An introduction to R

Data visualisation

Samy Zitouni

October 2024

Program for today

- We will learn how to merge datasets, to add new information
- We will produce a set of graphs with `ggplot2`
- We will try to interpret the graphs

Merging two or more datasets

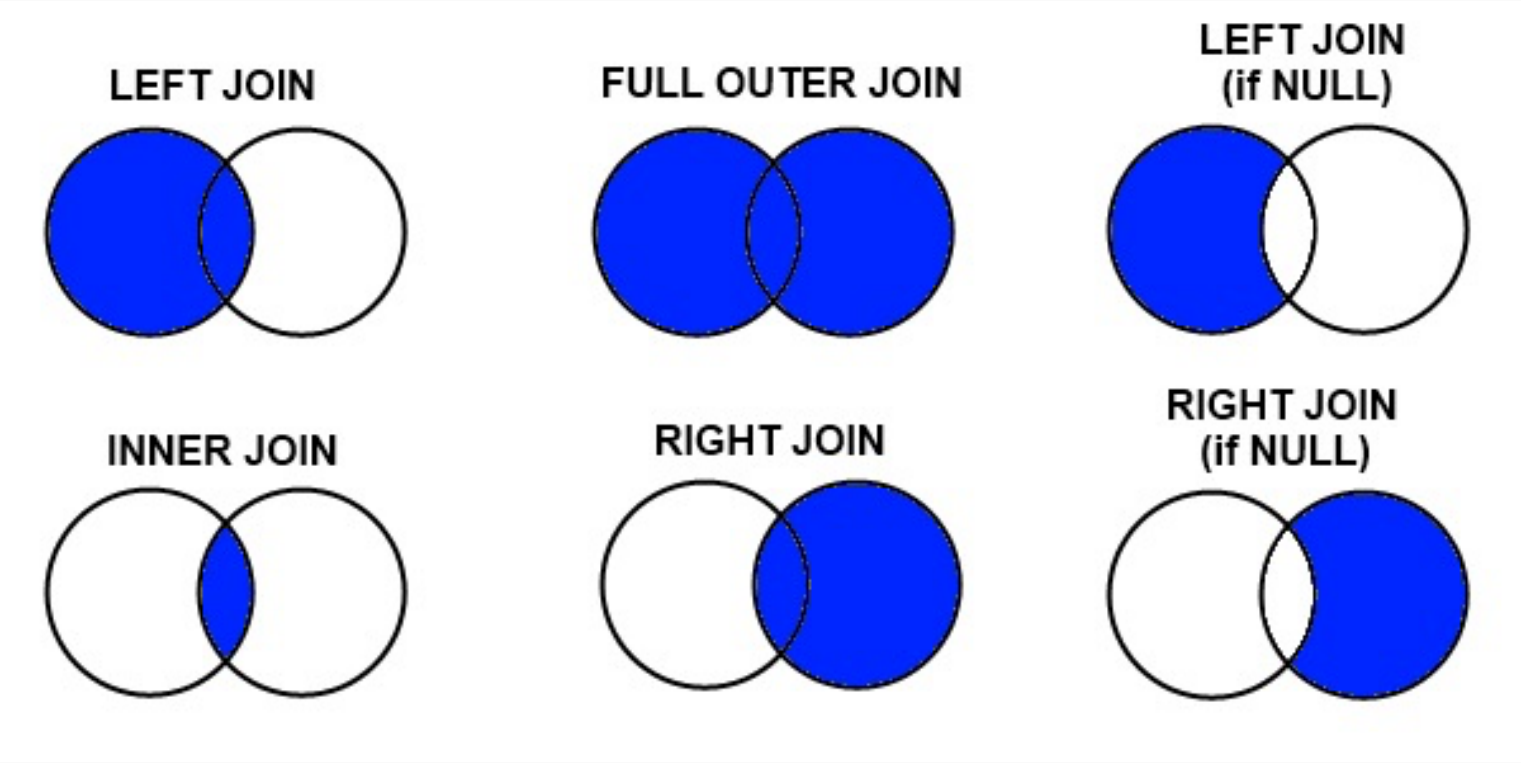
About merging

- Sometimes, a **single** dataframe does not contain all the needed information
- You may need to find it elsewhere

Some basics

- To match, you need to find a common identifier in the two databases
- Then decide what you want at the end to decide whether you use `left_join()`, `right_join()`, `full_join()` or `inner_join()`
- Common identifier : and id, a date, a region, a postcode, sometimes several columns: depends on the context !!

About merging



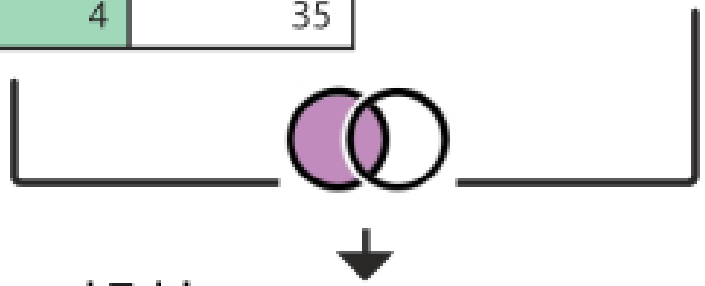
About merging

Left Table

Date	CountryID	Units
1/1/2024	1	40
1/2/2024	1	25
1/3/2024	3	30
1/4/2024	4	35

Right Table

ID	Country
1	USA
2	Canada
3	Panama



Merged Table

Date	CountryID	Units	Country
1/1/2024	1	40	USA
1/2/2024	1	25	USA
1/3/2024	3	30	Panama
1/4/2024	4	35	*null*

About merging

Date	Customer ID	Order total
Monday	1	10 €
Monday	2	23 €
Monday	3	10,89 €
Monday	4	45 €
Monday	5	3 €
Thursday	1	4 €
Thursday	2	67 €
Friday	1	56 €
Friday	2	90 €
Friday	3	8,99 €

Date	Customer ID	Name
Monday	1	Samy
Monday	2	Edgar
Monday	3	Khadija
Monday	4	Tania
Monday	5	Franck
Thursday	1	Julia
Thursday	2	Mathilde

Date	Customer ID	Order total	Name
Monday	1	10 €	Samy
Monday	2	23 €	Edgar
Monday	3	10,89 €	Khadija
Monday	4	45 €	Tania
Monday	5	3 €	Franck
Thursday	1	4 €	Julia
Thursday	2	67 €	Mathilde
Friday	1	56 €	NaN
Friday	2	90 €	NaN
Friday	3	8,99 €	NaN

Let's practice

For now, we are going to work with the World Inequality Database

- Download the files `{wid.csv}` and `{gdp.csv}` I sent you and place it in your input folder
- Check columns and rename if necessary

Example

```
# Imports
wid <- read.csv(paste0(path_input, 'wid.csv'))
gdp <- read.csv(paste0(path_input, 'gdp.csv'))
View(wid)
View(gdp)
colnames(gdp) #Bad column naming

colnames(gdp) <- c(
  'year',
  'yearCode',
  'country',
  'countryCode',
  'gdp')

# If you want to change only a few:

gdp <- gdp %>%
    rename(c('year' = 'Time'))
```

Let's pratice

Let's try to merge the datasets straightforward. Look at countrys names to be sure they are the same synthax:

Try to merge

```
# Lentgh of identifiers (unique)
length(unique(wid$country))
length(unique(gdp$country))
length(intersect(gdp$country, wid$country))

# Try to left_join
jwid <- wid %>% left_join(
    gdp %>% select(country, year, gdp),
    by = c('country', 'year')
    )
```

Does this work ?

