Escola d'Enginyeria UAB

UAB

# SEMINARI 6. *Advanced Systems II*

## 1. OBJECTIUS

Aquest seminari està enfocat a Sistemes Avançats II, en ell veure'm de manera pràctica algunes de les eines que vam veure a la classe de teoria de sistemes avançats I.

A la primera part, farem servir una de les tècniques que vam veure per mostras varies proporcions de cop i aprendrem a visualitzar-la : el *treemaps*.

A la segona part, veurem com fer un correlograma que ens permet veure la correlació entre varies variables numèriques.

En la tercera part, finalment, veurem un scatter plot matrix (SPLOM) dels que vam veure també a teoria. Aquest gràfic, com vam veure, ens permet veure també correlacions entre parelles de variables d'un dataframe amb múltiples variables (però no gaire gran).

Recordeu carregar les llibreries necessàries com sempre.

## 2. PART 1. *Advanced systems II*

En aquesta primera part, com ja hem anunciat a l'inici, veurem una eina que vam veure en teoria, i hem repassat just abans, els *treemaps*. Com sempre mireu les llibreries i paquets que necessiteu abans de començar.

**EXERCICIS:**

**1.- Utilitzant el que es va explicar sobre treemaps en R (inclòs com recordatori en les transparències d'avui), se us demana fer un treemap utilitzant el dataframe mtcars. On:**

- **L'àrea de les rajoles/caselles (tiles) i la seva 'etiqueta' vingui donada pel desplaçament (*disp*) de cada cotxe.**
- **El color de cada 'pare' vingui donat pel país d'origen del cotxe. És a dir per la variable que hem creat abans (`mtcars.country`)**
  **Feu ús de:** `mtcars.country <- c(rep("Japan", 3), rep("US",4), rep("Europe", 7),rep("US",3), "Europe", rep("Japan", 3), rep("US",4), rep("Europe", 3), "US", rep("Europe", 3))`

**a) Prepareu el mapeig (*aes*) del vostre ggplot i afegiu la geometria adient que hem vist per fer treemaps**

Seguint :

## 1.1. Visualizing many proportions at once

2. In order to create a treemap, the data must be converted to desired format using treemapify(). The important requirement is, you need to identify in your data:
   - One variable each describes the area of the tiles ('value')
   - One variable for fill color ('parent')
   - One variable that has the tile's label ('id')
   - The parent group ('parent')

   And map them by using aes:

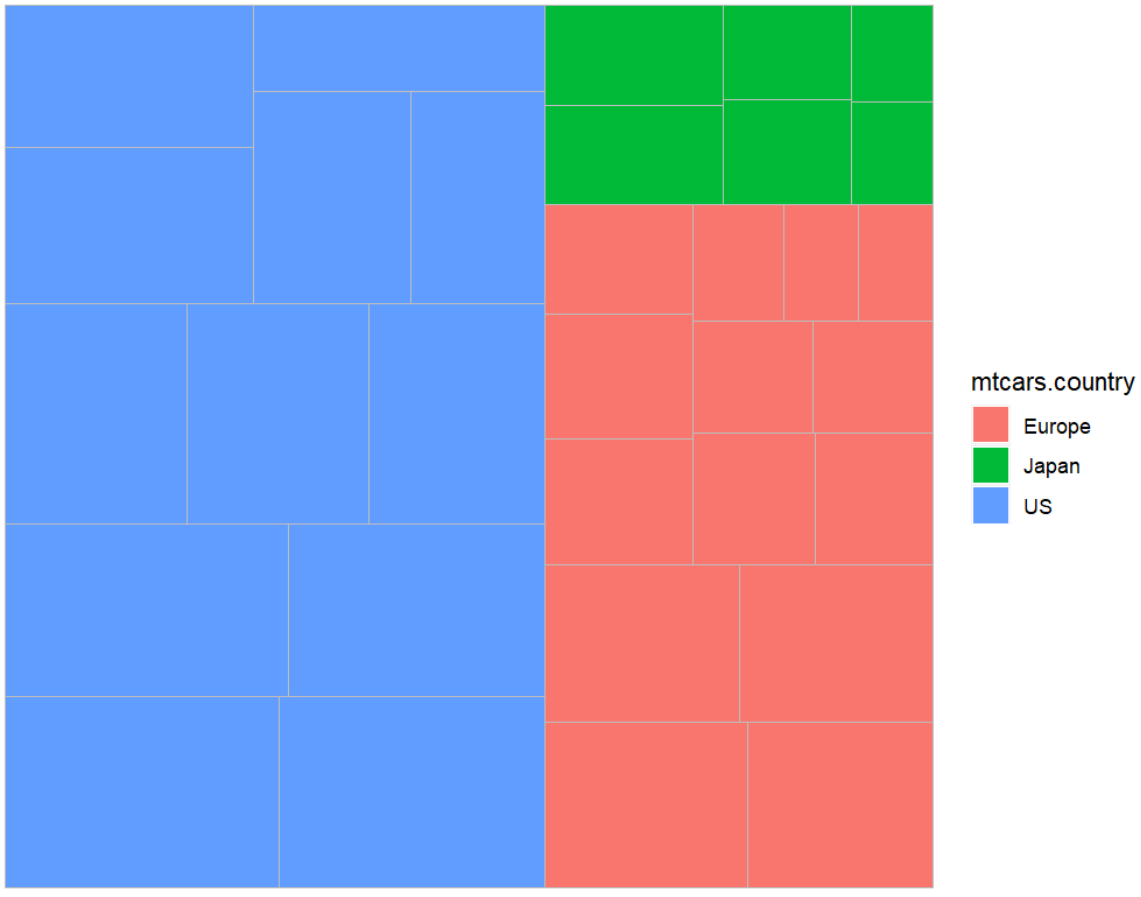
```
> ggplot(Proglangs, aes(area=value, fill=parent,
label=id, subgroup=parent)) + geom_treemap()
```

3. Use geom_treemap



En el nostre cas tenim:

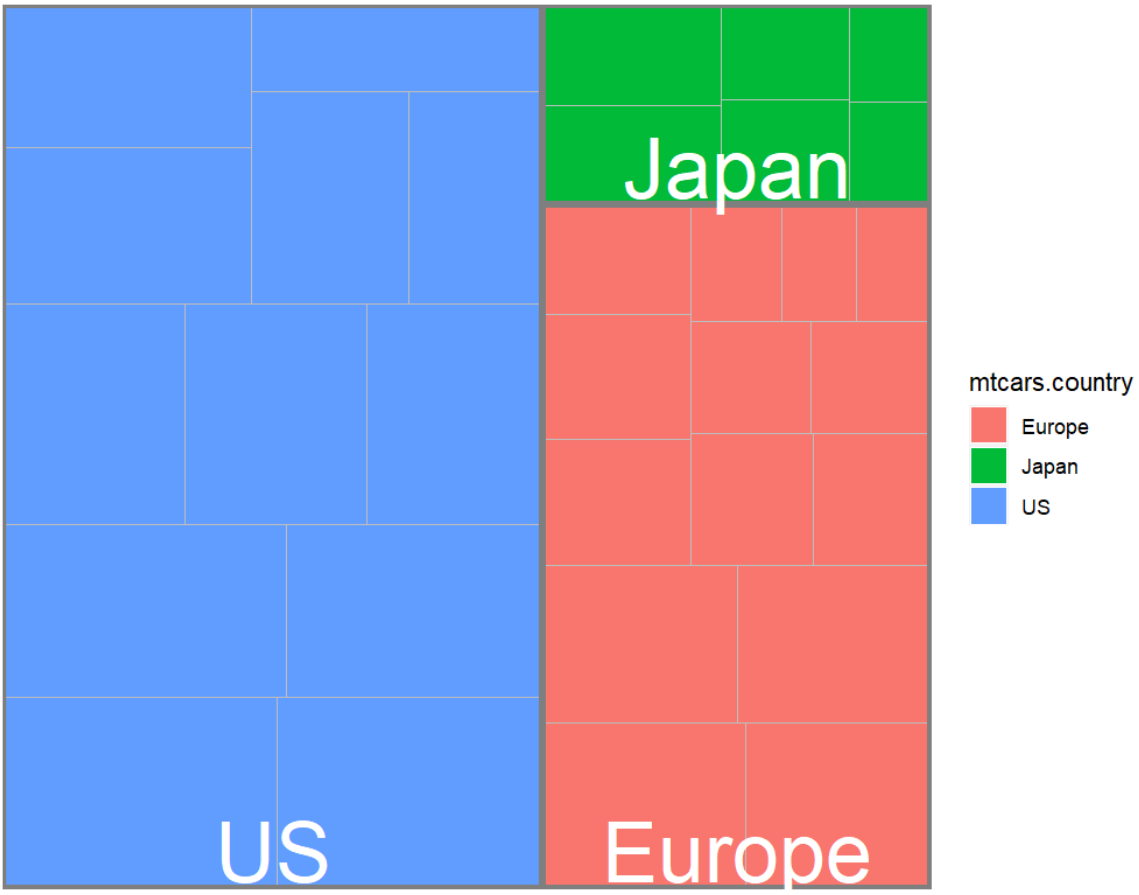
> ggplot(mtcars, **aes(area=disp, fill=mtcars.country, subgroup=mtcars.country))+geom_treemap()**




**b) Poseu el contorn de cada subgrup i la etiqueta/text del mateix en blanc**

Seguint les diapositives 9,10 i 11 de la primera part de la classe d'avui:

> ggplot(mtcars, aes(area=disp, fill=mtcars.country, subgroup=mtcars.country))+geom_treemap()+ **geom_treemap_subgroup_border() +geom_treemap_subgroup_text(color='white')**




**c) Ara que ja teniu l'estructura i etiquetats als 'pares'. Creeu una variable `mtcars.id` seguint:**

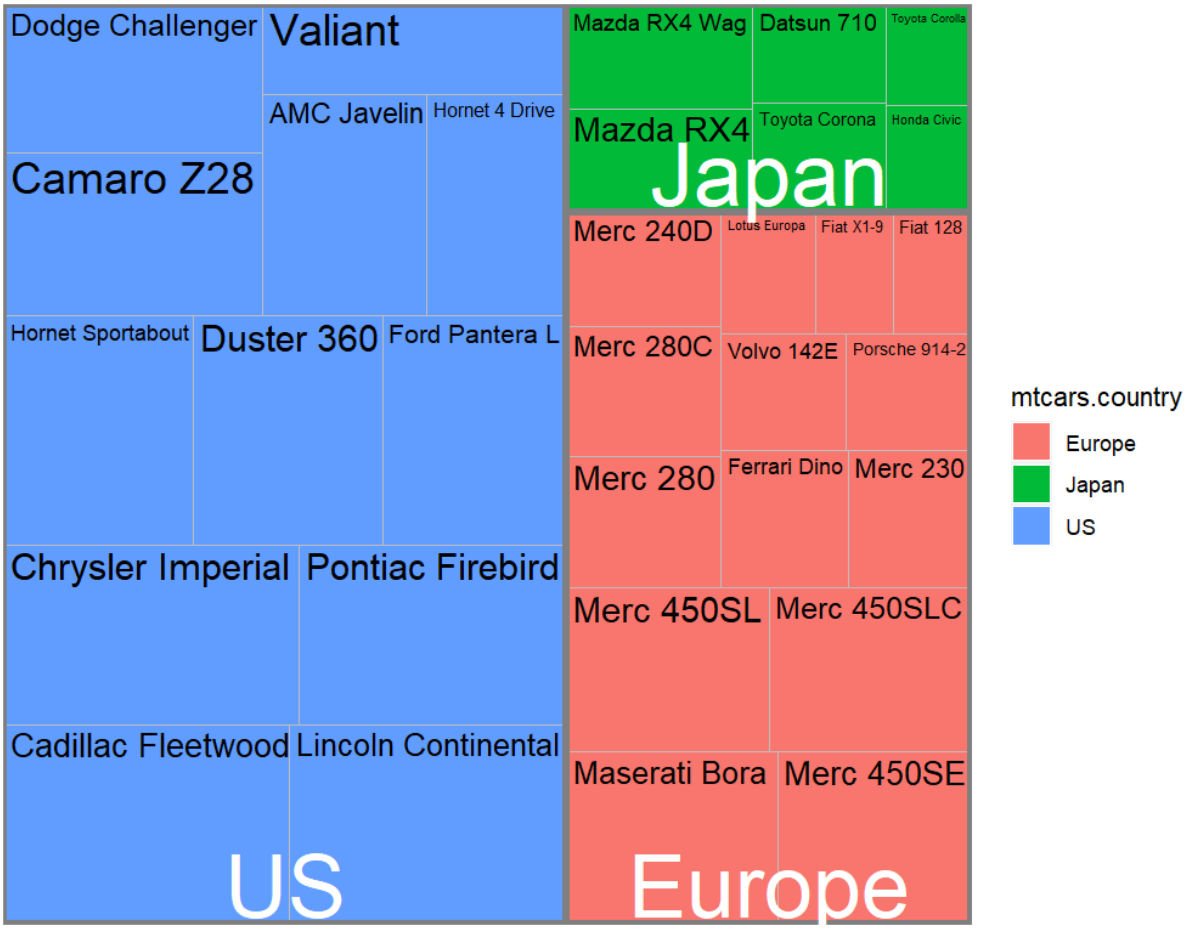
**`> mtcars.id=row.names(mtcars)`**

**Fixeu-vos que us agafa els noms de les marques dels cotxes (files del dataframe)**

**Afegiu en negre el text d'aquesta variable en el vostre treemap.**

Seguint la diapositiva 12 de la primera part de la classe:

> ggplot(mtcars, aes(area=disp, fill=mtcars.country, subgroup=mtcars.country))+geom_treemap()+ geom_treemap_subgroup_border() +geom_treemap_subgroup_text(color='white')+**geom_treemap_text (aes(label=mtcars.id))**




**d) Podeu treure la llegenda amb `theme(legend.position = "none")`**

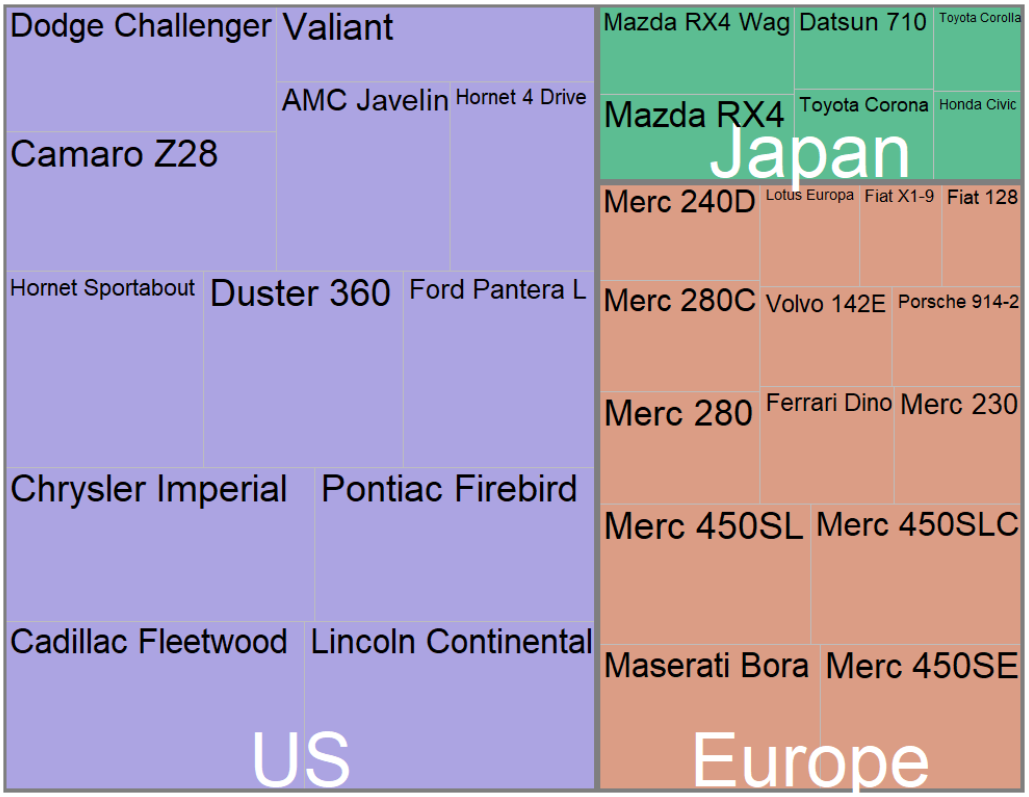
**Milloreu l'edició amb colors menys cridaners. Feu ús de**

```
? scale_fill_discrete_qualitative
```

**Proveu aquestes paletes per exemple, que funcionen bé amb aquest layer: `Pastel 1, Dark 2, Dark 3, Set 2, Set 3, Warm, Cold, Harmonic, Dynamic`**

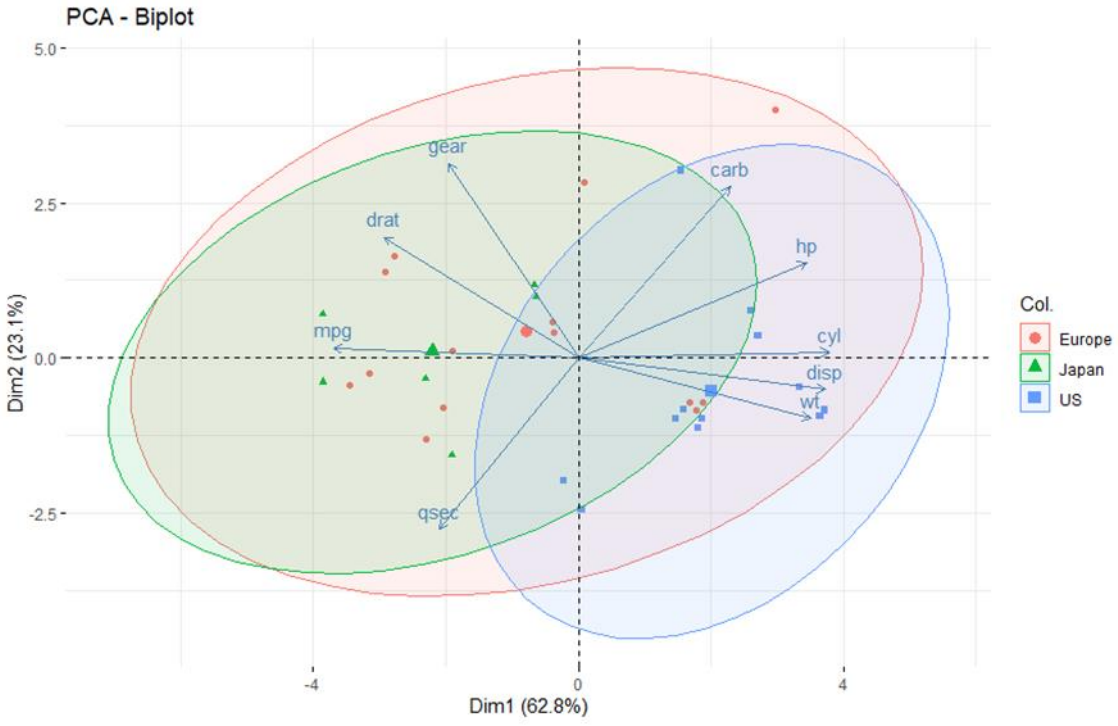
```
> ggplot(mtcars, aes(area=disp, fill=mtcars.country, subgroup=mtcars.country))+geom_treemap()+ geom_treemap_subgroup_border() +geom_treemap_subgroup_text(color='white')+geom_treemap_text (aes(label=mtcars.id))+scale_fill_discrete_qualitative(palette = "Dynamic")+theme(legend.position = "none")
```

Si no us funciona: `scale_fill_discrete_qualitative`, feu ús de `scale_fill_brewer`




També podríeu millorar-lo fent una lletra més petita en algunes etiquetes, posant un títol millor en la llegenda, si la deixeu, etc. La idea del seminari d'avui era aprendre a fer treemaps, però podeu fer proves per personalitzar el vostre gràfic.

**e) Quines conclusions en podeu extreure? Observeu alguna equivalència amb el PCA que vam fer en l'exercici pràctic de la última classe de teoria?**



**Possibles conclusions:** Els cotxes japonesos clarament tenen un desplaçament menor. Els de US fan un desplaçament molt major.

**Equivalència amb PCA:** En el PCA ja havíem vist que els cotxes nord-americans es caracteritzen per tenir valors elevats de *cyl*, *disp* i *wt*. En aquest treemap hem definit l'àrea de les rajoles/caselles segons el desplaçament (*disp*). Podem veure altre cop que els cotxes d'US són els que fan un desplaçament major.

## 3. PART 2. Correlograma

En aquesta segona part del seminari, anem a veure algunes eines de les que vam veure a teoria per veure relacions entre variables numèriques quan tenim moltes variables. Concretament veurem com fer un Correlograma. Farem servir el *dataframe* de R *mtcars*, però sense modificar. És a dir, el dataframe que conté 32 observacions i 11 variables sobre cotxes, sense NA's. Torneu-vos a familiaritzar amb el dataframe (str(mtcars)).

### EXERCICIS:

**1.- Volem fer un correlograma utilitzant el *dataframe mtcars* que ens mostri la correlació entre les sis variables:**

- **mpg: quilòmetres per galó (EUA)**
- **disp: Desplaçament**
- **hp: potència bruta**
- **drat: relació dels eixos posteriors**
- **wt: pes (1000 lliures)**
- **qsec: 1/milla de temps**

**a) Per això comenceu convertint el vostre *dataframe* a una *tibble* de nom *df_tb***

Com vam veure a l'últim seminari, podem fer la transformació utilitzant as.tibble de la llibreria tidyr.

```
> df_tb <- as_tibble(mtcars)
```

```
> df_tb
```

```
> df_tb
# A tibble: 32 x 11
     mpg   cyl  disp    hp  drat    wt  qsec    vs
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1  21       6  160    110  3.9   2.62  16.5     0
 2  21       6  160    110  3.9   2.88  17.0     0
 3  22.8     4  108     93  3.85  2.32  18.6     1
 4  21.4     6  258    110  3.08  3.22  19.4     1
 5  18.7     8  360    175  3.15  3.44  17.0     0
 6  18.1     6  225    105  2.76  3.46  20.2     1
 7  14.3     8  360    245  3.21  3.57  15.8     0
 8  24.4     4  147.    62  3.69  3.19  20       1
 9  22.8     4  141.    95  3.92  3.15  22.9     1
10  19.2     6  168.   123  3.92  3.44  18.3     1
# ... with 22 more rows, and 3 more variables:
#   am <dbl>, gear <dbl>, carb <dbl>
>
```

**b) Apliqueu la següent comanda new <-df_tb[, c(1,3,4,5,6,7)]. Què fa aquesta comanda?**

> **new <-df_tb[, c(1,3,4,5,6,7)]**

> **new** #al veure una *tibble* veiem la informació essencial

```
> new <-df_tb[, c(1,3,4,5,6,7)]
> new
# A tibble: 32 x 6
     mpg  disp    hp  drat    wt  qsec
   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1  21    160    110  3.9   2.62  16.5
 2  21    160    110  3.9   2.88  17.0
 3  22.8  108     93  3.85  2.32  18.6
 4  21.4  258    110  3.08  3.22  19.4
 5  18.7  360    175  3.15  3.44  17.0
 6  18.1  225    105  2.76  3.46  20.2
 7  14.3  360    245  3.21  3.57  15.8
 8  24.4  147.    62  3.69  3.19  20
 9  22.8  141.    95  3.92  3.15  22.9
10  19.2  168.   123  3.92  3.44  18.3
# ... with 22 more rows
> 
```

Aquesta comanda està seleccionant les columnes 1, 3, 4, 5, 6 i 7 de la *tibble* inicial. Ens estem de fet quedant amb les variables quantitatives.

**c) Intenteu crear una *tibble* igual que *new* a partir de *df_tb* usant eines de la llibreria dplyr que hem vist en classes anteriors**

En l'apartat anterior estem agafant columnes, per tant farem servir select:

> new2 <-**select**(df_tb, 'mpg','disp','hp','drat','wt','qsec')

També es pot posar:

> new2 <-select(df_tb, **c(1,3,4,5,6,7)**)

Ens hem quedat, com volíem, amb les variables: *mpg*, *disp*, *hp*, *drat*, *wt*, *qsec.*

**d) Ara que ja tenim les dades preparades, feu la matriu de correlació (que anomenarem *cormat*), arrodonint els valors a dos decimals. Per això utilitzeu les comandes que hem vist avui a la primera part de la classe (diapositives secció 6). Com és aquesta matriu?**

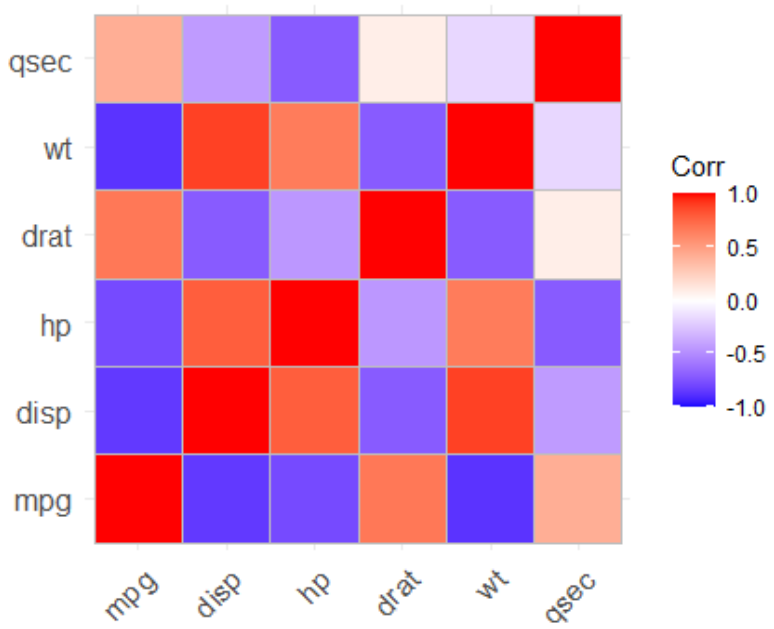
> cormat <- **round**(**cor**(new),2)

```
> cormat
       mpg  disp    hp  drat    wt  qsec
mpg   1.00 -0.85 -0.78  0.68 -0.87  0.42
disp -0.85  1.00  0.79 -0.71  0.89 -0.43
hp   -0.78  0.79  1.00 -0.45  0.66 -0.71
drat  0.68 -0.71 -0.45  1.00 -0.71  0.09
wt   -0.87  0.89  0.66 -0.71  1.00 -0.17
qsec  0.42 -0.43 -0.71  0.09 -0.17  1.00
> 
```

Com bona matriu de correlació, la matriu *cormat* té valors uns a la diagonal (on cada variable es relaciona amb sí mateixa), és simètrica, i els seus valors estan compresos entre -1 i 1. Recordem que com més alt és el valor absolut més correlacionades estan les variables.

**e) Si no el teniu instal·lat, instal·leu el paquet que hem vist a la primera part de la classe d'avui (diapositiva 13) per treballar amb matrius de correlació (ignoreu els warnings). Després, carregueu la llibreria necessària (ignorant també el warning), i useu la funció `ggcorrplot()` per visualitzar la vostra matriu de correlació**

```
> install.packages("ggcorrplot")

> library(ggcorrplot)

> ggcorrplot(cormat)
```




La funció `ggcorrplot` proporciona una solució per reordenar la matriu de correlació i mostra el nivell de significació al correlograma.

**f) Afegiu els coeficients de correlació fent servir l'argument '`lab=TRUE`' de la funció `ggcorrplot()`. Si teniu dubtes feu servir l'ajuda de R (`?ggcorrplot`). Compareu la vostra visualització amb la matriu `cormat` que heu creat en l'apartat (d). Ha fet algun canvi la funció `ggcorrplot()` a l'hora de visualitzar la matriu?**

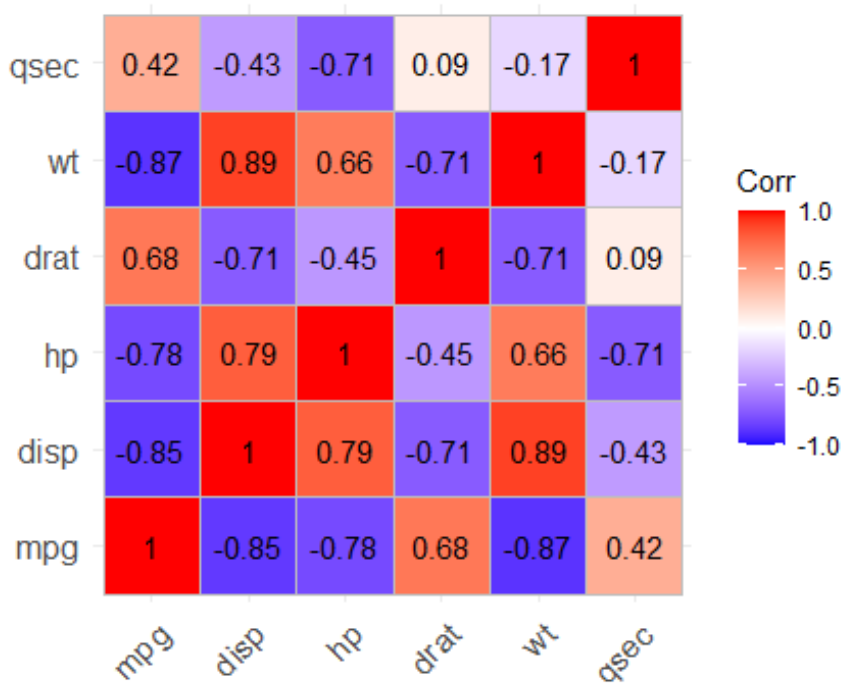
Fent ? ggcorrplot, veiem entre altres els arguments possibles de la funció

lab
logical value. If TRUE, add correlation coefficients on the plot.

