4. Visualización

En este apartado nos enfocaremos en entender por qué visualizar datos y cómo construirlos utilizando `ggplot()`.

4.1. Pregunta-problema

Caracterizar la territorialidad del voto en la Provincia de Buenos Aires.

```
library(tidyverse)

data <- read_csv("data/encuentro_1/ARG_elecciones.csv")

head(data,2)
```

```
# A tibble: 2 x 10
  id         seccion Elecciones Partido Porcentaje Votos Participacion electores
  <chr>      <chr>   <chr>      <chr>        <dbl> <dbl>         <dbl>     <dbl>
1 BUENOS AI~ Adolfo~ BALLOTAGE~ BLANCO        1.19   134          79.2     14171
2 BUENOS AI~ Albert~ BALLOTAGE~ BLANCO        1.6    123          84.2      9147
# i 2 more variables: votantes <dbl>, Provincia <chr>
```

4.2. ¿Por qué visualizar?

> **Atención**
>
> La visualización de datos es parte arte y parte ciencia y, como bien dice Claus Wilke, el desafío es realizar correctamente el arte sin desfigurar la ciencia (y viceversa).

Hay tres razones centrales por las que visualizamos la información:

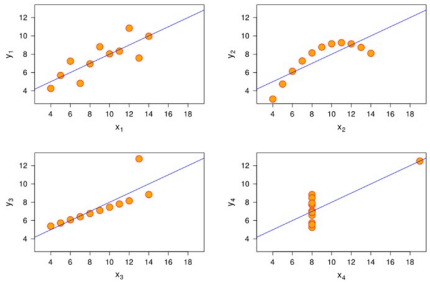
- **Explorar los datos**: hay relaciones que podemos malinterpretar si sólo miramos métricas resumen.
- **Expresar relaciones complejas**: no siempre las tablas nos van a permitir ver con claridad cuando hay mucha información involucrada.
- **Comunicar**: en general, construimos información para contársela a otras personas. Probablemente sea más fácil de contar una historia con un gráfico que con una tabla, por ejemplo.

Explorar los datos

Un gran ejemplo para mostrar lo importante de visualizar los datos es el llamado **Cuarteto de Anscombe**.

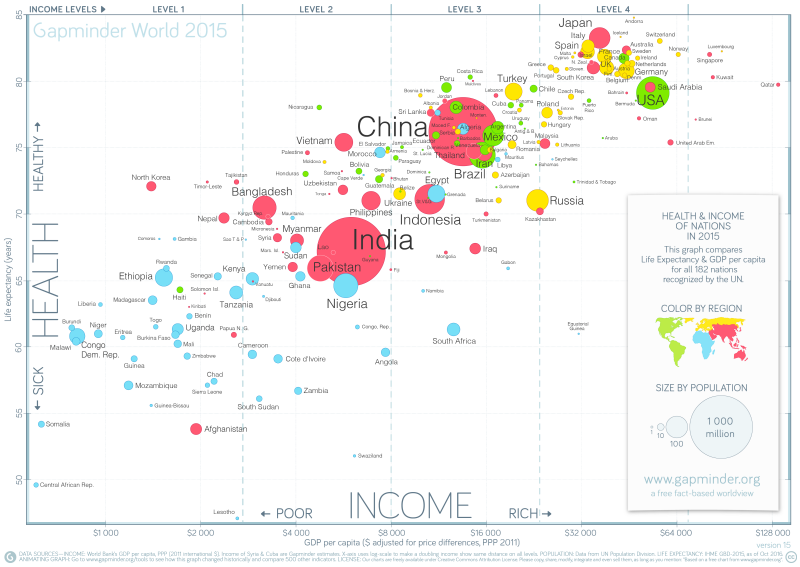
Cuarteto de Anscombe

Propiedad	Valor
Media de cada una de las variables x	9.0
Varianza de cada una de las variables x	11.0
Media de cada una de las variables y	7.5
Varianza de cada una de las variables y	4.12
Correlación entre cada una de las variables x e y	0.816
Recta de regresión	$y = 3 + 0.5x$



Expresar relaciones complejas

Hans Rosling fundó el proyecto Gapminder y popularizó la siguiente visualización. Aquí está disponible con la explicación del autor.



