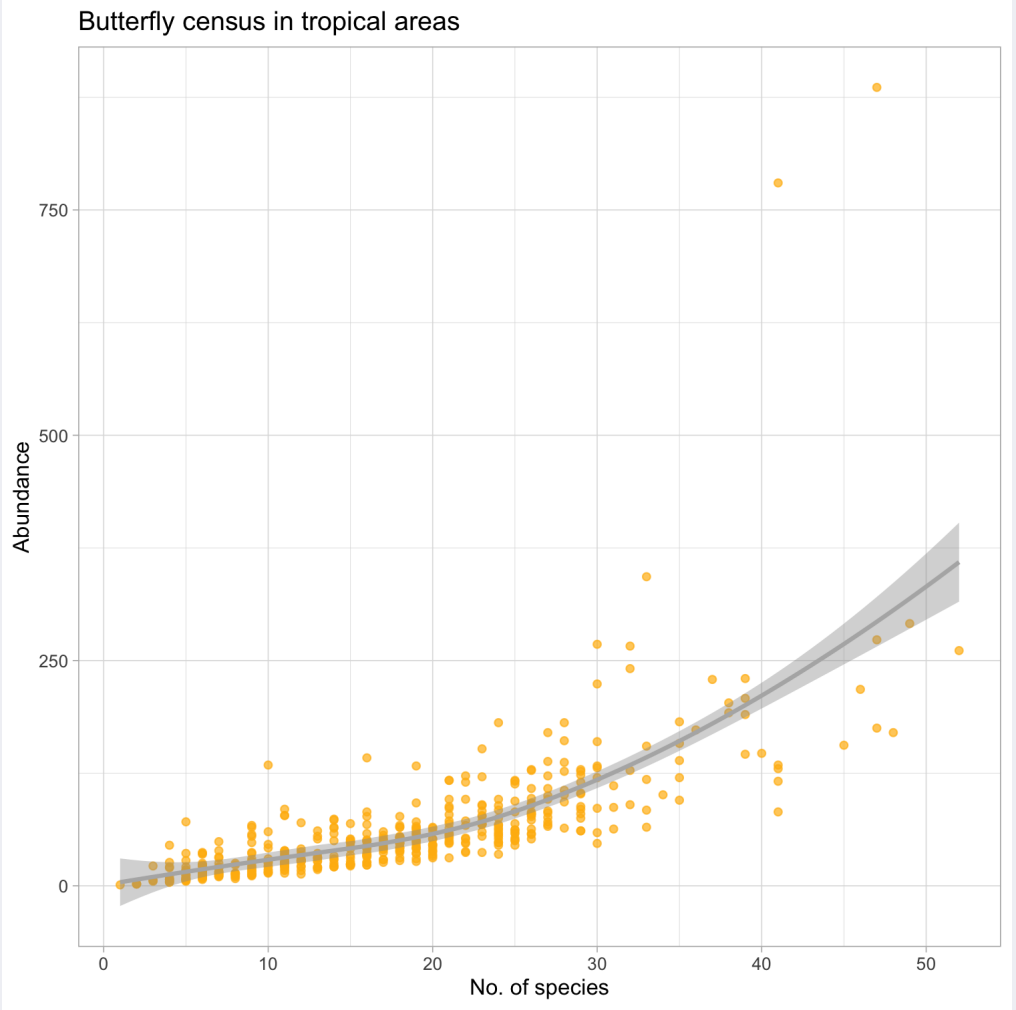
```
library(patchwork)
```

10. Composicion de figuras

Nombrar los plots como objetos

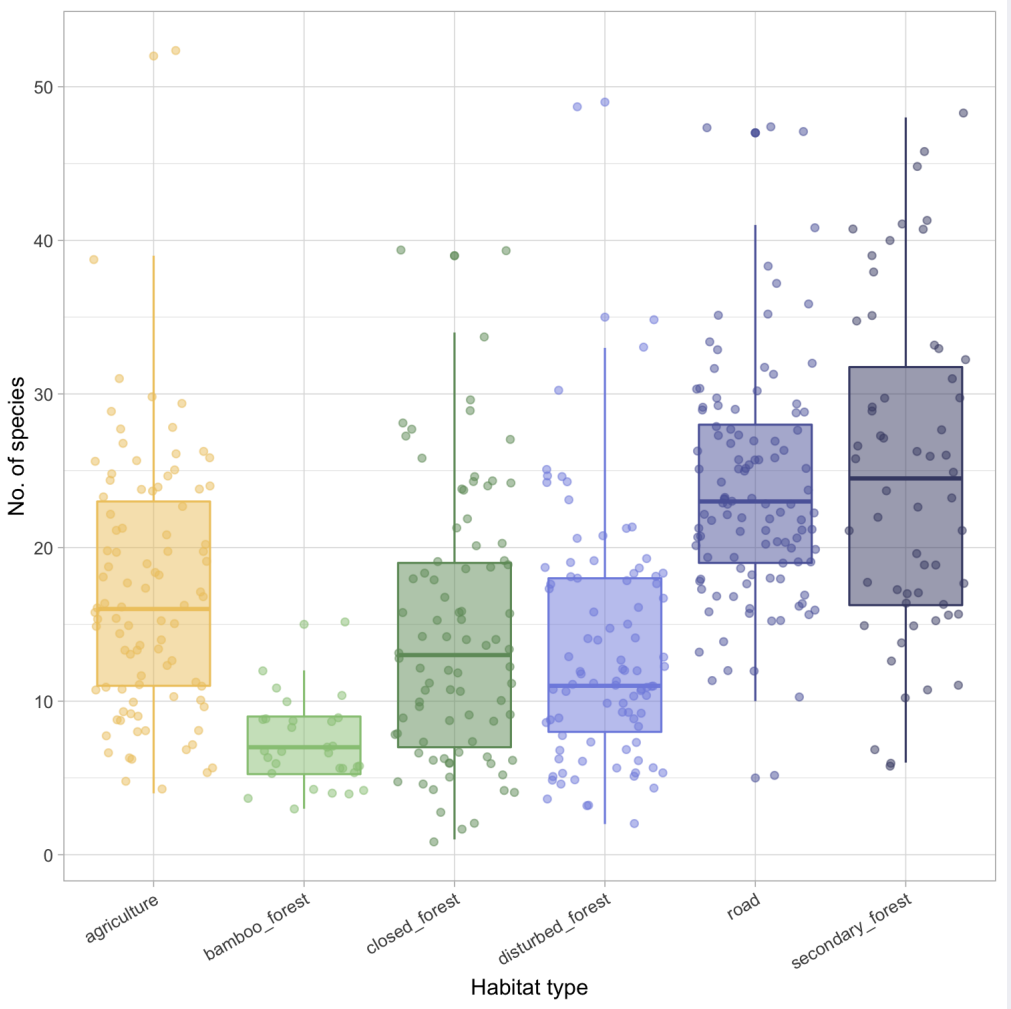
```
p1 <- ggplot(but_sum,
       aes(x = n_species,
           y = abundance)) +
  geom_point(color = "#FFB90F",
             alpha = 0.7) +
  geom_smooth(color = "grey70") +
  theme_light() +
  labs(x = "No. of species",
       y = "Abundance")

p1 + labs(
  title = "Butterfly census in tropical areas"
)
```

```
## `geom_smooth()` using method = 'loess' and formula
```