R: Visualization of a correlation matrix using ggplot2 — Find in Topic

| | |
|---|---|
| show.legend | logical, if TRUE the legend is displayed. |
| legend.title | a character string for the legend title. lower triangular, upper triangular or full matrix. |
| show.diag | logical, whether display the correlation coefficients on the principal diagonal. |
| colors | a vector of 3 colors for low, mid and high correlation values. |
| outline.color | the outline color of square or circle. Default value is "gray". |
| hc.order | logical value. If TRUE, correlation matrix will be hc.ordered using hclust function. |
| hc.method | the agglomeration method to be used in hclust (see ?hclust). |
| lab | logical value. If TRUE, add correlation coefficient on the plot. |
| lab_col, lab_size | size and color to be used for the correlation coefficient labels. used when lab = TRUE. |
| p.mat | matrix of p-value. If NULL, arguments sig.level, insig, pch, pch.col, pch.cex is invalid. |
| sig.level | significant level, if the p-value in p-mat is bigger than sig.level, then the corresponding correlation coefficient is regarded as insignificant. |
| insig | character, specialized insignificant correlation coefficients, "pch" (default), "blank". If "blank", wipe away the |

> ggcorrplot(cormat, **lab=TRUE**)




Fixeu-vos que en la visualització, les files de la matriu a la visualització ens surten (per defecte) '*ordenades a l'inversa de com tenim realment la matriu*' a l'apartat (d):

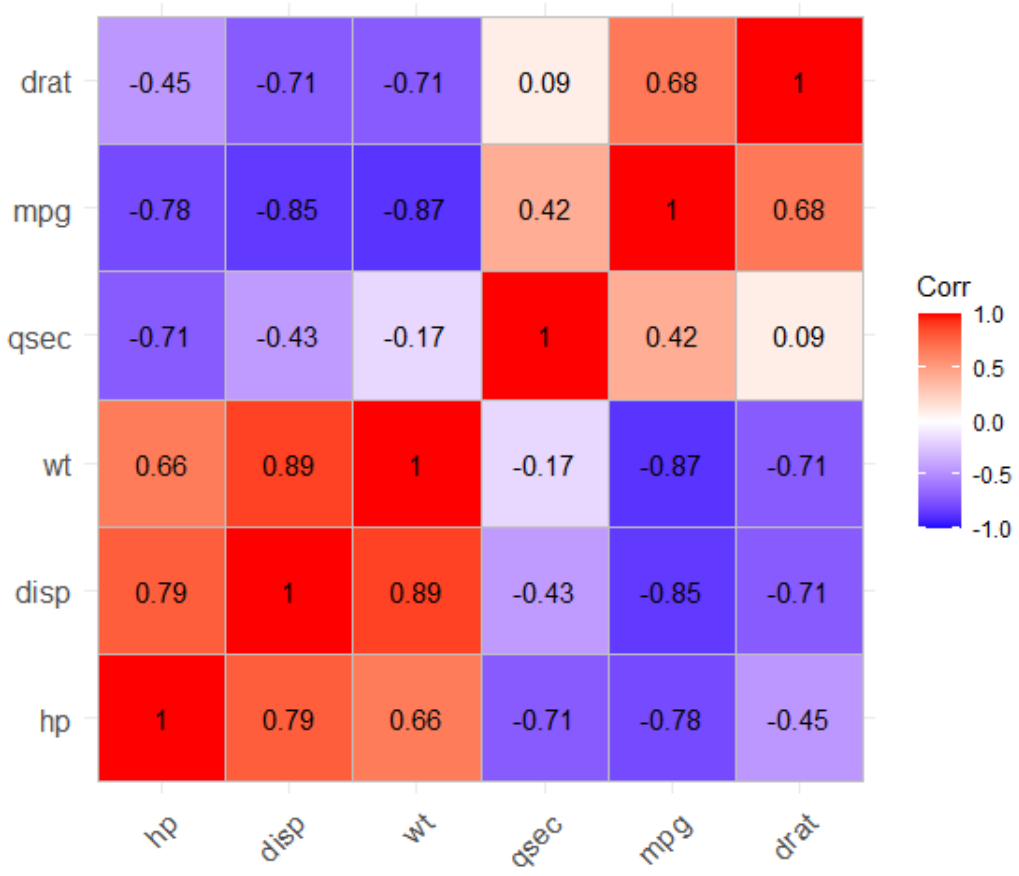
```
> cormat
        mpg  disp    hp  drat    wt  qsec
mpg    1.00 -0.85 -0.78  0.68 -0.87  0.42
disp  -0.85  1.00  0.79 -0.71  0.89 -0.43
hp    -0.78  0.79  1.00 -0.45  0.66 -0.71
drat   0.68 -0.71 -0.45  1.00 -0.71  0.09
wt    -0.87  0.89  0.66 -0.71  1.00 -0.17
qsec   0.42 -0.43 -0.71  0.09 -0.17  1.00
>
```

Per tant els 1's estan ara a l'anti-diagonal de la matriu, on cada variable es relaciona amb si mateixa.

**g) Ara afegiu l'argument 'hc.order=TRUE', què passa?**

Si afegim l'argument 'hc.order=TRUE':

> ggcorrplot(cormat, lab=TRUE, **hc.order = TRUE**)




Aquest argument fa ús de la funció `hclust()` de R. `hclust()` és una funció d'anàlisi jeràrquica de clústers sobre un conjunt de diferències i mètodes per analitzar-lo.

Al posar 'hc.order=TRUE', se'ns han reordenat a l'eix *x* i *y* les diferents variables per clústers.

**h) Per defecte la funció `ggcorrplot()` utilitza quadrats per a visualitzar la matriu de correlació. Feu ús de l'ajuda de R i mostreu cercles enlloc de quadrats. No hi poseu els coeficients de correlació aquest cop (és a dir, no activeu l'argument 'lab=TRUE').**

Al fer `?ggcorrplot`:

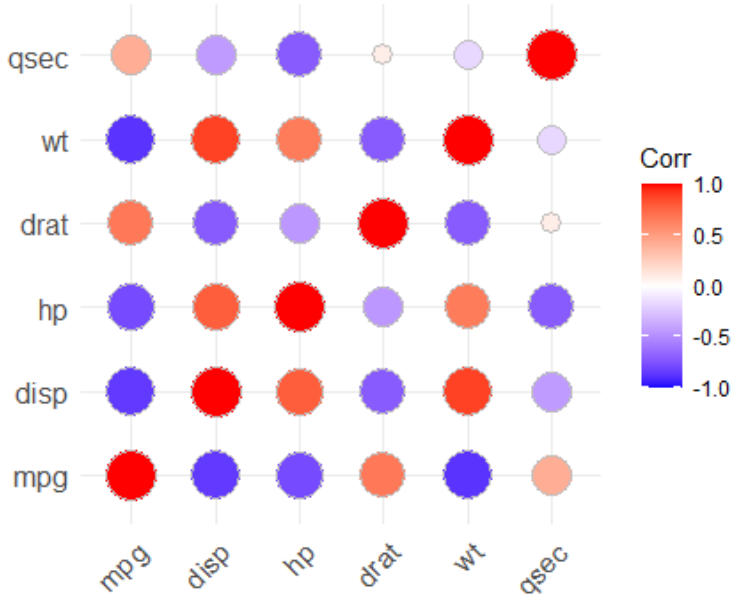
method
character, the visualization method of correlation matrix to be used. Allowed values are "square" (default), "circle".

`cor_pmat(x, ...)`

**Arguments**

| | |
|---|---|
| `corr` | the correlation matrix to visualize |
| `method` | character, the visualization method of correlation matrix to be used. Allowed values are "square" (default), "circle". |
| `type` | character, "full" (default), "lower" or "upper" display. |
| `ggtheme` | ggplot2 function or theme object. Default value is 'theme_minimal'. Allowed values are the official ggplot2 themes including theme_gray, theme_bw, theme_minimal, theme_classic, theme_void, .... Theme objects are also allowed (e.g., 'theme_classic()'). |
| `title` | character, title of the graph. |
| `show.legend` | logical, if TRUE the legend is displayed. |
| `legend.title` | a character string for the legend title. lower triangular, upper triangular or full matrix. |
| `show.diag` | logical, whether display the correlation coefficients on the principal diagonal. |
| `colors` | a vector of 3 colors for low, mid and high correlation values. |
| `outline.color` | the outline color of square or circle. Default value is "gray". |
| `hc.order` | logical value. If TRUE, correlation matrix will be hc.ordered using hclust function. |

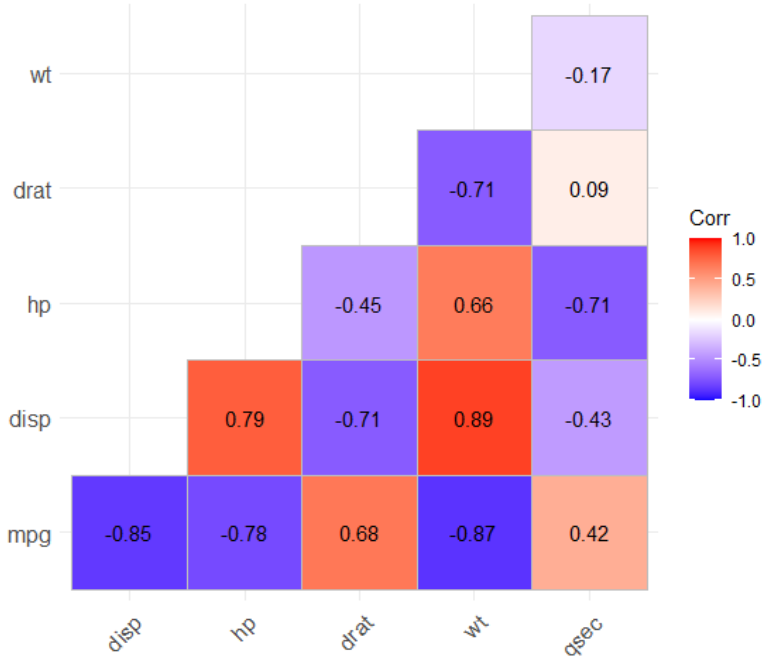
> ggcorrplot(cormat, **method='circle'**)



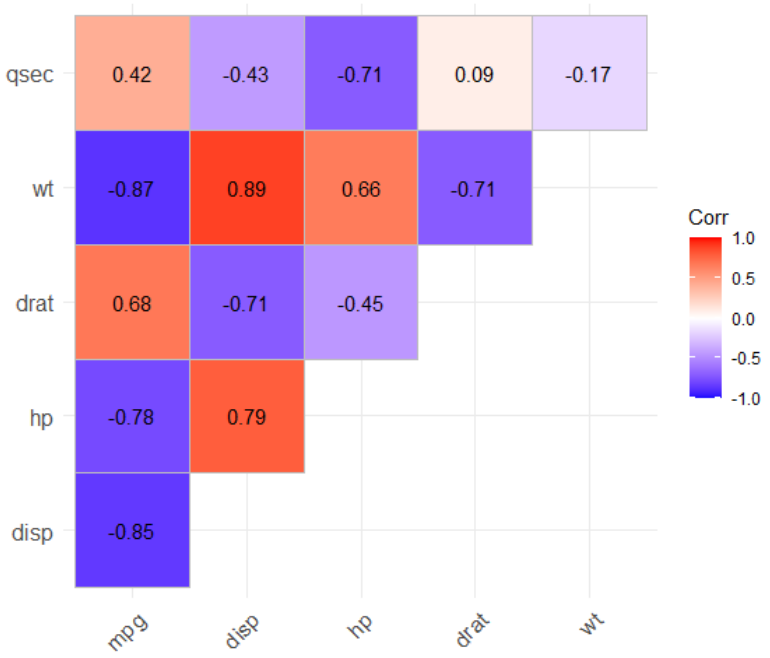

Podem veure que en aquest cas a més els cercles són proporcionals al valor absolut de correlació. Quan més a prop de 1, més grans, quan més a prop de 0, més petits. Per tant, en el cas dels cercles, no ens cal tenir els valors per tenir una intuïció de com de correlacionades estan les nostres variables.

**i) Donada la simetria de la matriu i que l'anti-diagonal està formada per 1's, podem mostrar just la part de sobre ("upper") o de sota ("lower") de l'anti-diagonal. Això ens permetrà reduir la "carrega de visualització de la informació" (i serà d'agrair per la gent que vegi la nostra gràfica). Indirectament centrarem la seva atenció en el realment important. Per això podem jugar amb l'argument type de la funció `ggcorrplot()`. Feu altre cop `?ggcorrplot` i introduïu aquest argument a la visualització de l'apartat (f).**

> ggcorrplot(cormat, lab=TRUE, **type = "lower"**)



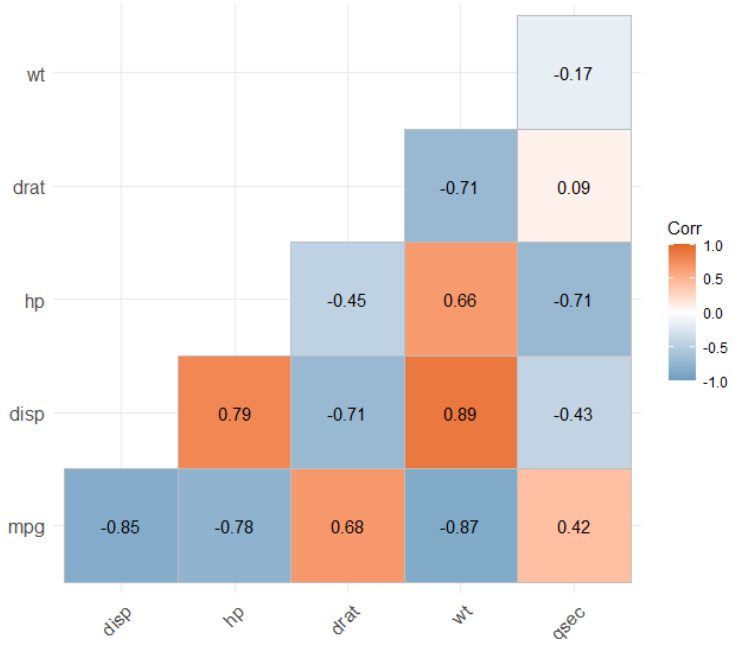

> ggcorrplot(cormat, lab=TRUE, **type = "upper"**)




**NOTA: (a) i (b) no són necessaris, podríem haver començat directament en (c), però ho hem fet per practicar amb l'ús de *tibbles*.**

**Podeu canviar els colors, per exemple per uns amb una tonalitat més suau. Per això fem us de l'argument color:**

> ggcorrplot(cormat, lab=TRUE, type="lower", **colors = c("#6D9EC1", "white", "#E46726")**)




**Material extra: Si voleu aventurar-vos a canviar colors, podeu fer us de la cheatsheet** https://www.nceas.ucsb.edu/sites/default/files/2020-04/colorPaletteCheatsheet.pdf

## 4. PART 3. Scatter Plot Matrix (SPLOM)

Per aquesta tercera part, treballarem amb el dataframe iris, amb el que ja hem treballat en ocasions anteriors i veurem com es crea un SPLOM.

Iris proporciona les mesures (en cm) de les variables longitud i amplada dels sèpals i dels pètals respectivament per 50 flors de cadascuna de les 3 espècies d'Iris (150 en total). Les espècies d'iris són: la Versicolor, la Virginica i la Setosa.

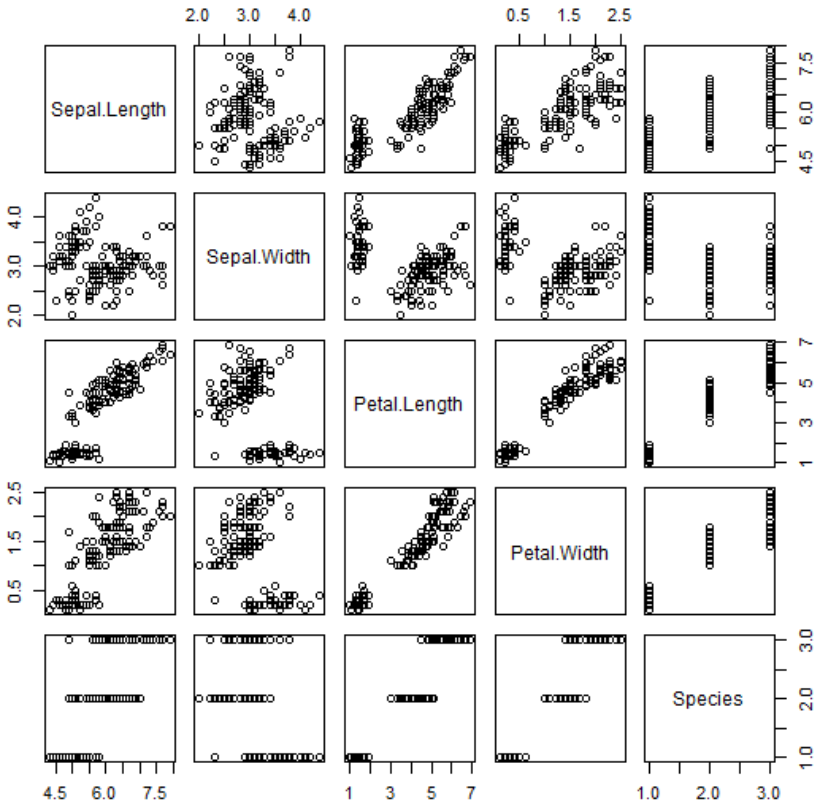

Iris és un dataframe amb 150 observacions (files) i 5 variables (columnes): **Sepal.Lenght**, **Sepal.Width**, **Petal.Length**, **Petal.Width** i **Species**. Els valors de les variables referents a les respectives longituds i amplades (mètriques) estan en centímetres.

**EXERCICIS:**

**1.- Partint del dataframe iris**

**a) Feu us de la funció `pairs` de R per fer una visualització SPLOM.**

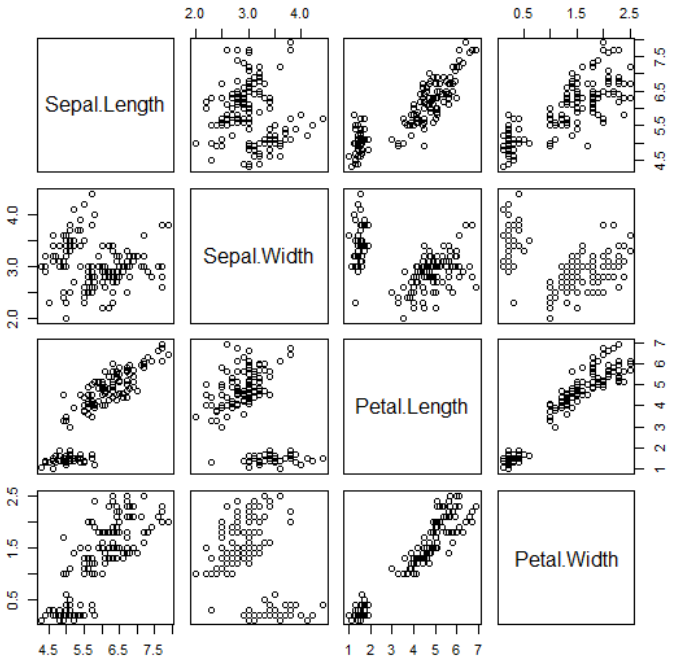
```
> pairs (iris)
```




**b) Veient el gràfic de l'apartat (a), eliminaríeu alguna variable? Per què? Si és el cas traieu-la i del SPLOM**

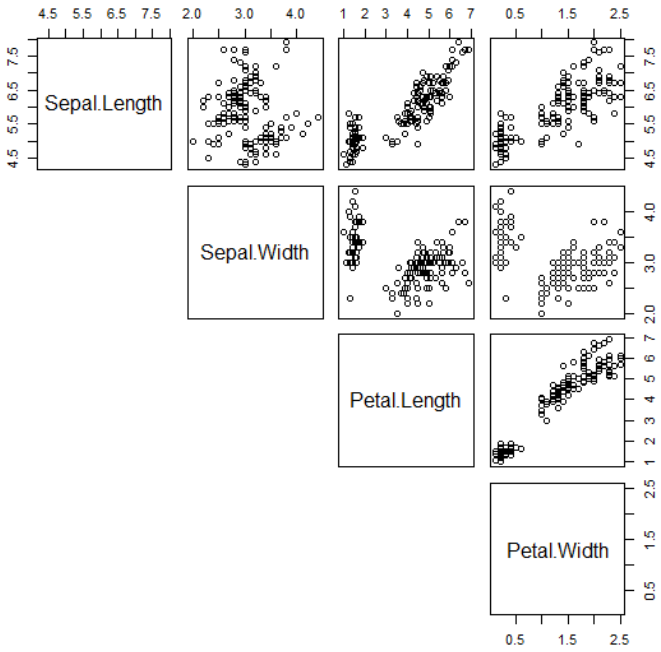
Veiem clarament que fer el scatter plot entre la variable Species (variable categòrica) i la resta de variables de mètriques no ens dóna massa informació. La traiem:

```
> pairs (iris [1:4])
```



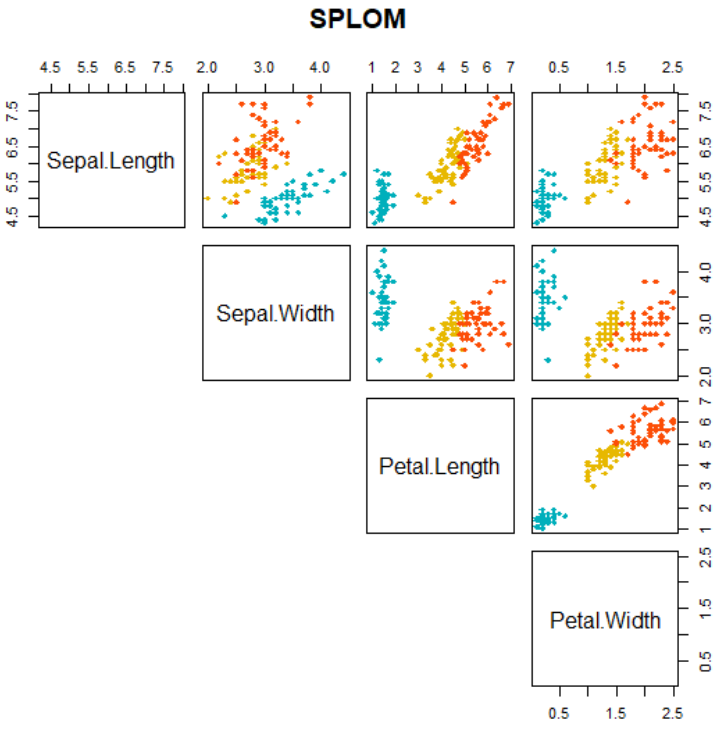
**c) Com és una matriu simètrica, elimineu la diagonal inferior del gràfic fent ús de : `lower.panel = NULL`**

```
> pairs(iris[1:4], lower.panel = NULL)
```




**c) Posa color per especies. Per tal d'afegir aquesta informació. Pots fer ús de `col=c("#00AFBB", "#E7B800", "#FC4E07")[iris$Species]`, o triar altres colors. Pots a més omplir el punt de color, afegint `pch=#` (on # és un número, per exemple , 18 o 19 són bones opcions). Afegeix títol també amb `main="SPLOM"`.**

```
> pairs(iris[,1:4],lower.panel=NULL, col = c("#00AFBB", "#E7B800", "#FC4E07")[iris$Species], pch=18, main="SPLOM" )
```



**2.- Ara anem a fer un scatter plot matrix (SPLOM) fent us de la funció ggpairs() del paquet GGally de ggplot2. Per això:**

**a) Com sempre primer instal·leu el paquet necessari, en aquest cas GGally i carregueu la llibreria (necessiteu també la llibreria ggplot2).**

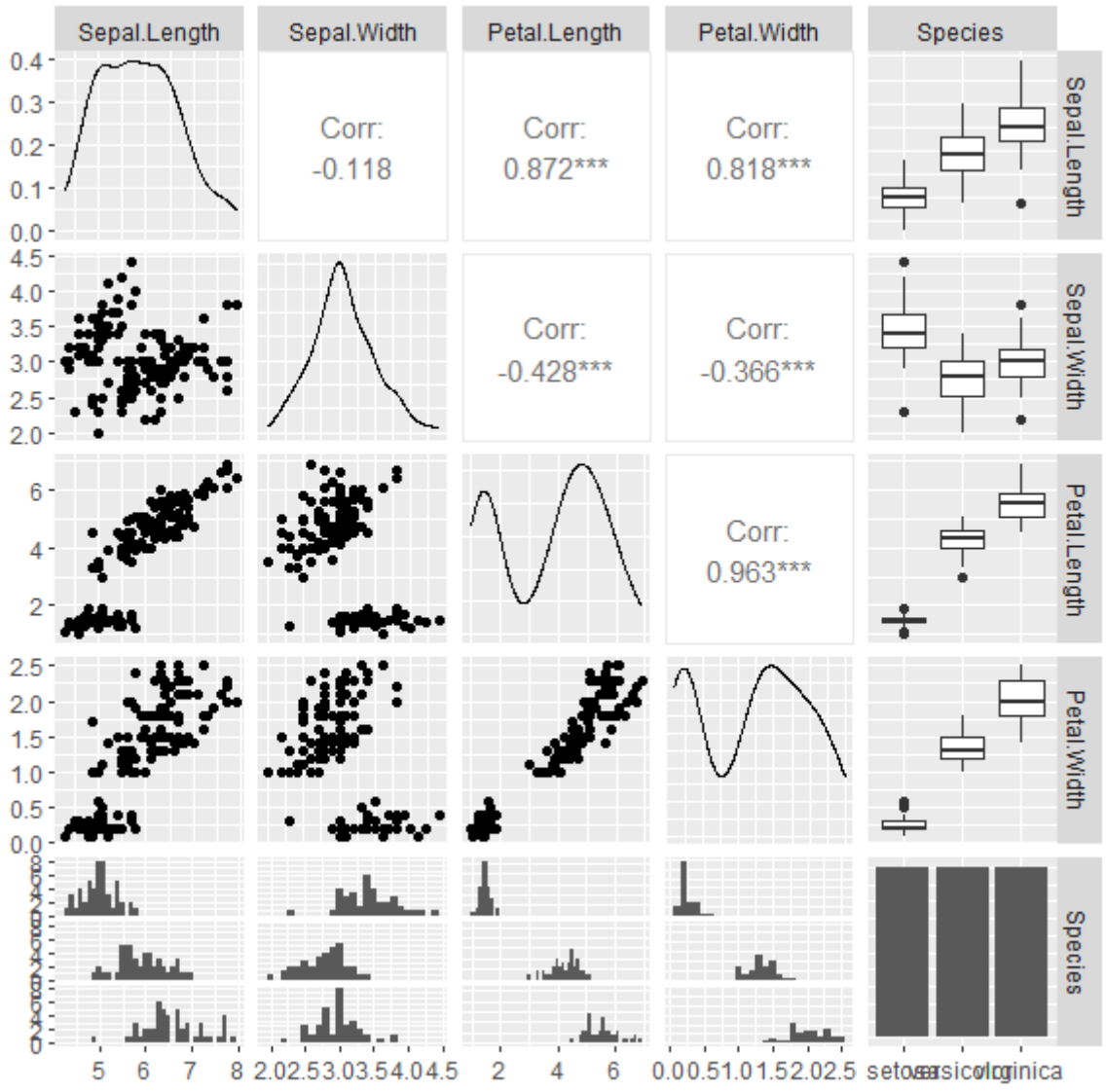
```
> install.packages("GGally")
> library(ggplot2)
> library (GGally)
```

**b) Un cop tingueu la llibreria carregada, feu `ggpairs(iris)`. Què observeu en la part esquerra, diagonal i part dreta del gràfic, respectivament?**

Els diagrames de dispersió (scatterplot) de cada parell de variables numèriques es dibuixen a la part esquerra de la figura. La correlació de Pearson es mostra a la dreta. La distribució variable està disponible a la diagonal.