Comunicar

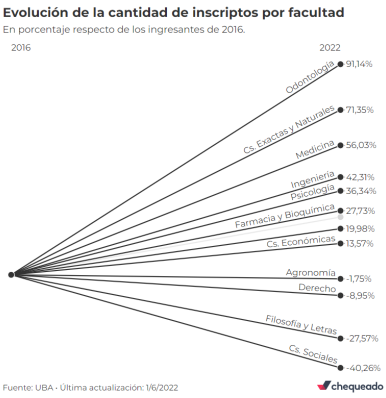
4.3. Construyendo un gráfico en `ggplot()`

Hagamos una tabla con los votos al PJ en las distintas elecciones en la Provincia de Buenos Aires.

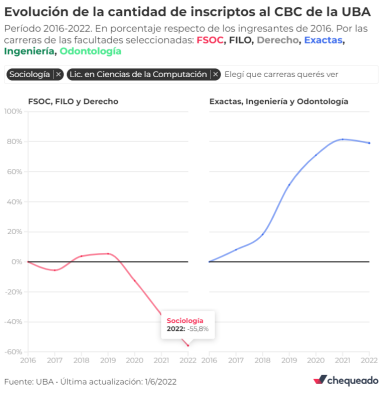
```
cols_pj <- c("FRENTE PARA LA VICTORIA","UNION POR LA PATRIA","FRENTE DE TODOS")
orden <- c("P2011", "G2011",
"P2015", "G2015","B2015",
"P2019", "G2019",
"P2023", "G2023","B2023")

tab_pj <- data %>%
  filter(Provincia=="BUENOS AIRES") %>%
  separate_wider_delim(Elecciones, " ", names = c("tipo_eleccion", "anio_eleccion"), cols_remov
  group_by(Elecciones, tipo_eleccion, anio_eleccion, Partido) %>%
  summarise(votos = sum(Votos),
            electores = sum(electores),
            votantes = sum(votantes)) %>%
  mutate(votos_per = votos / votantes,
         participacion = votantes / electores,
```

Dos ejemplos de visualizaciones que tienen muy en claro qué es lo que quieren comunicar Una disposición de la información que acompaña y refuerza el mensaje.



(a)



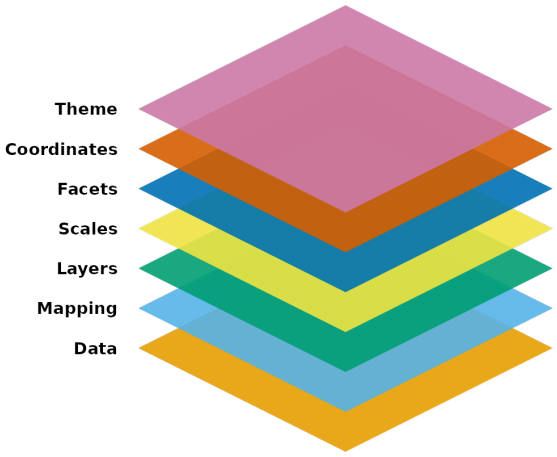
(b)

Figure 1: Fuente: Chequeado

```
        Elecciones = str_replace(Elecciones, "PASO ","P"),
        Elecciones = str_replace(Elecciones, "GENERALES ","G"),
        Elecciones = str_replace(Elecciones, "BALLOTAGE ","B"),
        Elecciones = factor(Elecciones, orden)
        ) %>%
  filter(Partido %in% cols_pj)
tab_pj
```

```
# A tibble: 10 x 9
# Groups:   Elecciones, tipo_eleccion, anio_eleccion [10]
   Elecciones tipo_eleccion anio_eleccion Partido       votos electores votantes
   <fct>      <chr>         <chr>         <chr>         <dbl>     <dbl>    <dbl>
 1 B2015      BALLOTAGE     2015          FRENTE PARA~ 4.83e6  11756541  9700855
 2 B2023      BALLOTAGE     2023          UNION POR L~ 4.92e6  13133726 10017387
 3 G2011      GENERALES     2011          FRENTE PARA~ 4.70e6  10574461  8715437
 4 G2015      GENERALES     2015          FRENTE PARA~ 3.42e6  12033279  9494724
 5 G2019      GENERALES     2019          FRENTE DE T~ 5.03e6  11995955  9882295
 6 G2023      GENERALES     2023          UNION POR L~ 4.22e6  13124435 10199399
 7 P2011      PASO          2011          FRENTE PARA~ 4.22e6  10818764  8540638
 8 P2015      PASO          2015          FRENTE PARA~ 3.24e6  11866173  8686139
 9 P2019      PASO          2019          FRENTE DE T~ 4.66e6  12348284  9279760
10 P2023      PASO          2023          UNION POR L~ 2.83e6  13115144  8902113
# i 2 more variables: votos_per <dbl>, participacion <dbl>
```

