[Support] Training Course : Discovery of Dplyr & GGplots packages

Data Center of University of Brittany

Paul Pinard

Extensions / Packages :

- Packages to charge / install before starting the tutorial

```
# install.packages("*****")

library(datasets)

library(dplyr)

library(ggplot2)

library(tidyr)

library(DT)

library(catdata)

library(car)
```

Practice Dataset :

- Dataset Mtcars & Starwars will be used as exemple for the hole training course

```
data(mtcars)
starwars <- as.data.frame(starwars)[,1:11] %>%
                  mutate(gender = as.factor(gender)) %>%
                  mutate(across(where(is.character), as.factor))
```

- For practice, you can either use your dataset or the practice one given to you.
- This dataset is :
    - Fictionnal
    - Realised from real ones as base

```
data_training <- read_excel("data_entrainement_eng.xlsx") %>%
                                  mutate_if(is.character, as.factor)
```

Dplyr & managing your dataset :

- Why use Dplyr ?
  - Easier to change values from one to multiple columns
  - Create new variables
  - Create summaries
  - Filtering the dataset
  - And so on…
- How ?
  - Need a dataset
  - At the end of each line, needs to add a « **PIPE** » (%>%)
    - *Except the last line where there should be nothing (eg. Dataset importation code !)*
- Which functions ?
  - Mutate
  - Filter
  - Select
  - Group_by
    - Usually used with the function « summarise » or « mutate »
  - Left / Right / inner / Full_join
  - Arrange

➢ How the tutorial will work from this point onward :

❖ An exemple with a question and two answers (one WITH dplyr and one WITHOUT)

❖ A new question to answer using dplyr and the dataset you choose

✓ ***Exemple 1*** *: Filter only characters who have brown eyes and light skin color.*

```
starwars[starwars$skin_color == "light" & starwars$eye_color == "brown"
, ]

starwars %>% filter(skin_color == "light", eye_color == "brown")
```

✓ ***Exercice 1*** *: Filter only individuals who are over 1m70 and who live in the countryside.*

```
# TO_DO
```

✓ ***Exemple 2*** *: Order by height (from largest to smallest) then, subsequently, by mass.*

```
starwars[order(-starwars$height, starwars$mass), ]

starwars %>% arrange(desc(height), mass)
```

✓ ***Exercice 2 :*** *Order by car brand, then by employment situation (descending)*

✓ `# TO_DO`

✓ ***Exemple 3*** *: Keep only the "name", "height", "sex" and "skin_color" columns*

```
starwars[,c("name","height","sex","skin_color")]

starwars %>% dplyr::select(name,height,sex,skin_color)
```

✓ ***Exercice 3 :*** *Keep the columns corresponding to age, place of living, weight and nationality*

```
# TO_DO
```

✓ ***Exemple 4*** *: Rename columns currently in English to French*

```
colnames(starwars)[colnames(starwars) == "name"] <- "NOM_PERSONAGE"

data("starwars")
starwars %>% rename(NOM_PERSONAGE = name)
```

✓ ***Exercice 4 :*** *Lowercase the name of 3 columns of your choice*

✓ ***Bonus :*** *Change all columns to lowercase (ask if there is a problem)*

✓ *# TO_DO*

✓ ***Exemple 5*** *: Create 2 new variables (Height (in meters) & BMI)*

```
starwars$hight_m <- starwars$height/100
starwars$IMC <- starwars$mass / (starwars$hight_m^2)

starwars %>% mutate(height_m = height / 100,
                    IMC = mass / (height_m^2))
```

✓ ***Exercice 5 :***

- *Change the "height" variable from meters to Feet **WITHOUT** creating a new variable*
- *For the Weight variable, change it from Kg to Lbs **BY** creating a new variable*
  - *Conversion (1m = 3.28 ft / 1kg = 2.20 Lbs)*

✓ ***Bonus :*** *Round calculations to 2 decimal (Hint : function round())*

 # TO_DO

✓ ***Exemple 6 :*** *Calculate the average of the "Height" variable based on the sex of the individual*

```
starwars_filtered <- starwars[complete.cases(starwars$height), ]
moyenne_taille_sexe <- tapply(starwars_filtered$height,
starwars_filtered$sex, mean)
```

```
starwars %>%
  group_by(sex) %>%
  summarise(Moyenne_taille_sexe = mean(height, na.rm = TRUE))
```