**b.1) Si us ha sortit un warning referent al binwidth useu:**

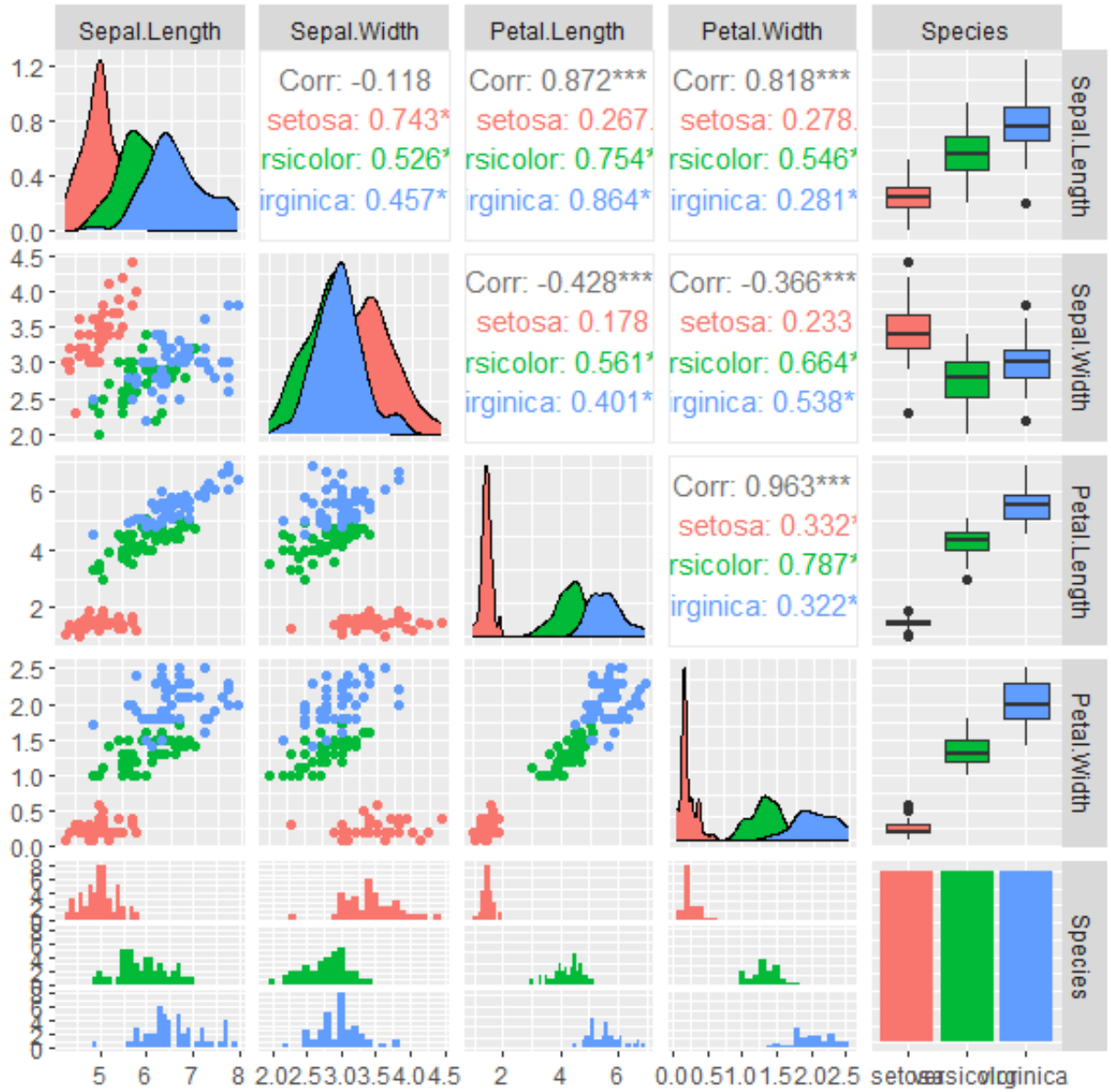
```
>ggpairs(iris, lower=list(combo=wrap("facethist", binwidth=0.1)))
```



**c) Fent servir l'estètica (`aes()`), completeu les *** de la següent comanda per tal de pintar el gràfic obtingut en l'apartat (b) segons l'espècie (*Species*). Pista: Com ha de ser la variable *Species* per poder usar l'estètica del color?**

```
> ggpairs(iris, aes(color=***),..)
```

```
> ggpairs (iris,aes(color=as.factor(Species)),lower=list(combo=wrap("facet
hist", binwidth=0.1)))
```



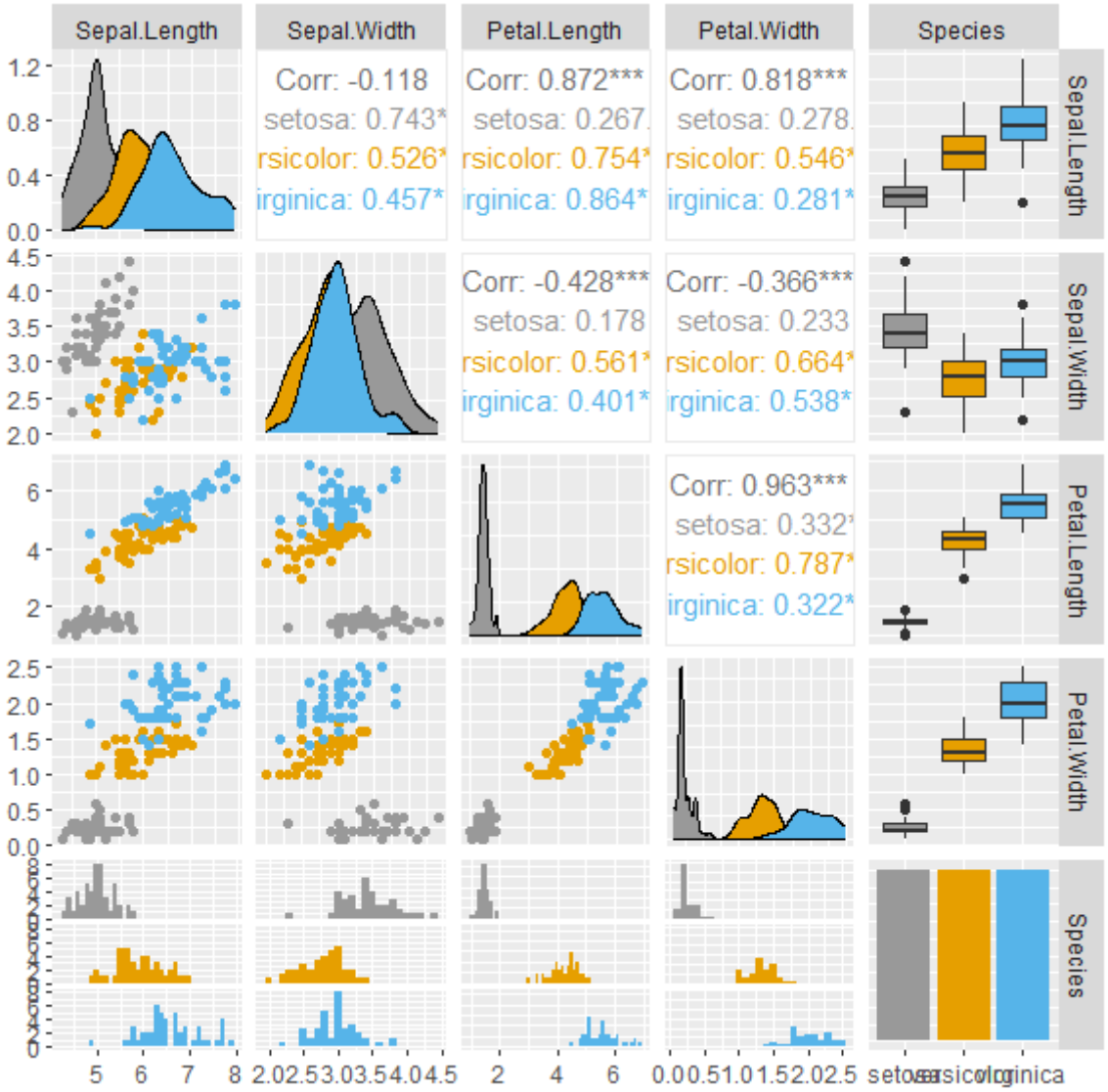

**d) Poseu un altre color, per exemple manualment posant el color donat en: `c("#999999", "#E69F00", "#56B4E9")`**

Per facilitar, assignem el gràfic a una variable

```
>gràfic<-ggpairs
(iris,aes(color=as.factor(Species)),lower=list(combo=wrap("facethist",
binwidth=0.1)))
```

Fixem-nos que alguns gràfics requereixen color i altres fill, per tant, haurem d'usar:

```
>grafic+scale_color_manual(values=c("#999999", "#E69F00", "#56B4E9"))+scale_fill_manual(values=c("#999999", "#E69F00", "#56B4E9"))
```




Podem millorar-ho usant el canal `theme.` Però com vam veure a classe de teoria, no usarem aquests gràfics amb moltes variables (4 com a màxim potser). Ara, ens poden ser de gran ajuda com a gràfiques exploratòries.

**e) Ara anem a categoritzar la longitud del pètal en dos subgrups:**

- **Direm que pertany al grup "llarg", si la longitud del pètal (`Petal.Length`) està per sobre de la longitud mitjana del pètal.**
- **Anàlogament, direm que és "curt", si la longitud del pètal està per sota de la longitud mitjana del pètal.**

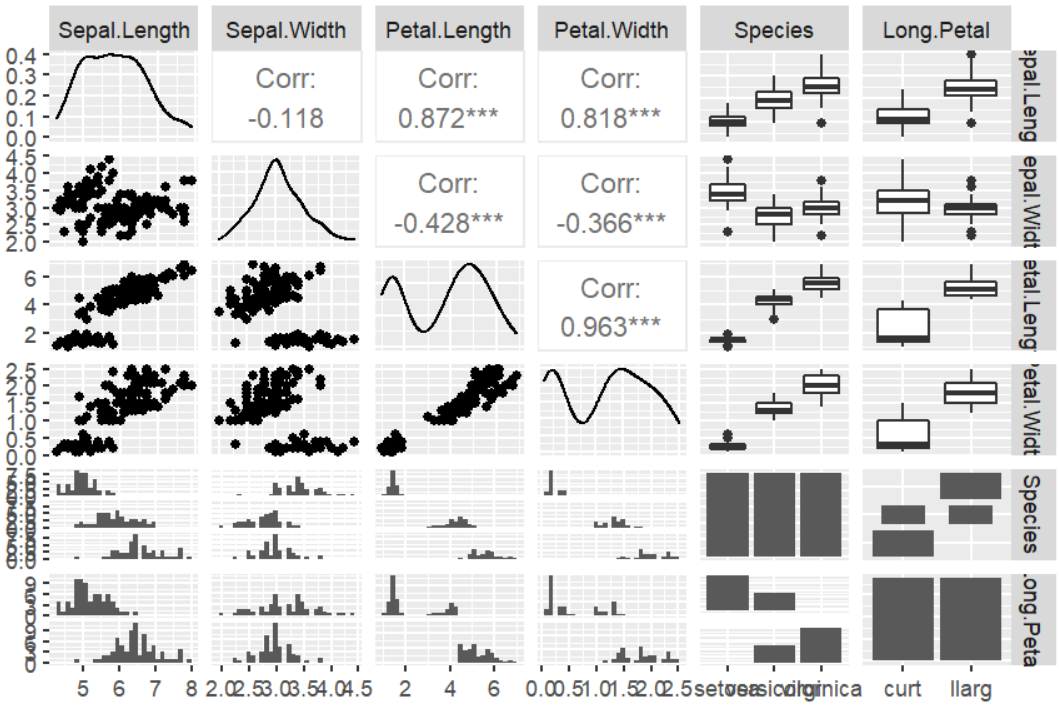
**Tot i que us dono la comanda per fer aquest apartat (e), ja que conté part d'estadística que no hem vist en aquest curs, fixeu-vos com ho fem:**

```
iris$Long.Petal<-
as.factor(ifelse(iris$Petal.Length>median(iris$Petal.Length), "llarg",
"curt"))
```

**En R, les dades poden ser forçades per transformar-les a un altre tipus. Aquí hem fet una coerció explícita per categoritzar, usant la funció `as.factor`. Indirectament, ja vam veure això amb la llibreria `mtcars` i la variable cilindres en el mòdul 3.**

**(Podeu trobar més informació sobre coerció explícita en la secció 4.7.1 d'aquest document:** https://bookdown.org/jboscomendoza/r-principiantes4/coercion.html )

**f) Feu us de ggpairs per mostrar el dataframe *iris* modificat en (d)**




**En aquesta nova visualització que us sortirà podreu observar:**

- **Uns gràfics de densitat i uns gràfics de barres a la diagonal que reflecteixen les distribucions marginals de les variables. Veureu, que per a les trames quantitatives-quantitatives, hi ha una forta associació positiva entre la longitud i l'amplada dels pètals, que també es recolza en una correlació de 0,872.**
- **També observareu un panell de mosaic entre les espècies d'iris i els grups de longitud de pètals mostrant ambdues distribucions condicionals; per exemple, en el panell de la fila 5, columna 6, us ha d'aparèixer la distribució dels grups de longitud de pètals per espècies.**

Podeu veure més exemples amb `ggpairs()` usant el dataframe `iris` en: https://r-charts.com/es/correlacion/ggpairs/.

Judit Chamorro Servent
Bellaterra, Abril 2025

Escola d'Enginyeria UAB

UAB

# SEMINARI 7. *Advanced Systems II*

## 1. OBJECTIUS

Introduir la creació de mapes amb R i concretament amb ggplot2. Els mapes són una eina de sistemes avançats de visualització. Ja vam veure a la classe de teoria alguns exemples d'ús, i avui crearem alguns mapes més.

**NOTA:** Si vau fer els exercicis durant la segona part de la classe teòrica del Tema 9, podeu passar directament a l'exercici 4 de l'apartat 4.

## 2. PAQUETS A INSTAL·LAR (per parts 3 i 4 del seminari)

**Hi ha molts paquets R disponibles a CRAN (la xarxa d'arxius completa R, que és el dipòsit principal dels paquets R). Per aquest seminari, haureu d'instal·lar alguns paquets addicionals**

**# Aquests són paquets que necessitareu (a més de ggplot2 i dplyr), però que probablement ja teniu. (No us molesteu a instal·lar-los si ja els teniu):**

```
> install.packages (c("devtools", "stringr"))
```

**#cliqueu no si us demana restart R**

**# També necessitareu, alguns paquets de mapes estàndard:**

```
> install.packages (c ("maps", "mapdata","ggmap"))
```

## 3. COM FEM MAPES

**Normalment podem descompondre la creació d'un mapa en dos "problemes": Utilitzar dades d'una certa font per dibuixar el mapa (background) i afegir al mapa informació de les metadades d'una altra font.**

**Tractarem majorment amb classes *sp* per dates espacials (per defecte) o directament classes *sf* (simple features).**

**Ara que tenim els paquets instal·lats, carreguem les llibreries que no tinguem carregades:**

```
> library(ggplot2)
> library(ggmap)
> library(maps)
> library(mapdata)
```

- **El paquet de mapes (maps) conté una gran quantitat de contorns de continents, països, estats i comtats que han estat en R durant molt de temps (pot ser doncs, que no estigui totalment actualitzat o que no estigui 100% correcte i això, ho haurem de tenir en compte). També conté esquemes i punts de ciutats, etc. Exemples: nz, State, world, etc.**
- **El paquet *mapdata* conté uns quants esquemes més, amb una resolució més alta.**
- **Tot i que el paquet "maps", inclou una funció de traçat, optarem per utilitzar ggplot2 per representar els mapes.**

**Recordeu que ggplot2 funciona en marcs de dades (*dataframes*). Per tant, necessitarem passar les dades dels mapes a un format *dataframe* que ggplot2 entengui.**

**ggplot2 proporciona la funció `map_data()`.**

**Penseu-ho com una funció que converteix una sèrie de punts al llarg d'un esquema en un marc de dades d'aquests punts.**

**Sintaxi: `map_data ("nom")` on "nom" és una cadena amb el nom d'un mapa al paquet de mapes o dades de mapes**

**Exemple:**

```
> usa <- map_data("usa")
> str(usa)
'data.frame':   7243 obs. of  6 variables:
 $ long     : num  -101 -101 -101 -101 -101 ...
 $ lat      : num  29.7 29.7 29.7 29.6 29.6 ...
 $ group    : num  1 1 1 1 1 1 1 1 1 1 ...
 $ order    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ region   : chr  "main" "main" "main" "main" ...
 $ subregion: chr  NA NA NA NA ...
>
```

**L'estructura d'aquests dataframes són senzilles:**

- ***long* - és la longitud. A l'oest del meridià primer aquestes són negatives.**
- ***lat* – és la latitud.**
- ***order* - només mostra en quin ordre ggplot ha de "connectar els punts"**
- ***region* & subregion - indiquen quina regió o subregió envolta un conjunt de punts.**
- ***group* - és molt important. Les funcions de `ggplot2` poden adoptar un argument de grup que controla (entre altres coses) si els punts adjacents haurien d'estar connectats per línies. Si formen el mateix grup, es connecten, però si estan en grups diferents, no ho fan.**
  **Essencialment, tenir punts en diferents grups significa que `ggplot` "aixeca el bolígraf" quan va entre ells.**

## 4. POLYGON MAPS

**Quan tenim la longitud i latitud dels mapes, una manera senzilla de dibuixar mapes, com veurem en l'exercici 1, és utilitzant `geom_polygon()`. Aquesta geometria ens ajuda a dibuixar contorns per diferents regions.**

**`geom_polygon()` dibuixa línies entre punts i "les tanca" (és a dir, dibuixa una línia des de l'últim punt fins al primer punt).**

### EXERCICIS

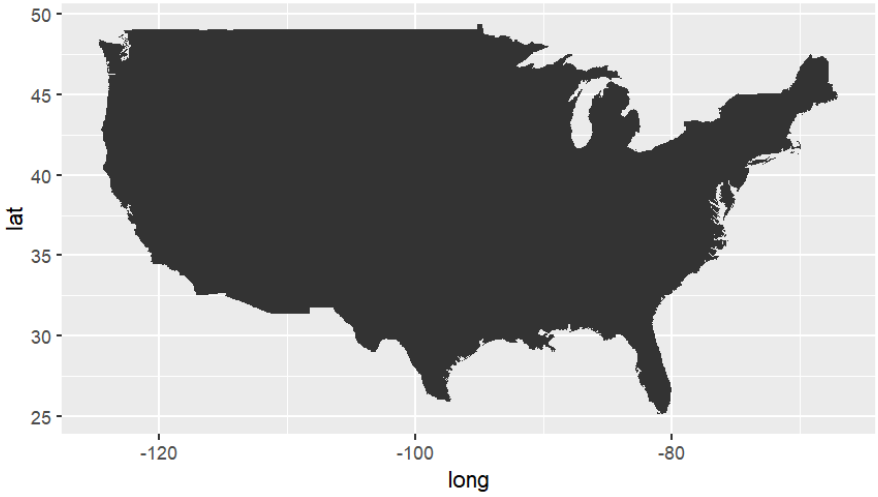
**1.- Anem a testejar la comanda `geom_polygon()` per dibuixar mapes quan tenim dades espacials que fan ús de la longitud-latitud. Per fer ús de ggplot, recordeu que necessitem les dades, el mapeig (al nostre sistema de coordenades com a mínim) i la geometria. En aquest cas, el nostre sistema de coordenades és `x =`**

***long*** **i** `y = lat`**. Per tant, com sempre, heu de fer un mapeig (usant l'estètica aes) per representar les dades en el vostre sistema de coordenades. A més, fent ús de *group* en el vostre mapeig, agrupeu les dades espacials de manera que es connectin els punts que han d'estar connectats per línies. Finalment, si agregueu l'argument *coord_quickmap()*us permet ajustar els eixos per assegurar-nos que la longitud i la latitud es representen a la mateixa escala.**

**(a) Seguint les indicacions que se us ha donat a l'enunciat, dibuixeu el mapa de USA**

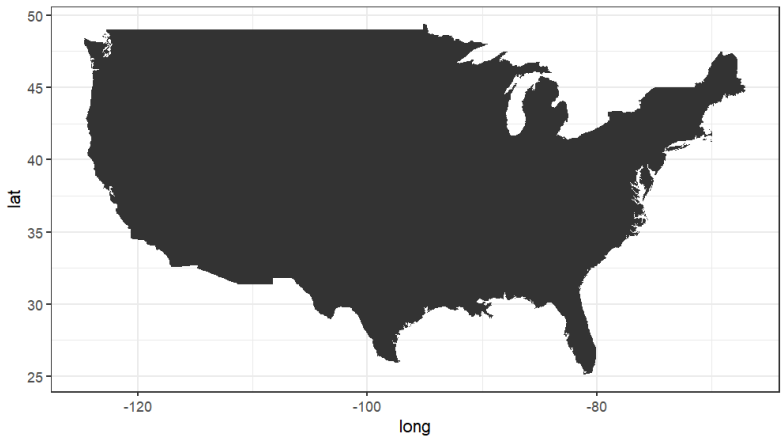
**NOTA:** Recordeu assignar primer a la variable de nom *usa*, un marc de dades (*dataframe*) de USA. Això podeu fer-ho gràcies a la funció `map_data` que heu vist més amunt:

```
> usa <- map_data("usa")
>ggplot(usa)+ aes(x=long, y = lat, group = group)+geom_polygon()+coord_quickmap()
```




**(b) Utilitzeu el clàssic layer de llum fosca per a `ggplot2` adequat per als mapes: `theme_set(theme_bw())`**

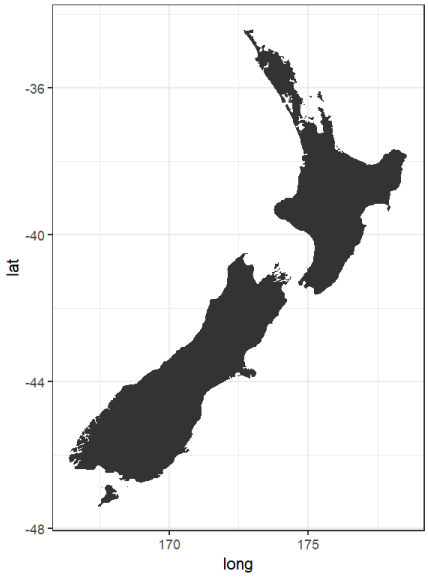
```
> ggplot(usa)+ aes(x=long, y = lat, group = group)+geom_polygon()+coord_quickmap()+theme_set(theme_bw())
```



**(c) Dibuixeu el mapa de Nova Zelanda (*nz*) anàlogament**

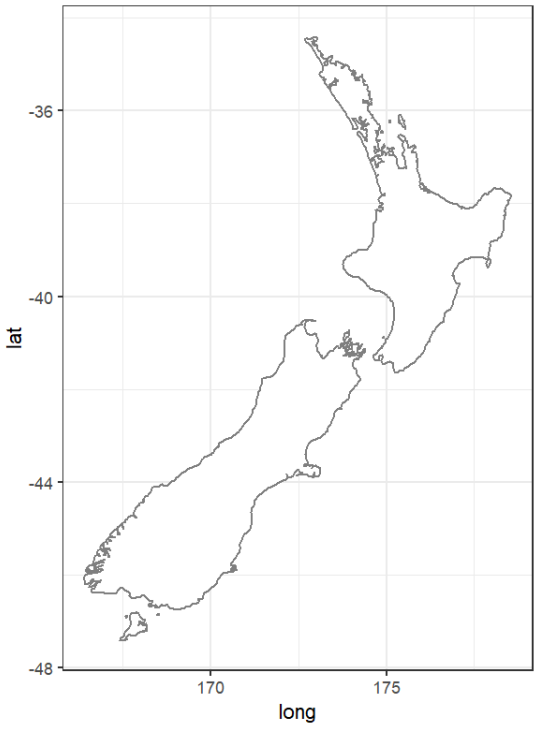
> nz<- **map_data("nz")**

> ggplot (**nz**) +aes(x=long,y=lat,group=group) + geom_polygon() + **coord_quickmap() + theme_set(theme_bw())**




**(d) Ara en canvi feu us de *`fill`* i *`colour`* que ja hem usat en altres geometries per omplir de blanc ("`white`") l'interior del mapa i posar els contorns en gris ("`grey50`")**

> ggplot (nz) +aes(x=long,y=lat,group=group) +geom_polygon**(fill="white", colour="grey50")**+coord_quickmap()



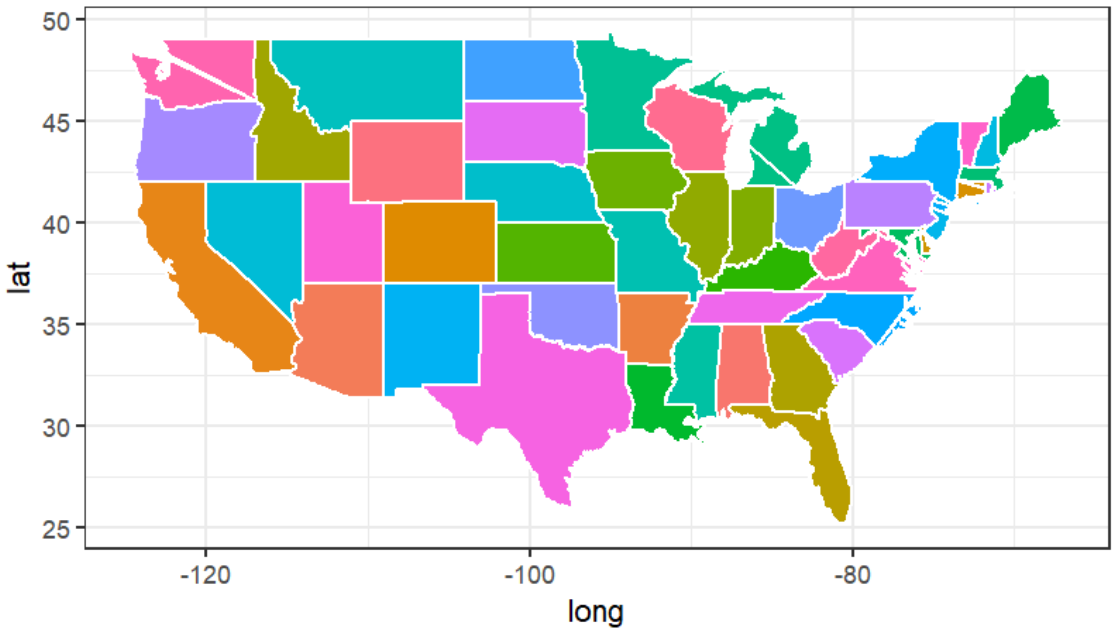
**2.- Fent ús de la funció, map_data, també podem obtenir un dataframe que ens indiqui les fronteres estatals del mapa 'usa':**

```
>  states <- map_data("state")
```

```
~/
> head(states)
       long      lat group order  region subregion
1 -87.46201 30.38968     1     1 alabama      <NA>
2 -87.48493 30.37249     1     2 alabama      <NA>
3 -87.52503 30.37249     1     3 alabama      <NA>
4 -87.53076 30.33239     1     4 alabama      <NA>
5 -87.57087 30.32665     1     5 alabama      <NA>
6 -87.58806 30.32665     1     6 alabama      <NA>
>
```

**Pinteu de colors els estats (atribut=*region*) i poseu les fronteres en blanc. Afegiu** `guides(fill=FALSE)` **per tal d'eliminar la llegenda.**
**NOTA: En algunes versions de R surt un *warning*. Si és el cas, useu:**
`guides(fill= "none")`

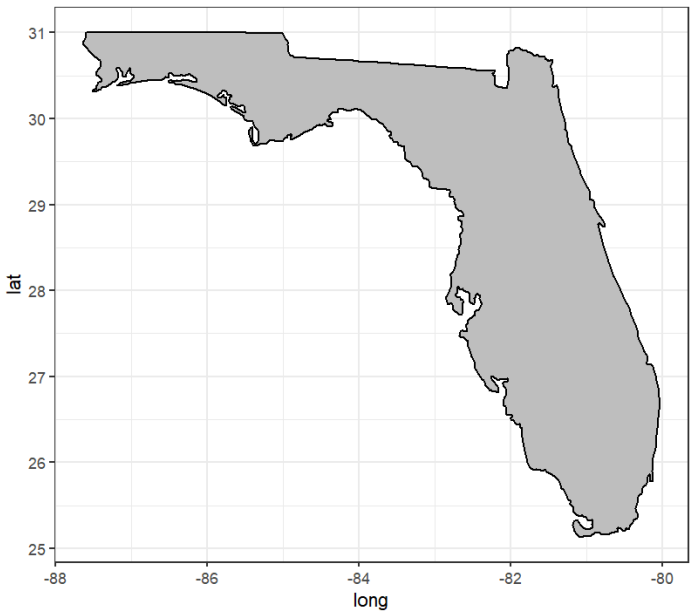
```
>  ggplot(data = states) + geom_polygon(aes(x = long, y = lat, fill =
region), color = "white")+coord_quickmap()+guides(fill=FALSE)
> ggplot(data = states) + geom_polygon(aes(x = long, y = lat, fill =
region), color = "white")+coord_quickmap()+guides(fill="none")
```




**3.- També podem crear un subconjunt amb la informació d' un estat (atribut='region') que ens interessi, fent servir:**
`fo_df <- subset(states, region == "florida") # estat de Florida`

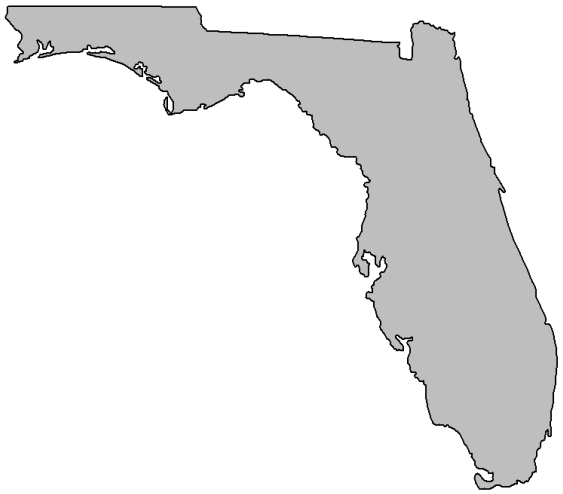
**(a) Dibuixeu el mapa de l'estat de Florida, pinteu l'interior en gris i el contorn en negre.**

```
>  ggplot(data = fo_df, mapping = aes(x = long, y = lat)) +
coord_quickmap()+ geom_polygon(color = "black", fill = "gray")
```