La librería estrella de la visualización en `Tidyverse` funciona a través de **capas**. Cada una se corresponde con funciones diferentes dentro de la visualización.

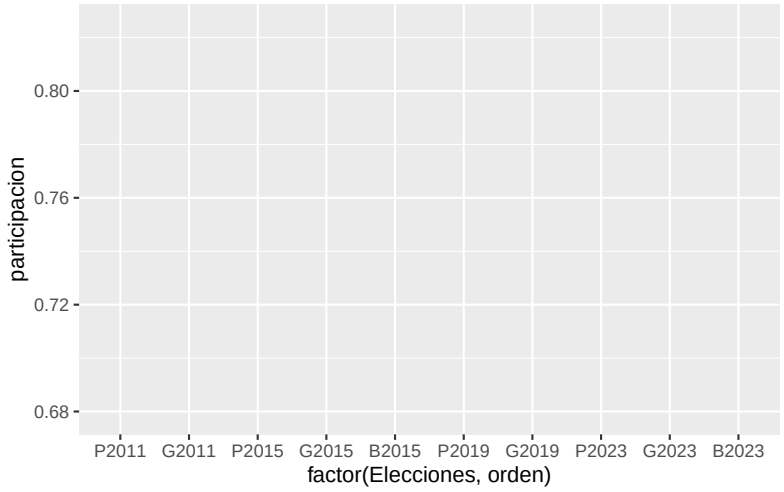


Con `ggplot()` simplemente vamos a establecer un lienzo vacío. En este caso, ya recibe la tabla con la información.

```
tab_pj %>%
  ggplot()
```

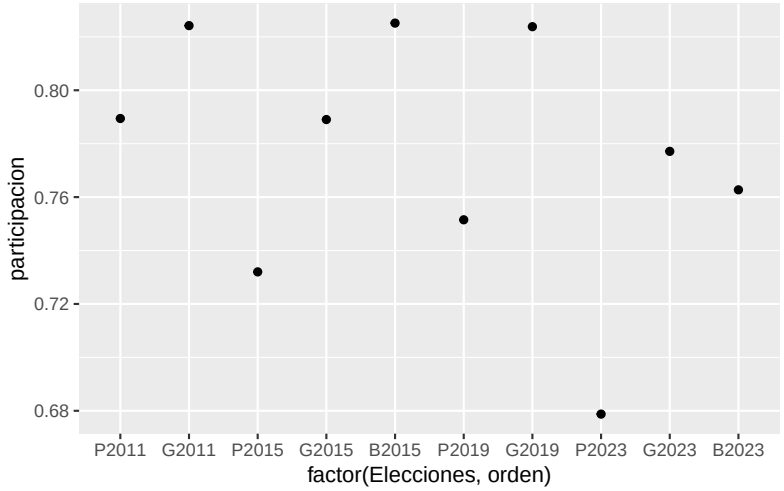

Luego definimos las **asignaciones estéticas**: la relación entre las variables y ciertos elementos de los gráficos (ejes/coordenadas o distintos atributos como color, tamaño, forma, etc.).

```
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=participacion))
```