Let's practice

On previous slide, there is a type problem: on one dataframe, the column `year` is an `character`, on the other it is a `character`

To solve it

```
jwid <- wid %>%
    mutate(year = as.character(year)) %>% # Modify the type of variable year
    left_join(gdp %>% select(c(country, year, gdp)), by = c('country', 'year'))
```

Look at NAs distribution

Filter

	country	continent	year	fshare	top1	inc_head	gdp
1	Algeria	Africa	2010	0.0992	0.1003	12610.627	*NA*
2	Algeria	Africa	2011	0.1120	0.0991	12619.984	*NA*
3	Algeria	Africa	2012	0.1201	0.0991	12634.026	*NA*
4	Algeria	Africa	2013	0.1206	0.0991	12531.988	*NA*
5	Algeria	Africa	2014	0.1160	0.0991	12546.430	238942664192.59
6	Algeria	Africa	2015	0.1221	0.0991	12533.389	187493855609.345
7	Algeria	Africa	2016	0.1232	0.0991	12955.519	180763839522.151
8	Algeria	Africa	2017	0.1282	0.0991	12740.548	189880896903.073
9	Algeria	Africa	2018	0.1265	0.0991	12590.207	194554483655.528
10	Algeria	Africa	2019	0.1248	0.0991	12533.531	193459662090.677
11	Angola	Africa	2010	0.2664	0.1744	11180.762	*NA*
12	Angola	Africa	2011	0.2707	0.1851	11251.621	*NA*
13	Angola	Africa	2012	0.2682	0.1958	11898.082	*NA*
14	Angola	Africa	2013	0.2631	0.2065	12194.377	*NA*
15	Angola	Africa	2014	0.2614	0.2172	12523.929	135966802586.713
16	Angola	Africa	2015	0.2610	0.2278	12344.216	90496420506.5957
17	Angola	Africa	2016	0.2658	0.2385	11565.139	52761617225.9253
18	Angola	Africa	2017	0.2677	0.2491	11044.407	73690154990.7312
19	Angola	Africa	2018	0.2717	0.2598	10306.950	79450688259.3664

Showing 1 to 20 of 1,610 entries, 7 total columns

On joining

- Usually, `left_join()` keeps the structure of your data, it only add the columns you want to add with observations where the identifier was found, and `NAs` for observations that were not found with the identifier
- It is very used to add data to what you already have
- For micro data, identifiers can be hard to find, or require a lot of uniformization work for th two databases.
- For macro data, it is usually easier !

Data visualisation basics : ggplot2

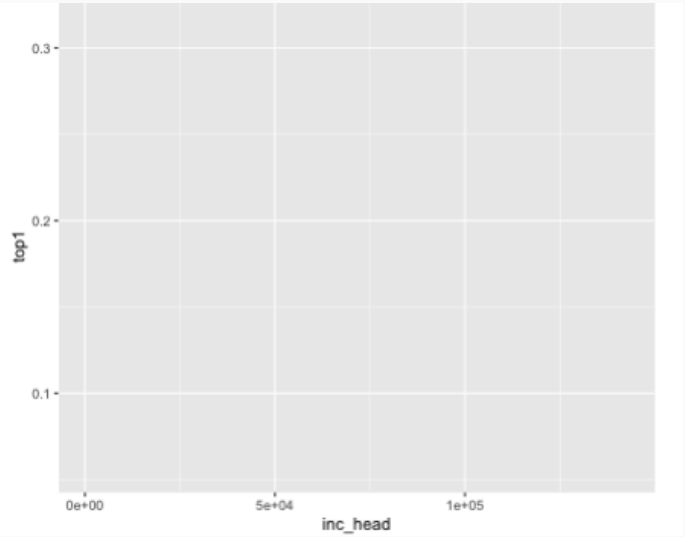
- The function `ggplot()` creates a canvas to draw on

```
library(ggplot2)

ggplot() # Creates a base canva
```

Data visualisation basics : ggplot2

- The function `ggplot()` creates a canvas to draw on
- `data =` specifies the data to work on
- `aes(x =, y =, ...)` gives the placement of variables of `data` on different axis

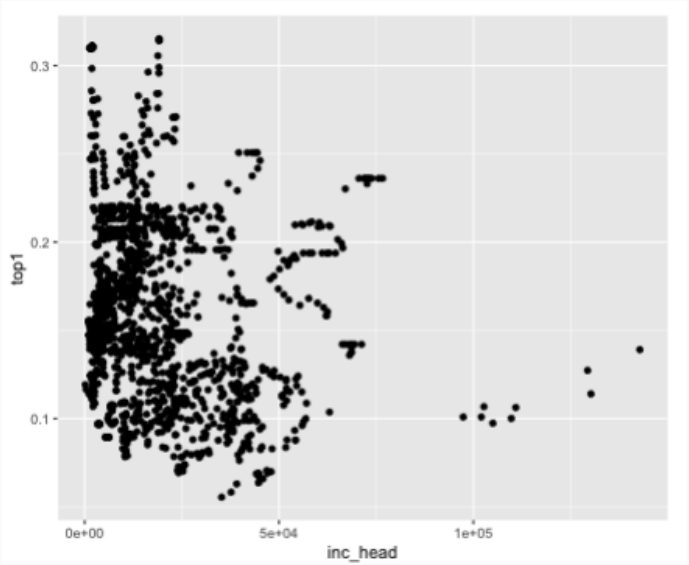
```
library(ggplot2)

ggplot(data = wid,
aes(x = inc_head
    y = top1)) # Specify data and axis
```



Data visualisation basics : ggplot2

- The function `ggplot()` creates a canvas to draw on
- `data =` specifies the data to work on
- `aes(x =, y =, ...)` gives the placement of variables of `data` on different axis
- `geom_point()` tells us wich type of graphics wewant to apply to the aesthetics (here, scatter plot)

```
library(ggplot2)

ggplot(data = wid,
aes(x = inc_head
    y = top1)
) + # Specify data and axis
geom_point()
```



ggplot2 recap

The structure is always the same

1. Data with `data =` in `ggplot()`
2. aesthetics with `aes(x =, y =, ...)` in `ggplot()`
3. Geometry with `+ geom_point(), +geom_lines()` and so on, that you add with `+`
4. Style (lines, grid, dots, colors, scales, legends, etc.) that we will see through this class

Integration to dplyr

We can use the pipe operator %>% to pass a dataframe in a ggplot()

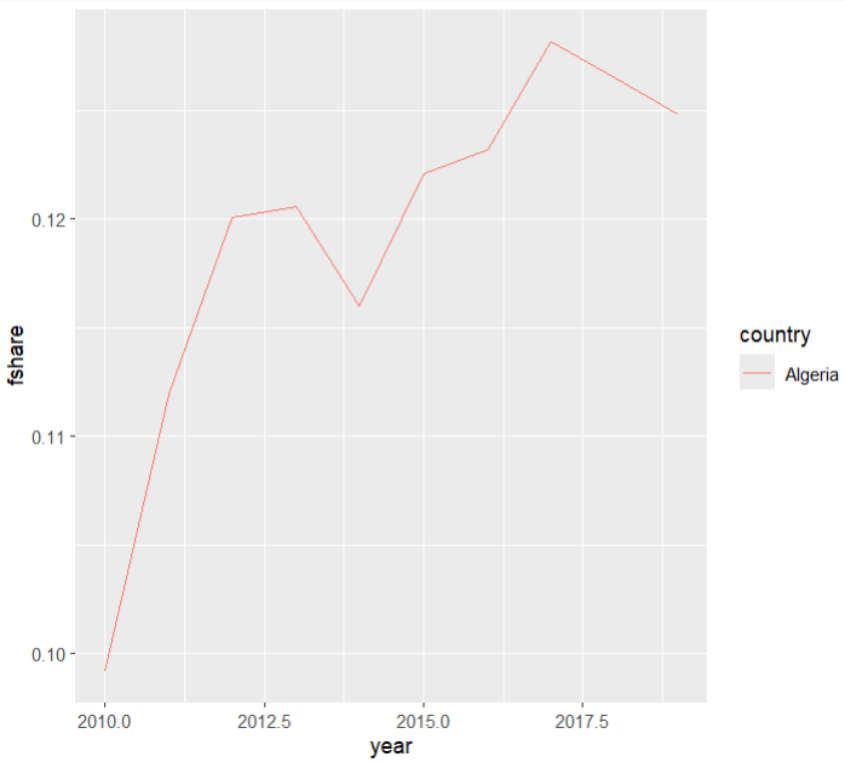
Example

```
wid %>%
    ggplot(aes(x = inc_head, y = top1)) + geom_point()
```

Integration to dplyr

Another example

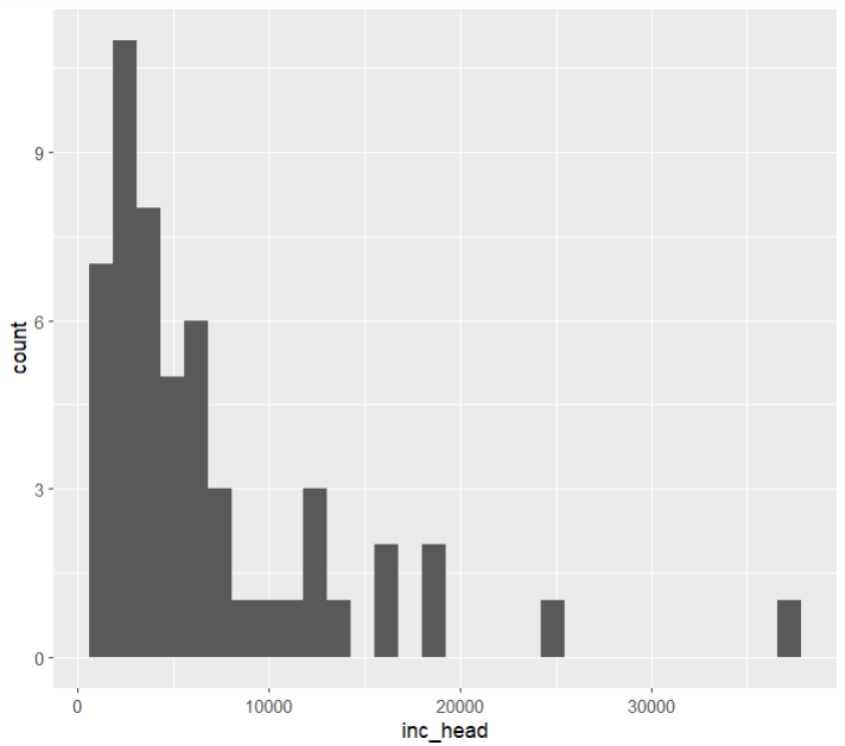
```
wid %>%
    filter(country == 'Algeria') %>%
    ggplot(aes(x = year, y = fshare)) +
    geom_line()
```