**(b) Afegiu l'argument `theme_nothing()`. Què us fa?**

```
> ggplot(data = fo_df, mapping = aes(x = long, y = lat)) +
coord_quickmap()+ geom_polygon(color = "black", fill =
"gray")+theme_nothing()
```



Treu els eixos i graella

**(c) Repetiu el procés amb California, i assigneu a una variable de nom 'CA_base' un ggplot d'un mapa com el de l'apartat anterior però de California (amb background buit). NOTA: Primer extraieu la informació de l'estat en un subconjunt com s'ha fet abans**

**Nota 2: Els noms de les regions van en minúscules**

```
> CA_df <- subset(states, region == "california")
```

```
> CA_base <- ggplot(data = CA_df, mapping = aes(x = long, y = lat,
group = group)) + coord_quickmap()+ geom_polygon(color = "black", fill
= "gray")+ theme_nothing()
> CA_base
```

**(d) Extraieu ara en un subconjunt els comptats de California (`"county"`):**

```
> counties <- map_data("county")
> CA_county <- subset(counties, region == "california")
```

```
> head(CA_county)
          long      lat group order     region subregion
6965 -121.4785 37.48290   157  6965 california   alameda
6966 -121.5129 37.48290   157  6966 california   alameda
6967 -121.8853 37.48290   157  6967 california   alameda
6968 -121.8968 37.46571   157  6968 california   alameda
6969 -121.9254 37.45998   157  6969 california   alameda
6970 -121.9483 37.47717   157  6970 california   alameda
>
```

**(e) Afegiu a la vostra variable CA_base creada en l'apartat anterior un `geom_polygon()` que contingui les dades del subconjunt CA_county, i el color de les fronteres entre comptats estigui blanc (deixeu `fill=NA`).**

```
> CA_base + geom_polygon(data = CA_county, fill = NA, color = "white")
```

Podeu tornar a situar la frontera estatal, fent:

```
> CA_base + geom_polygon(data = CA_county, fill = NA, colour =
"white") + geom_polygon(fill = NA, colour = "black")
```

**Podeu trobar altres paquets de R i llibreries per fer mapes i l'explicació del seu us a:** Chapter 9 Making maps with R | Geocomputation with R (geocompx.org), **i a:** https://www.earthdatascience.org/courses/earth-analytics/spatial-data-r/make-maps-with-ggplot-in-R/

**O aprofundir més seguint el material en que s'ha basat aquesta part de la classe:**

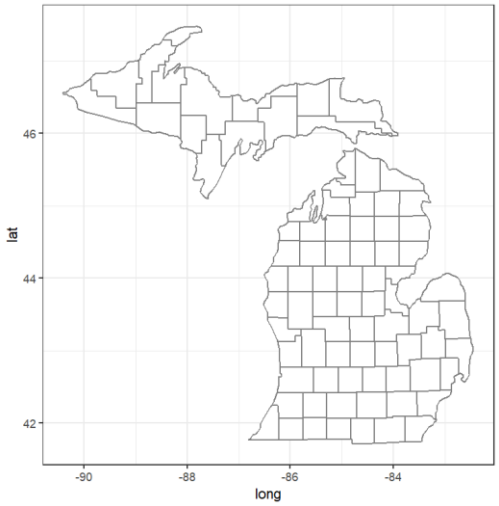
Making Maps With R · Reproducible Research. (eriqande.github.io)

**3.- En aquest exercici carreguem un mapa de Michigan que porta incorporats els contorns dels comptats de Michigan**

```
mi_counties <- map_data("county", "michigan")
```

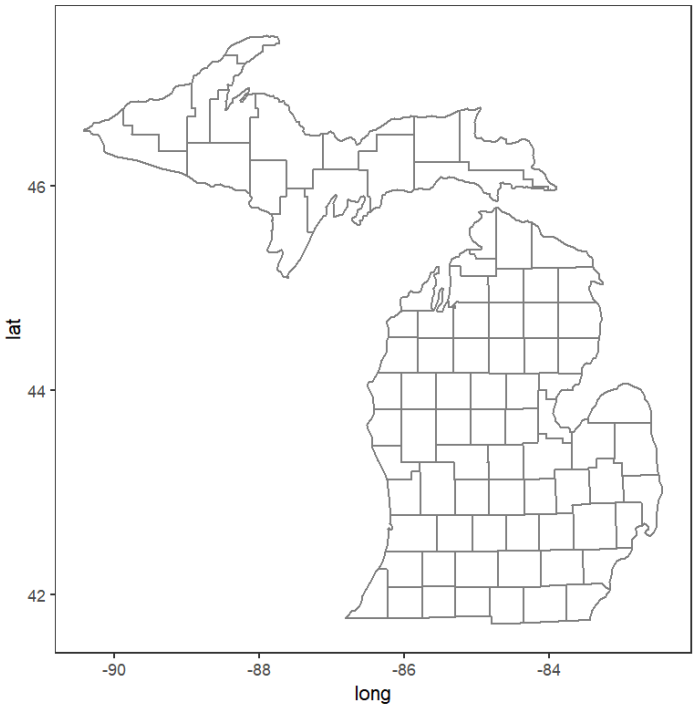
**a) Feu us de `geom_polygon()` per dibuixar el mapa. Poseu el fons del mapa blanc i el contorn en gris (`"grey50"`).**

```
> mich<- ggplot(mi_counties)+aes(long, lat, group = group) +geom_polygon(fill = "white", colour = "grey50") + coord_quickmap()
> mich
```




**b) Afegiu `theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())`**

```
> mich+theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())
```




## 5. PAQUET "SIMPLE FEATURES MAPS" (SF)

**Hi ha algunes limitacions amb l'enfocament descrit en els apartats 3 i 4 anteriors. La principal limitació és el fet que el format de dades simple "longitud-latitud" no**

**s'acostumen a utilitzar en els mapes del món real. Les dades vectorials per als mapes es codifiquen normalment mitjançant l'estàndard de "funcions simples" (sf, de les sigles en anglès) produït per l'Open Geospatial Consortium.**

**El paquet `sf` desenvolupat per Edzer Pebesma (https://github.com/r-spatial/sf) ofereix un conjunt d'eines excel·lent per treballar amb aquestes dades, i les funcions `geom_sf()` i `coord_sf()` de ggplot2 estan dissenyades per treballar conjuntament amb el paquet `sf`.**

**Per a trames simples, només necessitareu `geom_sf()` (una geometria inusual que dibuixarà diferents objectes geomètrics en funció de les sf que hi hagi a les dades: podeu obtenir punts, línies o polígons). Per afegir text i etiquetes, podeu utilitzar `geom_sf_text()` i `geom_sf_label()`, respectivament.**

**Hi ha tres maneres de proporcionar l'estètica de la geometria amb `geom_sf()` :**

1. **No feu res: per defecte `geom_sf()` assumeix que s'emmagatzema a la columna de geometria.**
2. **Passeu explícitament un objecte `sf` a l'argument de dades. Això utilitzarà la columna de geometria primària, sense importar com es digui.**
3. **Subministreu-ne una de vostra amb `aes(geometry = my_column)`**

**Si no s'especifica el sistema de referència de coordenades (CRS) en el qual s'han de projectar totes les dades abans de representar-les, utilitzarà el CRS definit a la primera capa `sf` de la trama.**

**`coord_sf()` garanteix que totes les capes utilitzen un CRS comú. Podeu especificar-lo mitjançant el paràmetre `crs`, o bé `coord_sf()`. Ho agafarà de la primera capa que defineix un CRS.**

**Per introduir aquestes funcions, anem a utilitzar el paquet `ozmaps` de Michael Summer (https://github.com/mdsumner/ozmaps/ ). Són dades que es poden dibuixar o accedir directament com a objectes de característiques simples. El paquet inclou funcions senzilles per a mapes de països o estats d'Austràlia i conjunts de dades de regions administratives de l'Oficina d'Estadística d'Austràlia <https://www.abs.gov.au/>. Les capes inclouen les divisions electorals i àrees de govern local, etc. Tot i que totes elles estan simplificades a partir de les fonts originals, contenen suficients detall per permetre la cartografia d'un municipi local.**

**Per utilitzar el paquet ozmaps anem a carregar-lo. Carregarem també el paquet sf i cridarem ambdues llibreries, `ozmaps` i `sf`:**

```
> install.packages("ozmaps")
```

**# la versió actualitzada de *github***

```
> devtools::install_github("mdsumner/ozmaps")
```
**#cliqueu 1 quan us demani**

**# Ignoreu els warnings de carregar llibreries.**

```
> install.packages("sf")
```

**Ara que tenim els paquets instal·lats, carreguem les llibreries:**

```
> library(ozmaps)
> library(sf)
```

**Assigneu:**

```
> oz_states <- ozmaps::ozmap_states
```

**Verifiqueu que teniu la *sf*:**

```
> oz_states
Simple feature collection with 9 features and 1 field
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: 105.5507 ymin: -43.63203 xmax: 167.9969 ymax: -9.229287
Geodetic CRS:  GDA94
# A tibble: 9 x 2
  NAME                                                        geometry
* <chr>                                                  <MULTIPOLYGON [°]>
1 New South Wales              (((150.7016 -35.12286, 150.6611 -35.117~
2 Victoria                     (((146.6196 -38.70196, 146.6721 -38.702~
3 Queensland                   (((148.8473 -20.3457, 148.8722 -20.3757~
4 South Australia              (((137.3481 -34.48242, 137.3749 -34.468~
5 Western Australia            (((126.3868 -14.01168, 126.3625 -13.982~
6 Tasmania                     (((147.8397 -40.29844, 147.8902 -40.302~
7 Northern Territory           (((136.3669 -13.84237, 136.3339 -13.839~
8 Australian Capital Territory (((149.2317 -35.222, 149.2346 -35.24047~
9 Other Territories            (((167.9333 -29.05421, 167.9188 -29.034~
>
```

**1.- Donades les dades en aquest format, podem utilitzar `geom_sf()` i `coord_sf()` per dibuixar un mapa útil sense especificar cap paràmetre o fins i tot, sense necessitat de declarar explícitament cap estètica/mapeig.**

**(a) Dibuixeu el mapa de oz_states tot traient la graella del fons. Cal fer ús de `coord_sf()`? Si la resposta és negativa argumenta per què?**

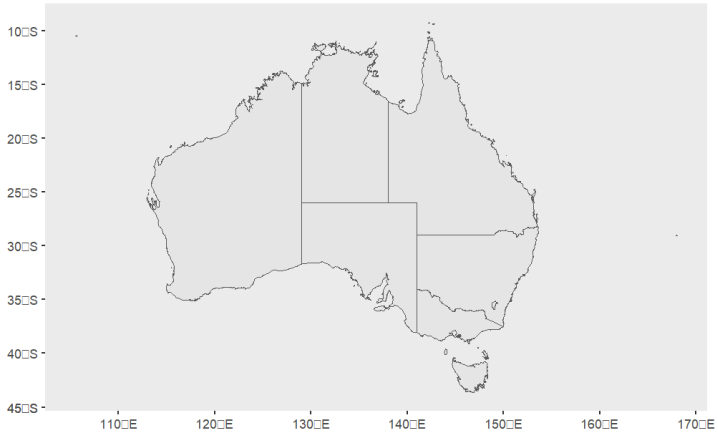
Si ens fixem oz_states utilitza el sistema de coordenades CRS

```
> oz_states
Simple feature collection with 9 features and 1 field
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: 105.5507 ymin: -43.63203 xmax: 167.9969 ymax: -9.229287
Geodetic CRS:  GDA94
# A tibble: 9 x 2
  NAME                                                        geometry
* <chr>                                                  <MULTIPOLYGON [°]>
1 New South Wales              (((150.7016 -35.12286, 150.6611 -35.117~
2 Victoria                     (((146.6196 -38.70196, 146.6721 -38.702~
3 Queensland                   (((148.8473 -20.3457, 148.8722 -20.3757~
4 South Australia              (((137.3481 -34.48242, 137.3749 -34.468~
5 Western Australia            (((126.3868 -14.01168, 126.3625 -13.982~
6 Tasmania                     (((147.8397 -40.29844, 147.8902 -40.302~
7 Northern Territory           (((136.3669 -13.84237, 136.3339 -13.839~
8 Australian Capital Territory (((149.2317 -35.222, 149.2346 -35.24047~
9 Other Territories            (((167.9333 -29.05421, 167.9188 -29.034~
>
```

Com hem dit abans: “Si no s'especifica el sistema de referència de coordenades (CRS) en el qual s'han de projectar totes les dades abans de representar-les, utilitzarà el CRS definit a la primera capa `sf` de la trama.” Per tant, afegir o no el `coord_sf` no varia, en aquest cas, la visualització :

```
>      ggplot(oz_states)+geom_sf()+theme(panel.grid.major      =
element_blank(), panel.grid.minor = element_blank())
```

```
>   ggplot(oz_states)+geom_sf()+coord_sf()+theme(panel.grid.major
= element_blank(), panel.grid.minor = element_blank())
```




**(b) En segon lloc, extraieu els límits electorals d'una forma simplificada utilitzant la funció `ms_simplify()` del paquet `rmapshaper`. Nota: Instal·leu primer el paquet `rmapshaper`. Aquí teniu més informació d'aquest paquet: [https://cran.r-project.org/web/packages/rmapshaper/rmapshaper.pdf](https://cran.r-project.org/web/packages/rmapshaper/rmapshaper.pdf)**

**La funció ms_simplify ens permet simplificar polígons o línies.**

**Per fer aquest apartat, us avancem la resposta, simplement heu de fer:** `oz_votes <- rmapshaper::ms_simplify(ozmaps::abs_ced)`

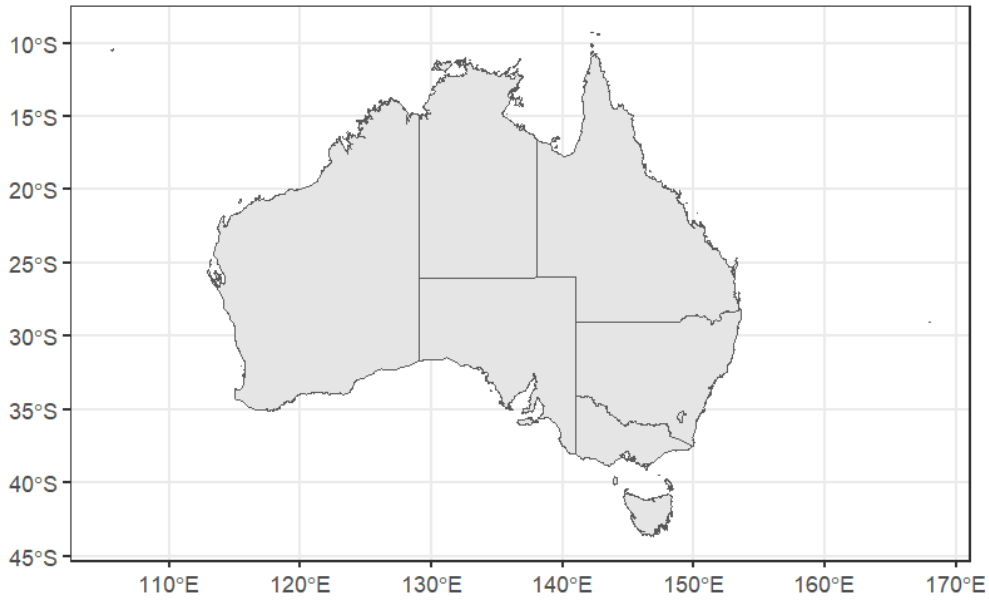
**Compareu** `ozmaps::abs_ced` **i** `oz_votes`**. Què ha simplificat** `ms_simplify`**?**

```
> ozmaps::abs_ced
Simple feature collection with 151 features and 1 field
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: 96.81703 ymin: -43.65856 xmax: 167.9969 ymax: -9.219937
Geodetic CRS:  GDA94
First 10 features:
        NAME                    geometry
1      Banks MULTIPOLYGON (((151.0156 -3...
2     Barton MULTIPOLYGON (((151.1681 -3...
3  Bennelong MULTIPOLYGON (((151.0511 -3...
4    Berowra MULTIPOLYGON (((151.1786 -3...
5   Blaxland MULTIPOLYGON (((151.0301 -3...
6  Bradfield MULTIPOLYGON (((151.1001 -3...
7     Calare MULTIPOLYGON (((149.1198 -3...
8    Chifley MULTIPOLYGON (((150.8666 -3...
9       Cook MULTIPOLYGON (((151.1548 -3...
10    Cowper MULTIPOLYGON (((153.1503 -3...
> oz_votes
Simple feature collection with 151 features and 1 field
Geometry type: GEOMETRY
Dimension:     XY
Bounding box:  xmin: 105.5507 ymin: -43.63203 xmax: 153.6299 ymax: -9.229287
Geodetic CRS:  GDA94
First 10 features:
        NAME                    geometry
1      Banks POLYGON ((151.0156 -33.9820...
2     Barton POLYGON ((151.1681 -33.9201...
3  Bennelong POLYGON ((151.0511 -33.8242...
4    Berowra POLYGON ((151.1597 -33.6605...
5   Blaxland POLYGON ((151.0301 -33.8351...
6  Bradfield POLYGON ((151.1001 -33.7535...
7     Calare POLYGON ((149.1198 -33.8878...
8    Chifley POLYGON ((150.8666 -33.6525...
9       Cook POLYGON ((151.1548 -33.9666...
10    Cowper POLYGON ((153.1503 -30.2398...
>
```

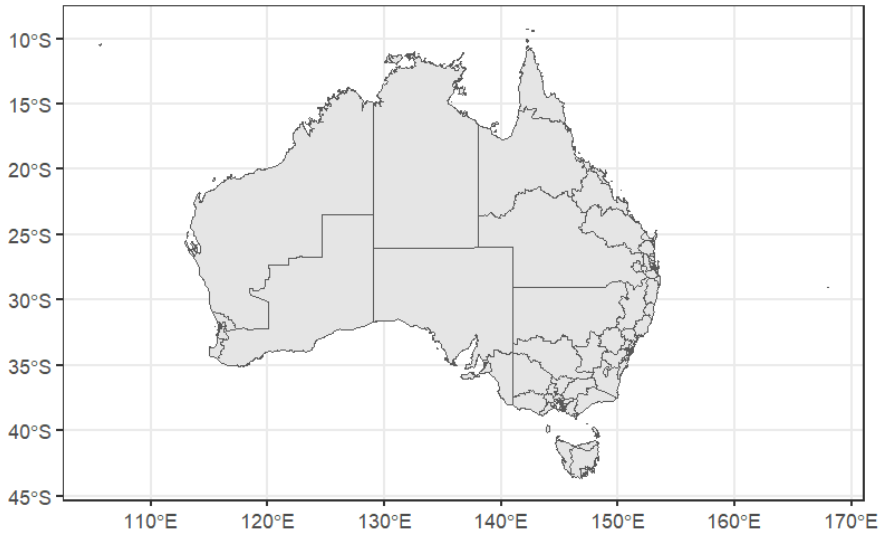
Ens ha simplificat clarament els polígons.

**(c) Ara que teniu ambdós conjunts de dades per representar les fronteres estatals (oz_states) i electorals (oz_votes) en un mateix mapa, afegiu dues capes `geom_sf()`, una per les dades contingudes en oz_states i l'altra per les dades contingudes en oz_votes**

```
> ggplot() + geom_sf(data = oz_states)
```



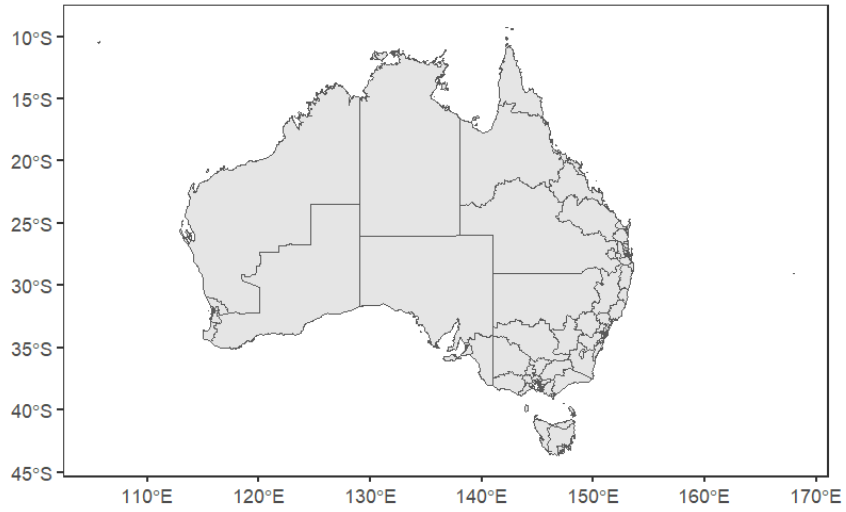
```
> vots<- ggplot() + geom_sf(data = oz_states) +geom_sf(data = oz_votes)
```




**Nota: Fixeu-vos no cal `coord_sf()`**

```
> vots+coord_sf()
```

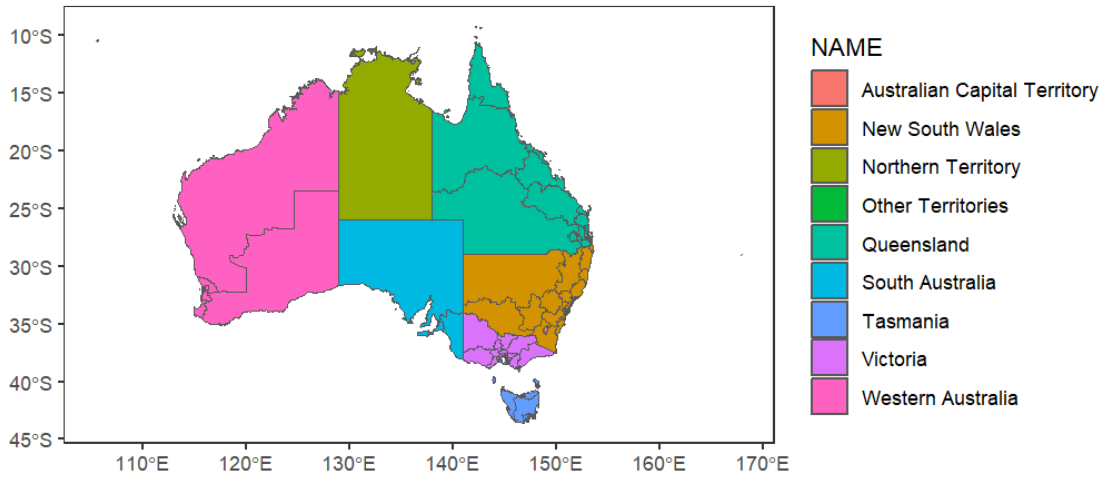
```
> vots + theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())
```




**(d) Afegiu l'estètica necessària per pintar l'interior (`fill`) dels estats segons el seu nom (NAME).**

**NOTA: Per evitar el conflicte amb la informació de `oz_votes`, poseu en la capa `geom_sf()` de `oz_votes`: `geom_sf(data = oz_votes, fill=NA)`.**

```
> ggplot() + geom_sf(data = oz_states, aes(fill=NAME))
+geom_sf(data = oz_votes, fill=NA) +
coord_sf()+theme(panel.grid.major = element_blank(),
panel.grid.minor = element_blank())
```




**IMPORTANT: En aquest cas, la variable NAME és una variable categòrica i no transmet cap informació addicional, però el mateix enfocament es podria utilitzar per visualitzar altres tipus de metadates d'àrea. Per exemple, si oz_states tingués una columna addicional que especifiques el nivell de persones empleades o en atur a cada estat, podríem assignar l'estètica de de color a aquesta variable.**

**2.- En aquest exercici anem a dibuixar un mapa electoral de Sydney partint del mapa que tenim. Per això hem de fer tres passos:**

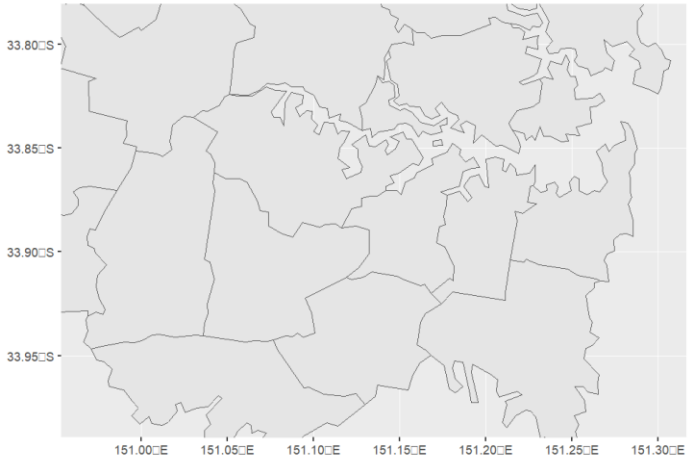
**(a) Primer, hem d'extreure les dades del mapa pels electors rellevants. Copieu el següent filtratge:**

```
> sydney_map <- ozmaps::abs_ced %>% filter(NAME %in% c("Sydney", "Wentworth", "Warringah", "Kingsford Smith", "Grayndler", "Lowe", "North Sydney", "Barton", "Bradfield", "Banks", "Blaxland", "Reid", "Watson", "Fowler", "Werriwa", "Prospect", "Parramatta", "Bennelong", "Mackellar", "Greenway", "Mitchell", "Chifley", "McMahon"))
```

NOTA: Recordeu carregar la llibreria necessària per `filter` si no ho heu fet

**(b) Després hem de buscar la regió que ens interessa, Sydney. Per això, simplement feu zoom a la regió de Sydney especificant `xlim = c(150.97, 151.3), ylim = c(-33.98, -33.79)` a `coord_sf()`**

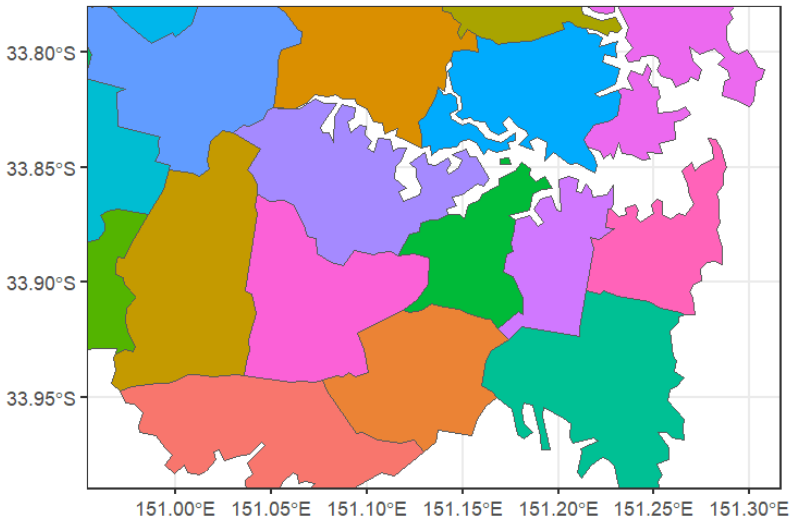
```
> ggplot(sydney_map) +geom_sf() + coord_sf(xlim = c(150.97, 151.3), ylim = c(-33.98, -33.79))
```




**(c) Tot utilitzant la regió de Sydney que heu creat en l'apartat (b), pinteu cada subregió segons el seu NAME, poseu dins del `geom_sf` l'argument `show.legend = FALSE`, per amagar la llegenda**

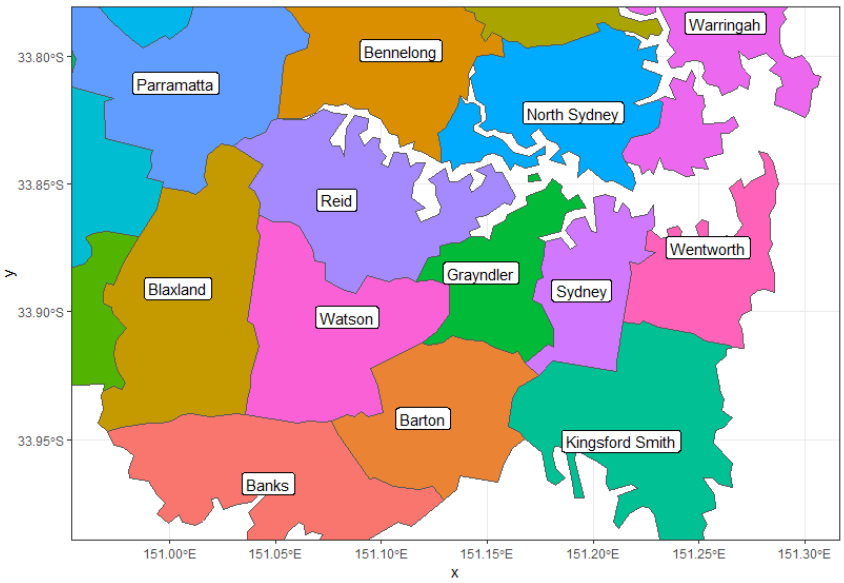
```
> syd <- ggplot(sydney_map) +geom_sf(aes(fill = NAME), show.legend = FALSE) + coord_sf(xlim = c(150.97, 151.3), ylim = c(-33.98, -33.79))
```

```
> syd
```




**(d) Finalment, utilitzeu `geom_sf_label()` per superposar les etiquetes amb el nom de cada electorat (poseu l'estètica `label=NAME`)**

```
> syd +geom_sf_label(aes(label = NAME))
```