✓ ***Exercice 6 :*** *Calculate the mean and standard deviation of age based on family situation AND place of living (Hint : function standard deviation = sd())*

 # TO_DO

✓ ***Exercice Bonus 1 :*** *Change all "Character" type columns into "factor"*

✓ *# TO_DO*

✓ ***Exercice Bonus 2 :*** *Sort car brands in descending order based on Gender and "Owner" variable*

 # TO_DO

✓ ***Exercice Bonus 3 :***

- *Obtain the number of different employment situations depending on nationality*
- *Put this number as a percentage*

 # TO_DO

- To always emphasize the comparison between R Base and Dplyr, the following two pieces of code are examples one **WITH** and the other **WITHOUT** dplyr to show complete practical uses. They have the same purpose. Chronologically, the code goes :
  - *Deleting rows with a NA in the "Height" column*
  - *Selection of certain variables that we keep*
  - *Creating a new variable based on an existing one (**ifelse**)*
  - *Recovery of an average size and weight depending on the species*
  - *Join between our original and the table of averages*
  - *Rearrangement of columns in another direction*
  - *Changing column names (**only for code without dplyr**)*

```
# WITHOUT Dplyr
starwars_sans_na <- subset(starwars, !is.na(height))
starwars_sans_na_select <- subset(starwars_sans_na_select, select = c(name, h
eight, mass, species))
starwars_sans_na_select$height_category <- ifelse(starwars_sans_na_select$hei
ght < 170, "Short", ifelse(starwars_sans_na_select$height < 180, "Medium", "T
all"))

Especes_resume <- aggregate(cbind(height, mass) ~ species, data = starwars_sa
ns_na_select, FUN = function(x) mean(x, na.rm = TRUE))

starwars_final <- merge(starwars_filtered, species_summary, by = "species")
starwars_final <- starwars_final[,c("species", "name", "height.x", "mass.x",
"height.y", "mass.y")]

colnames(starwars_final) <- c("species", "name", "height ", "mass", "Moyenne_
Espece_Taille", "Moyenne_Espece_Poids"))
```

```
# WITH Dplyr
starwars %>%
  filter(!is.na(height)) %>%
  dplyr::select(name, height, mass, species) %>%
  mutate(height_category = ifelse(height < 170, "Short",

                                  ifelse(height < 180, "Medium", "Tall"))) %>%
  group_by(species) %>%
  summarise(mean_height = mean(height, na.rm = TRUE),
            mean_mass = mean(mass, na.rm = TRUE)) %>%
  left_join(starwars, by = "species") %>%
  dplyr::select(rank, species, name, height, mass, mean_height, mean_mass)
```

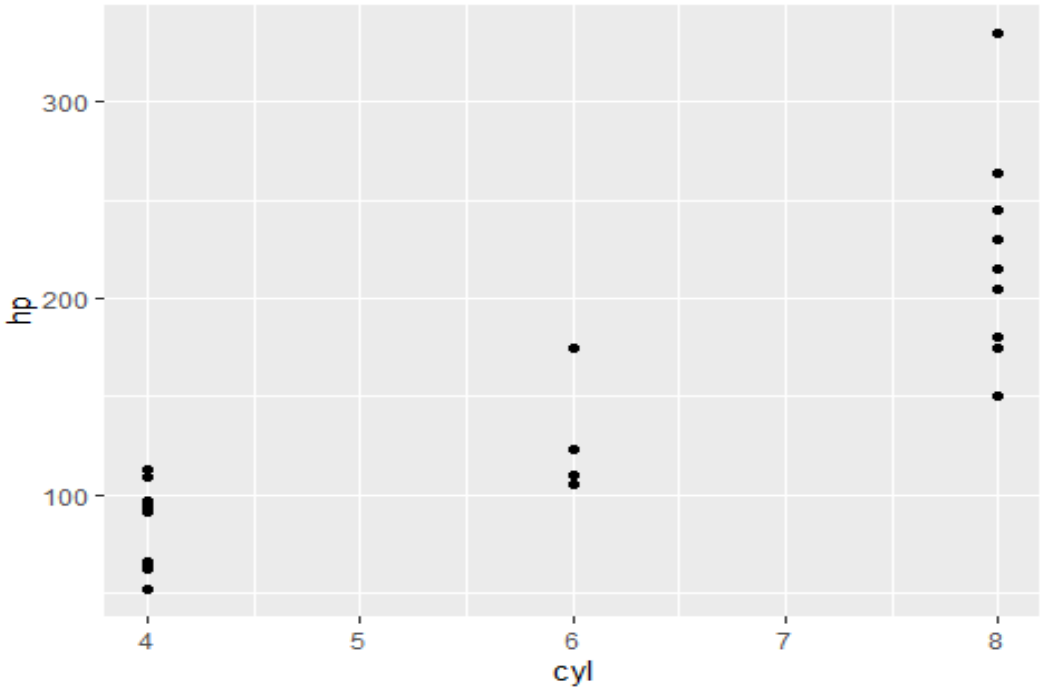
GGplot & graphics creation :