Integration to dplyr

Another one

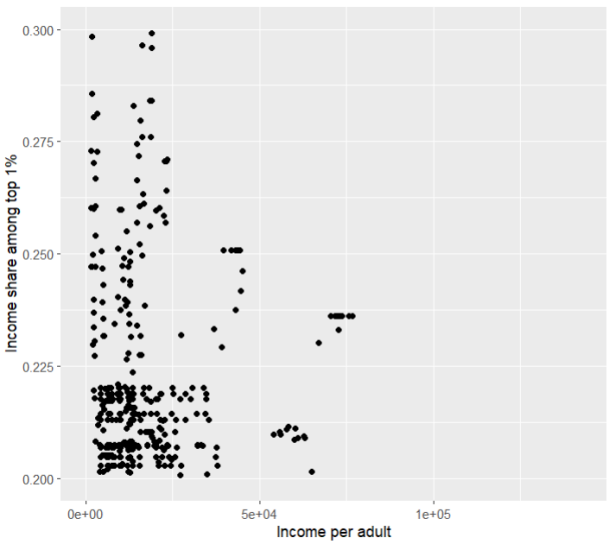
```
wid %>%
    filter(continent == 'Africa', year == '2013') %>%
    ggplot(aes(x = inc_head)) +
    geom_histogram()
```



More about aesthetics

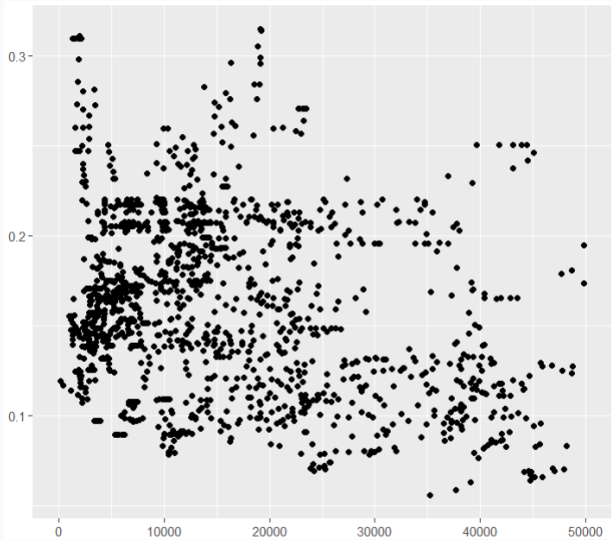
- You can add with + some basic modifications

```
wid %>%
  ggplot(aes(inc_head, top1)) +
  geom_point() +
  xlab("Income per adult") +
  ylab("Income share among top 1%") +
  ylim(0.2, 0.3)
```



- Changing scales changes how you see the data

```
wid %>%
  ggplot(aes(x = inc_head, y = top1)) +
  geom_point() +
  xlab(NULL) +
  ylab(NULL) +
  xlim(0, 50000)
```



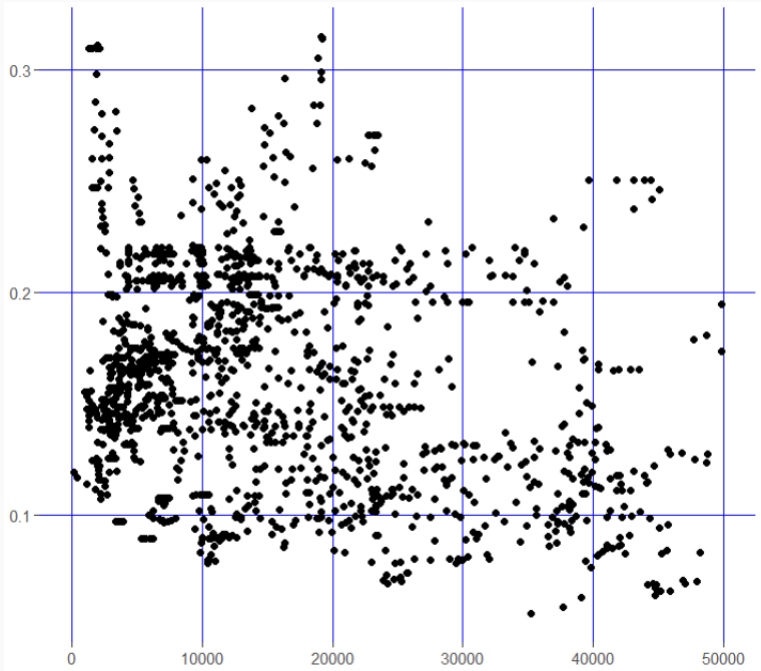
More about aesthetics

Example : personnal settings

```
wid %>%
  ggplot(aes(x = inc_head, y = top1)) +
  geom_point() +
  xlab(NULL) +
  ylab(NULL) +
  xlim(0, 50000) +
  theme(
    panel.grid.major = element_line(colour = 'blue'),
    panel.background = element_blank()
  )
```

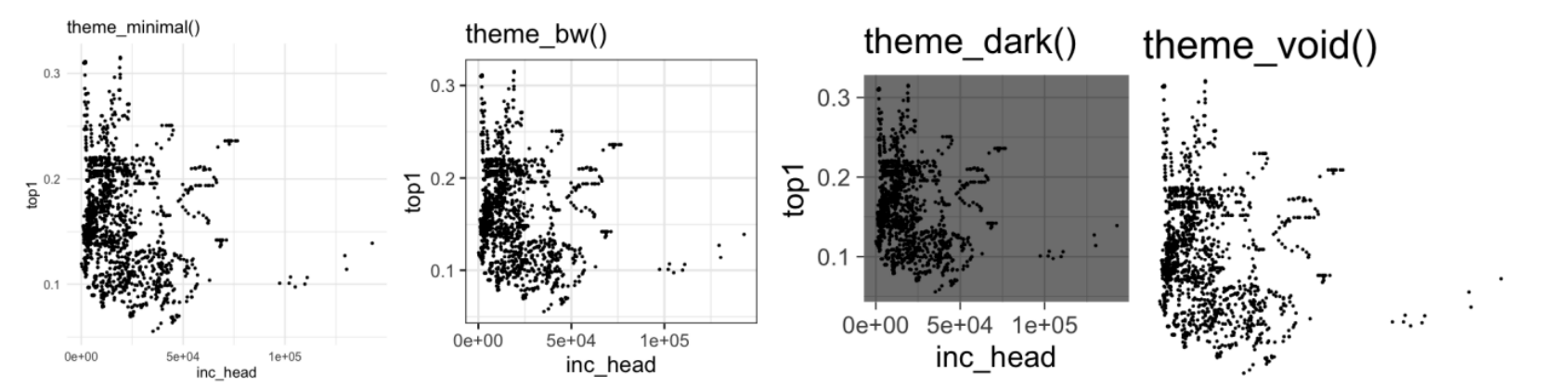
Theme

- You can change the theme of your plot as you want
- There are pre saved themes you can use, or ou can also **set**



About aesthetics

Pre existsing themes for quick rendering



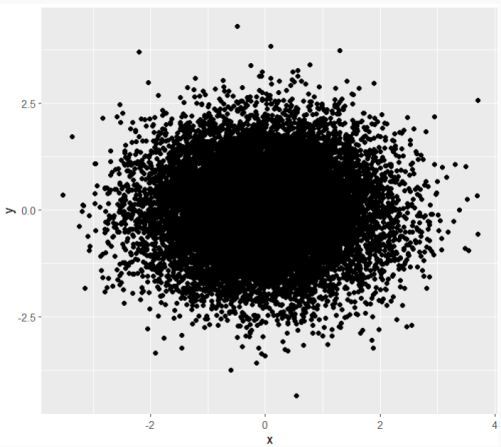
I won't cover it in these slides, but

- You can set your own theme personnalizing every detail as you want it to be
- Although it i available, it is **not super useful**. Always opt for something neat and clear for the audience !
- Some theme recommandations : `this link`