**3.- Ens podria interessar per exemple mostrar les ubicacions de les capitals australianes al mapa anterior.**

**Suposem que tenim les dades de les ciutats junt a la seva latitud i longitud. Afegiu una altra geometria, per exemple, `geom_point()` que mostri aquesta informació.**

**(a) Construïu el següent dataframe fent una `tibble`, i assigneu-lo a una variable de nom oz_capitals. Se us dona el codi :**

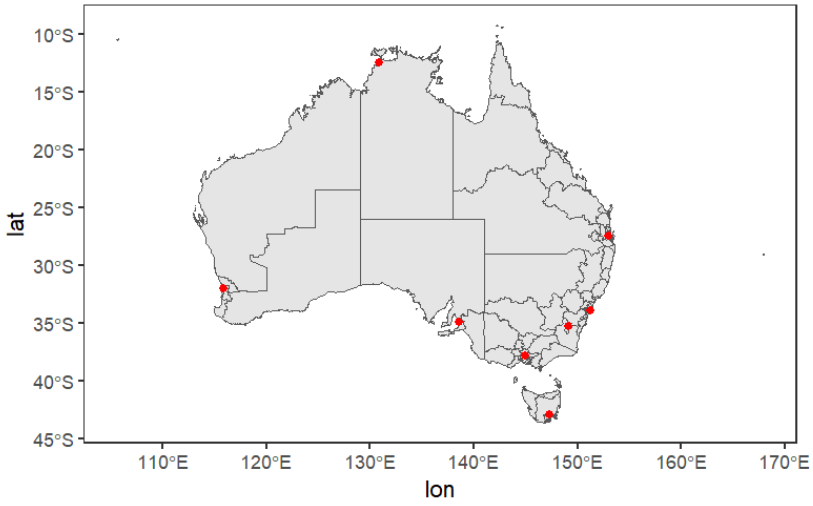
```
> oz_capitals <- tibble::tribble(
  ~city,  ~lat,     ~lon,
  "Sydney",  -33.8688, 151.2093,
  "Melbourne",-37.8136, 144.9631,
  "Brisbane", -27.4698, 153.0251,
  "Adelaide", -34.9285, 138.6007,
  "Perth",    -31.9505, 115.8605,
  "Hobart",   -42.8821, 147.3272,
  "Canberra", -35.2809, 149.1300,
  "Darwin",   -12.4634, 130.8456,
)
```

```
> oz_capitals
# A tibble: 8 x 3
  city        lat   lon
  <chr>     <db1> <db1>
1 Sydney    -33.9  151.
2 Melbourne -37.8  145.
3 Brisbane  -27.5  153.
4 Adelaide  -34.9  139.
5 Perth     -32.0  116.
6 Hobart    -42.9  147.
7 Canberra  -35.3  149.
8 Darwin    -12.5  131.
>
```

**(b) Afegiu la informació del dataframe amb punts de color vermell al mapa d'Australia que contenia les fronteres estatals (oz_states) i electorals (oz_votes)**

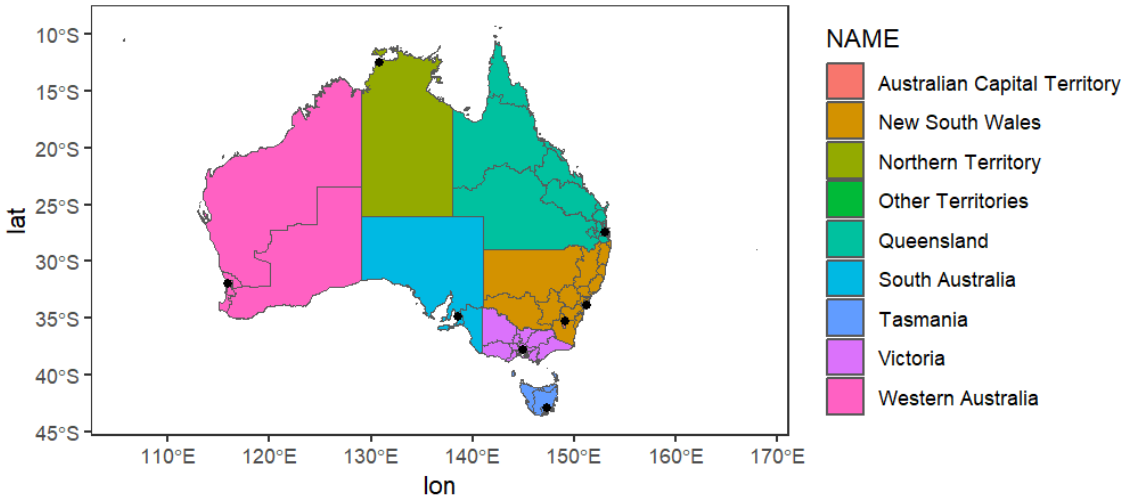
Seguint el que hem fet en l'exercici 1 d'aquesta secció i afegint `geom_point()`

```
> ggplot() + geom_sf(data = oz_states) +geom_sf(data = oz_votes) + geom_point(data = oz_capitals, mapping = aes(x = lon, y = lat), colour = "red")+coord_sf()+theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank())
```




O inclús:

```
>ggplot() + geom_sf(data = oz_states, aes(fill=NAME))
+geom_sf(data = oz_votes, fill=NA) + geom_point(data =
oz_capitals, mapping = aes(x = lon, y = lat), colour =
"black")+coord_sf()+theme(panel.grid.major = element_blank(),
panel.grid.minor = element_blank())
```




**4.- Per aquest exercici anem a fer servir un arxiu georeferenciat amb les comunes de la Ciutat Autònoma de Buenos Aires. L'arxiu està disponible online en format *geojson*, un format estàndard de representació de dades geogràfiques que és fàcil d'usar**

```
> Barris <-
st_read('https://bitsandbricks.github.io/data/CABA_barrios.geojson')
```

```
> head (Barris)
Simple feature collection with 6 features and 4 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -58.50617 ymin: -34.63064 xmax: -58.41192 ymax: -34.57829
Geodetic CRS:  WGS 84
            BARRIO COMUNA PERIMETRO    AREA                       geometry
1        CHACARITA     15  7725.695 3118101 POLYGON ((-58.45282 -34.595...
2         PATERNAL     15  7087.513 2229829 POLYGON ((-58.46558 -34.596...
3     VILLA CRESPO     15  8132.699 3613584 POLYGON ((-58.42375 -34.597...
4 VILLA DEL PARQUE     11  7705.390 3399596 POLYGON ((-58.49461 -34.614...
5          ALMAGRO      5  8537.901 4050752 POLYGON ((-58.41287 -34.614...
6        CABALLITO      6 10990.964 6851029 POLYGON ((-58.43061 -34.607...
>
```

**Familiaritzeu-vos amb les variables**

**El dataframe té 48 files i 5 columnes. Una fila per barri i una columna per cada variable disponible (Comuna- unitat administrativa, perímetre i àrea que ens informen sobre les dimensions del polígon cobert per cada barri, i 'geometry' de tipus espacial i que conté les dades amb les seves coordenades geogràfiques)**

**Carreguem les dades de la població en cada barri:** `poblacio <- read.csv("https://bitsandbricks.github.io/data/caba_pob_barrios_2010.csv")`

**Familiaritzeu-vos també amb les variables**

```
> head(poblacio)
     BARRIO POBLACION
1 AGRONOMIA      13912
2   ALMAGRO     131699
3 BALVANERA     138926
4  BARRACAS      89452
5  BELGRANO     126267
6      BOCA      45113
>
```

tenim el nom de cada barri i la seva població.

**D'altra banda, anem a carregar dades provinents dels registres del Sistema Únic d'Atenció Ciutadana (o SUACI), una plataforma del Govern de la Ciutat que administra els contactes iniciats per ciutadans. Al portal de dades obertes de la Ciutat es publica cada un dels contactes rebuts per SUACI any a any.**

```
> suaci2018 <- read.csv('https://bitsandbricks.github.io/data/gcba_suaci_2018.csv')
```

```
> head(suaci2018)
     BARRIO CONTACTOS
1 AGRONOMIA      7378
2   ALMAGRO     31420
3 BALVANERA     28616
4  BARRACAS     23106
5  BELGRANO     46936
6      BOCA     11495
>
```

**Familiaritzeu-vos amb les noves variables altre cop**

**Veiem que la taula té 48 files (una per cada barri de la ciutat) i 2 columnes (una amb el nom de barri, i una altra amb la quantitat total de contactes registrats el 2018).**

**Com el nostre dataframe de contactes a SUACI i poblacio tenen una columna amb el mateix nom que conté les mateixes categories, sumar la informació de població a suaci2018 és tan fàcil com usar la funció de *dplyr* `left_join`:** `new <- left_join(suaci2018, poblacio)`

```
> head(new)
     BARRIO CONTACTOS POBLACION
1 AGRONOMIA       7378      13912
2   ALMAGRO      31420     131699
3 BALVANERA      28616     138926
4  BARRACAS      23106      89452
5  BELGRANO      46936     126267
6      BOCA      11495      45113
>
```

**(a) Sumeu la informació de Barris per a que us quedi:**

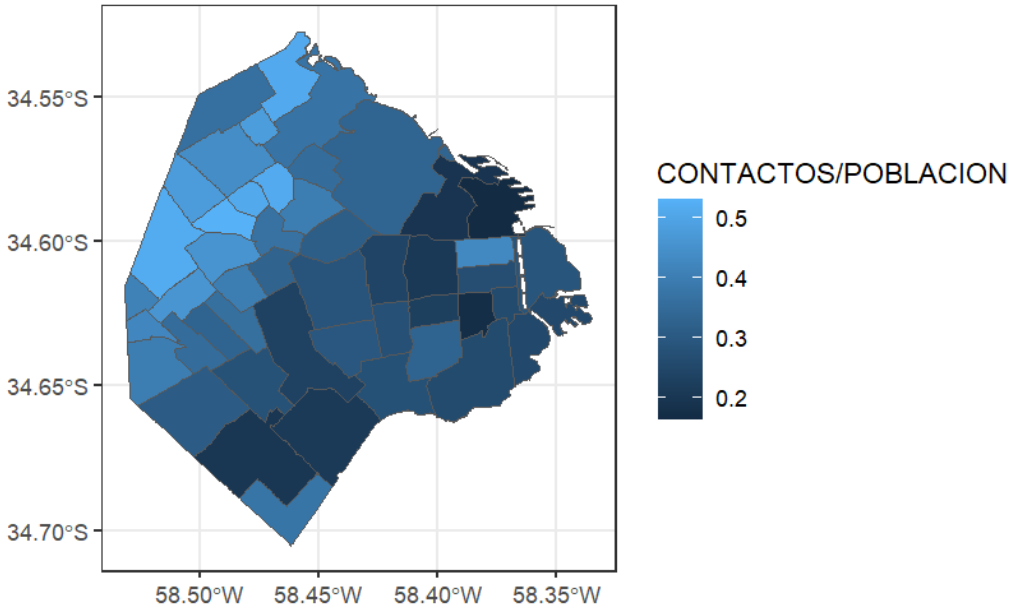
```
Simple feature collection with 6 features and 6 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -58.50617 ymin: -34.63064 xmax: -58.41192 ymax: -34.57829
Geodetic CRS:  WGS 84
            BARRIO COMUNA PERIMETRO    AREA CONTACTOS POBLACION
1        CHACARITA     15  7725.695 3118101     11076     27761
2         PATERNAL     15  7087.513 2229829      7255     19717
3     VILLA CRESPO     15  8132.699 3613584     25748     81959
4 VILLA DEL PARQUE     11  7705.390 3399596     25224     55273
5          ALMAGRO      5  8537.901 4050752     31420    131699
6        CABALLITO      6 10990.964 6851029     50301    176076
                       geometry
1 POLYGON ((-58.45282 -34.595...
2 POLYGON ((-58.46558 -34.596...
3 POLYGON ((-58.42375 -34.597...
4 POLYGON ((-58.49461 -34.614...
5 POLYGON ((-58.41287 -34.614...
6 POLYGON ((-58.43061 -34.607...
>
```

La comanda seria molt semblant a l'anterior:

```
> new <- left_join(Barris, new)
```

**(b) Feu una comanda com la que heu fet abans per dibuixar el mapa amb `geom_sf`. Però ara les dades són el vostre nou dataframe creat en l'apartat (a). En aquest apartat se us demana dibuixar un mapa amb els barris de la ciutat que mostri la quantitat de sol·licituds de la ciutadania per càpita. És a dir que el gràfic estigui pintat segons la ratio entre contactes i població (contactes/població). PISTA: Feu un mapeig pintant el gràfic segons la relació desitjada**

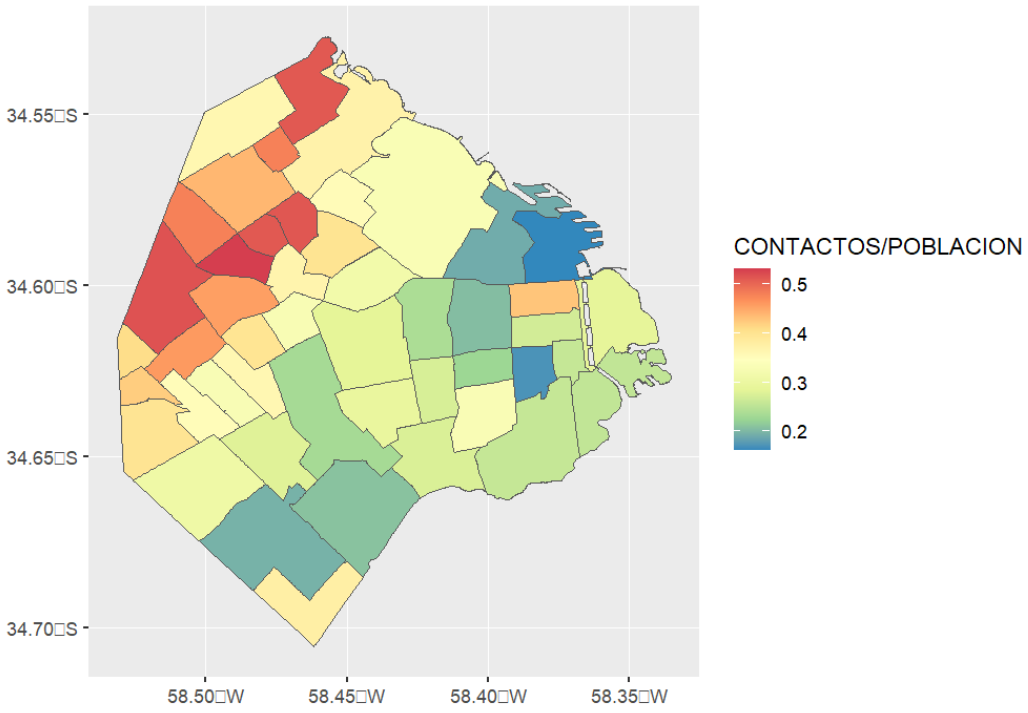
```
> ggplot(new) +geom_sf(aes(fill = CONTACTOS / POBLACION))+coord_sf()
```




**(c) Finalment, definirem la paleta de colors a utilitzar en l'estètica fill. Per exemple, podeu triar una escala anomenada "Spectral", que va de el blau al vermell i és molt usada quan es vol ressaltar la divergència d'una variable (és una possibilitat però heu vist més sobre el bon us dels colors a la classe de Colors d'en Guillermo).**

**Per fer aquest canvi afegiu a la vostra comanda de l'exercici (b) l'escala amb:** `scale_****_distiller(palette = "Spectral").` **Trieu adequadament** `****`

```
> ggplot(new) + geom_sf(aes(fill = CONTACTOS / POBLACION))+coord_sf()+scale_fill_distiller(palette = "Spectral")
```



# 6. EXTRA: (MINI) Introducció a mapes interactius

**Mapview proporciona funcions per crear de manera molt ràpida i convenient visualitzacions interactives de dades espacials. El seu objectiu principal és omplir el buit de representació interactiva ràpida (no de qualitat de presentació) per examinar i investigar visualment tant els aspectes de les dades espacials, les geometries com els seus atributs. També es pot considerar una API dirigida per dades per al paquet `leaflet` ja que renderitzarà automàticament els tipus de mapes correctes, segons el tipus de dades (punts, línies, polígons). A més, fa ús de funcionalitats de representació avançades que permeten visualitzar dades molt més grans del que és possible amb `leaflet`.**

**Finalment, podeu usar `st_as_sf` per convertir un objecte que vosaltres creeu a un objecte sf. I `st_crs`, per recuperar el sistema de referència de coordenades d'un objecte sf o sfc i estableix o substitueix el sistema de referència de coordenades recuperat de l'objecte.**

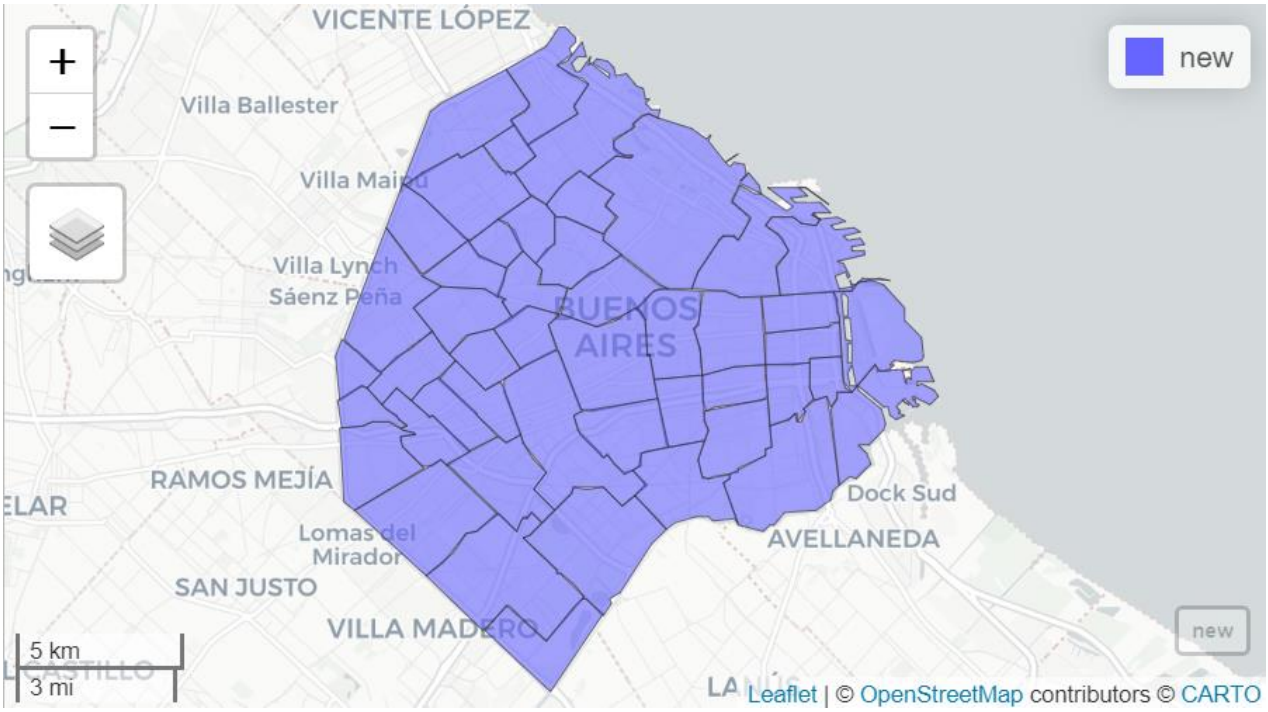
**Instal·leu el paquet necessari i carregueu la llibreria**

```
> install.packages("mapview")
```

```
>library(mapview)
```

**Utilitzeu mapview amb les dades sf que heu creat en l'últim exercici.**

```
mapview(new)
```




**Podeu crear els vostres propis mapes interactius. Copieu per exemple:**

```
d <- data.frame(
    place = c("London", "Paris", "Madrid", "Rome"),
```

```
    long = c(-0.118092, 2.349014, -3.703339, 12.496366),
    lat = c(51.509865, 48.864716, 40.416729, 41.902782),
    value = c(200, 300, 400, 600))
dsf <- st_as_sf(d, coords = c("long", "lat"))
st_crs(dsf) <- 4326
mapview(dsf)
```


**Busca la longitud i latitud de Barcelona i afegeix les dades al mapa per value =500**

```
d <- data.frame(
    place = c("London", "Paris", "Madrid", "Barcelona", "Rome"),
    long = c(-0.118092, 2.349014, -3.703339, 2.15899, 12.496366),
    lat = c(51.509865, 48.864716, 40.416729, 41.38879, 41.902782),
    value = c(200, 300, 400, 500, 600))
dsf <- st_as_sf(d, coords = c("long", "lat"))
st_crs(dsf) <- 4326
mapview(dsf)
```


Judit Chamorro Servent
Bellaterra, Abril 2025

# SEMINARI 9.

# *Interacció i Animació en R* (Respostes)

## 1. OBJECTIUS

**Conèixer i utilitzar eines d'interacció i animació en R com `plotly`, `gganimate`, `shiny` per a crear entorns interactius i d'animació en Visualització de Dades.**

## 2. PART 1. Entorns de Navegació

**En aquesta primera part crearem entorns interactius de navegació amb plotly, a partir de `ggplot()` o directament amb `plot_ly()`. Treballarem amb el data set `Poblacio_Mundial_1960-2022.csv` de la població mundial des del 1960 al 2022 segmentat per país i continent.**

**Abans de fer els exercicis, dos passos previs:**

**1. Carregar les llibreries que utilitzarem.**

```
> library(tidyverse)
> install.packages('plotly') # Càrrega paquet plotly (1a vegada)
> library(plotly)
>
```

**2. Llegir el dataset.**

```
> setwd("C:/Users/enric/Documents/R")
> PWorld <- read.csv('./Poblacio_Mundial_1960-2022.csv')
> str(PWorld)
>
```

```
> setwd("C:/Users/enric/R")
> PWorld <- read.csv('./Poblacio_Mundial_1960-2022.csv')
> str(PWorld)
'data.frame':   16695 obs. of  4 variables:
 $ Pais.Nom : chr  "Aruba" "Aruba" "Aruba" "Aruba" ...
 $ Continent: chr  "America" "America" "America" "America" ...
 $ Any      : int  1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 ...
 $ Poblacio : num  54608 55811 56682 57475 58178 ...
>
```

`Variables`:

- `Pais.Nom`: **Nom del país observat o nom de la zona geogràfica que agrupa varis països.**
- `Continent`: **Continent del país. En alguns registres agafa el valor "Zone" referint-se a una zona geogràfica.**
- `Any`: **Any de l'observació**
- `Poblacio`: **Nombre d'habitants del país i any de l'observació**

## EXERCICIS:

**1.- Mostra el codi per a definir una gràfica interactiva de barres i una gràfica interactiva de línies de l'evolució temporal de la població en els cinc continents, excloent les zones geogràfiques o polítiques(eliminar registres amb `Continent="Zone"`).**

**RESPOSTA:**

**PAS 1:** DATA MASSAGING. Filtrar (eliminar) zones geogràfiques i posteriorment agrupar per continents i sumar les poblacions:

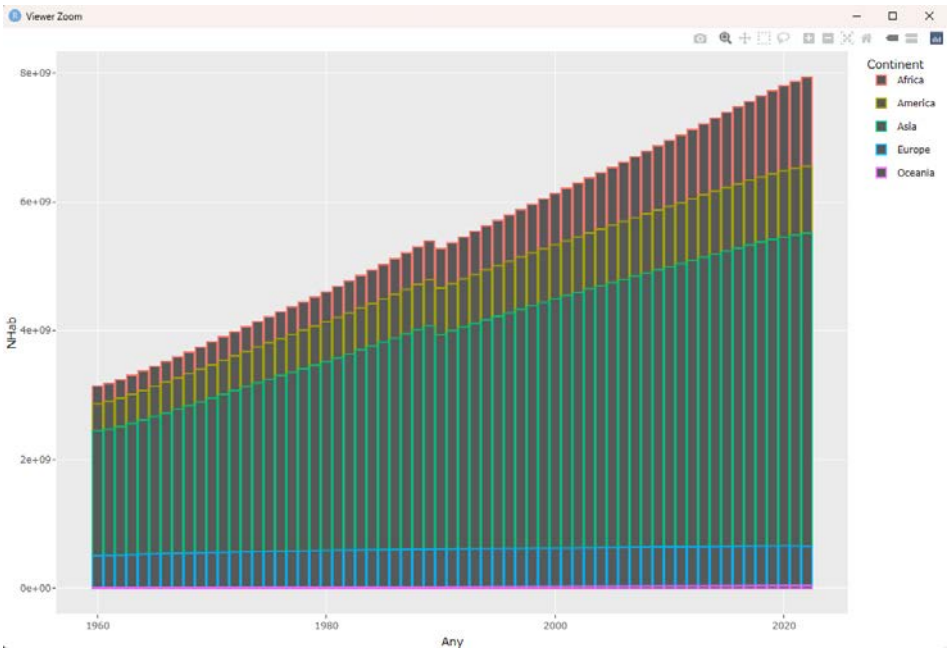
```
> PMundial <- PWorld %>% filter(Continent != "Zone") %>% group_by(Any,Continent) %>% summarize(NHab = sum(Poblacio))
> PMundial
```

```
> PMundial <- PWorld %>% filter(Continent != "Zone") %>% group_by(Any,Continent) %>% summarize(NHab=sum(Poblacio))
`summarise()` has grouped output by 'Any'. You can override using the `.groups` argument.
> PMundial
# A tibble: 315 × 3
# Groups:   Any [63]
     Any Continent       NHab
   <int> <chr>          <dbl>
 1  1960 Africa     276157583
 2  1960 America    422093874
 3  1960 Asia      1934891710
 4  1960 Europe     493657644
 5  1960 Oceania     15466870
 6  1961 Africa     282778774
 7  1961 America    431722162
 8  1961 Asia      1955905117
 9  1961 Europe     498333017
10  1961 Oceania     15795468
# i 305 more rows
# i Use `print(n = ...)` to see more rows
>
```

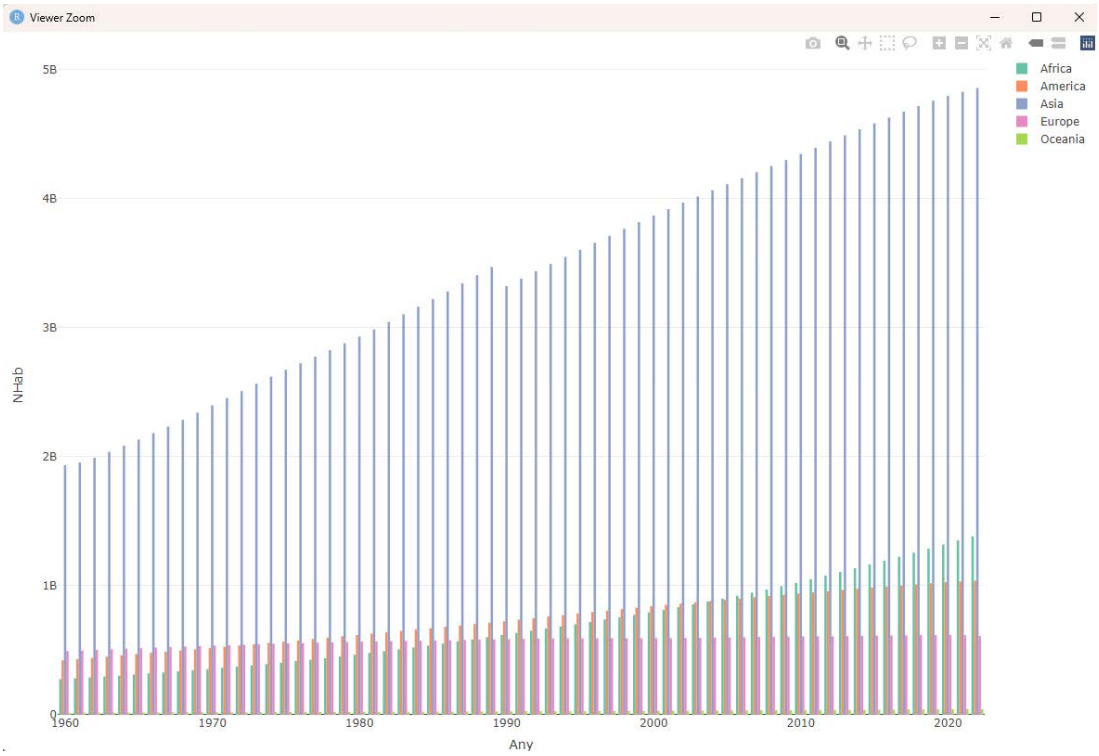
**PAS 2:** Pot haver diferents formes de visualitzar:

**DIAGRAMA DE BARRES INTERACTIVA:**

```
# DIAGRAMA AMB BARRES AMB ggplotly
> gplotPMundial<-ggplot(PMundial, aes(x=Any, y=NHab, color=Continent))
+ geom_col()
> ggplotly(gplotPMundial)
```



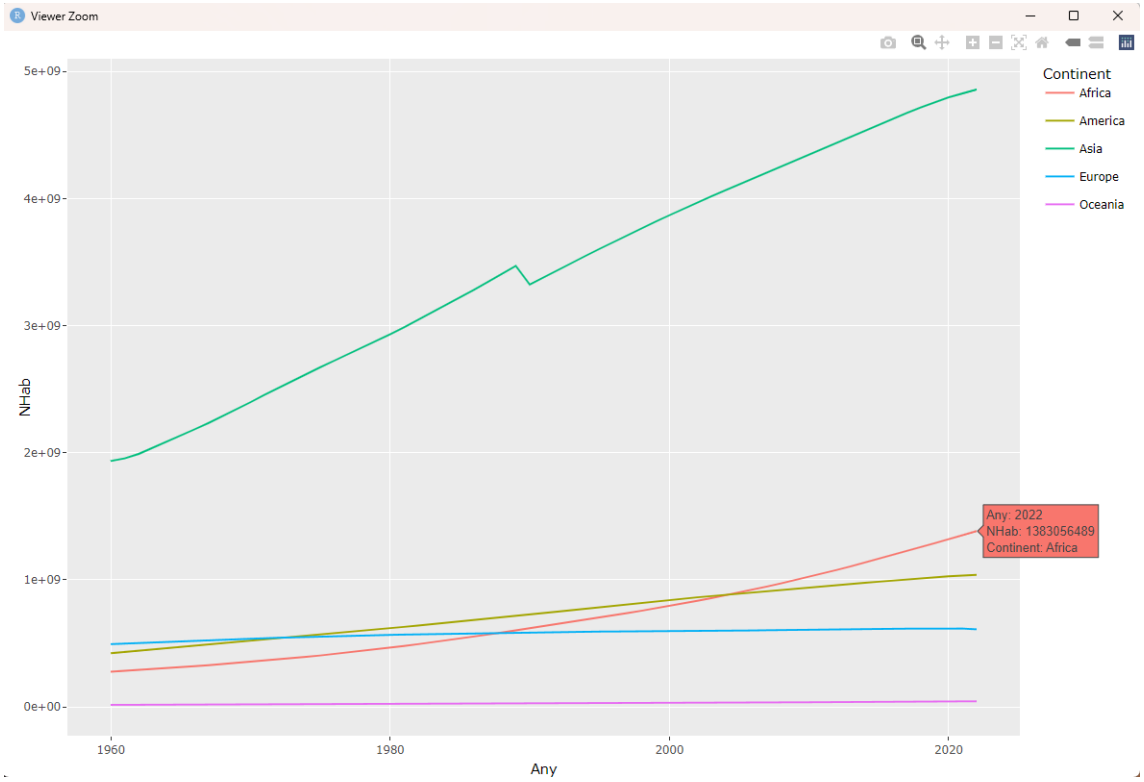

```
# DIAGRAMA AMB BARRES AMB plotly
> plot_ly(PMundial,x=~Any, y=~NHab, color=~Continent) %>% add_bars()

> plot_ly(PMundial,x=~Any, y=~NHab, color=~Continent, type ='bar')
```




**GRÀFICA DE LÍNIES INTERACTIVA:**

```
# GRÀFICA DE LÍNIES AMB ggplotly
> ggplotPMundial <- ggplot(PMundial, aes(x=Any, y=NHab,
color=Continent)) + geom_line()
> ggplotly(ggplotPMundial)
```

```
# GRÀFICA DE LÍNIES AMB plotly
> plot_ly(PMundial,x=~Any, y=~NHab, color=~Continent, type='scatter', mode='line')
```




**Respon a les següents preguntes veient la gràfica:**

**a) Descriu tres característiques importants que es veuen en la gràfica de línies.**

**RESPOSTA:**

1. La població del continent asiàtic predomina respecte els altres continents en tots els anys (1934M 1960 a 4858M el 2022). Té una davallada l'any 1990, però després es va recuperant.
2. La resta de continents té un creixement més moderat o una mica de decreixement. Destaca Africa a partir del s.XXI que te un lleuger creixement (794M el 2000 a 1383M el 2022), sobrepassant la població del continent americà.
3. El continent europeu té un creixement molt baix al llarg dels anys (49M 1960 a 61M 2022.
4. Amèrica tenia menys població que Europa el 1960 (422M America per 493M Europa) i acaba el 2022 amb una diferència de 40M (1033M Amèrica a 610M Europa)35-54 anys del 2006 al 2015, 55-74 anys del 1985 al 2002.