10. Composicion de figuras

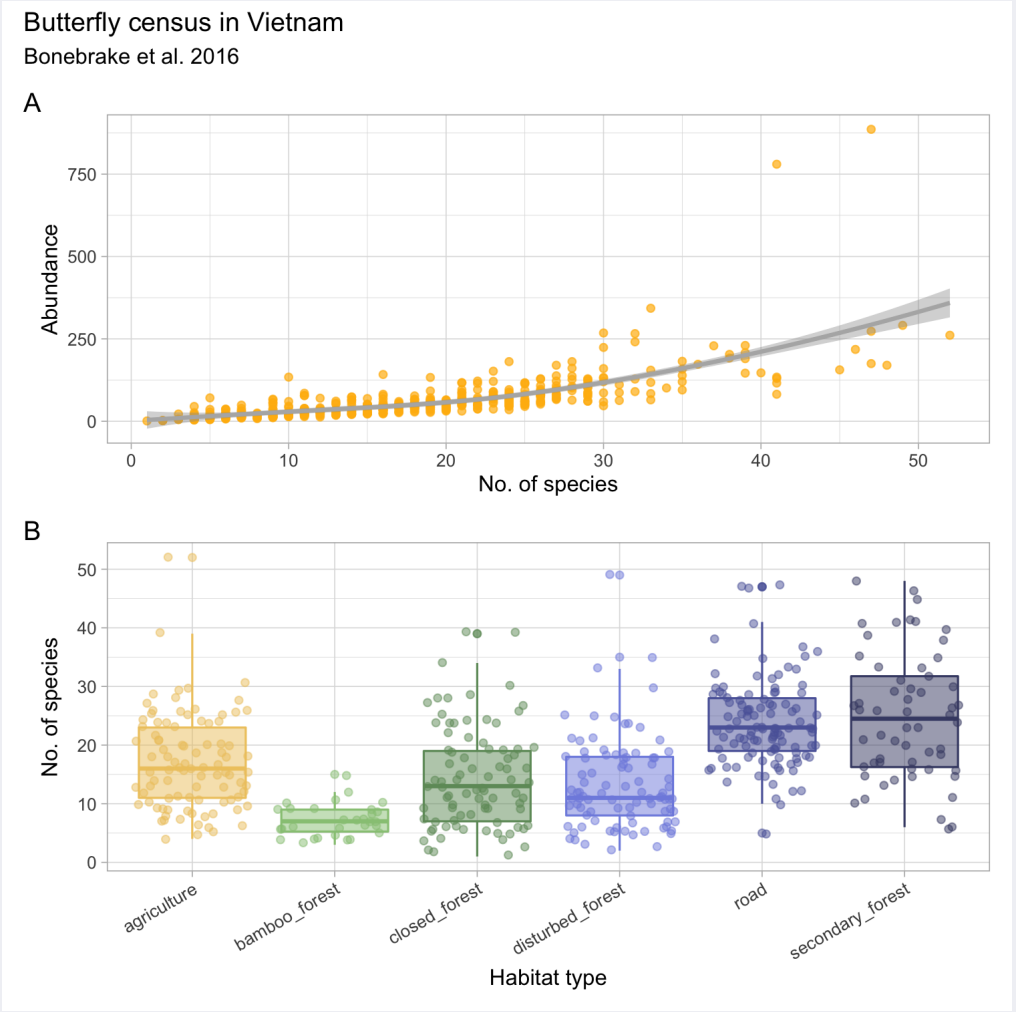
```
p2 <-
  ggplot(filter(but_sum, habitat != "unknown")
         aes(x = habitat,
             y = n_species,
             colour = habitat,
             fill = habitat)) +
  geom_boxplot(alpha = .5) +
  geom_jitter(alpha = .5) +
  MetBrewer::scale_fill_met_d("Derain") +
  MetBrewer::scale_color_met_d("Derain") +
  theme_light() +
  theme(legend.position = "none",
        axis.text.x = element_text(
          angle = 30, hjust = 1)) +
  labs(y = "No. of species",
       x = "Habitat type",)

p2
```



10. Composicion de figuras

```
p1 / p2 + plot_annotation(
  title = "Butterfly census in Vietnam",
  subtitle = "Bonebrake et al. 2016",
  tag_levels = 'A')
```

```
## `geom_smooth()` using method = 'loess' and formula
```



Guardar gráficos

```
plot_to_save <- p1 / p2 + plot_annotation(
  title = "Butterfly census in Vietnam",
  subtitle = "Bonebrake et al. 2016",
  tag_levels = 'A')
```