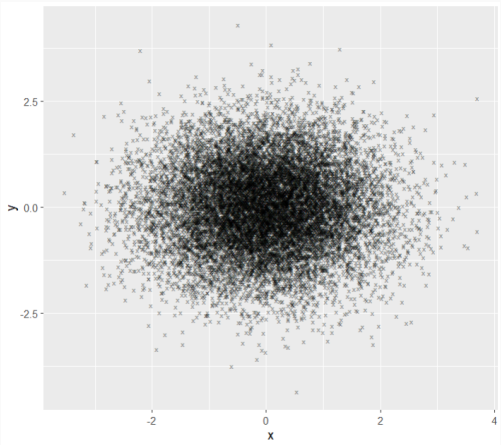
Use `geom_options` to be clearer

Use options in geometry to increase clarity

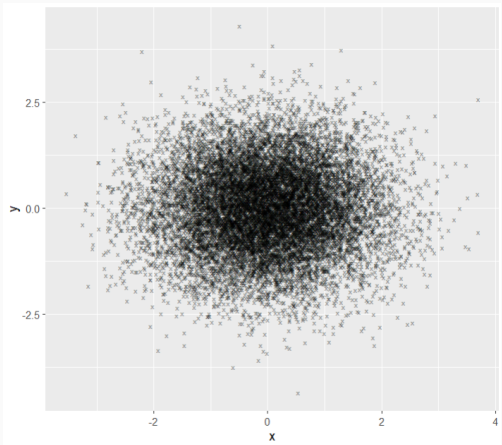
```
plot_base +
  geom_point()
```




```
plot_base +
  geom_point(size = .2)
```




```
plot_base +
  geom_point(shape = 'x',
    ↪    alpha = .5)
```

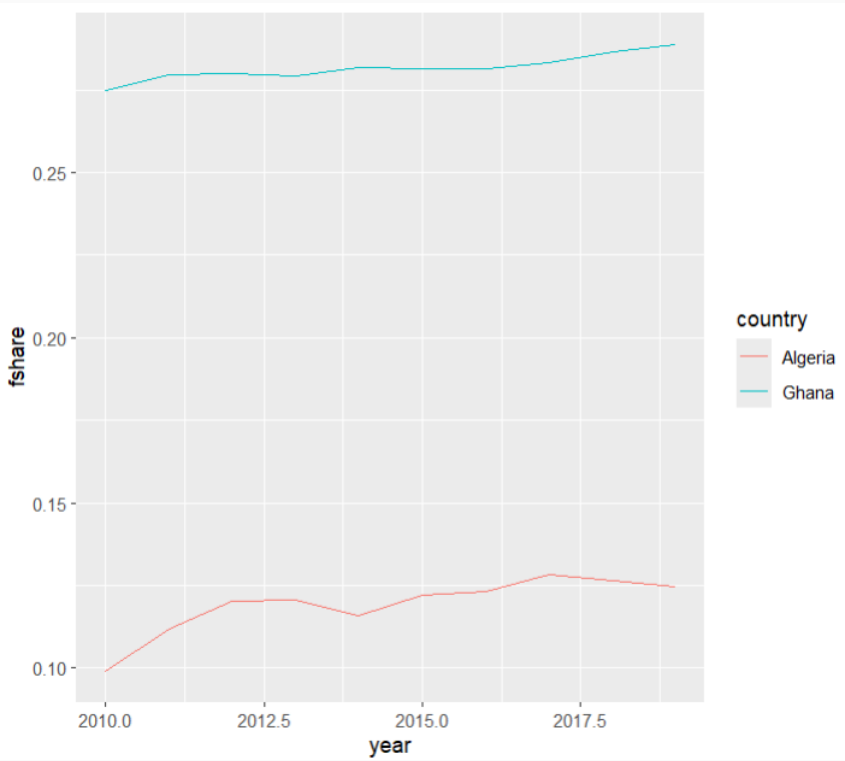


Additionnal dimensions with `color, shape, size`

We can add `color =` to our aesthetics to plot by country

Example

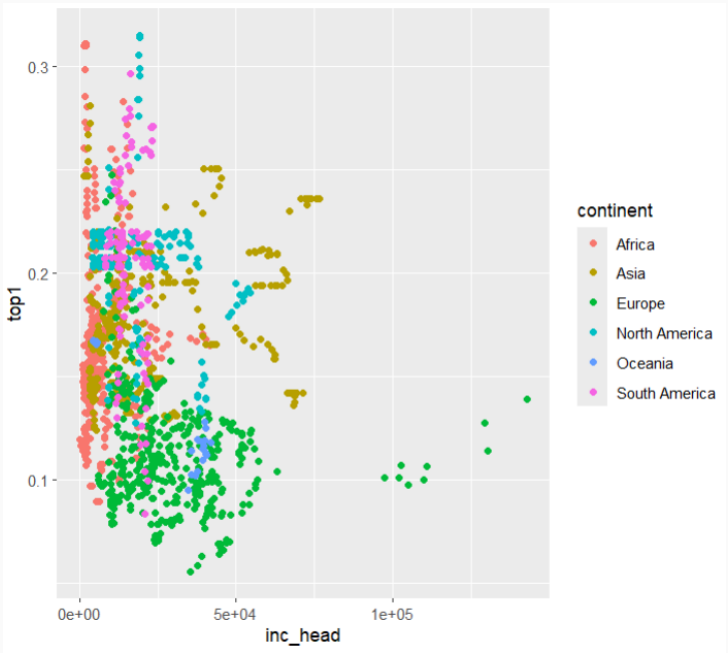
```
wid %>%
    filter(country %in% c('Algeria', 'Ghana')) %>%
    ggplot(aes(x = year, y = fshare, color = country)) +
    geom_line()
```



Beware

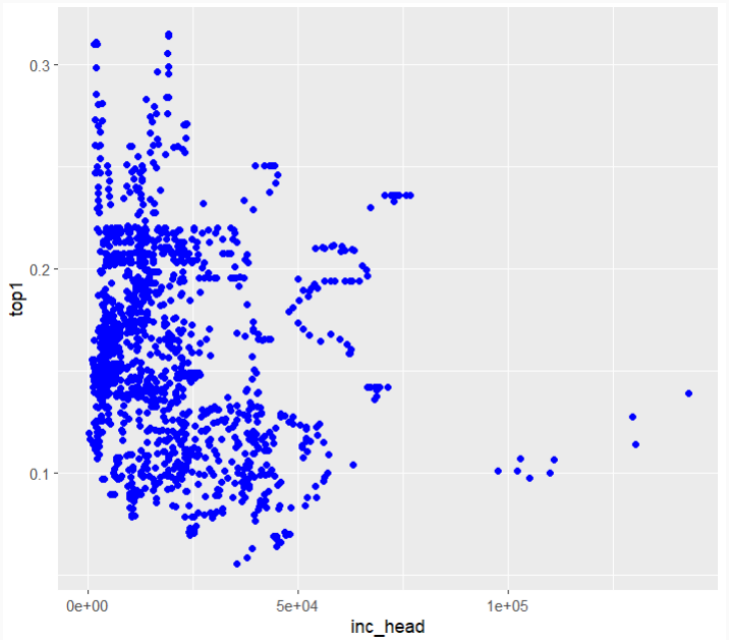
Difference between this..

```
wid %>%
  ggplot(aes(x = inc_head, y = top1,
    ↪    color = continent)) +
  geom_point()
```



... and this

```
wid %>%
  ggplot(aes(x = inc_head, y = top1)) +
  geom_point(color = 'blue')
```

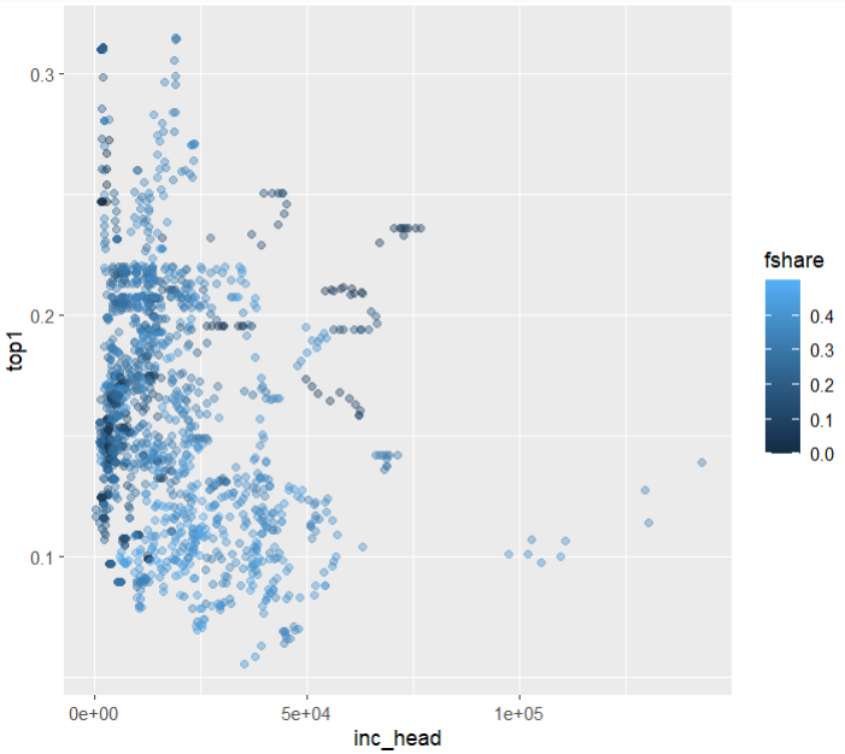


You can mix as you want

Color can also represent and continuous value, therefore showing different intensities