**b) Enumera els 3 continents amb menys població l'any 2012. Continent i població en ordre creixent.**

**RESPOSTA:**

1. Oceania – 37475300
2. Europe – 608142862

3. America - 958228468

**c) En quin parell d'anys any fa una davallada a població d'Asia?. Indica la població els dos anys.**

**RESPOSTA:**

- 1989 – 34070598904
- 1990 – 3323719552

**d) A partir de quin any la població del continent africà supera al d'Amèrica?..**

**RESPOSTA:** I fent un zoom en la zona propera al creuament Africa-America

L'any 2005:

- Africa – 899781392
- America – 891694507

**2.- Fes una gràfica interactiva de línies de l'evolució temporal de població dels països del continent americà.**

**RESPOSTA:**

**PAS 1:** DATA MASSAGING. Abans de crear la gràfica cal filtrar els registres amb valor de la variable Continent "America".

```
> PMAmerica <- PMundial %>% filter(Continent == "America")
> PMAmerica
```

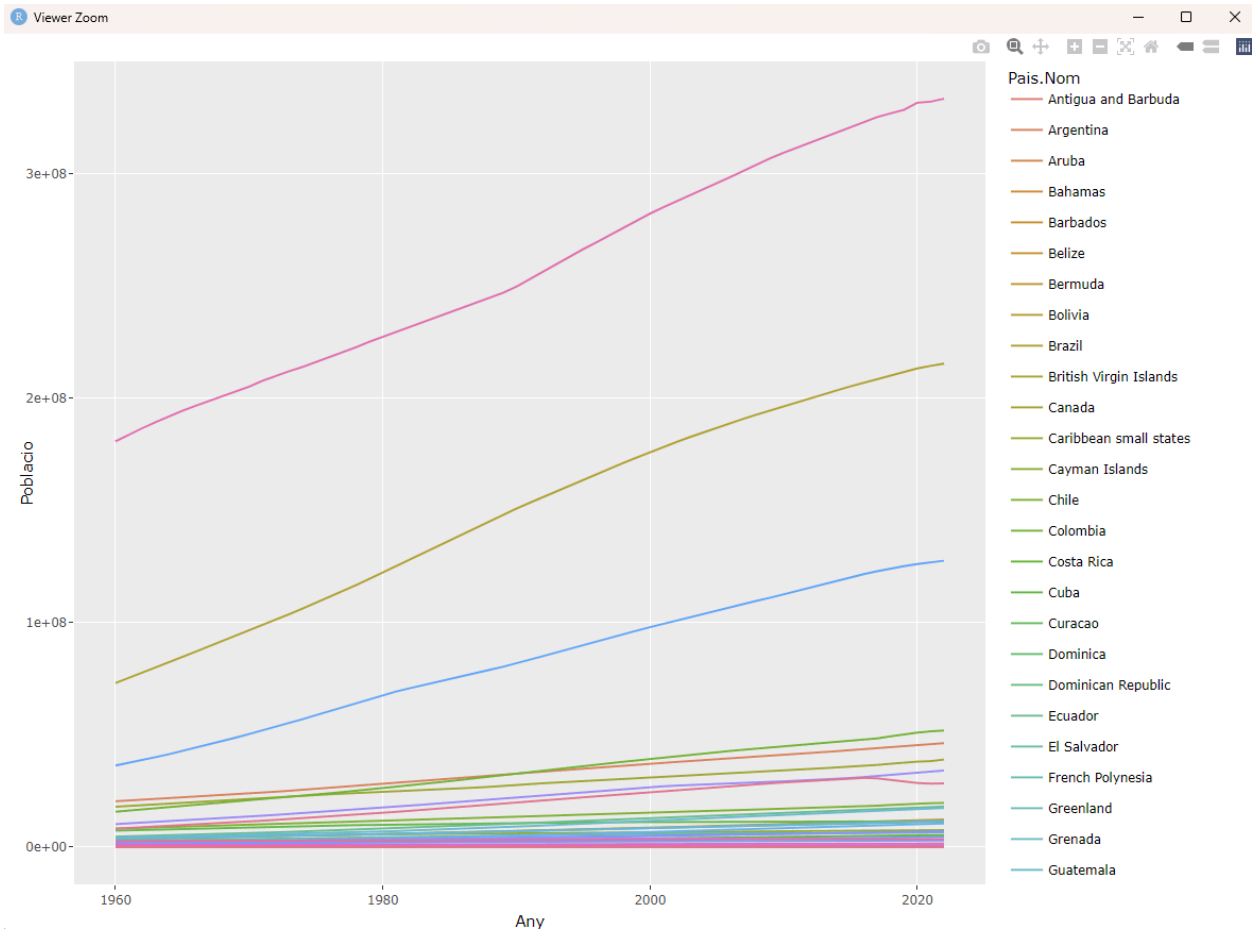
```
> PMAmerica <- PWorld %>% filter(Continent !="Zone", Continent == "America")
> PMAmerica
               Pais.Nom Continent  Any Poblacio
1                 Aruba   America 1960    54608
2                 Aruba   America 1961    55811
3                 Aruba   America 1962    56682
4                 Aruba   America 1963    57475
5                 Aruba   America 1964    58178
6                 Aruba   America 1965    58782
7                 Aruba   America 1966    59291
8                 Aruba   America 1967    59522
9                 Aruba   America 1968    59471
10                Aruba   America 1969    59330
11                Aruba   America 1970    59106
12                Aruba   America 1971    58816
```

**PAS 2:** Visualitzar per gràfica interactiva de línies

**GRÀFICA INTERACTIVA DE LÍNIES:**

```
# GRAFICA DE LÍNIES amb ggplotly
> gplotPMAmerica <- ggplot(PMAmerica, aes(x=Any, y=Poblacio,
color=Pais.Nom)) + geom_line()
> ggplotly(gplotPMAmerica)
```

```
# GRAFICA DE LÍNIES amb plotly
> plot_ly(PMAmerica,x=~Any, y=~Poblacio, color=~Pais.Nom,
type='scatter', mode='line')
```




Respon a les següents preguntes veient la gràfica:

**a) Quins tres països predominen en població durant tots els anys?. Trobes a faltar alguns països que per extensió podrien tenir més població?.**

**RESPOSTA:**

1) USA
2) Brazil,
3) Mexico

Es troba a faltar països com Canadà i Argentina que per extensió podrien tenir més població.

**b) A partir del 2016 hi ha un país americà que té un descens de població. Digues quin país és, quina població tenia els anys 2016 i 2022.**

**RESPOSTA:** Venezuela, i 28317M a 2022.

**c) Dels tres països amb més població entre 1960 i 2022, quin és el país amb més creixement de població entre 1960 i 2022?. Dona les diferencies en milions de persones dels tres.**

**RESPOSTA:**

1) USA: 333M (2022) - 180M (1960) = 153M
2) Brazil: 215M (2022) - 73M (1960) = 142M
3) Mexico: 127M (2022) - 36M (1960) = 91M

**3.- Defineix una gràfica interactiva de línies de població dels 10 països més poblats del món.**

**RESPOSTA:**

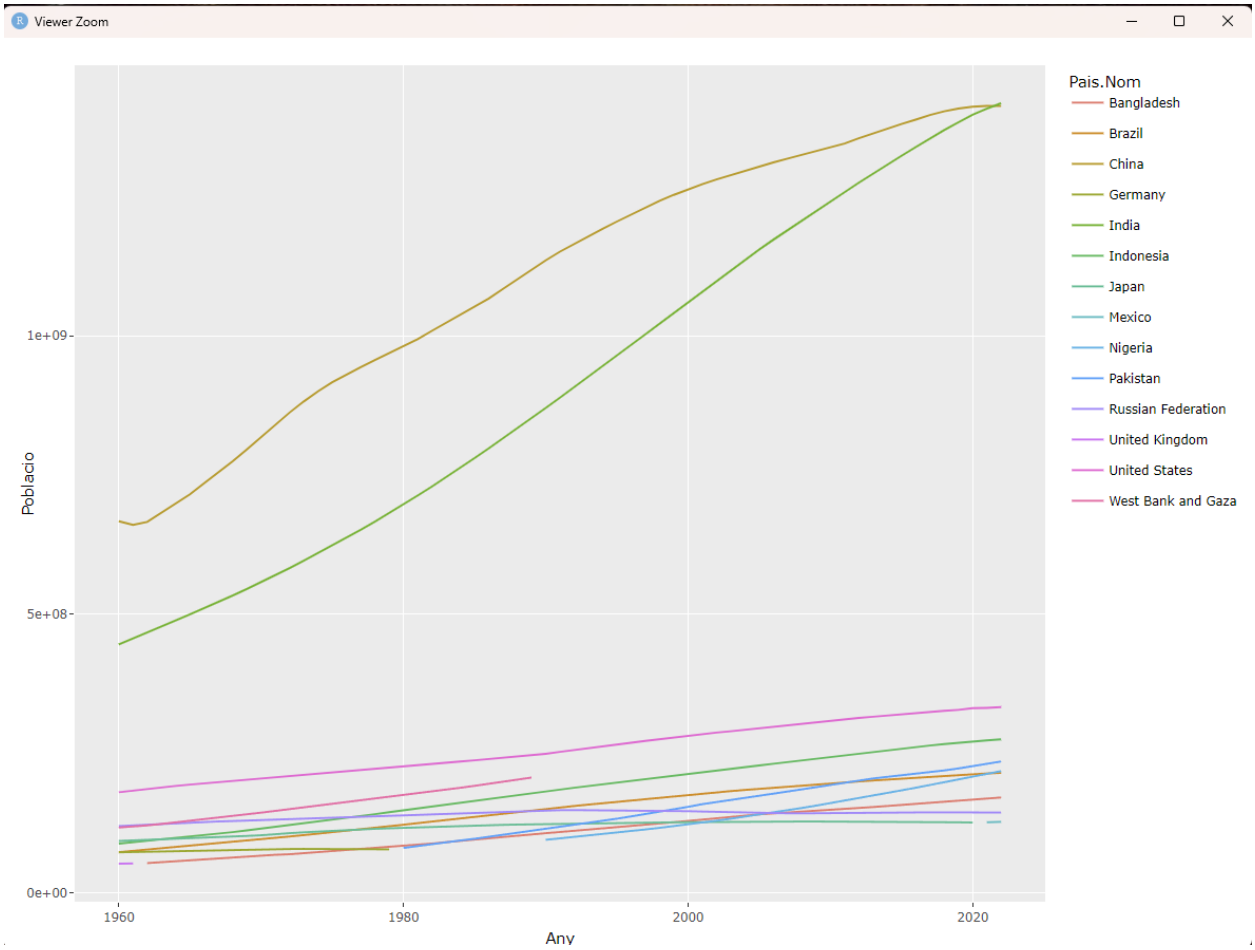
**PAS 1:** DATA MASSAGING: Afegim la variable rank que correspon al ranking decreixent per països del nombre de suïcidis i seleccionem els 10 primers de cada any segons la variable rank.

```
> PWorld_formatted <- PWorld %>% filter(Continent  != "Zone") %>% group_by(Any) %>% mutate(rank = rank(-Poblacio)) %>% filter(rank <= 10)
```

```
> PWorld_formatted <- PWorld %>% filter(Continent != "Zone") %>% group_by(Any) %>% mutate(rank = rank(-Poblacio)) %>%
                          filter(rank <=10)
>
> PWorld_formatted
# A tibble: 630 x 5
   Pais.Nom   Continent   Any Poblacio  rank
   <chr>      <chr>     <int>    <dbl> <dbl>
 1 Bangladesh Asia       1962 53461661    10
 2 Bangladesh Asia       1963 55094115    10
 3 Bangladesh Asia       1964 56774465    10
 4 Bangladesh Asia       1965 58500159    10
 5 Bangladesh Asia       1966 60265259    10
 6 Bangladesh Asia       1967 62104488    10
 7 Bangladesh Asia       1968 63995652    10
 8 Bangladesh Asia       1969 65866908    10
 9 Bangladesh Asia       1970 67541860    10
10 Bangladesh Asia       1971 68376204    10
# i 620 more rows
# i Use `print(n = ...)` to see more rows
>
```

**PAS 2:** Visualitzar la gràfica interactiva de línies:

```
# GRÀFICA AMB LÍNIES AMB ggplotly
> ggplotPM_formatted <- ggplot(PWorld_formatted, aes(x=Any, y=Poblacio, color=Pais.Nom)) + geom_line()
> ggplotly(ggplotPM_formatted)

# GRÀFICA AMB LÍNIES AMB plotly
>   plot_ly(PWorld_formatted,x=~Any,   y=~Poblacio,   color=~Pais.Nom, type='scatter', mode='line')
```




Respon a les següents preguntes veient la gràfica:

**a) Quins tres països predominen en població durant tots els anys?. Mirant la gràfica, quin país ha tingut un creixement més gran entre 1960 i 2022?.**

**RESPOSTA:**

1) China
2) Índia
3) USA

Índia és la que ha tingut un major creixement entre 1960 i 2022.

**b) Quin és el país més poblat del món a 2022?. Digues el país i la població en milions de persones.**

**RESPOSTA:** India amb 1417M a 2022

**c) Calcula la densitat de població (Poblacio / superfície en km2) l'any 2022 pels tres països més poblats del món i mira si es manté el ranking per densitat de població. població entre 1960 i 2022, quin és el país amb més creixement de població entre 1960 i 2022?. Dona les diferencies en milions de persones dels tres.**

**RESPOSTA:**

- INDIA: Població: 1.417.173.173 (2022) - Superfície - 3.287.000 km2 - Densitat: 431,14 hab./km2
- CHINA: Població: 1.412.175.000 (2022) - Superfície: 9.597.000 km2 - Densitat: 147,14 hab./km2
- USA: Població: 333.287.557 (2022) - Superfície: 9.867.000 km2 - Densitat: 33,77 hab./km2.

**4.- Visualitza dues gràfiques interactives de línies per a comparar-les.**

a) **Representa una gràfica interactiva de línies amb la població dels 15 països menys poblats de l'Asia. Quins són els tres països menys poblats el 2022 amb la diferència de població entre 2022 i 1960?.**

**RESPOSTA:**

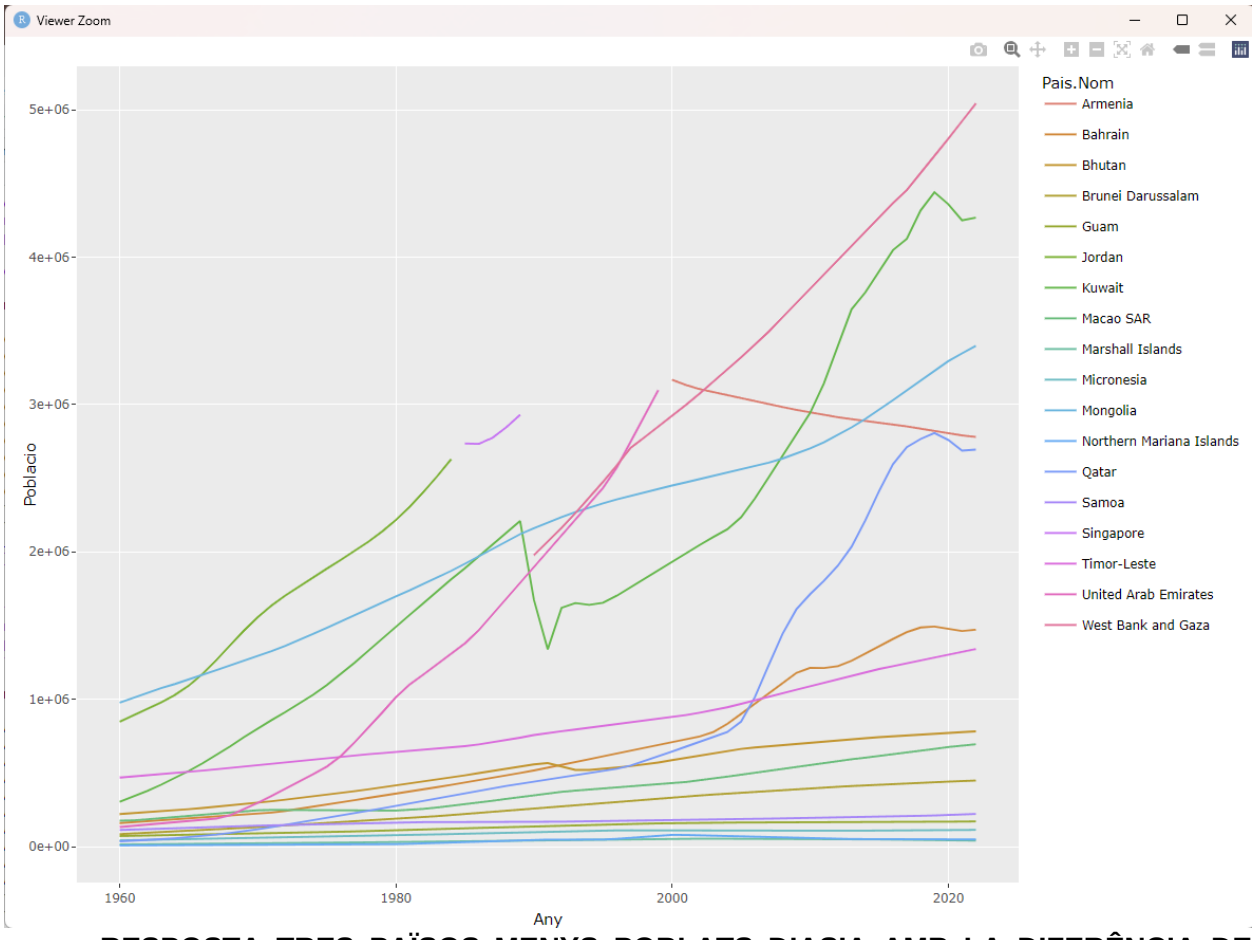
**PAS 1:** DATA MASSAGING: Afegim la variable rank que correspon al ranking decreixent per països del nombre de suïcidis i seleccionem els 10 primers de cada any segons la variable rank.

```
> PWorldAsia_formatted <- PWorld %>% filter(Continent != "Zone", Continent == "Asia") %>% group_by(Any) %>% mutate(rank = rank(Poblacio)) %>% filter(rank <= 15)
```

```
> PWorldAsia_formatted <- PWorld %>% filter(Continent != "Zone", Continent == "Asia") %>%
group_by(Any) %>% mutate(rank = rank(Poblacio)) %>%
                          filter(rank <=15)
> PWorldAsia_formatted
# A tibble: 945 x 5
   Pais.Nom             Continent   Any Poblacio  rank
   <chr>                <chr>     <int>    <dbl> <dbl>
 1 United Arab Emirates Asia       1960   133426     8
 2 United Arab Emirates Asia       1961   140984     8
 3 United Arab Emirates Asia       1962   148877     8
 4 United Arab Emirates Asia       1963   157006     8
 5 United Arab Emirates Asia       1964   165305     8
 6 United Arab Emirates Asia       1965   173797     8
 7 United Arab Emirates Asia       1966   182509     8
 8 United Arab Emirates Asia       1967   191404     8
 9 United Arab Emirates Asia       1968   213582     9
10 United Arab Emirates Asia       1969   253261    10
# i 935 more rows
# i Use `print(n = ...)` to see more rows
>
```

**PAS 2:** Visualitzar per gràfica interactiva de línies:

```
# GRÀFICA DE LÍNIES amb ggplotly
> ggplotAsia_formatted <- ggplot(PWorldAsia_formatted, aes(x=Any, y=Poblacio, color=Pais.Nom)) + geom_line()
> ggplotly(ggplotAsia_formatted)

# GRÀFICA DE LÍNIES amb plot_ly
>  plot_ly(PWorldAsia_formatted,x=~Any,  y=~Poblacio,  color=~Pais.Nom, type='scatter', mode='line')
```




**RESPOSTA TRES PAÏSOS MENYS POBLATS D'ASIA AMB LA DIFERÊNCIA DE POBLACIÓ ENTRE 2022 i 1960:**

1) MARSHALL ISLANDS: 41569 (2022) - 8702 (1960) = 32867
2) MICRONESIA: 114164 (2022) - 42986 (1960) = 71178
3) GUAM: 171774 (2022) - 72374 (1960) = 99400

**b) Representa una gràfica interactiva de línies amb la població dels 15 països menys poblats d'Europa. Quins són els tres països menys poblats el 2022 amb la diferència de població entre 2022 i 1960?.**

**RESPOSTA:**

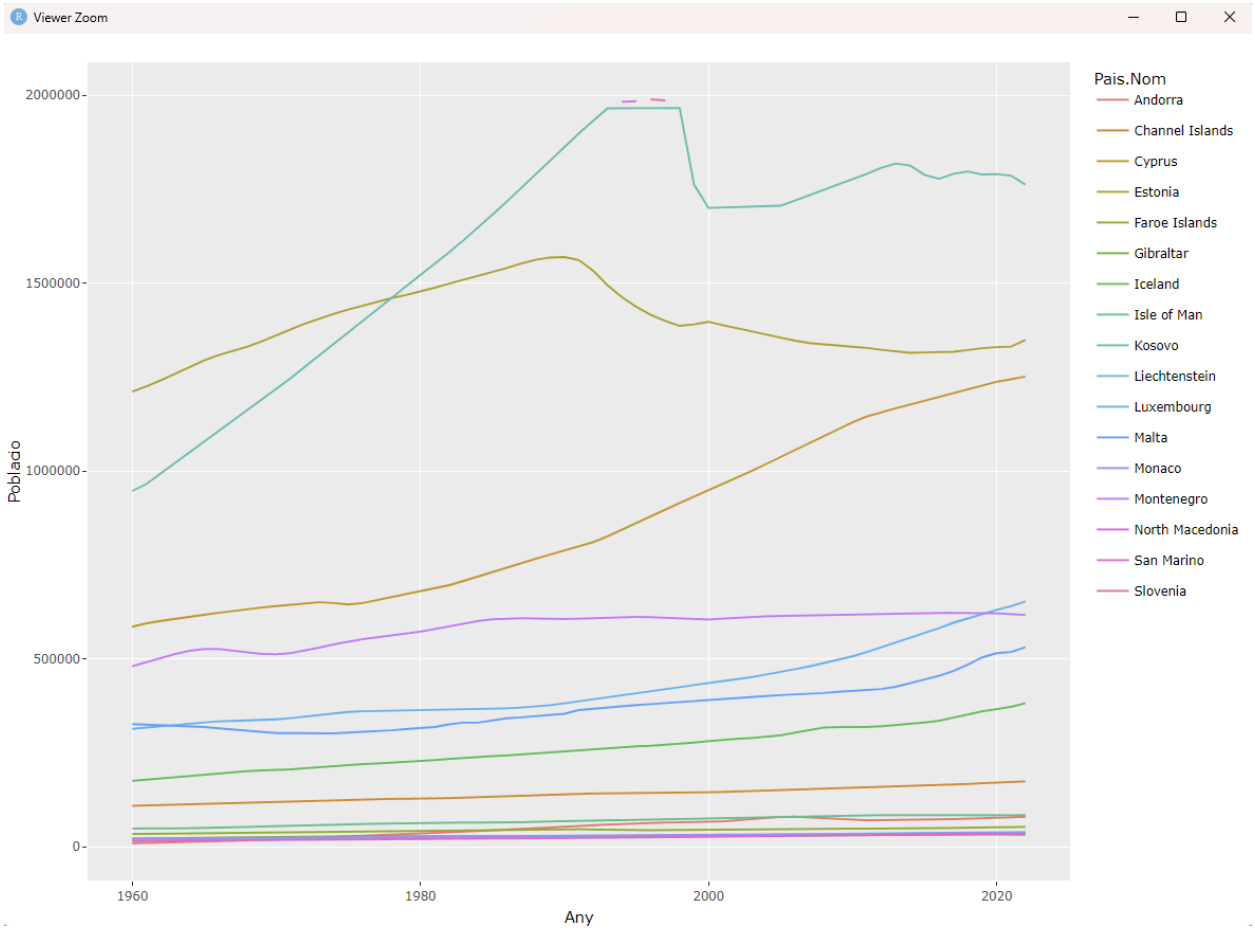
**PAS 1:** DATA MASSAGING: Afegim la variable rank que correspon al ranking decreixent per països del nombre de suïcidis i seleccionem els 10 primers de cada any segons la variable rank.

```
> PWorldEuropa_formatted <- PWorld %>% filter(Continent != "Zone", Continent == "Europe") %>% group_by(Any) %>% mutate(rank = rank(Poblacio)) %>% filter(rank <= 15)

> PWorldEuropa_formatted
```

```
> PWorldEuropa_formatted <- PWorld %>% filter(Continent != "Zone", Continent == "Europe") %>%
group_by(Any) %>% mutate(rank = rank(Poblacio)) %>%                          filter(rank <=15) %

> PWorldEuropa_formatted
# A tibble: 945 × 5
   Pais.Nom Continent   Any Poblacio  rank
   <chr>    <chr>     <int>    <dbl> <dbl>
 1 Andorra  Europe     1960     9443     1
 2 Andorra  Europe     1961    10216     1
 3 Andorra  Europe     1962    11014     1
 4 Andorra  Europe     1963    11839     1
 5 Andorra  Europe     1964    12690     1
 6 Andorra  Europe     1965    13563     1
 7 Andorra  Europe     1966    14546     1
 8 Andorra  Europe     1967    15745     1
 9 Andorra  Europe     1968    17079     1
10 Andorra  Europe     1969    18449     2
# i 935 more rows
# i Use `print(n = ...)` to see more rows
>
```