- Ups :
  - A lot of flexibility at the disposal of the user
  - Possibility of adding an *"unlimited"* number of variables
  - Ability to put several types of graphics into a single one
    - *Put a histogram and a curve or several histograms*
- Down :
  - Need to have 2 variables most of the times
    - *Histogram hist() from basic R can create a graph with a single input variable*
  - Can struggle if you have long texts or lots of groups

➢ Process further down :
- ❖ Creation of a simple graph which will be expanded later
- ❖ In the same way as the "**Pipe**", need to add a **"+"** at the end of each line except the last
- ❖ An example first, then an exercise
- ❖ The graph shows what the "example" code produces.

- Create a chart :
  - The exemple :
    - In X-axis, the number of cylinders
    - In Y-axis, the number of horsepowers
    - Representation by point
  - Using data_training :
    - In X-axis, the height
    - In Y-axis, the weight
    - Representation by point
    - Save it to an object named « Graph_Appr »

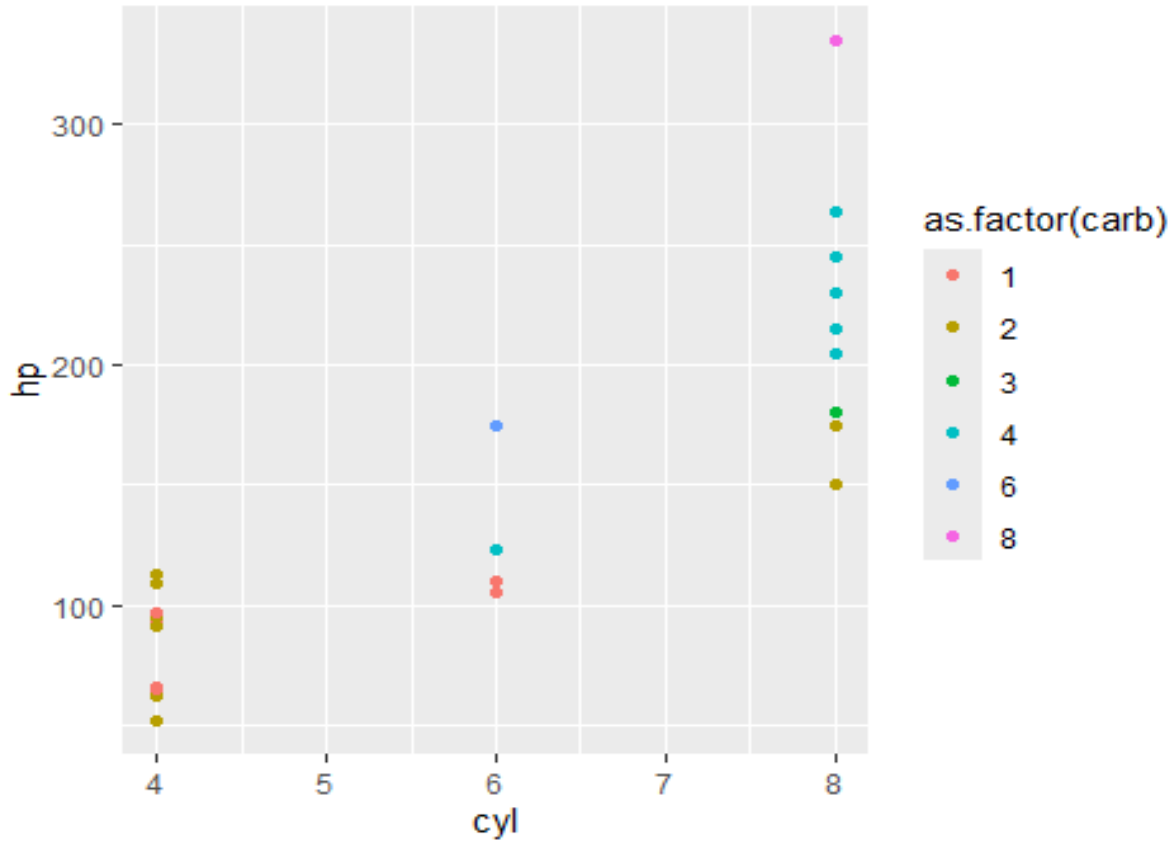
```
ggplot(data = mtcars, aes(x = cyl,y = hp)) +
  geom_point()

ggplot(data = mtcars) +
  geom_point(aes(x = cyl,y = hp))

ggplot() +
  geom_point(data = mtcars, aes(x = cyl,y = hp))
```