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

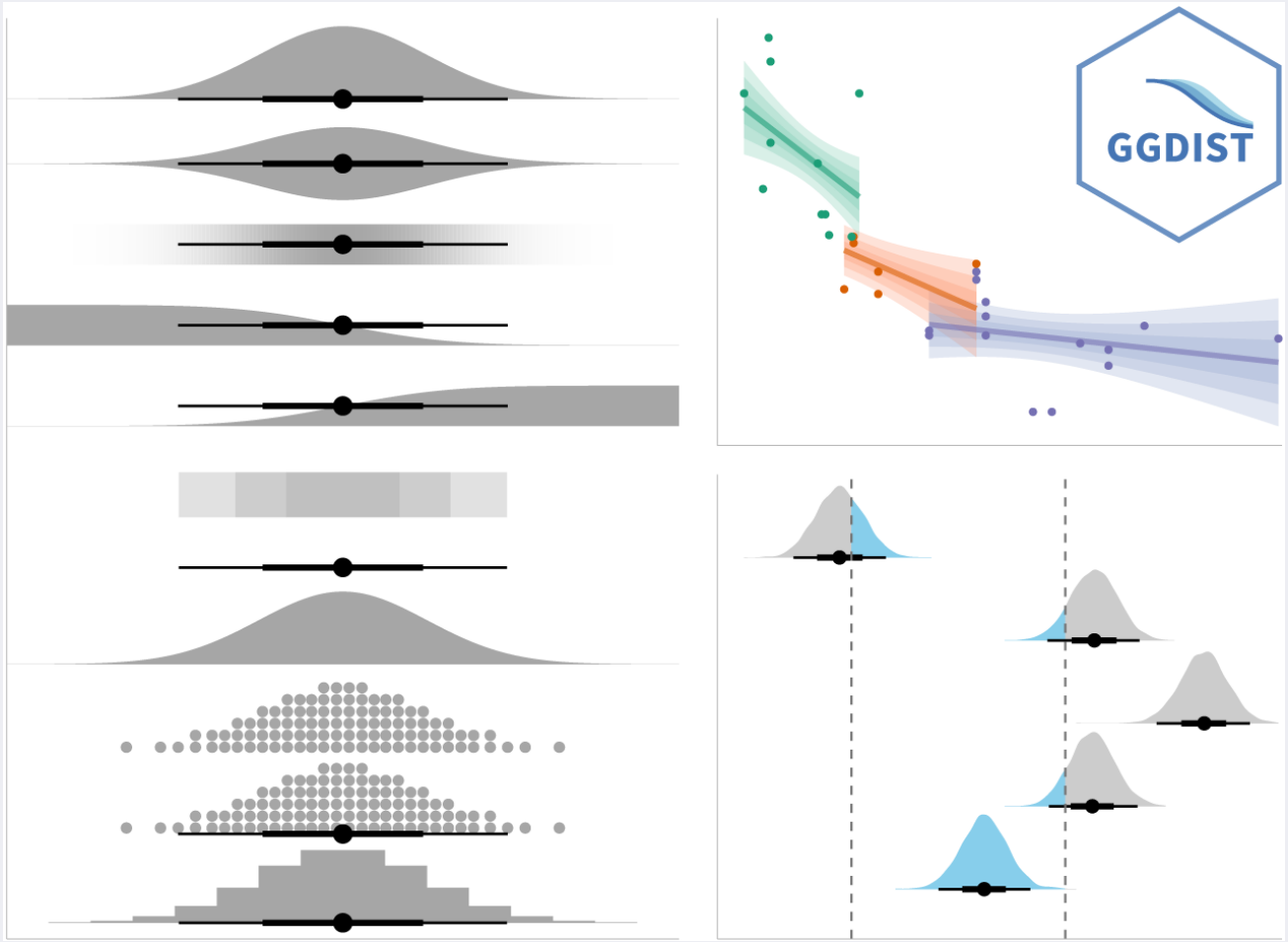
```
## Saving 20 x 17.8 cm image

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

Argumentos:

- width
- height
- units = ("in", "cm", "mm", "px")
- device = ("png", "pdf, "jpeg", "tiff", "svg"...)

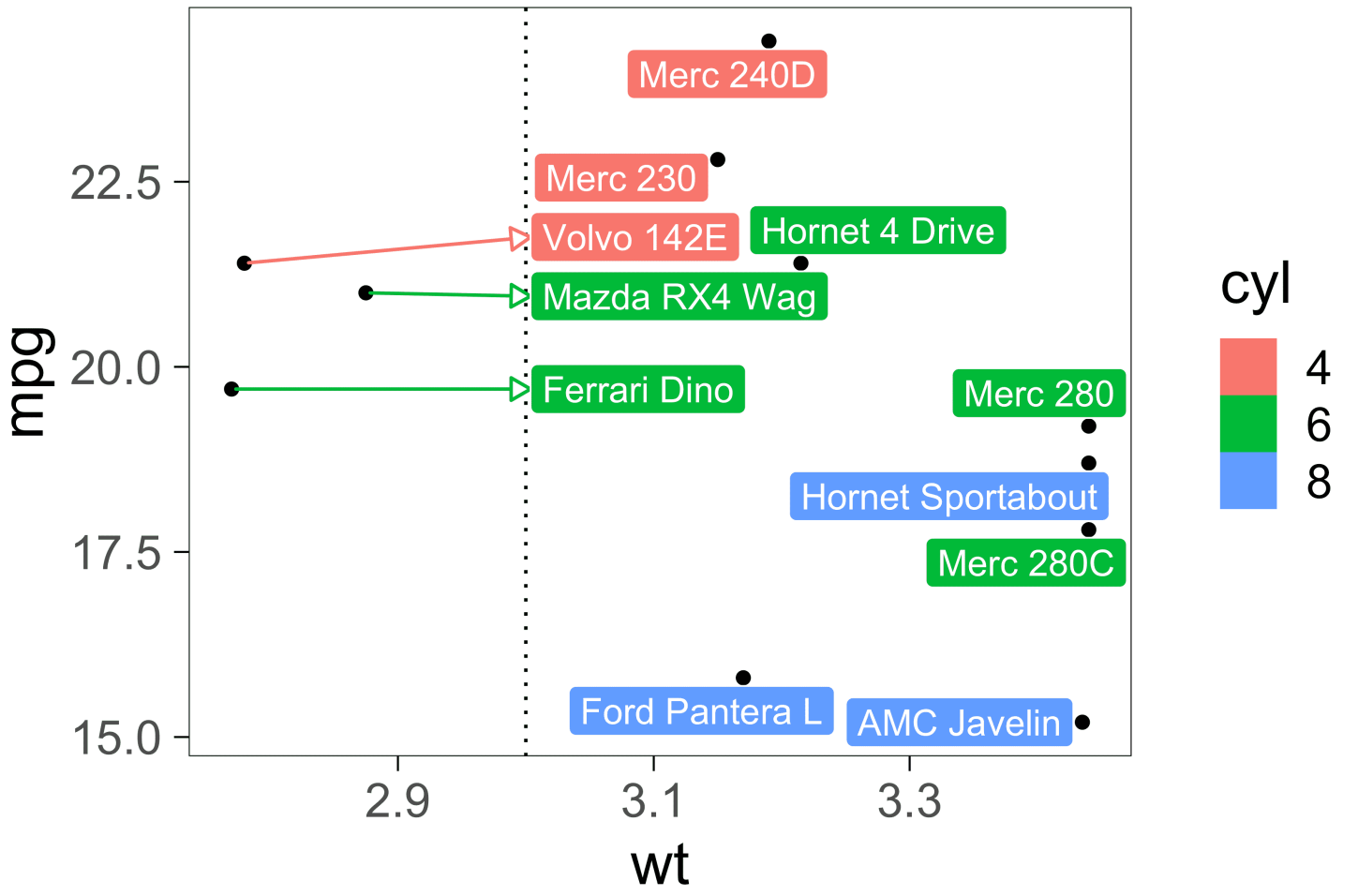