**PAS 2:** Visualitzar per gràfica interactiva de línies:

```
# GRÀFICA DE LÍNIES amb ggplotly
> ggplotEuropa_formatted <- ggplot(PWorldEuropa_formatted, aes(x=Any, y=Poblacio, color=Pais.Nom)) + geom_line()
> ggplotly(ggplotEuropa_formatted)

# GRÀFICA DE LÍNIES amb plot_ly
> plot_ly(PWorldEuropa_formatted,x=~Any, y=~Poblacio, color=~Pais.Nom, type='scatter', mode='line')
```



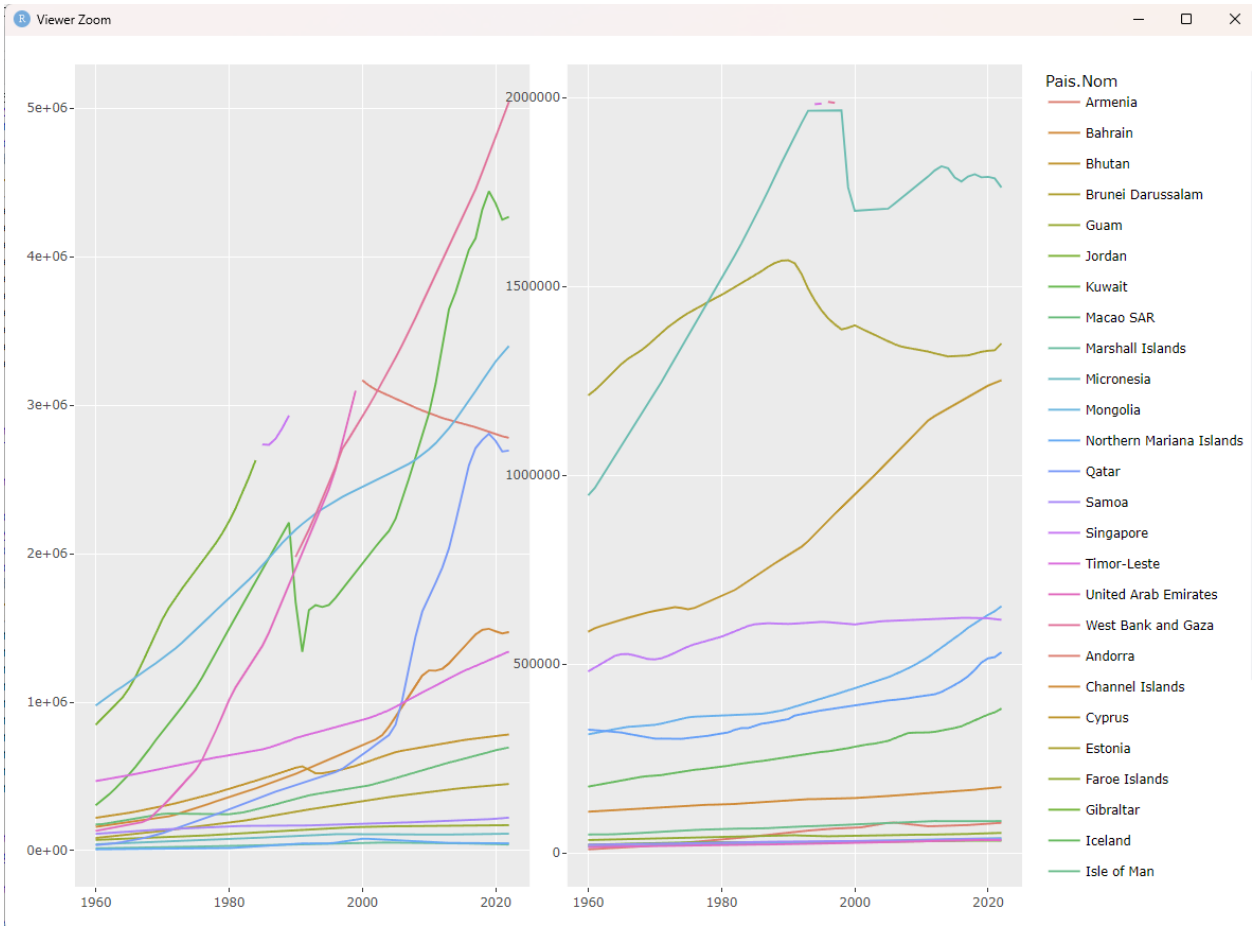

**RESPOSTA TRES PAÏSOS MENYS POBLATS D'EUROPA AMB LA DIFERÊNCIA DE POBLACIÓ ENTRE 2022 i 1960:**

1) GIBRALTAR: 32649 (2022) - 21822 (1960) = 10827
2) SANT MARINO: 33660 (2022) - 15556 (1960) = 18104
3) MONACO: 36469 (2022) - 21797 (1960) = 14672

**c) Dibuixa les dues gràfiques anteriors (Asia esquerra, Europa dreta) en una mateixa pantalla i comenta 3 idees que es poden deduir veient les dues gràfiques.**

**RESPOSTA:**

```
> subplot(ggplotAsia_formatted, ggplotEuropa_formatted)
```




**RESPOSTA:**

- L'ordre de magnitud de la gràfica de població (eix y) en Asia es més del doble que d'Europa.
- El nivell de creixement al llarg dels anys en Asia és superior al d'Europa.
- Hi ha més canvis en el ranking en els aïsos d'Asia que en els d'Europa, on hi ha més estabilitat en les posicions.
- Kosovo pateix una davallada de població l'any 1993 producte de la guerra dels Balcans.
- Kuwait pateix una davallada de població l'any 1990 i 1991 producte de la guerra del Kuwait i del Golf.

**5.- Guarda el darrer plot en un fitxer en format HTML en la carpeta de treball (wd) per a poder-se executar en un navegador. Digues què ocupa la versió estesa i la reduïda i prova d'executar-les en un navegador.**

**RESPOSTA:**

```
# Funció widget_size que guarda objecte plotly a format HTML
widget_file_size <- function(p) {
  d <- getwd()
  withr::with_dir(d, htmlwidgets::saveWidget(p, "index.html"))
  f <- file.path(d, "index.html")
  mb <- round(file.info(f)$size / 1e6, 3)
  message("File is: ", mb," MB")
}
```

```
# Generar fitxer index.html
> plot <- subplot(ggplotAsia_formatted, ggplotEuropa_formatted)
> widget_file_size(plot)    # Versió completa HTML
> widget_file_size(partial_bundle(plot)) # Versió reduida HTML
```

```
> widget_file_size <- function(p) {
+     d <- getwd()
+     withr::with_dir(d, htmlwidgets::saveWidget(p, "index.html"))
+     f <- file.path(d, "index.html")
+     mb <- round(file.info(f)$size / 1e6, 3)
+     message("File is: ", mb," MB")
+ }
>
> plot <- subplot(ggplotAsia_formatted, ggplotEuropa_formatted)
>
> widget_file_size(plot)
File is: 4.019 MB
> widget_file_size(partial_bundle(plot))
File is: 1.355 MB
>
```

També es pot exportar amb HTML amb el botó *Export* del visualitzador interactiu de R.

**6.- Visualitza un scatter plot 3D sobre anys, població i PIB combinant el dataset que treballem `Poblacio_Mundial_1960-2022.csv` amb el data set PIB_Mundial_1960_2020.csv que conté dades del PIB de països del món des del 1960 al 2020.**

**Comenta mirant la gràfica algunes idees que vegis. Mira si pots trobar algun tipus de correlació entre el PIB, Continent i la població dels països.**

**Us podeu carregar la llibreria plyr per a fer joins entre datasets.**

```
> install.packages('plyr') # Càrrega paquet plyr (1a vegada)
> library(plyr)
>
```

**RESPOSTA:**

**Ara llegim el data set `Poblacio_Mundial_1960-2022.csv`**

```
> setwd("C:/Users/enric/R")
> PWorld <- read.csv('./Poblacio_Mundial_1960-2022.csv')
>
```

**Després llegim el data set `PIB_Mundial_1960-2020.csv`**

```
> setwd("C:/Users/enric/R")
> PIBWorld <- read.csv('./PIB_Mundial_1960-2020.csv')
> str(PIBWorld)
>
```

**PAS 1:** DATA MASSAGING: Eliminem les entrades que tinguin "Continent="Zone", ens quedem només amb països, no zones geogràfiques.

```
PMon <- PWorld %>% filter(Continent != "Zone")
> SWorldPIB
```

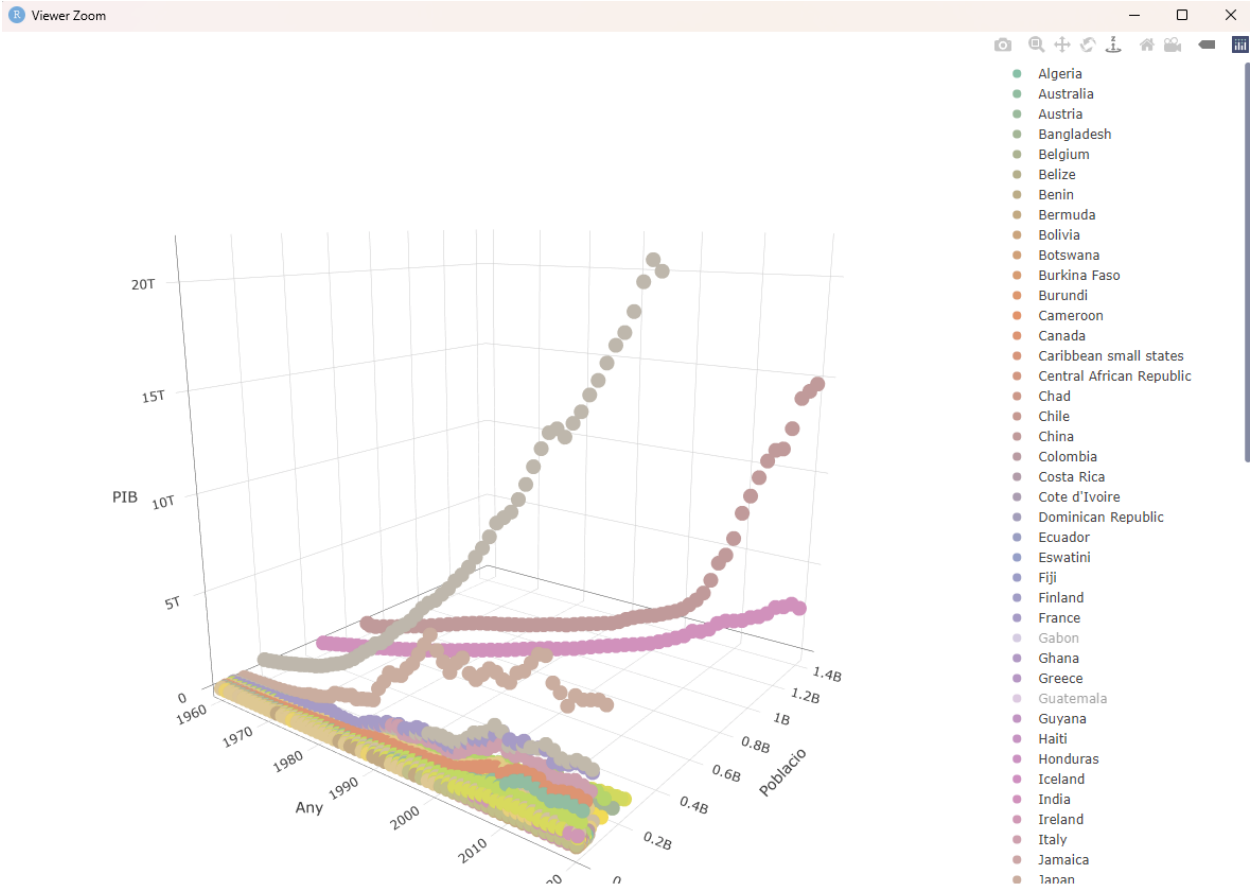
**PAS 2:** DATA MASSAGING: Fem un join entre els dos datasets: per país i any per a agrupar població i GDP (PIB) i eliminem registres amb PIB indefinit (PIB=="NA")..

```
> Resultat <- join(PMon,PIBWorld) %>% filter(PIB != "NA")
> str(Resultat)
```

```
> Resultat <- join(PMon,PIBWorld) %>% filter(PIB != "NA")
Joining by: Pais.Nom, Any
> str(Resultat)
'data.frame':   5124 obs. of  5 variables:
 $ Pais.Nom : chr  "Australia" "Australia" "Australia" "Australia" ...
 $ Continent: chr  "Oceania" "Oceania" "Oceania" "Oceania" ...
 $ Any      : int  1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 ...
 $ Poblacio : num  10276477 10483000 10742000 10950000 11167000 ...
 $ PIB      : num  1.86e+10 1.97e+10 1.99e+10 2.15e+10 2.38e+10 ...
>
```

**PAS 3:** SCATTER 3D PLOT:

```
> plot <- plot_ly(Resultat, x=~Any, y=~Poblacio, z=~PIB, color=~Pais.Nom, type='scatter3d', mode='markers')
> plot
```



**COMENTARIS:**

1. La majoria de països amb una baixa població mantenen un PIB baix al llarg dels anys. De tot aquest conjunt, destaquen una mica per PIB United Kingdom, France, Italy
2. Quatre països surten una mica de la norma: China, India, Unites States i Japan.
3. Dos països (China i Índia) destaquen per població, però no per PIB, en que predomina United States.
4. El que té una evolució més alta i predominant del PIB amb una mitjana població és US.
5. Japó té una evolució del PIB més modesta però amb una població força baixa. Potser és el millor país dins la majoria de països, tot i que la seva evolució del PIB ha tingut baixades els anys 1996 i 2013.
6. En segon lloc, la China ha tingut una alta evolució del PIB (no tant com United States) amb una població molt alta.
7. L'India ha tingut una evolució molt creixent de població, situant-se amb un PIB del pitjor dels 4 països predominants, i inclús més baix que algun país com UK i força igualat a France l'any 2020.

**PAS 4:** Guardar-el en html:

```
# Funció widget_size que guarda objecte plotly a format HTML
widget_file_size <- function(p) {
  d <- getwd()
  withr::with_dir(d, htmlwidgets::saveWidget(p, "scatterPlot3D.html"))
  f <- file.path(d, "scatterPlot3D.html")
  mb <- round(file.info(f)$size / 1e6, 3)
  message("File is: ", mb," MB")
}

# Generar fitxer scatterPlot3D.html
> widget_file_size(plot)    # Versió completa HTML
```

```
> widget_file_size(plot)        # Versió completa HTML
File is: 1.355 MB
Warning messages:
1: In RColorBrewer::brewer.pal(N, "Set2") :
  n too large, allowed maximum for palette Set2 is 8
Returning the palette you asked for with that many colors

2: In RColorBrewer::brewer.pal(N, "Set2") :
  n too large, allowed maximum for palette Set2 is 8
Returning the palette you asked for with that many colors

>
>
```

```
> widget_file_size(partial_bundle(plot)) # Versió reduida HTML
```

```
> widget_file_size(partial_bundle(plot)) # Versió reduida HTML
File is: 1.355 MB
Warning messages:
1: In RColorBrewer::brewer.pal(N, "Set2") :
  n too large, allowed maximum for palette Set2 is 8
Returning the palette you asked for with that many colors

2: In RColorBrewer::brewer.pal(N, "Set2") :
  n too large, allowed maximum for palette Set2 is 8
Returning the palette you asked for with that many colors

>
```

## *3.* PART 2. Animate Bar Race Ranking

**En aquesta segona part del seminari intentarem fer gràfiques el tipus *Animate Bar Race Ranking* sobre el data set del padró de noms (`Poblacio_Mundial_1960-2022.csv`).**

**RESPOSTA:**

**Primer de tot, instal·larem les llibreries que necessitem:**

```
> library(tidyverse)
> install.packages('gifski')  # Càrrega paquet gifski (primera vegada)
> library(gifski)             # Generació fitxer animació GIF
> install.packages('av')      # Càrrega paquet av (primera vegada)
> library(av)                 # Generació fitxer animació AVI
> install.packages('gganimate') # Càrrega paquet gganimate (1a vegada)
> library(gganimate)          # Generació de frames i fitxer animació
>
```

**Llegim el data set:**

```
> setwd("C:/Users/enric/Documents/R")
> PWorld <- read.csv('./Poblacio_Mundial_1960-2022.csv')
```

**EXERCICIS:**

**1.- Crea un *Animate Bar Race Ranking* sobre el primer quartil (Q1) dels països europeus MENYS poblats per any.**

**Respecte els valors donats en les transparències, modifica els valors `transition_lenght` (per exemple 4000) i `state_lenght` (per exemple 400) de la comanda `transition_states()` i el valor del paràmetre del nframes (per exemple a 500) de la comanda `animate(objecte,nframes,...)`. Com canvia l'animació?**

**COMENTARI: Comenta l'evolució d'Andorra al llarg dels anys.**

**PAS 1:** DATA MASSAGING: Eliminem les entrades que tinguin "Continent="Zone", ens quedem només amb països d'Europa.

```
> PMon <- PWorld %>% filter(Continent != "Zone", Continent ==
"Europe")
```

```
> PWorld <- read.csv('./Poblacio_Mundial_1960-2022.csv')
> PMon <- PWorld %>% filter(Continent != "Zone", Continent == "Europe")
> PMon
      Pais.Nom Continent  Any Poblacio
1      Albania    Europe 1960  1608800
2      Albania    Europe 1961  1659800
3      Albania    Europe 1962  1711319
4      Albania    Europe 1963  1762621
5      Albania    Europe 1964  1814135
6      Albania    Europe 1965  1864791
7      Albania    Europe 1966  1914573
8      Albania    Europe 1967  1965598
9      Albania    Europe 1968  2022272
10     Albania    Europe 1969  2081695
```

**PAS 2:** Afegim la variable *rank* que correspon al ranking creixent per països de la població (eliminem el signe negatiu del paràmetre de la funció rank()) i seleccionem els Q1 països per cada any segons la variable `rank`. El guardem en l'objecte *`PMon_formatted`*:

```
> PMon_formatted <- PMon %>%
  group_by(Any) %>%
  mutate(rank = rank(Poblacio)) %>%
  filter(rank <= 0.25 * max(rank)) #Q1
```

```
> PMon_formatted <- PMon %>%
+     group_by(Any) %>%
+     # The * 1 makes it possible to have non-integer ranks while sliding
+     mutate(rank = rank(Poblacio)) %>%
+     filter(rank <= 0.25 * max(rank)) #Q1
>
> PMon_formatted
# A tibble: 756 x 5
# Groups:   Any [63]
   Pais.Nom Continent   Any Poblacio  rank
   <chr>    <chr>     <int>    <dbl> <dbl>
 1 Andorra  Europe     1960     9443     1
 2 Andorra  Europe     1961    10216     1
 3 Andorra  Europe     1962    11014     1
 4 Andorra  Europe     1963    11839     1
 5 Andorra  Europe     1964    12690     1
 6 Andorra  Europe     1965    13563     1
 7 Andorra  Europe     1966    14546     1
 8 Andorra  Europe     1967    15745     1
 9 Andorra  Europe     1968    17079     1
10 Andorra  Europe     1969    18449     2
# i 746 more rows
# i Use `print(n = ...)` to see more rows
>
```

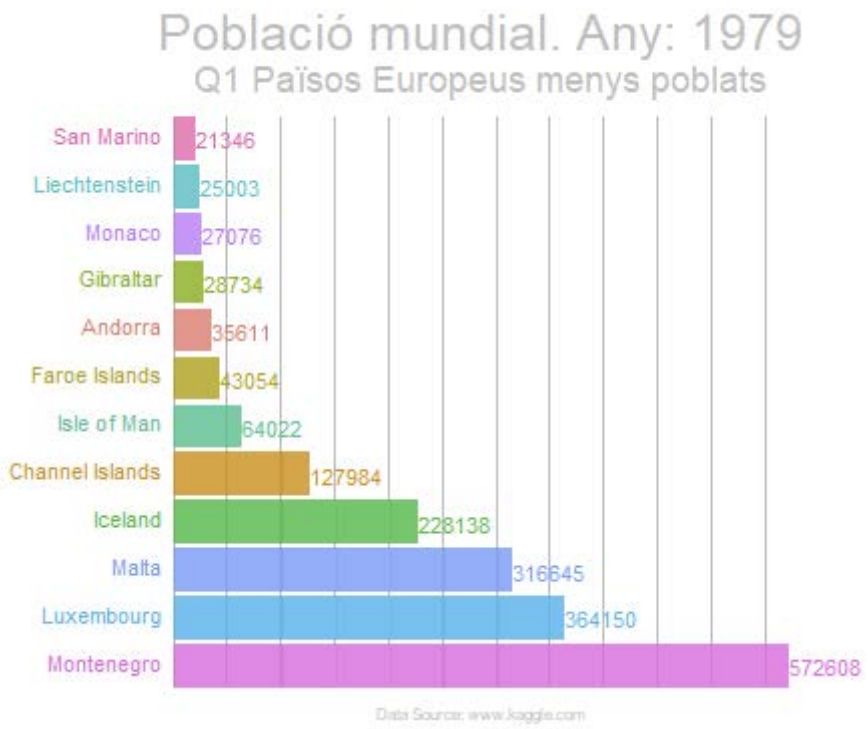
**PAS 3:** Basant-nos en la comanda explicada en les transparències, podem generar els frames estàtics de les gràfiques per anys:

```
> anim <- ggplot(PMon_formatted, aes(rank, group = Pais.Nom,
                fill = as.factor(Pais.Nom), color = as.factor(Pais.Nom))) +
    geom_tile(aes(y = Poblacio/2,
                height = Poblacio,
```

```
            width = 0.9), alpha = 0.8, color = NA) +
      geom_text(aes(y=0, label = paste(Pais.Nom," ")), vjust=0.2, hjust=1) +
      geom_text(aes(y = Poblacio, label = Poblacio, hjust = 0)) +
      coord_flip(clip = "off", expand = FALSE) +
       scale_y_continuous(labels = scales::comma) +
      scale_x_reverse() +
       guides(color = "none", fill = "none") +
      theme(axis.line=element_blank(),
      axis.text.x=element_blank(),
      axis.text.y=element_blank(),
      axis.ticks=element_blank(),
      axis.title.x=element_blank(),
      axis.title.y=element_blank(),
      legend.position="none",
      panel.background=element_blank(),
      panel.border=element_blank(),
      panel.grid.major=element_blank(),
      panel.grid.minor=element_blank(),
      panel.grid.major.x = element_line( linewidth=.1, color="grey" ),
      panel.grid.minor.x = element_line( linewidth=.1, color="grey" ),
      plot.title=element_text(size=25, hjust=0.5, face="bold", colour="grey",
      vjust=-1),
      plot.subtitle=element_text(size=18,      hjust=0.5,      face="italic",
      color="grey"),
      plot.caption     =element_text(size=8,     hjust=0.5,     face="italic",
      color="grey"),
      plot.background=element_blank(),
      plot.margin = margin(2,2, 2, 4, "cm")) +
      transition_states(Any,    transition_length    =    4,    state_length=1,
      wrap=FALSE) +
      view_follow(fixed_x = TRUE) +
      labs(title = 'Població mundial. Any: {closest_state}',
           subtitle  =  "Q1 Països Europeus menys poblats",
           caption  = "Data Source: www.kaggle.com")
> anim
```

**PAS 4:** A partir de l'objecte `anim`, creem els fitxers d'animació GIF i AVI:

```
# Exportar frames a fitxer GIF
> animate(anim, 200, fps = 20,  width = 1200, height = 1000,
        renderer   =   gifski_renderer("Q1Europeus.gif"),   end_pause   =   15,
        start_pause =  15)

# Exportar frames a fitxer AVI
> animate(anim, 200, fps = 20,  width = 1200, height = 1000,
        renderer = av_renderer("Q1Europeus.avi"), end_pause = 15, start_pause
        =  15)
```




**ALTERNATIVA: Llibreria ddplot**

**Primer de tot, instal·larem les llibreries que necessitem:**

```
> install.packages('ddplot')  # Càrrega paquet ddplot (primera vegada)
> library(ddplot)             # Generació Bar Chart Race
>
```

**PAS 1:** DATA MASSAGING: Ens quedem només amb països d'Europa.

```
> PMon <- PWorld %>% filter(Continent == "Europe")
```

**PAS 2:** Afegim la variable *rank* que correspon al ranking creixent per països de la població, i seleccionem els Q1 països de cada any segons la variable `rank`. El guardem en l'objecte *PMon_formatted*:

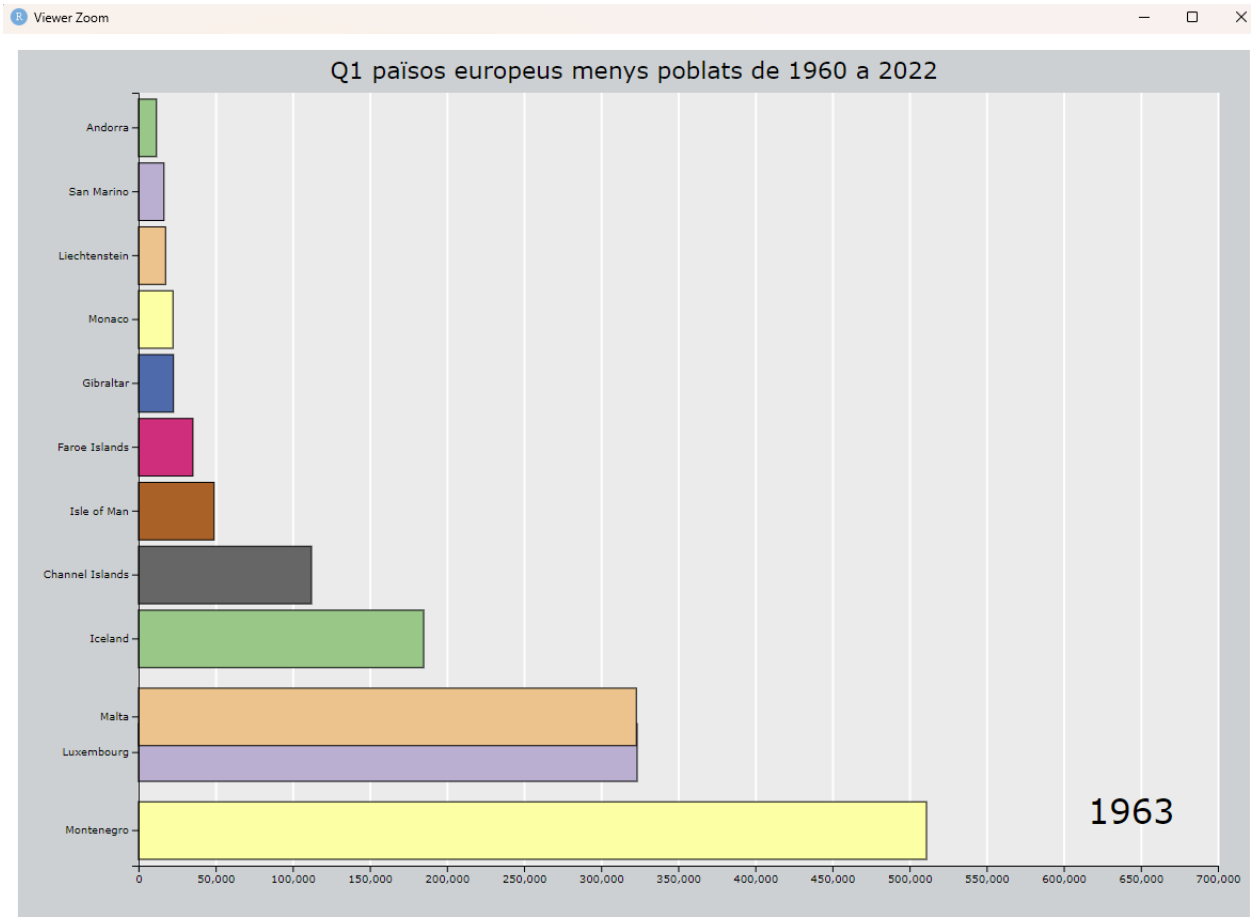
```
> PMon_formatted <- PMon %>%
  group_by(Any) %>%
  mutate(rank = rank(Poblacio)) %>%
  filter(rank <= 0.25 * max(rank))
```

**PAS 3:** Basant-nos en la comanda explicada en les transparències, podem generar els frames estàtics de les gràfiques per anys:

```
> barChartRace(PWorld_formatted, x="Poblacio", y="Pais.Nom", time="Any", sort
= "ascending", title = "Q1 països europeus menys poblats de 1960 a 2022")
```

**COMENTARI:** Andorra era el país menys poblat d'Andorra els anys 60. I els anys 70 va tenir un augment significatiu de població fins a passar al sisè menys poblat d'Europa.

En general no hi ha han hagut grans canvis de població en la majoria dels Q1 països menys poblats d'Europa entre 1960 i 2022.