- Add on the same chart :
  - The exemple :
    - Color according to the number of carburetor for each car
  - Using data_training :
    - Color for the sex of individuals
    - Different forms depending on the number of dependent children
      - ✓ *Hint : the parameter to use is 'shape'*

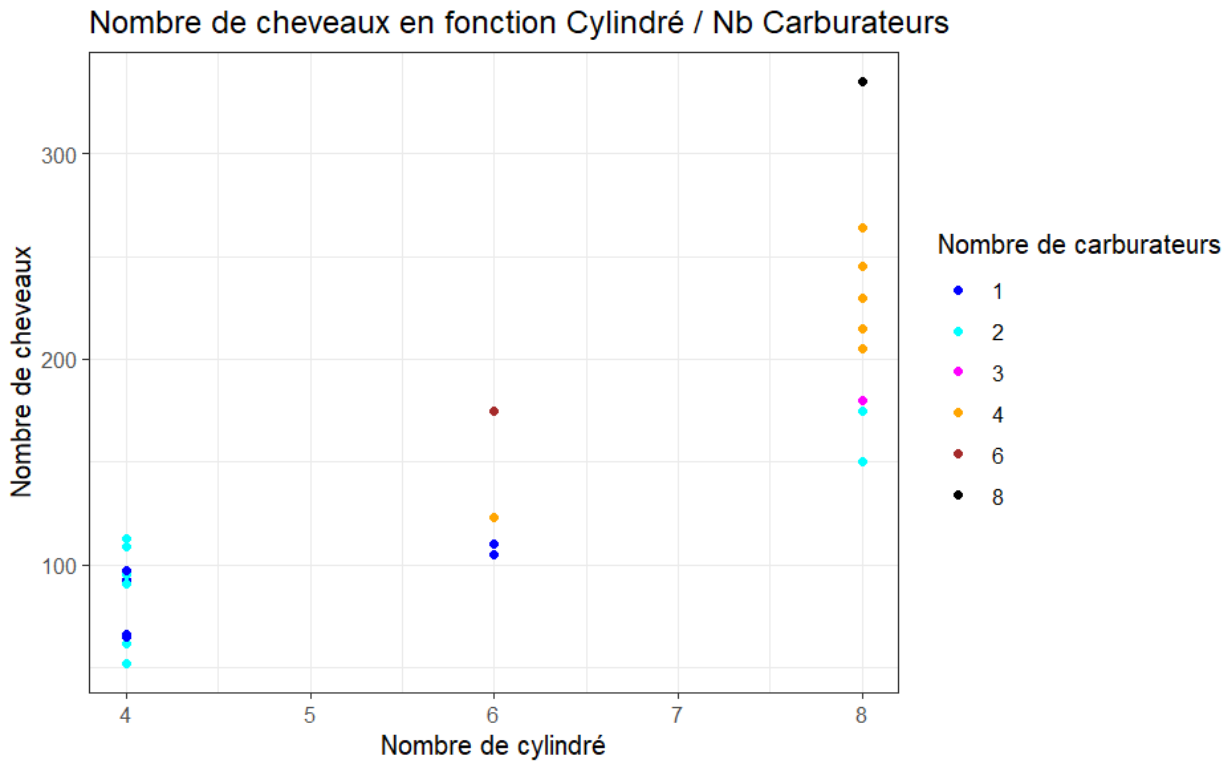
Nb : Whether it is to add color or shapes, do not forget to put the desired variable in factor !

```
ggplot(data = mtcars) +
  geom_point(aes(x = cyl,y = hp, color = as.factor(carb)))
```