Otros paquetes útiles - `library(ggdist)`

Graficar datos con distribuciones amplias (ej. Bayes posteriors).



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

Otros paquetes útiles - `library(ggrepel)`



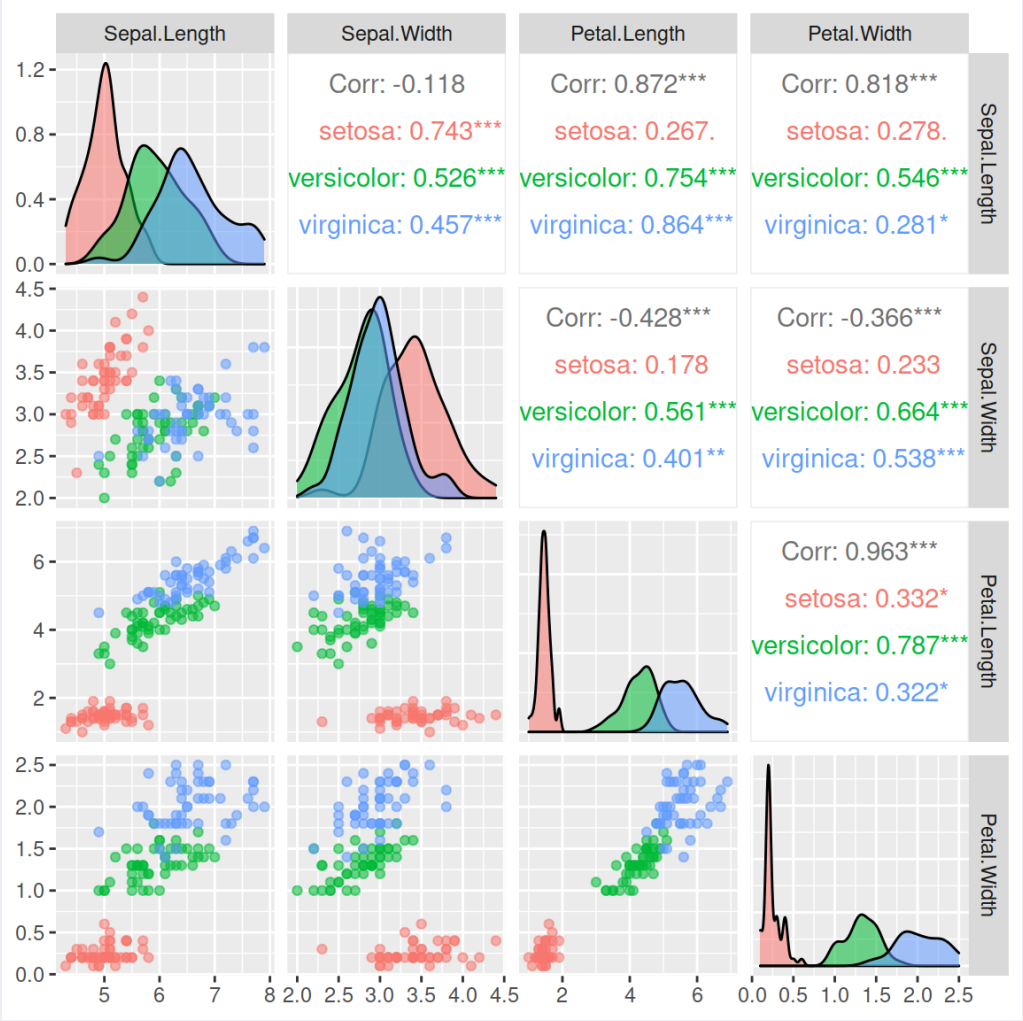
https://cran.r-project.org/web/packages/ggrepel/vignettes/ggrepel.html

Otros paquetes útiles - `library(GGally)`