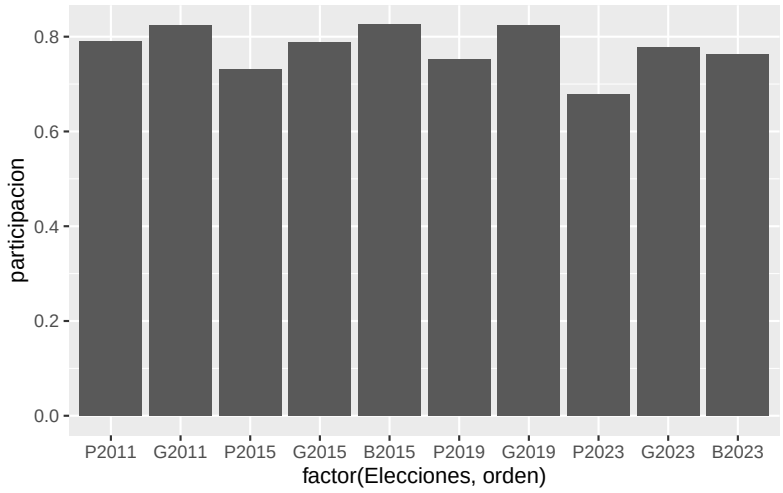


La siguiente definición es de los **elementos geométricos** con los que vamos a representar los datos definidos con anterioridad.

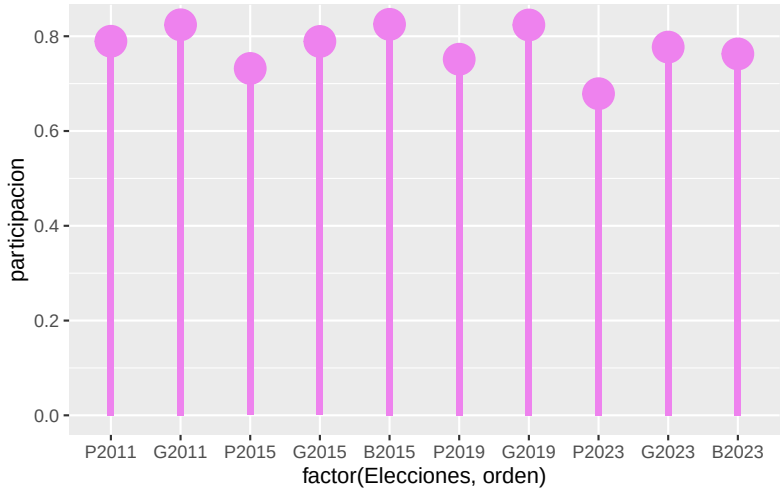
```
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=participacion))+
  geom_point()
```



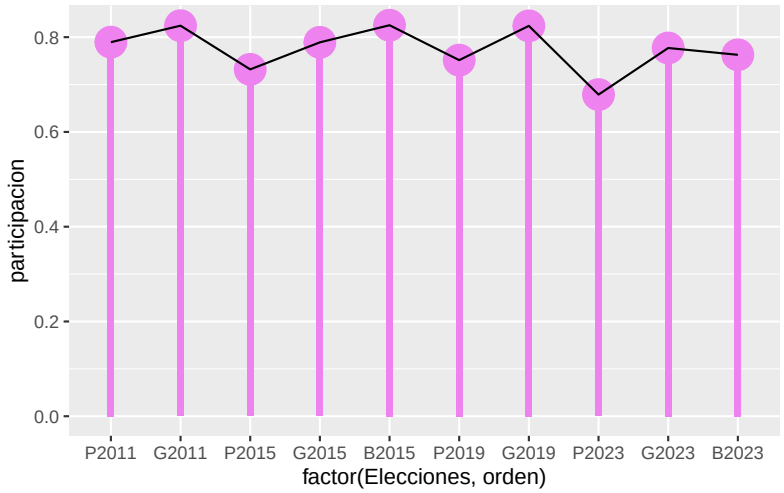
```r
# intentamos con otro tipo de gráfico
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=participacion))+
  geom_col()
```




```r
# intentamos con DOS gráficos
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=participacion))+
  geom_col(width=0.1, fill="violet")+
  geom_point(color="violet", size=7)
```