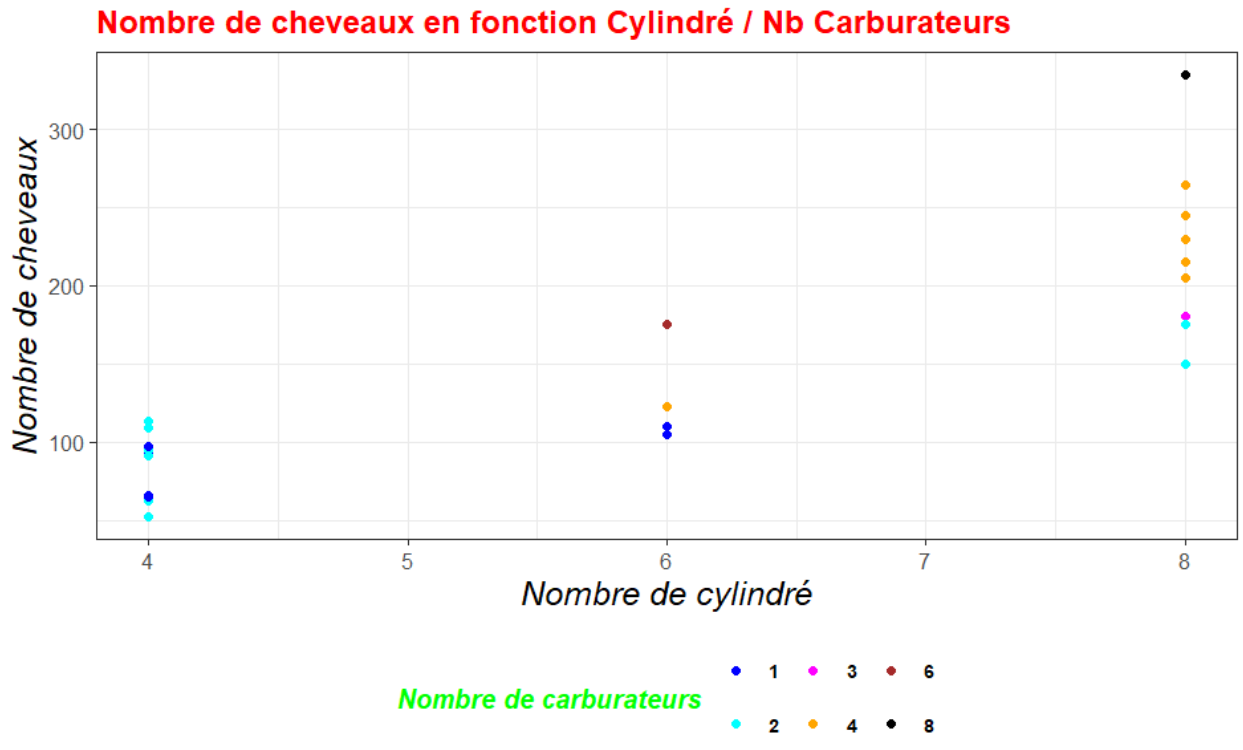
- Customize Chart :
  - The exemple :
    - Change chart title, X/Y axis and legend name
    - Change the colors set automatically by manual colors (blue, cyan, magenta, orange, brown, and black)
    - Add a theme (background of the chart)
  - Using data_training :
    - Change the name of BOTH legends
    - Add the theme of your choice
    - Manually change colors AND shapes
      - ✓ *For forms, hints :* `values = c(3,4,15,17,19)`
      - ✓ *https://www.sthda.com/english/wiki/ggplot2-point-shapes*

```
ggplot(data = mtcars) +
  geom_point(aes(x = cyl,y = hp, color = as.factor(carb))) +
  theme_bw() +
  labs(title = "Nombre de cheveaux en fonction Cylindré / Nb Carburateurs",
       x = "Nombre de cylindré",
       y = "Nombre de cheveaux",
       color = "Nombre de carburateurs")+
  scale_color_manual(values = c("blue", "cyan", "magenta", "orange", "brown",
"black"))
```