Example

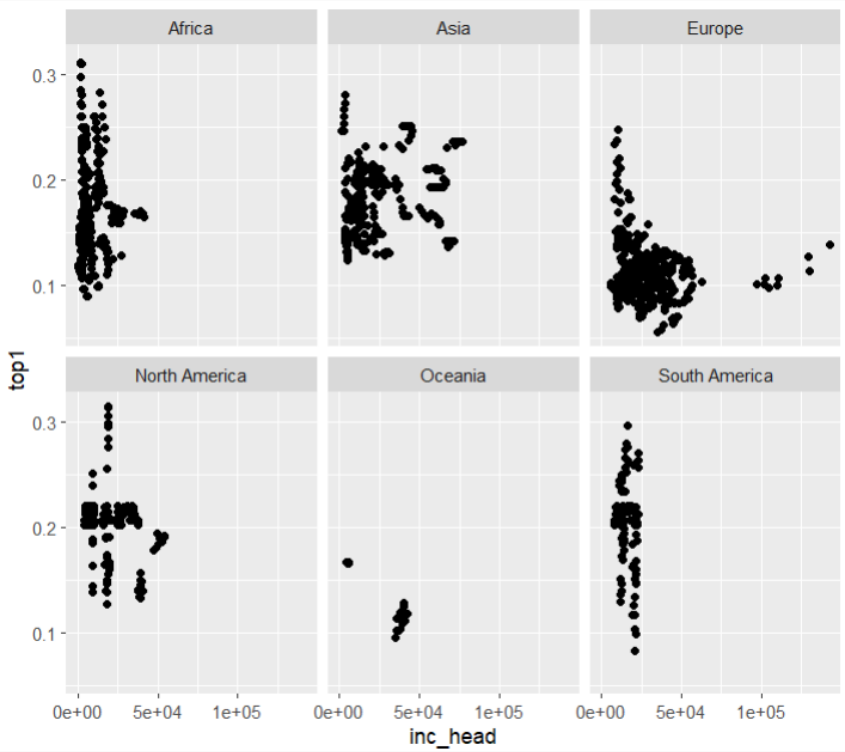
```
wid %>%
 ggplot(aes(x = inc_head, y = top1, color = fshare)) +
 geom_point(alpha = .4)
```



Facetting

You may need to separate plots

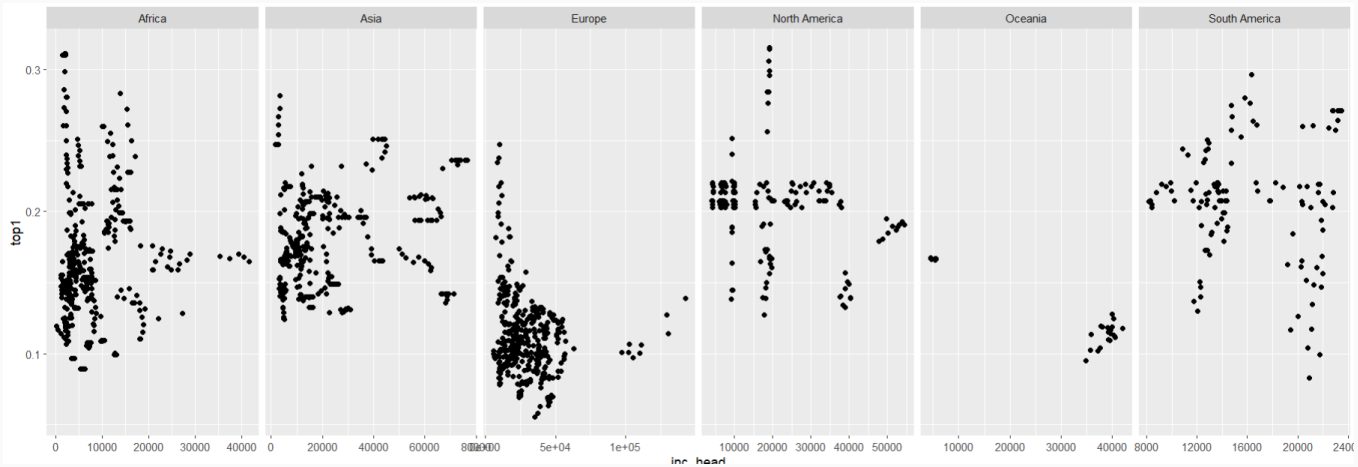
```
wid %>%
  ggplot(aes(x = inc_head, y = top1)) +
  geom_point() +
  facet_wrap(~continent)
```



Facetting

Adapt scale for each plot

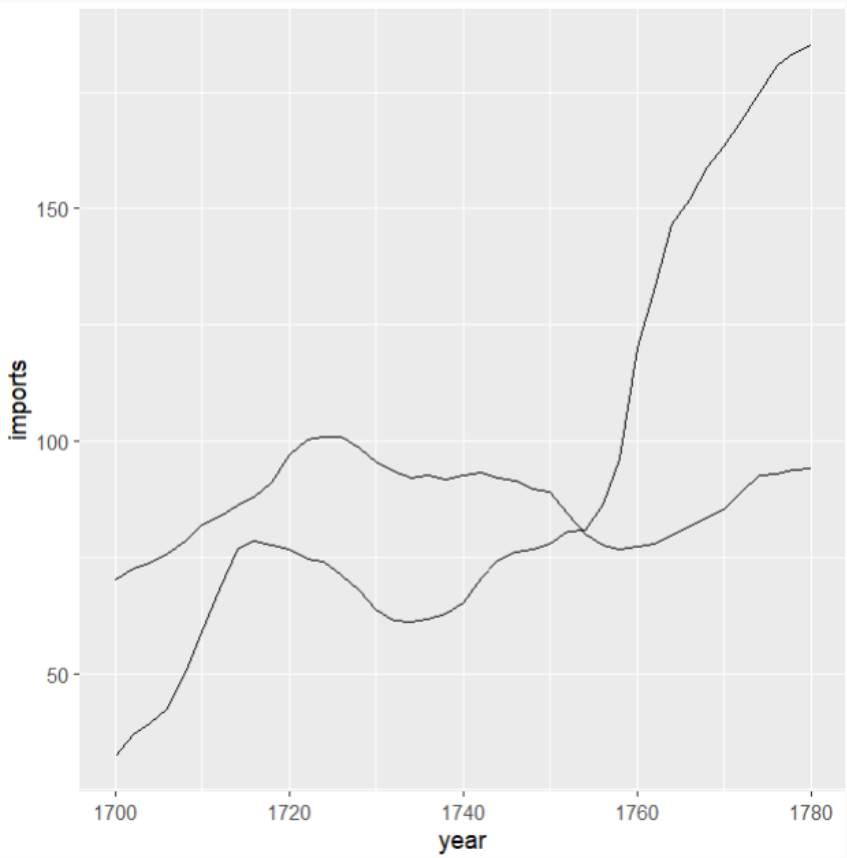
```
wid %>%
  ggplot(aes(x = inc_head, y = top1)) +
  geom_point() +
  facet_wrap(
    ~continent,
    ncol = 6,
    scale = 'free_x'
  )
```