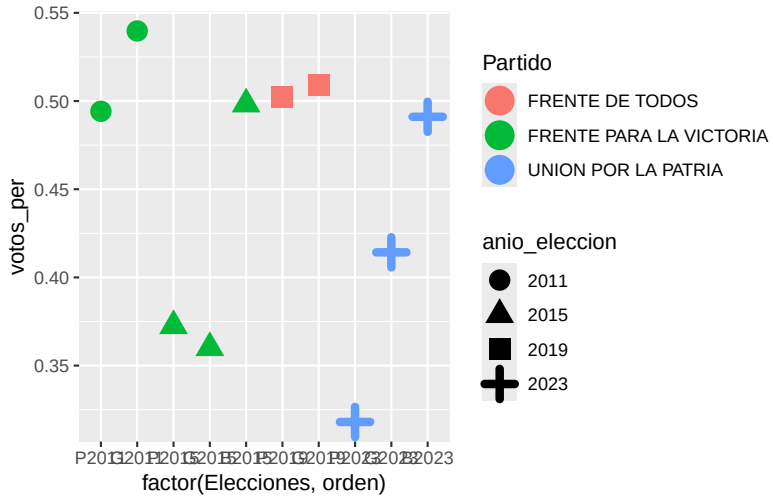


```r
# intentamos con TRES gráficos
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=participacion))+
  geom_col(width=0.1, fill="violet")+
  geom_point(color="violet", size=7)+
  geom_line(aes(group=1),color="black")
```

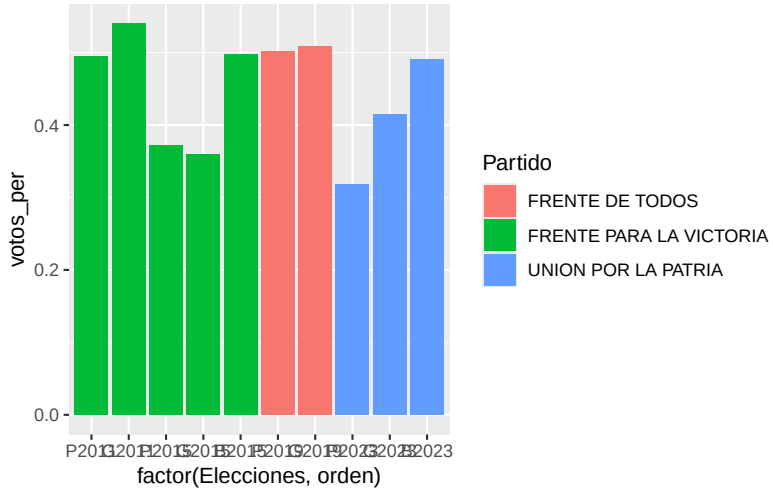


Se pueden agregar otros atributos. `Color` se utiliza para líneas y puntos; `fill` se utiliza para las áreas de los gráficos.

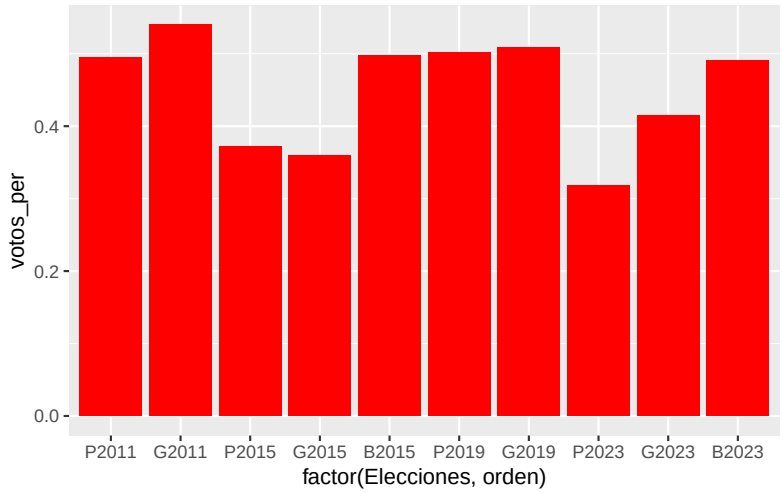
```r
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=votos_per, color=Partido, shape=anio_eleccion))+
  geom_point(size=3, stroke=3)
```




```
# intentamos con otro tipo de gráfico
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=votos_per, fill=Partido))+
  geom_col()
```