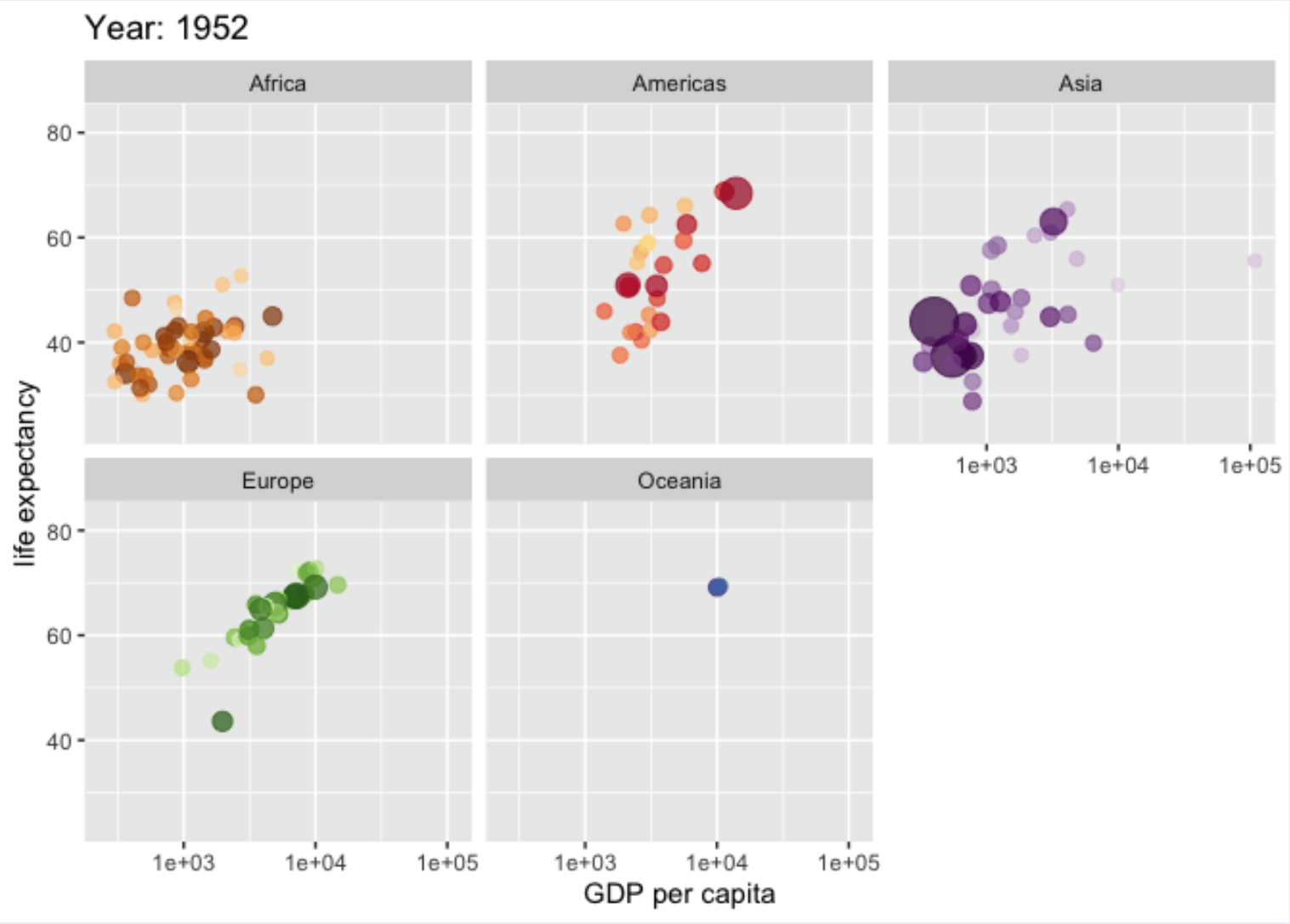
Para la exploración de datos y relación entre variables.

https://github.com/ggobi/ggally

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

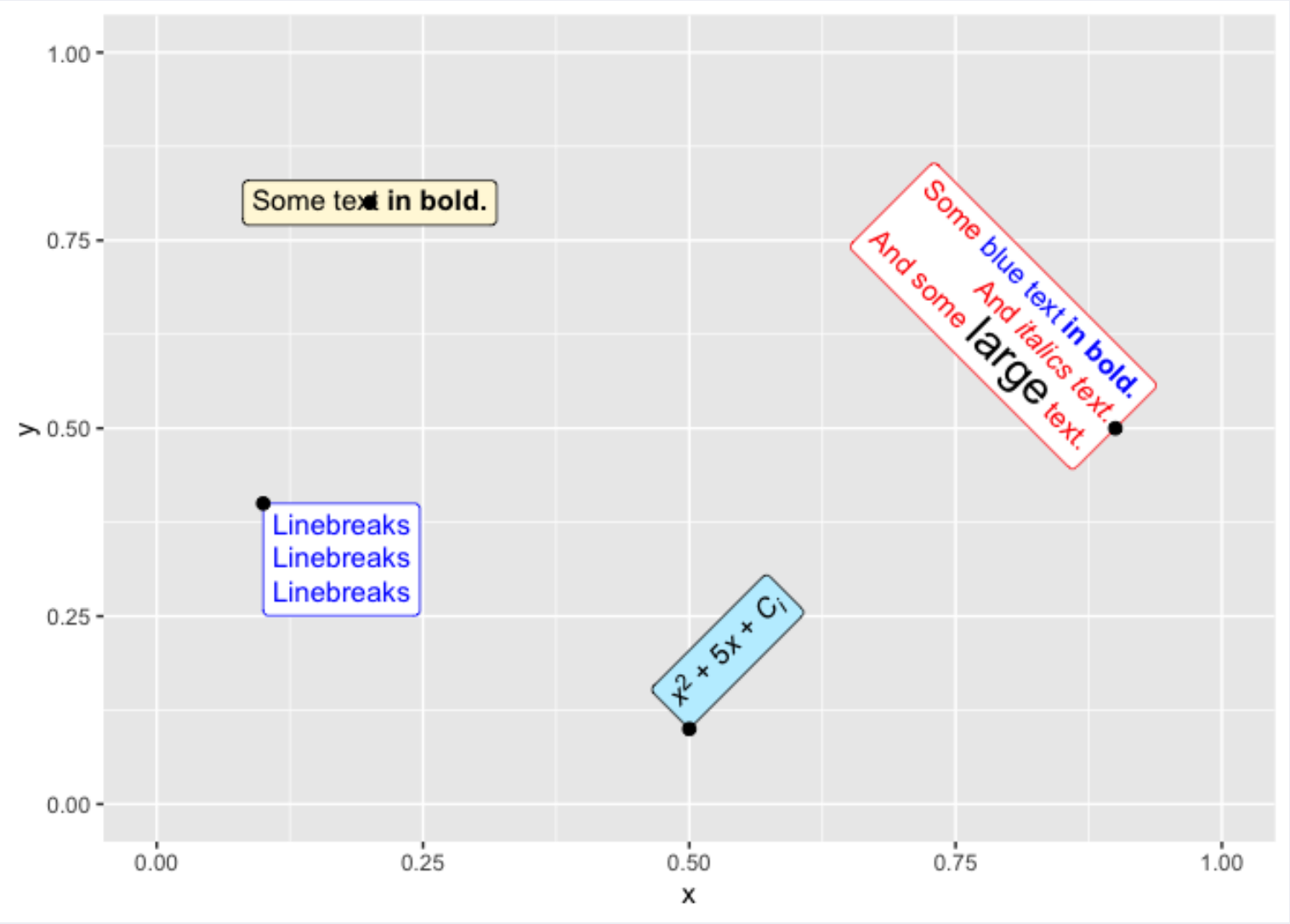


Otros paquetes útiles - `library(gganimate)`



https://gganimate.com/

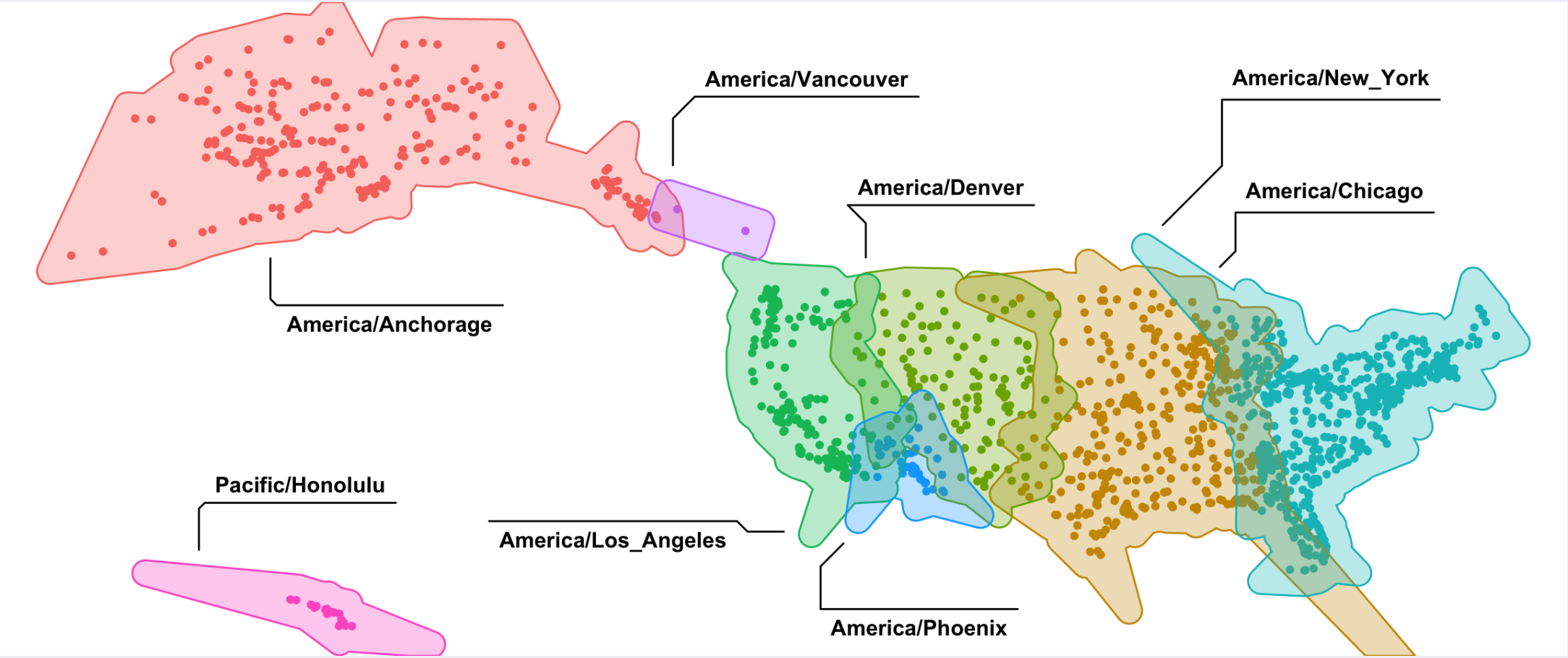