Layers

Add layers

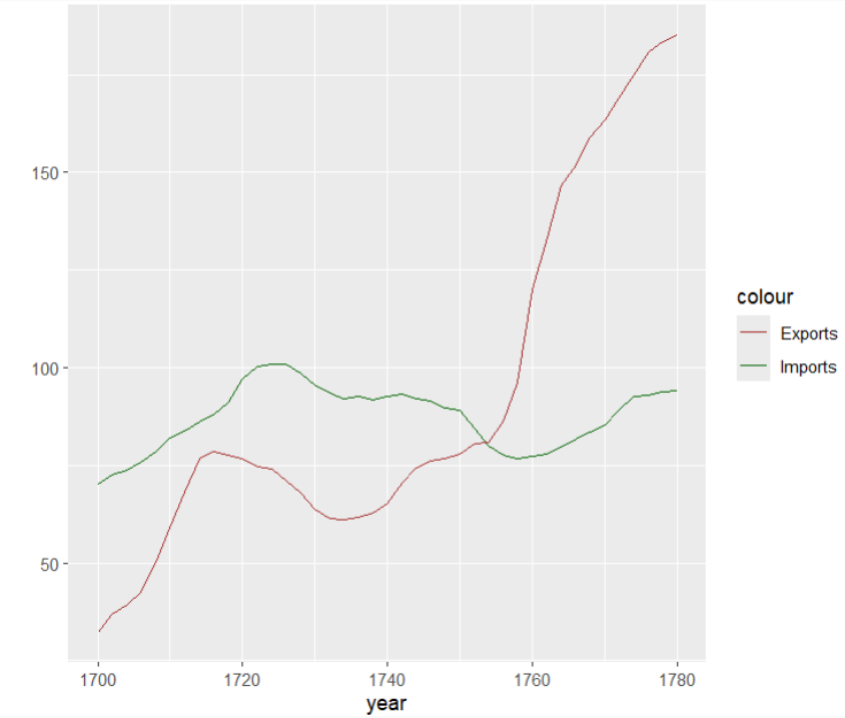
```
bl <- read.csv(paste0(path_input, '02_playfair-balance.csv'))
bl %>%
  ggplot() +
  geom_line(aes(x = year, y = imports)) +
  geom_line(aes(x =  year, y = imports))
```



Layers

Add layers

```
bl %>%
  ggplot() +
  geom_line(aes(x = year, y = imports, color = 'Imports')) +
  geom_line(aes(x =  year, y = imports, color = 'Exports')) +
  scale_color_manual(values = c('brown', 'darkgreen')) +
  xlab(NULL)
```

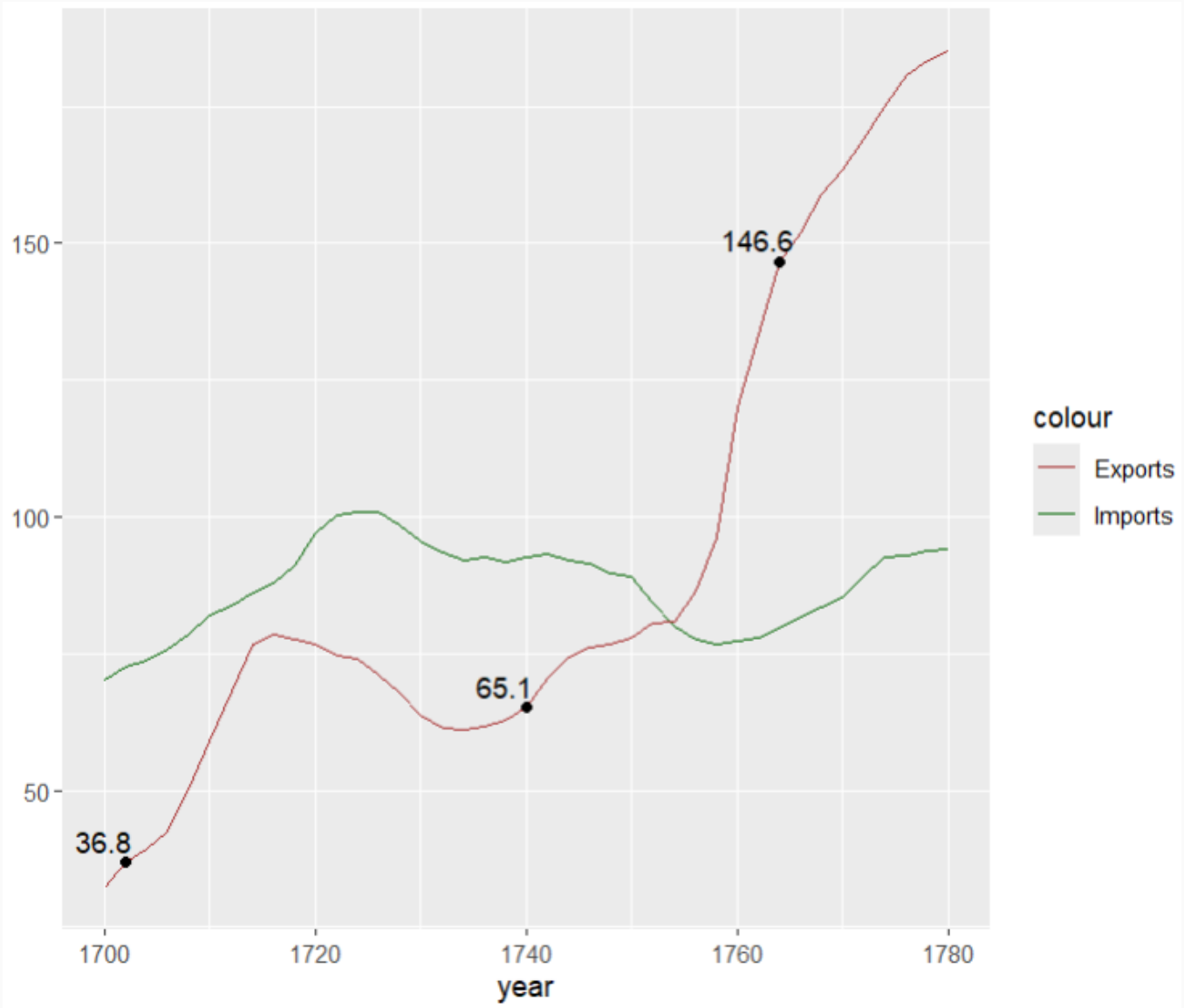


More layers with different datasets

Add layers

```
bl_plot <- bl %>% ggplot() +
  geom_line(aes(x = year, y = imports, color = 'Imports')) +
  geom_line(aes(x =  year, y = imports, color = 'Exports')) +
  scale_color_manual(values = c('brown', 'darkgreen')) +
  xlab(NULL)

# Text I want to print for some years only
some_years <- bl %>% filter(year %in% c(1702, 1740, 1764))

# Ad them to the plot
bl_plot <- bl_plot +
  geom_point(data = some_years) +
  geom_text(
    aes(x = year, y = exports),
    data = some_years,
    nudge_y = 4,
    nudge_x = -2
    )
```

More layers with different datasets



What you want to think about

1. Always think about what you want to show.
2. Be honest, don't lie with scales, don't hide data, because is changes what you want to show
3. Also think about the structure of your data, is it categorical ? Is it continuous ?