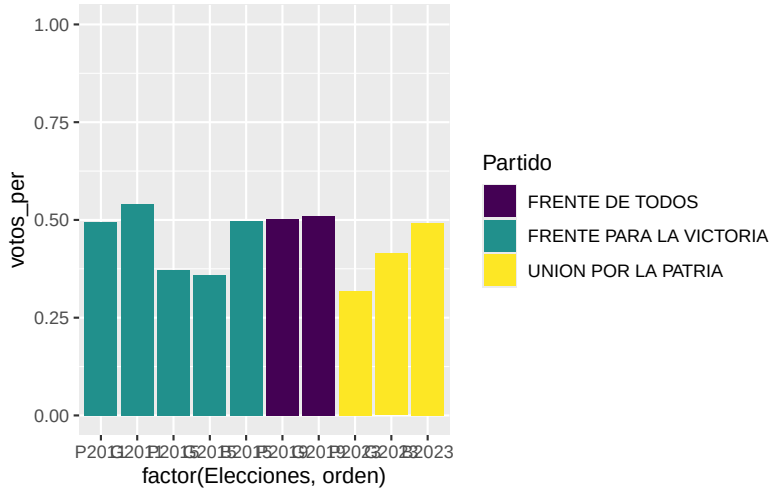



```
# si se define por fuera del aes() funciona como atributo general
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=votos_per))+
  geom_col(fill="red")
```



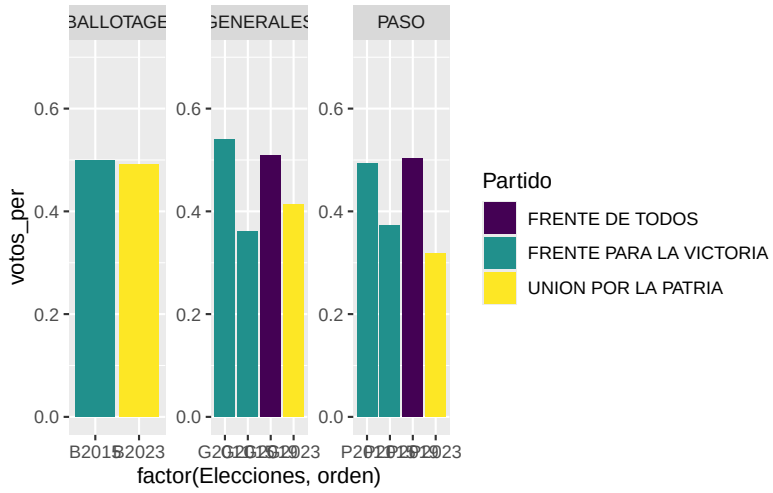
El siguiente elemento que podríamos incorporar tiene que ver con definir el tipo de paleta de colores a utilizar, los límites de los ejes u otras cuestiones que no afectan a lo esencial del gráfico pero ayudan a comunicar el punto.

```
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=votos_per, fill=Partido))+
  geom_col()+
  ylim(0,1)+ # ponemos límites entre 0 y 100%
  scale_fill_viridis_d() # viridis es una paleta de colores en particular
```



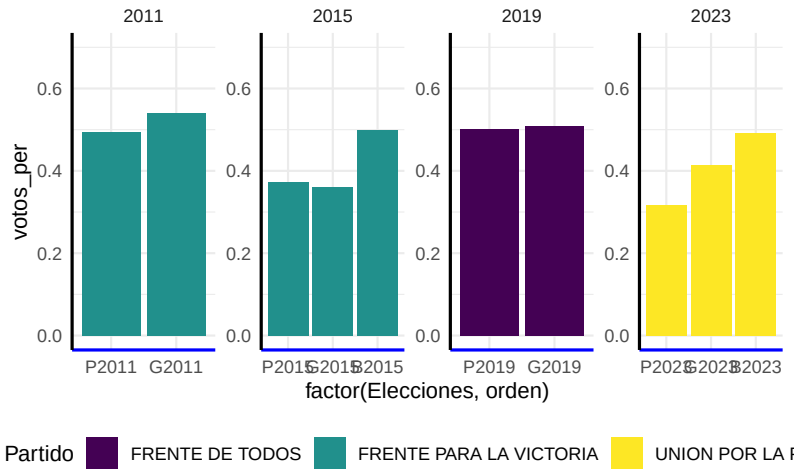
Cuando hablamos de **facetado** nos referimos a separar los gráficos en partes según alguna variable.

```
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=votos_per, fill=Partido))+
  geom_col()+
  ylim(0,0.7)+ # ponemos límites entre 0 y 100%
  scale_fill_viridis_d()+ # viridis es una paleta de colores en particular
  facet_wrap(~tipo_eleccion, scales = "free")
```