Otros paquetes útiles - `library(ggtext)`



https://github.com/wilkelab/ggtext

Otros paquetes útiles - `library(ggforce)`

Resaltar características de los datos.



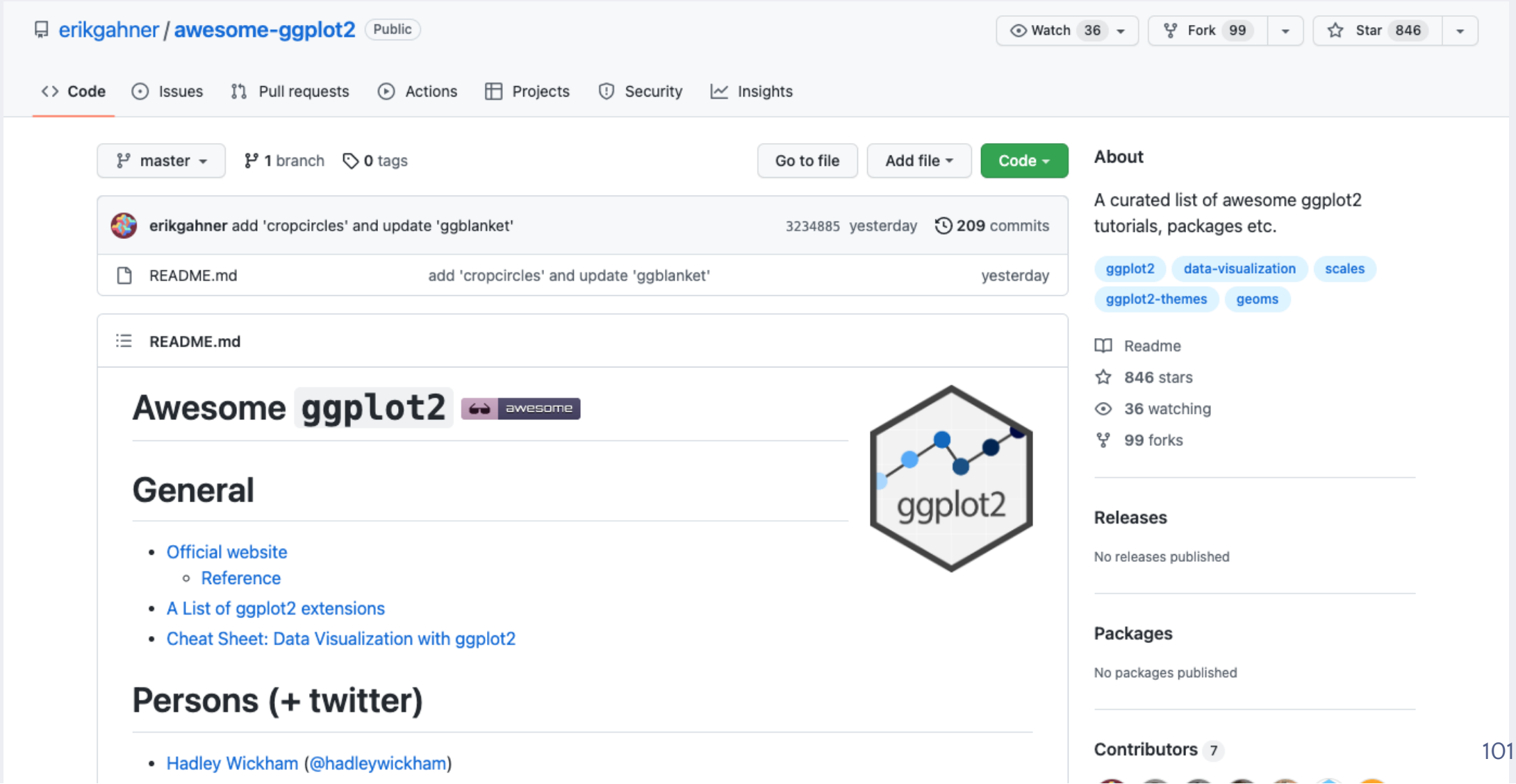
https://github.com/thomasp85/ggforce

Más extensiones de GGplot

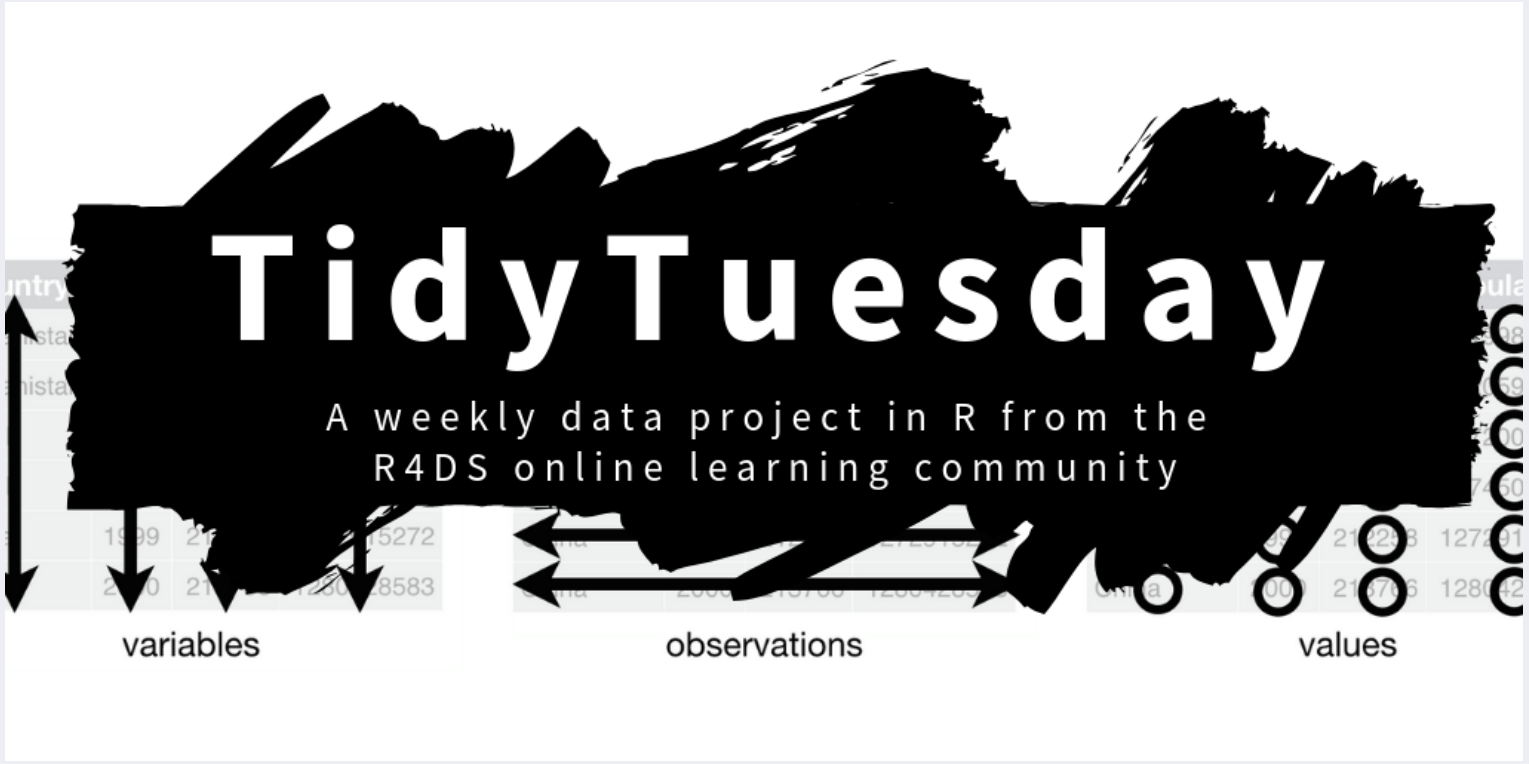
https://exts.ggplot2.tidyverse.org/gallery/