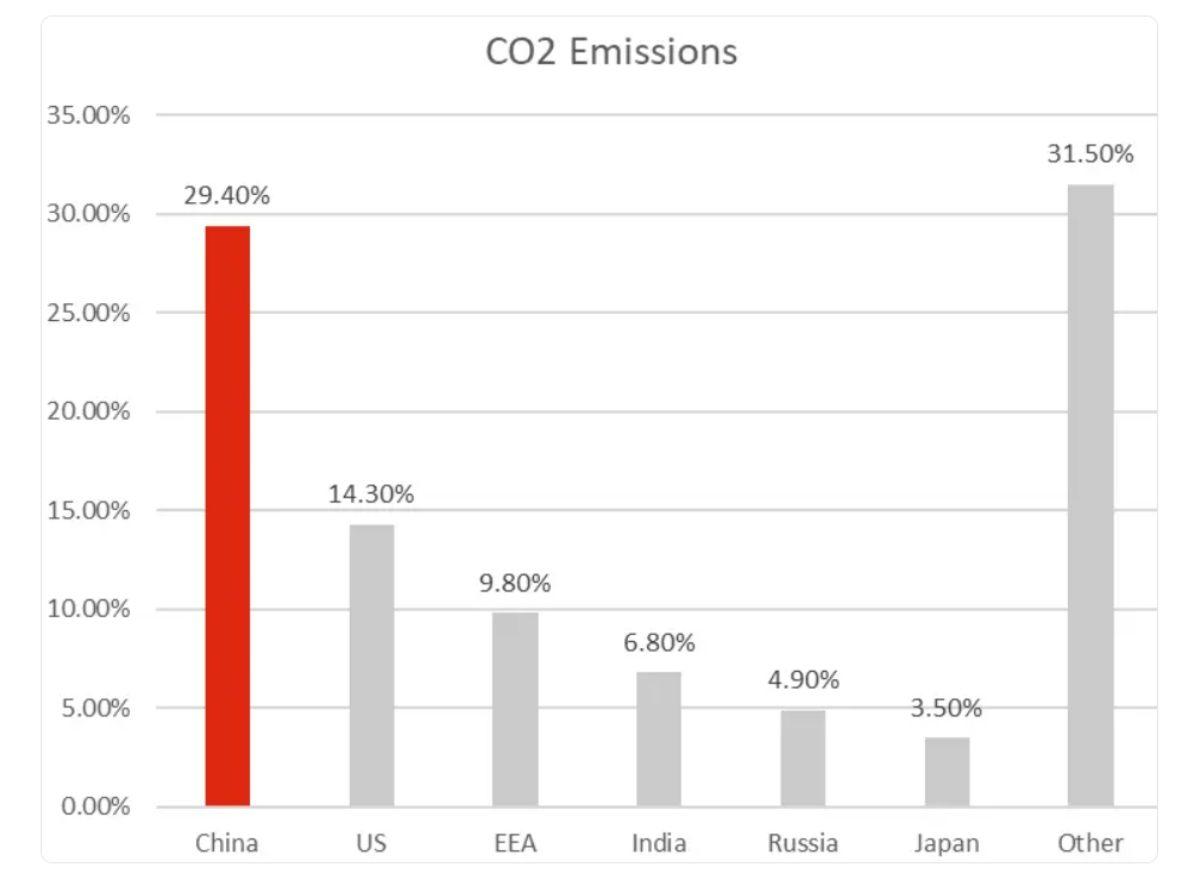
Some links

- To help you in the iterative decision process of choosing a graph : `this link`
- `Best graphs to think about`
- `Another great link`

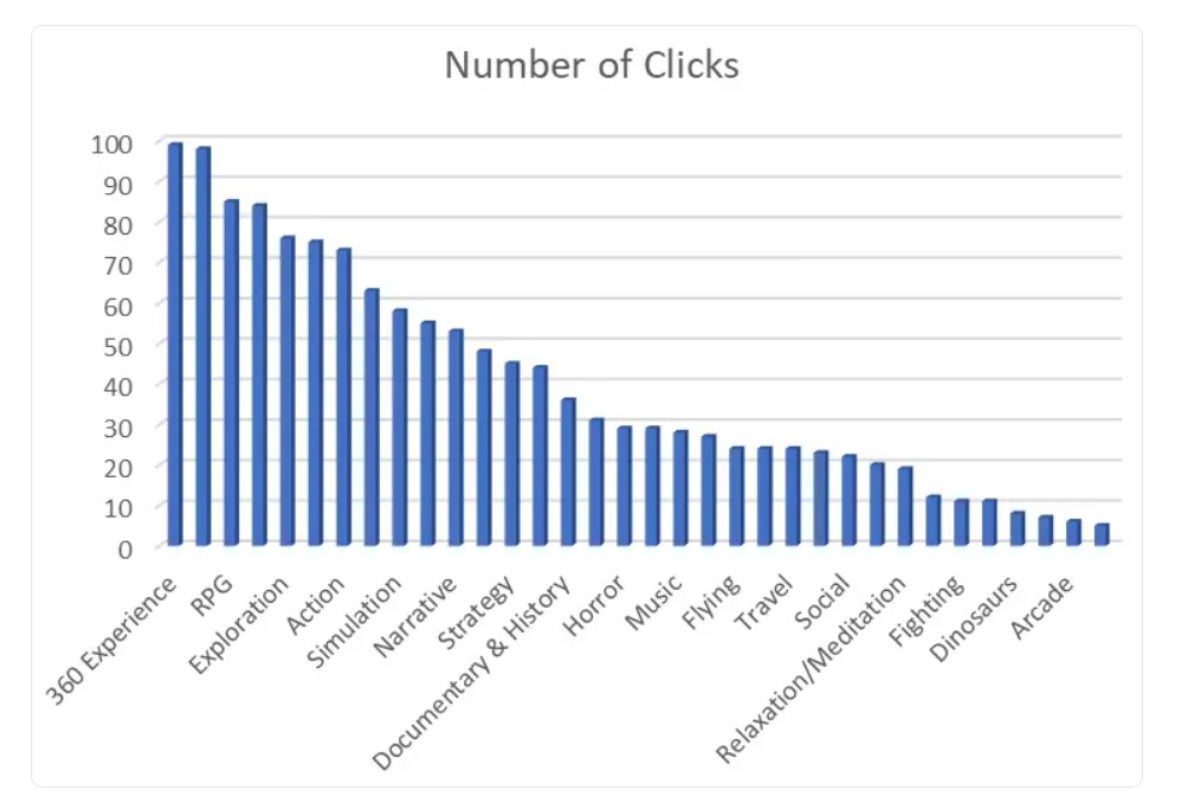
Some links

- Color palettes coolors

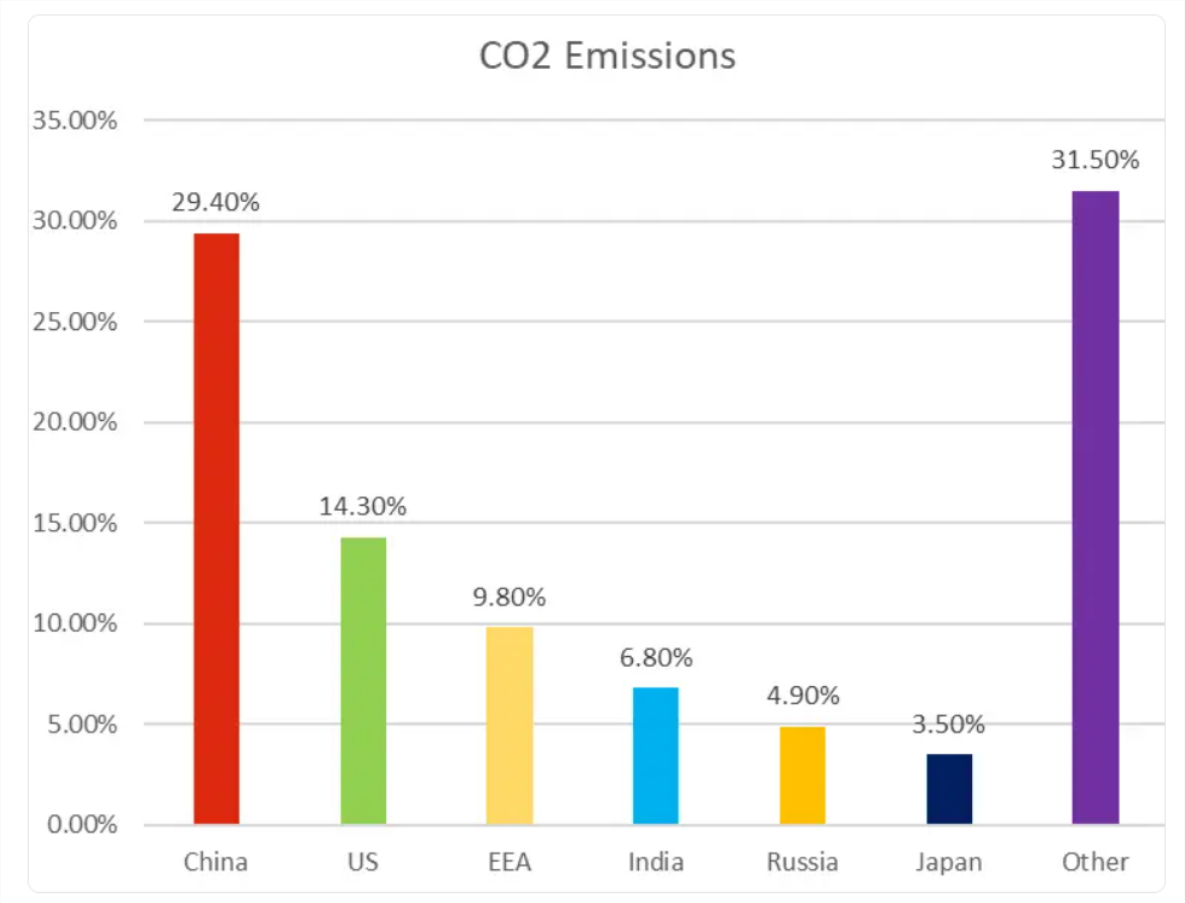
Good or bad ?