**2.- Defineix un *Animate Bar Race Ranking* del segon quartil (Q2) de països MÉS poblats d'Amèrica. Comenta com evolucionen els països en general en l'animació. En quin any apareix Cuba i en quina posició acaba l'any 2022?.**

**RESPOSTA:**

**PAS 1:** DATA MASSAGING: Fem el mateix pas que l'anterior. Ens quedem només amb països d'America.

```
> PMon <- PWorld %>% filter(Continent == "America")
```

**PAS 2:** Afegim la variable *rank* que correspon al ranking decreixent per països de la població, i seleccionem els Q2 països per cada any segons la variable `rank`. Guardem tot en l'objecte *PMon_formatted*:

```
> PMon_formatted <- PMon %>%
  group_by(Any) %>%
  # The * 1 makes it possible to have non-integer ranks while sliding
  mutate(rank = rank(-Poblacio)) %>%
  filter(rank >= 0.25 * max(rank) & rank <= 0.5 * max(rank)) # Q2
```

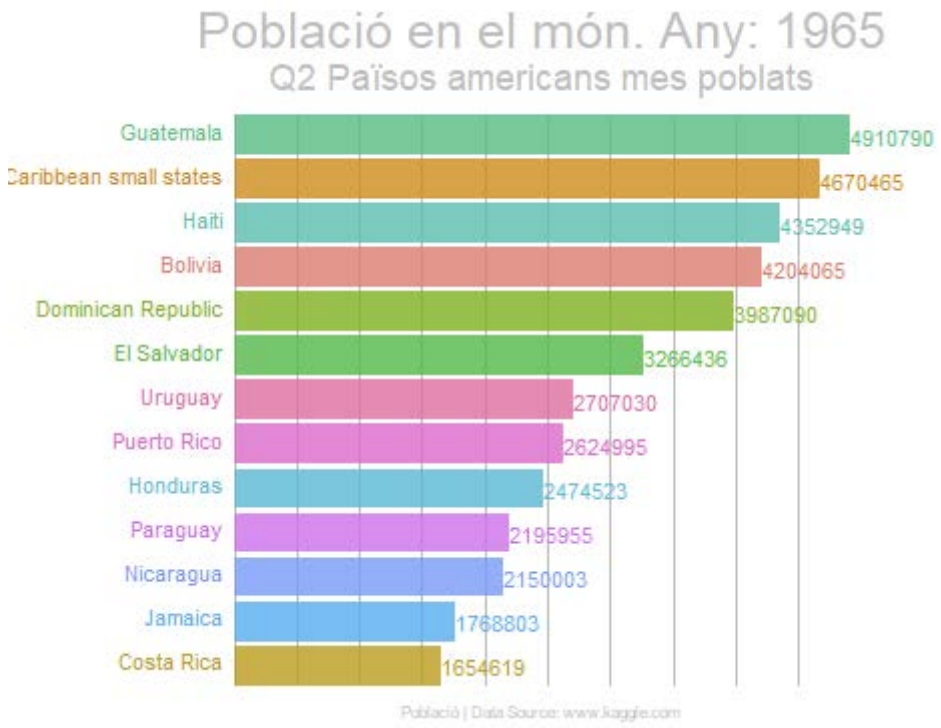
**PAS 3:** Basant-nos en la comanda explicada en les transparències, podem generar en l'objecte `anim` els frames estàtics de les gràfiques per anys:

```
> anim <- ggplot(PMon_formatted, aes(rank, group = Pais.Nom,
                fill = as.factor(Pais.Nom), color = as.factor(Pais.Nom))) +
  geom_tile(aes(y = Poblacio/2,
                height = Poblacio,
                width = 0.9), alpha = 0.8, color = NA) +
  geom_text(aes(y = 0, label = paste(Pais.Nom, " ")), vjust=0.2, hjust = 1) +
  geom_text(aes(y = Poblacio, label = Poblacio, hjust = 0)) +
  coord_flip(clip = "off", expand = FALSE) +
  scale_y_continuous(labels = scales::comma) +
  scale_x_reverse() +
  guides(color = "none", fill = "none") +
  theme(axis.line=element_blank(),
        axis.text.x=element_blank(),
        axis.text.y=element_blank(),
        axis.ticks=element_blank(),
        axis.title.x=element_blank(),
         axis.title.y=element_blank(),
        legend.position="none",
        panel.background=element_blank(),
        panel.border=element_blank(),
        panel.grid.major=element_blank(),
        panel.grid.minor=element_blank(),
        panel.grid.major.x = element_line( linewidth=.1, color="grey" ),
        panel.grid.minor.x = element_line( linewidth=.1, color="grey" ),
        plot.title=element_text(size=25,        hjust=0.5,        face="bold",
colour="grey", vjust=-1),
        plot.subtitle=element_text(size=18,     hjust=0.5,     face="italic",
color="grey"),
        plot.caption    =element_text(size=8,    hjust=0.5,    face="italic",
color="grey"),
        plot.background=element_blank(),
        plot.margin = margin(2,2, 2, 4, "cm")) +
  transition_states(Any, transition_length = 4, state_length=1, wrap=FALSE) +
  view_follow(fixed_x = TRUE) +
  labs(title = 'Població en el món. Any: {closest_state}',
       subtitle  =  "Q2 Països americans més poblats",
       caption  = "Data Source: www.kaggle.com")
```

**PAS 4:** A partir de l'objecte `anim`, creem els fitxers d'animació GIF i AVI:

```
# Exportar frames a fitxer GIF
> animate(anim, 600, fps = 20,  width = 1200, height = 1000,
        renderer   =   gifski_renderer("Q2Americans.gif"),   end_pause   =   15,
        start_pause =  15)

# Exportar frames a fitxer AVI
```

```
> animate(anim, 600, fps = 20,  width = 1200, height = 1000, renderer = av_renderer("Q2Americans.avi"), end_pause = 15, start_pause =  15)
```




**ALTERNATIVA: Llibreria ddplot**

**Primer de tot, instal·larem les llibreries que necessitem:**

```
> install.packages('ddplot')  # Càrrega paquet ddplot (primera vegada)
> library(ddplot)             # Generació Bar Chart Race
>
```

**PAS 1:** DATA MASSAGING: Eliminem les entrades que tinguin "Continent="Zone", ens quedem només amb països d'America.

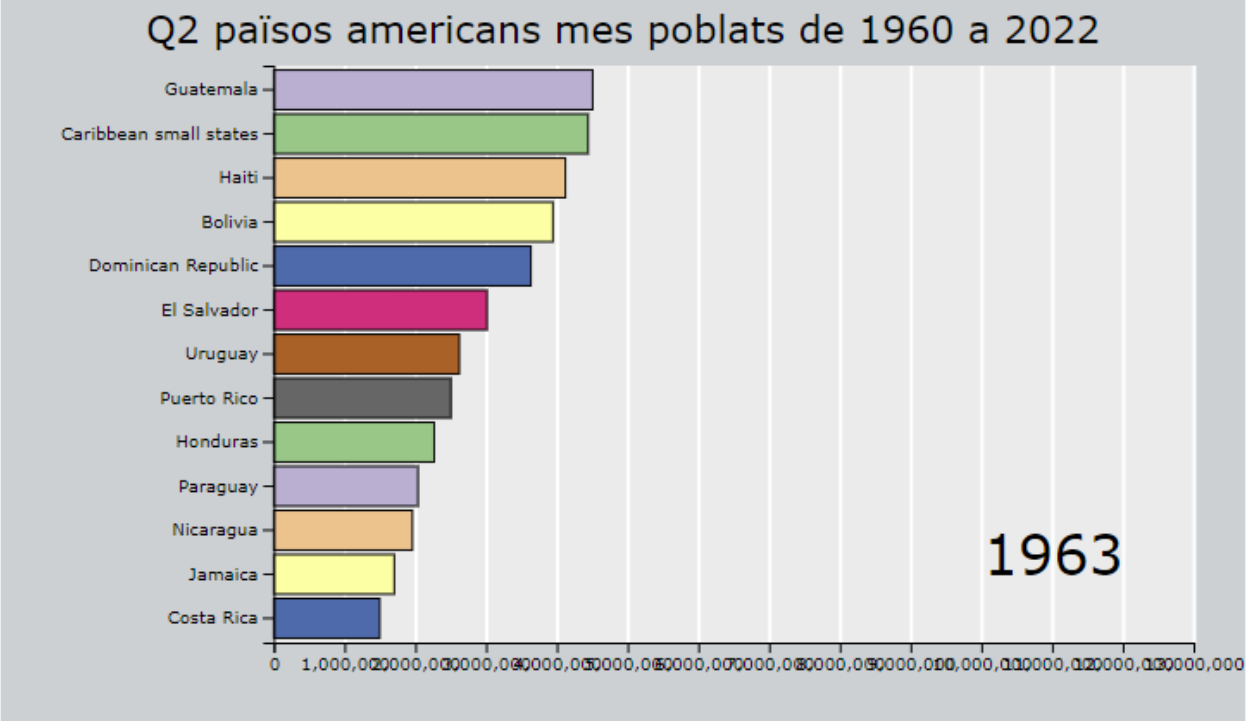
```
> PMon <- PWorld %>% filter(Continent == "America")
```

**PAS 2:** Afegim la variable *rank* que correspon al ranking decreixent per països de la població, i seleccionem els Q2 països de cada any segons la variable `rank`. El guardem en l'objecte `PMon_formatted`:

```
> PMon_formatted <- PMon %>%
  group_by(Any) %>%
  mutate(rank = rank(-Poblacio)) %>%
  filter(rank >= 0.25 * max(rank) & rank <= 0.5 * max(rank)) # Q2
```

**PAS 3:** Basant-nos en la comanda explicada en les transparències, podem generar els frames estàtics de les gràfiques per anys:

```
> barChartRace(PMon_formatted, x="Poblacio", y="Pais.Nom", time="Any", title = "Q2 països americans més poblats de 1960 a 2022")
```




**COMENTARI:** En general hi ha canvis en la zona alta, en tres primeres posicions del ranking dels Q2 països americans entre Guatemala, Haití, República Dominicana i Cuba. I en les darreres posicions predominen Costa Rica, panama, Uruguay, Puerto Rico.

En general tots els països creixen en població al llarg dels anys.

Cuba apareix l'any 1999 en primera osició i es manté uns anys en la primera posició, acabant l'any 2022 en la quarta posició.

**3.- Crea un *Animate Bar Race Ranking* sobre la primera meitat de països del Q1 MÉS poblats del món. Comenta tres aspectes significatius del que es veu en l'animació. Comenta l'evolució d'Espanya en aquesta llista.**

**RESPOSTA:**

**PAS 1:** DATA MASSAGING. Eliminem les entrades que tinguin "Continent="Zone", ens quedem només amb països, no zones geogràfiques.

```
> PMon <- PWorld %>% filter(Continent != "Zone")
```

**PAS 2:** Afegim la variable rank correspon al ranking creixent per població i seleccionem la primera meitat del Q1 de cada any segons la variable `rank`. El guardem en l'objecte *PMon_formatted*:

```
> PMon_formatted <- PWorld %>%
  group_by(Any) %>%
  mutate(rank = rank(-Poblacio)) %>%
  filter(rank <= 0.125 * max(rank))
```

**PAS 3:** Basant-nos en la comanda explicada en les transparències, podem generar en l'objecte `anim` els frames estàtics de les gràfiques per anys:

```
> anim <- ggplot(PMon_formatted, aes(rank, group = Pais.Nom,
```

```
            fill = as.factor(Pais.Nom), color = as.factor(Pais.Nom))) +
                   geom_tile(aes(y = Poblacio/2,
                   height = Poblacio,
                   width = 0.9), alpha = 0.8, color = NA) +
  geom_text(aes(y = 0, label = paste(Pais.Nom, " ")), vjust =0.2, hjust = 1) +
  geom_text(aes(y = Poblacio,label = Poblacio, hjust = 0)) +
  coord_flip(clip = "off", expand = FALSE) +
  scale_y_continuous(labels = scales::comma) +
  scale_x_reverse() +
  guides(color = "none", fill = "none") +
  theme(axis.line=element_blank(),
        axis.text.x=element_blank(),
        axis.text.y=element_blank(),
        axis.ticks=element_blank(),
        axis.title.x=element_blank(),
         axis.title.y=element_blank(),
        legend.position="none",
        panel.background=element_blank(),
        panel.border=element_blank(),
        panel.grid.major=element_blank(),
        panel.grid.minor=element_blank(),
        panel.grid.major.x = element_line( linewidth=.1, color="grey" ),
        panel.grid.minor.x = element_line( linewidth=.1, color="grey" ),
        plot.title=element_text(size=25,        hjust=0.5,        face="bold",
colour="grey", vjust=-1),
        plot.subtitle=element_text(size=18,     hjust=0.5,     face="italic",
color="grey"),
        plot.caption    =element_text(size=8,    hjust=0.5,    face="italic",
color="grey"),
        plot.background=element_blank(),
        plot.margin = margin(2,2, 2, 4, "cm")) +
  transition_states(Any, transition_length = 4, state_length=1, wrap=FALSE) +
  view_follow(fixed_x = TRUE) +
  labs(title = 'Població en el món. Any: {closest_state}',
       subtitle  =  "Top 15 Països Mundials",
       caption  = "Població | Data Source: www.kaggle.com")
```

**PAS 4:** A partir de l'objecte `anim`, creem els fitxers d'animació GIF i AVI:

```
# Exportar frames a fitxer GIF
> animate(anim, 200, fps = 20,  width = 1200, height = 1000,
        renderer  =  gifski_renderer("MeitatQ1Mundial.gif"),  end_pause  =  15,
        start_pause =  15)


# Exportar frames a fitxer AVI
> animate(anim, 200, fps = 20,  width = 1200, height = 1000,
        renderer   =   av_renderer("MeitatQ1Mundial.avi"),   end_pause   =   15,
        start_pause =  15)
```


**ALTERNATIVA: Llibreria ddplot**

**Primer de tot, instal·larem les llibreries que necessitem:**

```
> install.packages('ddplot')  # Càrrega paquet ddplot (primera vegada)
> library(ddplot)             # Generació Bar Chart Race
>
```

**PAS 1:** DATA MASSAGING. Eliminem les entrades que tinguin "Continent="Zone", ens quedem només amb països, no zones geogràfiques.

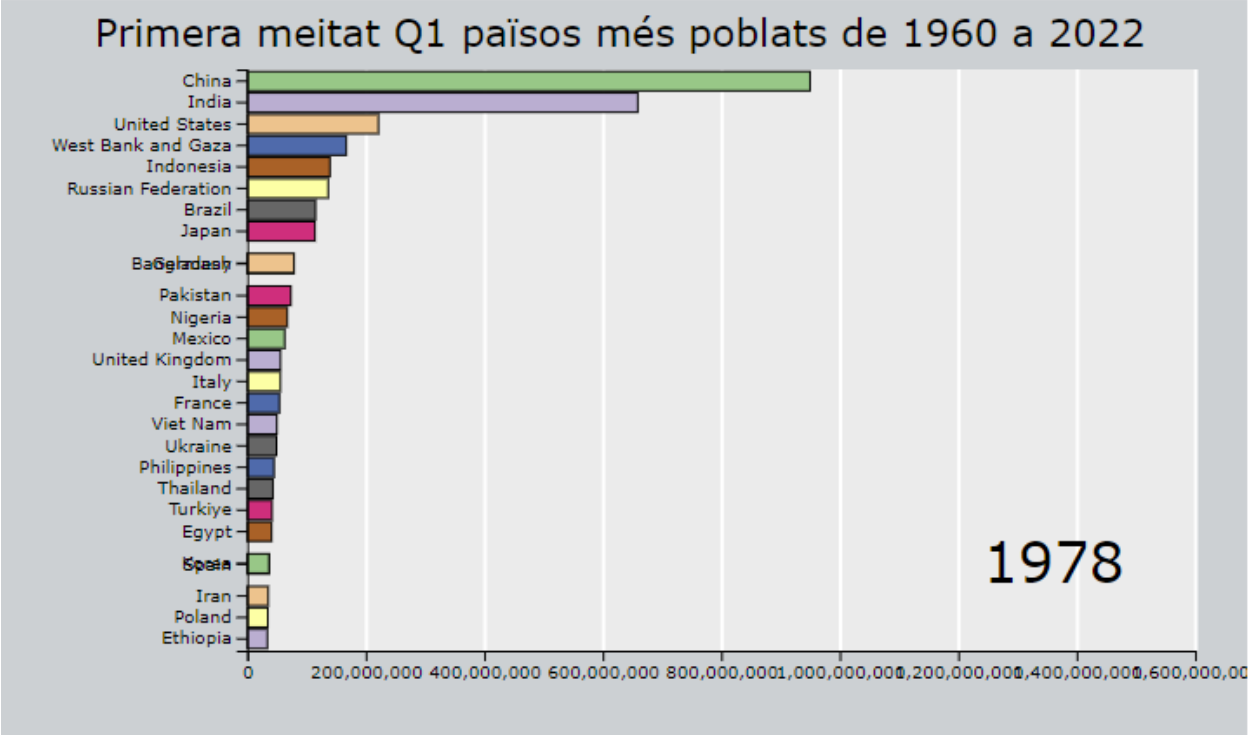
```
> PMon <- PWorld %>% filter(Continent != "Zone")
```

**PAS 2:** Afegim la variable rank correspon al ranking creixent per població i seleccionem la primera meitat dels Q1 països més poblats de cada any segons la variable `rank`. El guardem en l'objecte *PMon_formatted*:

```
> PMon_formatted <- PWorld %>%
  group_by(Any) %>%
  mutate(rank = rank(-Poblacio)) %>%
  filter(rank <= 0.125 * max(rank))
```

**PAS 3:** Basant-nos en la comanda explicada en les transparències, podem generar els frames estàtics de les gràfiques per anys:

```
> barChartRace(PWorld_formatted, x="Poblacio", y="Pais.Nom", time="Any", title
= "Primera meitat Q1 països més poblats de 1960 a 2022")
```




**COMENTARIS:**

1. China s'ha mantingut al llarg dels anys com el país més poblat del món fins l'any 2022.
2. India ha fet un creixement fort de població el s.XXI fins a arribar a superar China l'any 2022.
3. Altres països asiàtics com Indonesia, Pakistan, Bangladesh han tingut augments significatius de població passant a posicions entre els 8 primers.
4. Pel que fa als països africans només Nigeria ha tingut un augment significatiu de població, seguit d'Ethiopia.
5. Russian Federation, el país amb més extensió del món ha anat baixant posicions al llarg dels anys.
6. Pel que fa a països americans, USA es manté en la tercera posició pel que fa a població, seguit de més lluny de Mèxic que ha anat pujant al llarg dels anys fins l'11a posició.
7. Els països europeus (Alemanya, UK, Italy) han anat baixant posicions en el rànking en favor dels països asiàtics.
8. Espanya apareix des dels anys 160 en posicions mitja-baixa de la taula i desapareix l'any 1993. I torna a aparèixer esporàdicament els anys 2009, 2010 i 2011.

## *4.* PART 3. Aplicacions interactives amb `Shiny`

**En aquesta tercera part del seminari implementarem algunes gràfiques interactives amb la llibreria `Shiny`**

**RESPOSTA:**

**Primer de tot, instal·larem les llibreries `Plotly` i `Shiny`.**

```
> library (plotly)
```

```
> install.packages('shiny') # Càrrega paquet shiny (1a vegada)
> library (shiny)
```

**I seguirem treballant amb el mateix data set:**

```
> setwd("C:/Users/enric/Documents/R")
> PWorld <- read.csv('./Poblacio_Mundial_1960-2022.csv')
> str(PWorld)
```

## EXERCICIS:

**1.- Reprodueix l'aplicació shiny amb entrada per desplegable que dibuixi la gràfica de línies de població per països, de forma que es puguin seleccionar els països en el desplegable. Posa com a país per defecte el nom 'Spain'. Utilitza la funció `plot_ly()` o `ggplot()`, la que vulguis.**

**Comenta la gràfica de la població dels països de molta extensió com Germany, France, Italy, Poland i Ukraine amb la d'Espanya. Compara aquests països amb Espanya i comenta la seva evolució.**

**RESPOSTA:**

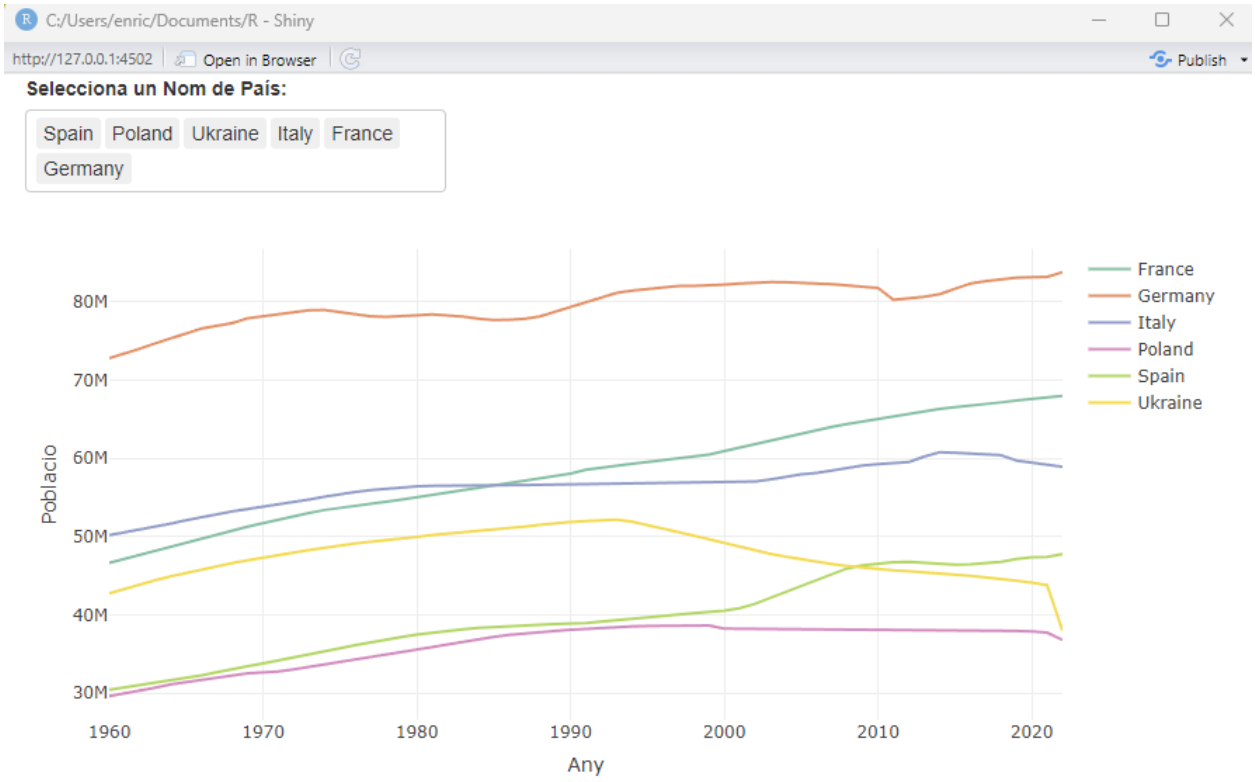
**PAS 1:** DATA MASSAGING. Ens quedem només amb països europeus:

```
> PEuropa <- PWorld %>% filter(Continent == "Europe")
```

**PAS 2:** Crear l'aplicació shiny:

```
# Aplicació Shiny amb plot_ly
ui <- fluidPage(  selectizeInput(
                              inputId = "NomsPais",
                              label = "Selecciona un Nom de País:",
                              choices = unique(PEuropa$Pais.Nom),
                              selected = "Spain",
                              multiple = TRUE
                                 ),
                  plotlyOutput(outputId = "plot")
           )

server <- function(input, output, ...)
      {     output$plot <- renderPlotly (
            {     plot_ly(PEuropa, x = ~Any, y=~Poblacio, color=~country) %>%
                  filter(Pais.Nom %in% input$NomsPais) %>%
                  group_by(Pais.Nom) %>%
                  add_lines()
            })
      }
shinyApp(ui, server)



# Aplicació Shiny amb ggplot
```

```
ui <- fluidPage(  selectizeInput(
                              inputId = "NomsPais",
                              label = "Selecciona un Nom de País:",
                              choices = unique(PEuropa$Pais.Nom),
                              selected = "Spain",
                              multiple = TRUE
                              ),
                        plotlyOutput(outputId = "plot")
          )

server <- function(input, output, ...)
      {     output$plot <- renderPlotly(
            {     plotNomP <- ggplot(filter(PEuropa,Pais.Nom %in%
                              input$NomsPais),
                              aes(x=Any,y=Poblacio,color=Pais.Nom))          +
                              geom_path()
                  ggplotly(plotNomP)
            })
      }
shinyApp(ui, server)
```




**COMPARATIVA PAÏSOS ALEMANYA, POLAND, ITALY, FRANCE, UKRAINE AMB ESPANYA:**

1. Germany i France es mantenen per sobre de la població d'Espanya al llarg de tots els anys, essent la població alemanya 2,5 vegades més gran que l'espanyola i la francesa 1,5 vegades aproximadament
2. La població d'Ukraine ha sigut superior a la d'Espanya, però l'any 2009 la població espanyola sobrepassa la d'Ukraine.que havia baixat 7M de 1994 (52M) a 2009 (45M), quedant en 38M l'any 2022.
3. Poland fins l'any 2000 ha tingut una població una mica inferior a l'espanyola, però a partir del 2000 la població espanyola ha tingut un major creixement i la polaca ha disminuitr una mica o s'ha estancat, baixant 1M de persones del 2021 al 2022.
4. La població italiana sempre ha estat superior a l'espanyola amb un creixement similar fins l'any 2012, però a partir de l'any 2014 ha patit un cert descens d'uns 2M entre 2014 a 2022.
5. Destaca l'augment de la població espanyola els primers 10 anys del .XXI, quedant després l'augment més estancat.
6. Ukraine ha patit una davallada de població a partir del 1994 (desaparició URSS) i especialment a partir del 2021 per la guerra d'Ukraine. De ser el país més poblat dels 5 fins el 1993 amb distància, a passat a la quarta posició dels cinc.
7. Poland ha tingut un creixement similar a Ukraine fins el 1993, tenint una lleugera davallada de població partir d'aquell any, quedant actualment en cinquena posició.

**2.- Reprodueix l'aplicació shiny amb entrada per desplegable que dibuixi la gràfica de línies de la població per països de tot el món, de forma que es puguin seleccionar els països en el desplegable. Posa com a país per defecte el nom 'Spain'. Utilitza la funció `plot_ly()` o `ggplot()`, la que vulguis.**

**Comenta la gràfica de la població d'Argentina, Canada, Ethiopia i South Africa amb la d'Espanya. Compara aquests països amb Espanya i comenta la seva evolució.**

**RESPOSTA:**

**PAS 1:** DATA MASSAGING. Eliminem les entrades que tinguin "Continent="Zone", ens quedem només amb països, no zones geogràfiques.

```
> PMon <- PWorld %>% filter(Continent != "Zone")
```

**PAS 2:** Crear l'aplicació shiny:

```
# Aplicació Shiny amb plot_ly
ui <- fluidPage(  selectizeInput(
                              inputId = "NomsPais",
                              label = "Selecciona un Nom de País:",
                              choices = unique(PMon$Pais.Nom),
                              selected = "Spain",
                              multiple = TRUE
                                 ),
                  plotlyOutput(outputId = "plot")
          )

server <- function(input, output, ...)
      {     output$plot <- renderPlotly (
```

```
            {     plot_ly(SWorldP, x = ~Any, y = ~Poblacio,
                          color=~Pais.Nom) %>%
                  filter(Pais.Nom %in% input$NomsPais) %>%
                  group_by(Pais.Nom) %>%
                  add_lines()
            })
      }
shinyApp(ui, server)


# Aplicació Shiny amb ggplot
ui <- fluidPage(  selectizeInput(
                          inputId = "NomsPais",
                          label = "Selecciona un Nom de País:",
                          choices = unique(PMon$Pais.Nom),
                          selected = "Spain",
                          multiple = TRUE
                          ),
                    plotlyOutput(outputId = "plot")
           )

server <- function(input, output, ...)
      {     output$plot <- renderPlotly(
            {     plotNomP <- ggplot(filter(SWorldP,country %in%
                          input$NomsPais),
                          aes(x=Any,   y=Poblacio,   color=Pais.Nom))   +
                          geom_path()
                  ggplotly(plotNomP)
            })
      }
shinyApp(ui, server)
```


**COMPARATIVA PAÏSOS ARGENTINA, CANADA, ESPANYA, ETHIOPIA i SOUTH AFRICA:**

1. La població espanyola parteix en primera posició l'any 1960 amb 30M i va tenint un lleuger creixement de població, que creix una mica en el període 2001 a 2011 quedant en 47M l'any 2022.
2. Argentina i Canada tenen una població inferior a l'espanyola durant tots els anys, tot i tenir major superfície. El creixement d'aquests dos països ha sigut sempre superior a l'espanyol, especialment els darrers anys, de 2009 a 2022.
3. La població de Canadà ha sigut sempre inferior a la d'Argentina.
4. South Africa parteix en la darrera posició d'aquests països l'any 1960 amb 16M i va tenint al llarg dels anys un creixement molt superior a Espanya, superant-la l'any 1990 i quedant en segona posició l'any 2022 amb 59M d'habitants.
5. La població d'Ethiopia parteix de la segona posició l'any 1960 (21M) tenint un moderat creixement a partir de l'any 1980 fins a arribar a 123M l'any 2022. La població etíop sobrepassa l'espanyola l'any 1984.
6. La població canadenca ha tingut també un creixement notable semblant a la d'Argentina, doblant el Canadà la població en 60 anys (de 17M a 1960 a 38M a 2022), passant de la quarta posició el 1960 a la darrera el 2022.
7. Argentina ha tingut un comportament similar al Canadà però amb població superior a la canadenca, doblant també la població en 60 anys (de 20M a 1960 a 47M a 2022).

**3. Executant l'aplicació, busca tres països que t'agradin i compara la seva població durant el període 1960-2022.. Què pots deduir de les gràfiques?. Inspira't amb les qüestions de les preguntes 2 i 3 de la part 1.**

**RESPOSTA:**

Enric Martí Gòdia
Bellaterra, Maig 2025