Más extensiones de GGplot

https://github.com/erikgahner/awesome-ggplot2



#TidyTuesday

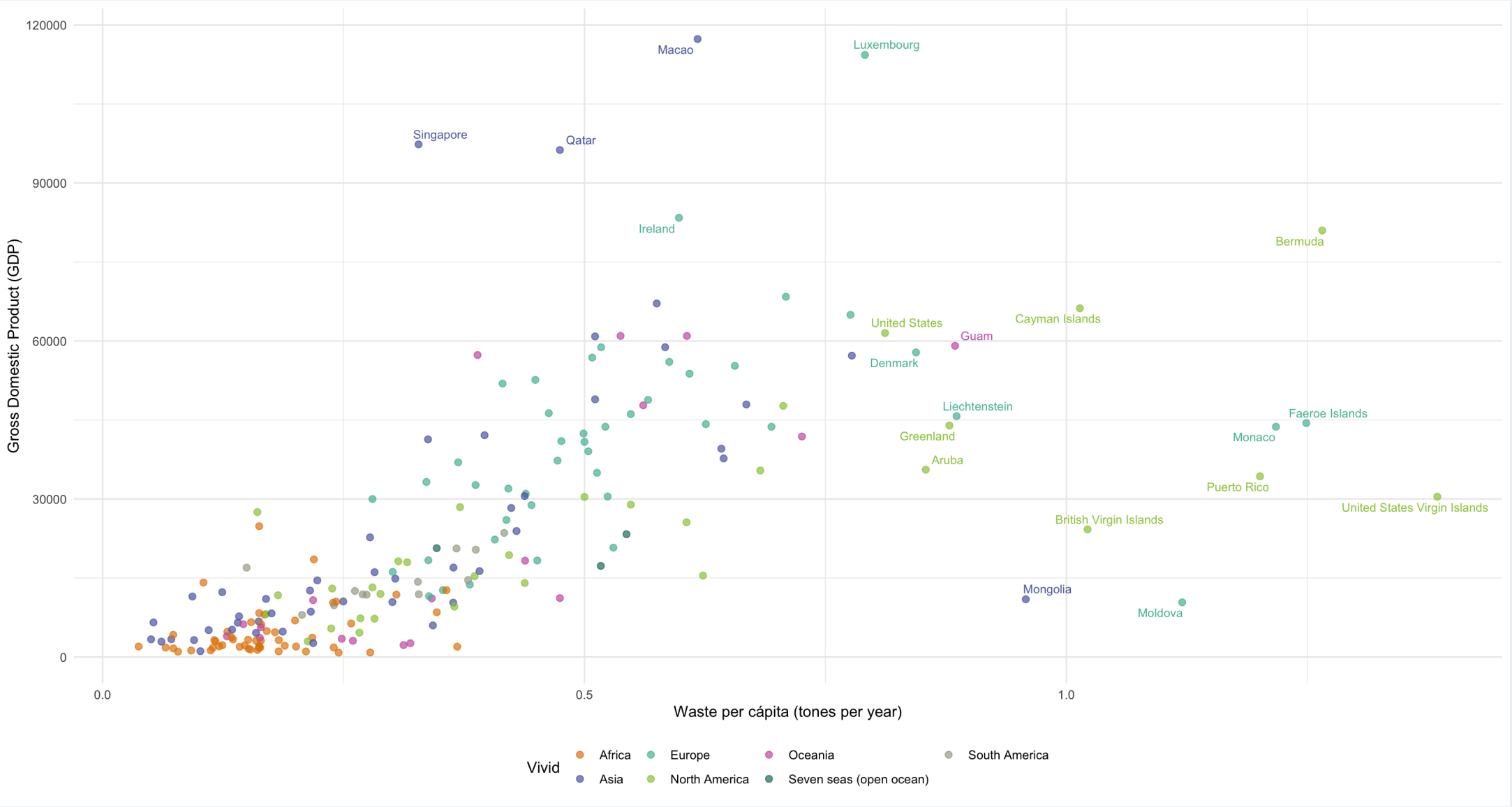


https://github.com/rfordatascience/tidytuesday

Recursos

- Referencia oficial 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"*

Ejemplo con el dataset de waste_world



Ejemplo con el dataset de waste_world

Leer datos:

```
waste_world <- read_csv(here("data/waste_world.csv"))
```

```
## Rows: 217 Columns: 18
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (6): iso_a3, region_id, country, income_id, country_name, continent
## dbl (12): gdp, population, total_waste, composition_food_organic_waste_percent, composition_glass_perce
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Ejemplo con el dataset de waste_world

Generar dataset para la figura:

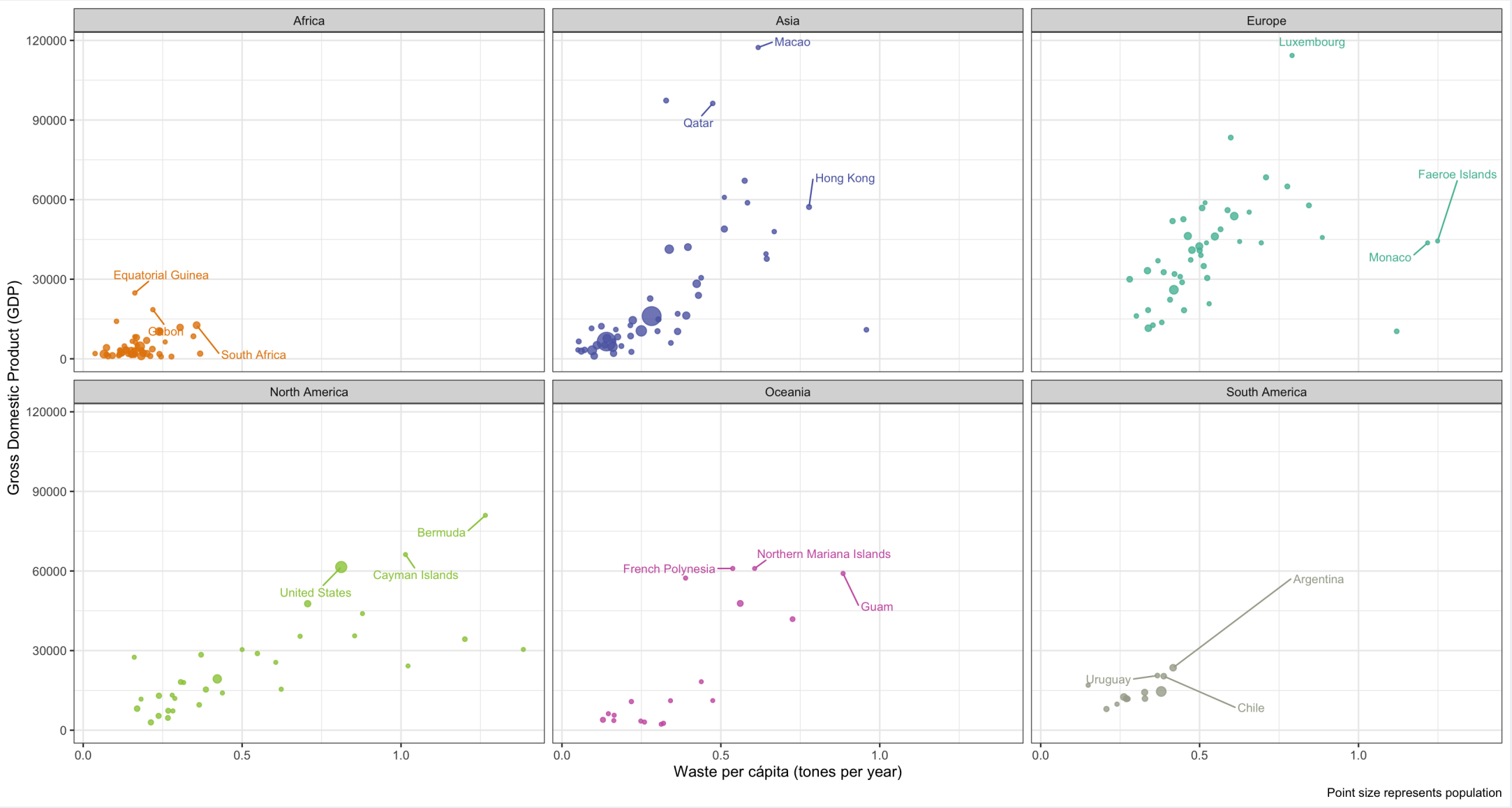
```
waste_world_mod <- waste_world %>%
  mutate(waste_per_capita = total_waste/population) %>%
  filter(!is.na(waste_per_capita) & !is.na(continent))
```

Ejemplo con el dataset de waste_world

Código de la gráfica:

```
ggplot(waste_world_mod, aes(
            x = waste_per_capita,
            y = gdp,
            color = continent)) +
  geom_point(alpha = 0.7,
             size = 2) +
  rcartocolor::scale_color_carto_d("Vivid") +
  ggrepel::geom_text_repel(data = waste_world_mod %>%  #mostar solo la etiqueta para aquellos paises
                             filter(waste_per_capita > 0.8 |
                                      gdp > 80000),
                           aes(x = waste_per_capita,
                               y = gdp,
                               label = country_name),
                           size = 3,
                           show.legend = FALSE) + #para que no muestre texto en la leyenda
  theme_minimal() +
  theme(legend.position = "bottom") +
  labs(x = "Waste per cápita (tones per year)",
       y = "Gross Domestic Product (GDP)")
```

Ejemplo con el dataset de waste_world



Ejemplo con el dataset de waste_world

Generar dataset para la gráfica 2:

```
waste_world_mod2 <- waste_world %>%
  mutate(waste_per_capita = total_waste/population) %>%
  filter(!is.na(waste_per_capita) & !is.na(continent)) %>%
  filter(continent != "Seven seas (open ocean)")
```

Ejemplo con el dataset de waste_world

Código de la gráfica 2:

```
ggplot(waste_world_mod2, aes(
            x = waste_per_capita,
            y = gdp,
            color = continent,
            size = population)) +
  geom_point(alpha = 0.8) +
  rcartocolor::scale_color_carto_d("Vivid") +
  ggrepel::geom_text_repel(data = waste_world_mod2 %>% #código para seleccionar que etiquetas mostra
                             group_by(continent) %>%
                             mutate(order = waste_per_capita*gdp) %>% #variable en la que basar qué
                             arrange(desc(order)) %>% #ordenar por continente los países que tengan
                             slice(1:3), #seleccionar los 3 países con valores más altos
                           aes(x = waste_per_capita,
                               y = gdp,
                               label = country_name),
                            force = 100, #repulsion entre etiquetas
                           size = 3) + #tamaño de la etiqueta
  theme_bw() +
  theme(legend.position = "none") +
  facet_wrap(~continent) +
  labs(x = "Waste per cápita (tones per year)",
       y = "Gross Domestic Product (GDP)",
       size = "Population",
       caption = "Point size represents population")
```