CO2 Emissions
35.00%
30.00%
25.00%
20.00%
15.00%
10.00%
5.00%
0.00%
29.40%
14.30%
9.80%
6.80%
4.90%
3.50%
31.50%
China
US
EEA
India
Russia
Japan
Other

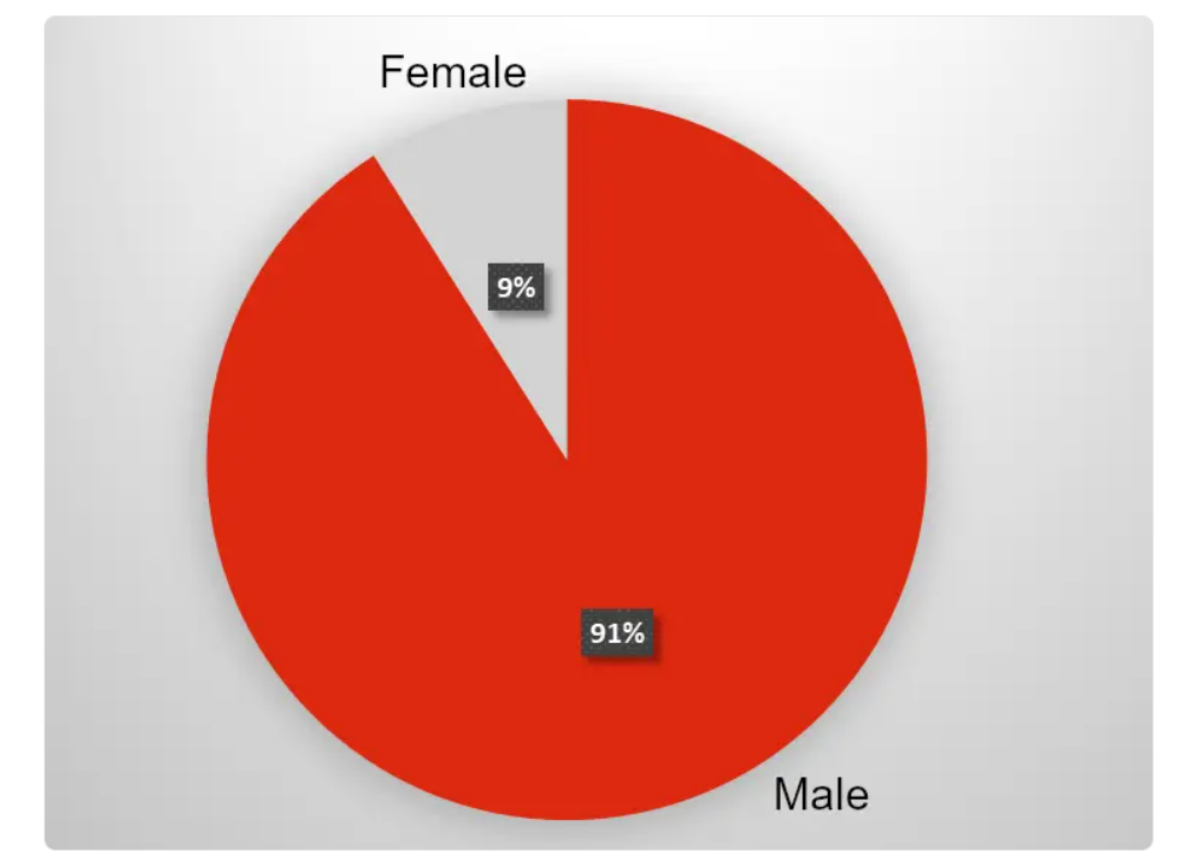
Good or bad ?


Number of Clicks
100
90
80
70
60
50
40
30
20
10
0
360 Experience
RPG
Exploration
Action
Simulation
Narrative
Strategy
Documentary & History
Horror
Music
Flying
Travel
Social
Relaxation/Meditation
Fighting
Dinosaurs
Arcade

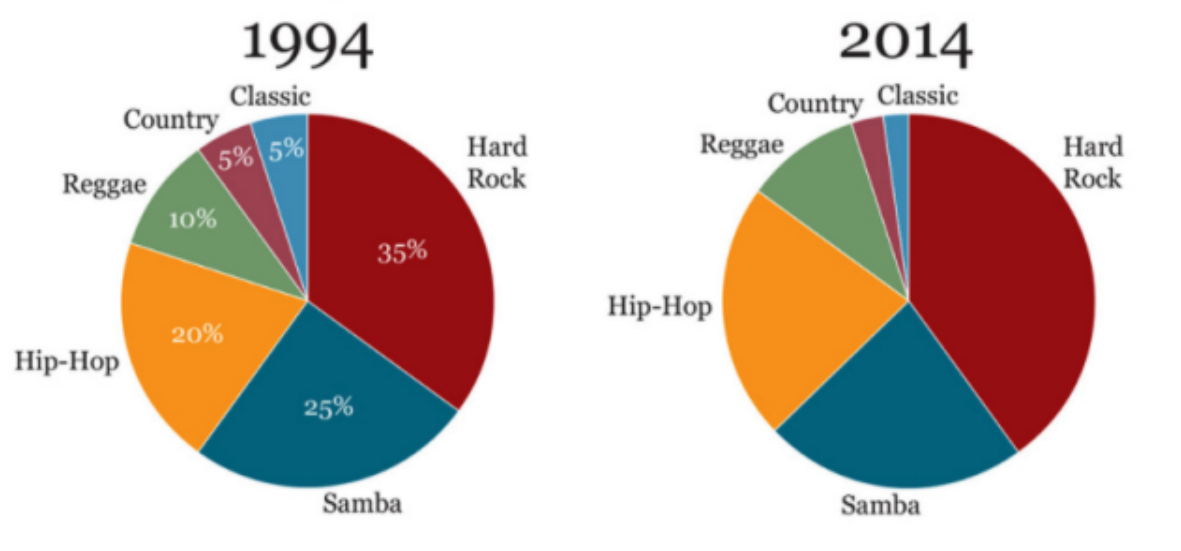
Good or bad ?


CO2 Emissions
35.00%
30.00%
25.00%
20.00%
15.00%
10.00%
5.00%
0.00%
29.40%
14.30%
9.80%
6.80%
4.90%
3.50%
31.50%
China
US
EEA
India
Russia
Japan
Other

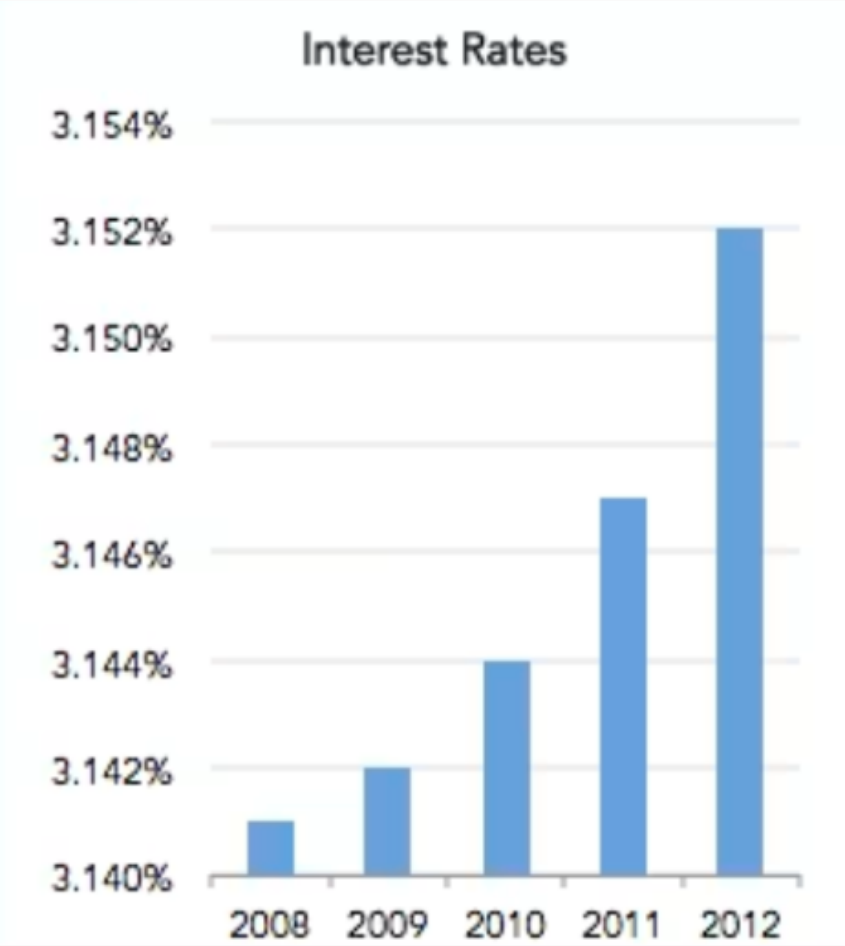
Good or bad ?



Good or bad ?


1994
Classic
Country
5%
5%
Hard
Rock
Reggae
10%
35%
20%
Hip-Hop
25%
Samba
2014
Country
Classic
Reggae
Hard
Rock
Hip-Hop
Samba

Good or bad ?


Interest Rates
3.154%
3.152%
3.150%
3.148%
3.146%
3.144%
3.142%
3.140%
2008
2009
2010
2011
2012