- Finishing touches :
  - The exemple :
    - Use the theme() argument for :
      - ✓ Change the position of the legend at the bottom of the chart, increase it's font size, put it in bold
      - ✓ Change axis titles to italic with size 14
      - ✓ Put the main title in red and bold
      - ✓ Put the legend title in bold italic and green
  - Using data_training :
    - Use your imagination !
    - For more examples, see below on the support (3 complete examples)

```
ggplot(data = mtcars) +
  geom_point(aes(x = cyl,y = hp, color = as.factor(carb))) +
  theme_bw() +
  labs(title = "Nombre de cheveaux en fonction Cylindré / Nb Carburateurs",
       x = "Nombre de cylindré",
       y = "Nombre de cheveaux",
       color = "Nombre de carburateurs")+
  scale_color_manual(values = c("blue", "cyan", "magenta", "orange", "brown",
"black")) +
  theme(legend.text = element_text(size = 8, face = "bold"),
        legend.position = "bottom",
        axis.title = element_text(face = "italic", size = 14),
        plot.title = element_text(face = "bold", color = "red"),
        legend.title = element_text(face = "bold.italic", color = "green"))
```



- A few more complete examples :
  - First Graph :
    - Summarizes information from 3 columns
    - Boxplots of the weights of individuals over 3 different time periods and with 6 groups per time
  - Second Graph :
    - Summarizes information from 5 columns
    - For each department of Brittany, a graph of the income earned by fictitious individuals according to their age, gender and education
  - Third Graph :
    - More useful for graphs with several Y in the same values of X (for example, comparing the salary between men and women according to their ages OR the amount of rain fell in 3-4 different cities over several years)
    - Data are full random and has no pratical applications like the last two graphs

Note: The data frame that is used for each chart can be copied from this document.

```
windowsFonts()
## [1] "TT Times New Roman"
## [1] "TT Arial"
## [1] "TT Courier New"

install.packages("extrafont")

library(extrafont)

font_import()

loadfonts(device = "win")
```

Data Frame Graph 1 :

```
crea_df <- data.frame(Periode = rep(c("0-10", "11-20", "21-31"), each = 12),
                      Group = rep(1:6, each = 2, times = 3),
                      Weight_Pers = round(runif(36, min = 60, max = 88), 2),
                      Poids_Pers = round(runif(36, min = 175, max = 194), 2))
%>%
                                    mutate(Periode = as.factor(Periode)) %>%
                                     mutate(Groupe = as.factor(Groupe))
```

Data Frame Graph 2 :