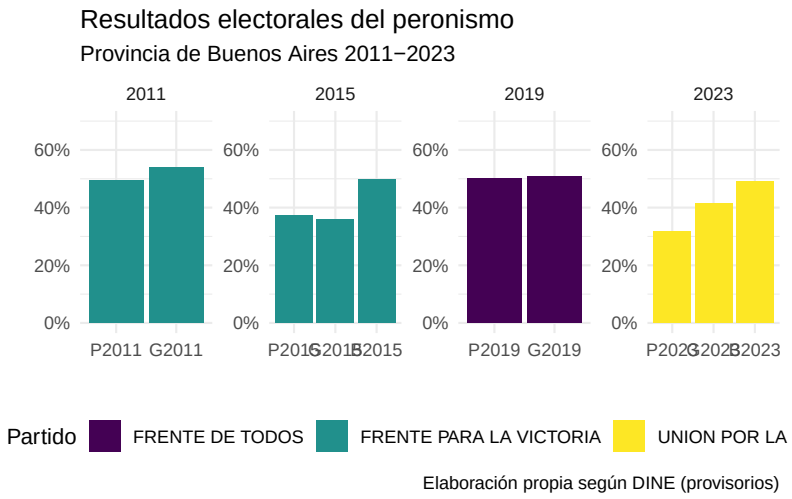


Lo último que nos interesa ver aquí son los temas, que comprenden los elementos visuales que no son controlados por la información propiamente dicha.

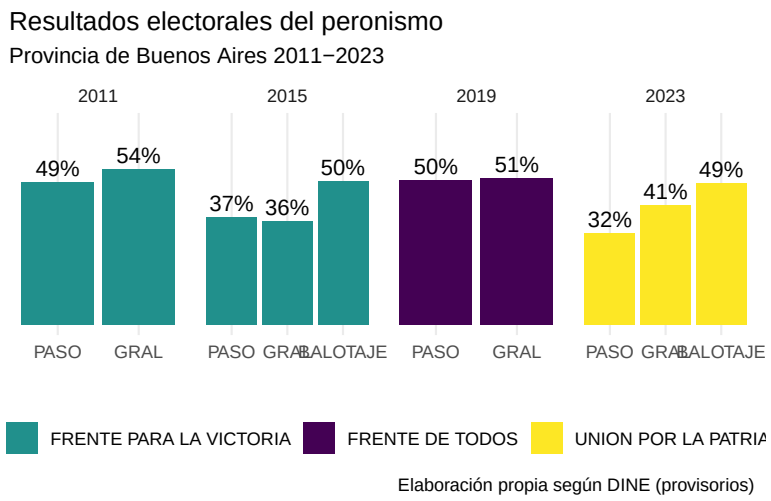
```
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=votos_per, fill=Partido))+
  geom_col()+
  ylim(0,0.7)+ # ponemos límites entre 0 y 100%
  scale_fill_viridis_d()+ # viridis es una paleta de colores en particular
  facet_wrap(~anio_eleccion, scales = "free", ncol=4)+
  theme_minimal()+ # agregamos definiciones generales
  theme(legend.position="bottom",
        axis.line = element_line(linewidth = 0.75),
        axis.line.x.bottom = element_line(colour = "blue")) # acá podemos definir elementos pa
```




```
# veamos una versión publicable
tab_pj %>%
  ggplot(aes(x=factor(Elecciones,orden), y=votos_per, fill=Partido))+
  geom_col()+
  scale_y_continuous(limits=c(0,0.7), labels = scales::percent_format(accuracy = 1))+
  scale_fill_viridis_d()+ # viridis es una paleta de colores en particular
  facet_wrap(~anio_eleccion, scales = "free", ncol=4)+
  theme_minimal()+ # agregamos definiciones generales
  theme(legend.position="bottom")+ # acá podemos definir elementos particulares
  labs(title="Resultados electorales del peronismo",
       subtitle="Provincia de Buenos Aires 2011-2023",
       x="", y="", caption = "Elaboración propia según DINE (provisorios)")
```




```
# cambiemos los límites por las etiquetas
tab_pj %>%
  mutate(tipo_eleccion = case_when(tipo_eleccion == "GENERALES" ~"GRAL",
                                   tipo_eleccion == "BALLOTAGE" ~"BALOTAJE",
                                   .default = as.character(tipo_eleccion))) %>%
  ggplot(aes(x=factor(tipo_eleccion,c("PASO","GRAL","BALOTAJE")), y=votos_per, fill=Partido))+
  geom_col()+
  geom_text(aes(label=paste0(round(votos_per*100),"%"), y=votos_per+0.05))+
  scale_y_continuous(limits=c(0,0.7), breaks=NULL)+
  scale_fill_viridis_d(breaks=c("FRENTE PARA LA VICTORIA", "FRENTE DE TODOS", "UNION POR LA PA
  facet_wrap(~anio_eleccion, scales = "free", ncol=4)+
  theme_minimal()+ # agregamos definiciones generales
  theme(legend.position="bottom")+ # acá podemos definir elementos particulares
  labs(title="Resultados electorales del peronismo",
       subtitle="Provincia de Buenos Aires 2011-2023",
       caption = "Elaboración propia según DINE (provisorios)",
       x="", y="", fill="")
```



4.4. El qué antecede al cómo

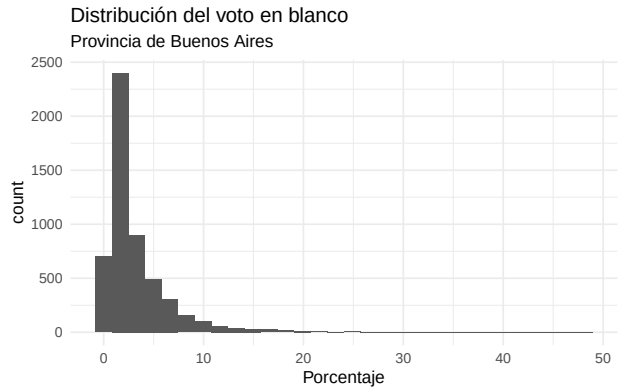
Elegir un gráfico no es una *mera* cuestión artística: hay distintos tipos de gráficos según *qué* se quiere mostrar. Para investigar un poco más, dejo dos recursos: una infografía sobre distintos tipos de gráficos y el proyecto Dataviz que funciona como una guía para elegir. Veamos algunos ejemplos.

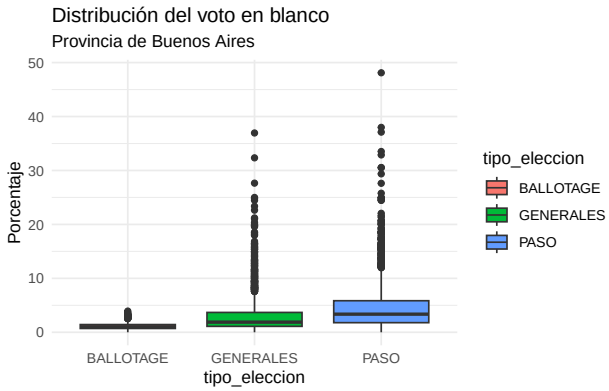
Distribuciones

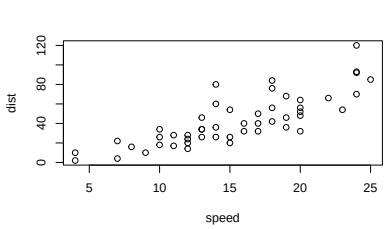
Si queremos ver una sola distribución (o dos) podemos usar un histograma o un gráfico de densidad. Para comparar entre varias, la mejor opción es el gráfico de cajas.

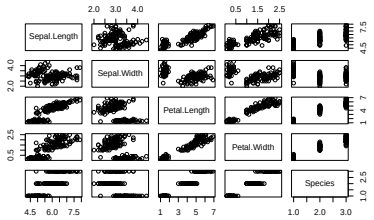
Distribuciones

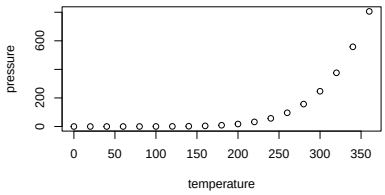
```
plot(cars)
plot(iris)
plot(pressure)
```











4.5. Yapa: Mapas

4.6. Para seguir practicando