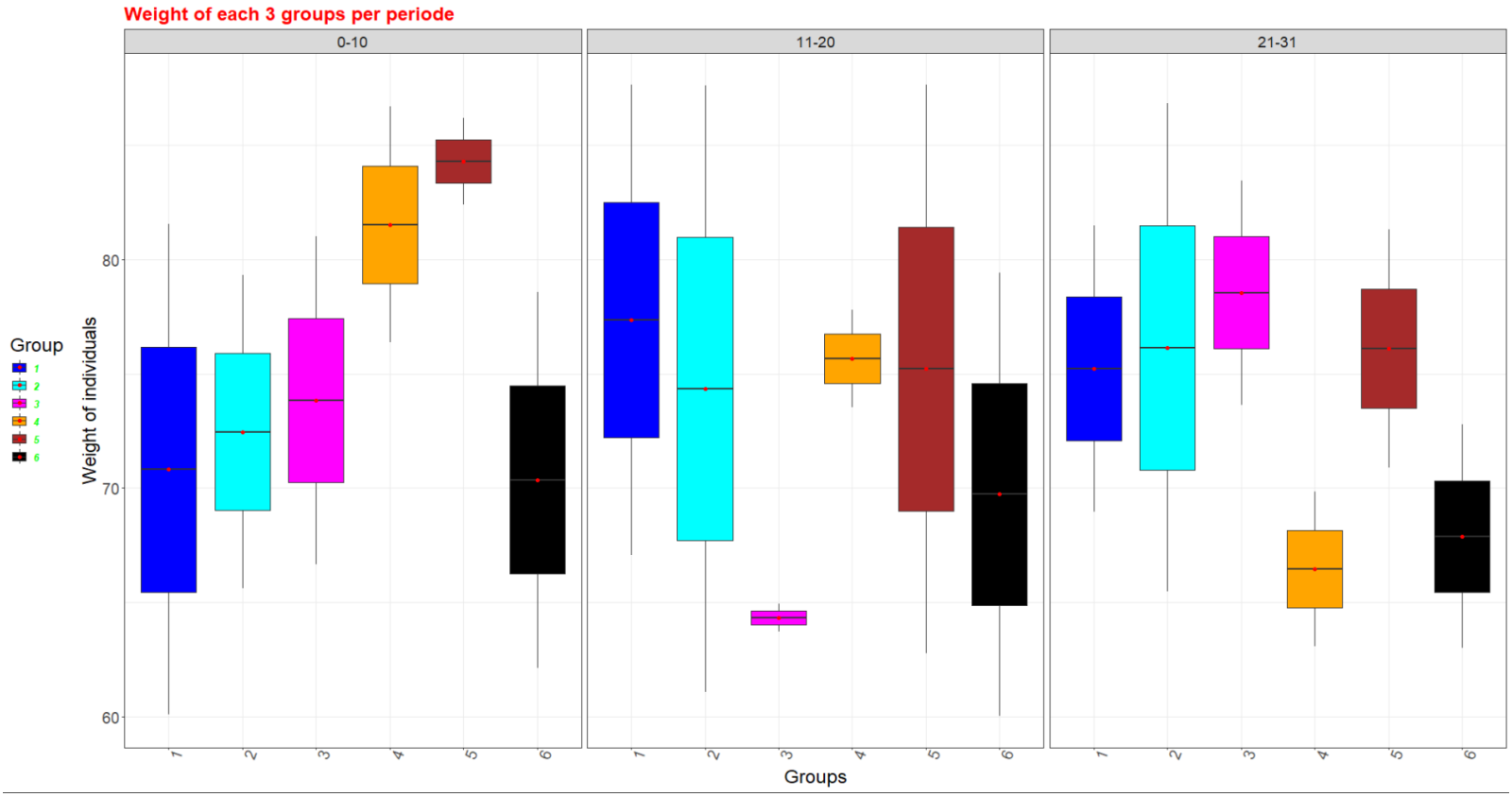
```
set.seed(1234)
n <- 150

age <- sample(18:80, n, replace = TRUE)
income <- rnorm(n, mean = 50000, sd = 20000)
gender <- sample(c("Homme", "Femme"), n, replace = TRUE, prob = c(0.5,0.5))
education <- sample(c("HighSchool", "Bachelor", "Master", "PhD"), n, replace
= TRUE)
Dep <- sample(c("Finistère", "Côte D'Armor", "Ille et Vilaine", "Morbihan"),
n, replace = TRUE)

data_plus <- data.frame(Age = age, Income = income, Genre = gender, Education
= education, Dep = Dep)
```

Data Frame Graph 3 :

```
data <- data.frame(x = 1:20,
                   y1 = rnorm(20, 0, 1),
                   y2 = rnorm(20, 0, 1),
                   y3 = rnorm(20, 0, 1),
                   y4 = rnorm(20, 0, 1))
```

Graph n°1:

```
ggplot(data = crea_df,aes(x=Group, y = Weight_Pers, fill=Group))+
          theme_bw()+
          geom_boxplot()+
          ggtitle("Weight of each 3 groups per periode")+
          stat_summary(geom = "point",col="red",fun = mean)+
          scale_fill_manual(values = c("blue", "cyan", "magenta", "orange", "
brown", "black"))+
          facet_wrap("Periode")+
          labs(x= "Groups", y= "Weight of individuals")+
          theme(axis.text.x=element_text(angle=70),
                axis.text = element_text(size = 15),
                legend.text = element_text(size = 10, face = "bold.italic", c
olor = "green"),
                legend.position = "left",
                title = element_text(size = 18),
                plot.title = element_text(size = 18, face = "bold", color = "
red"),
                strip.text = element_text(size=14))
```



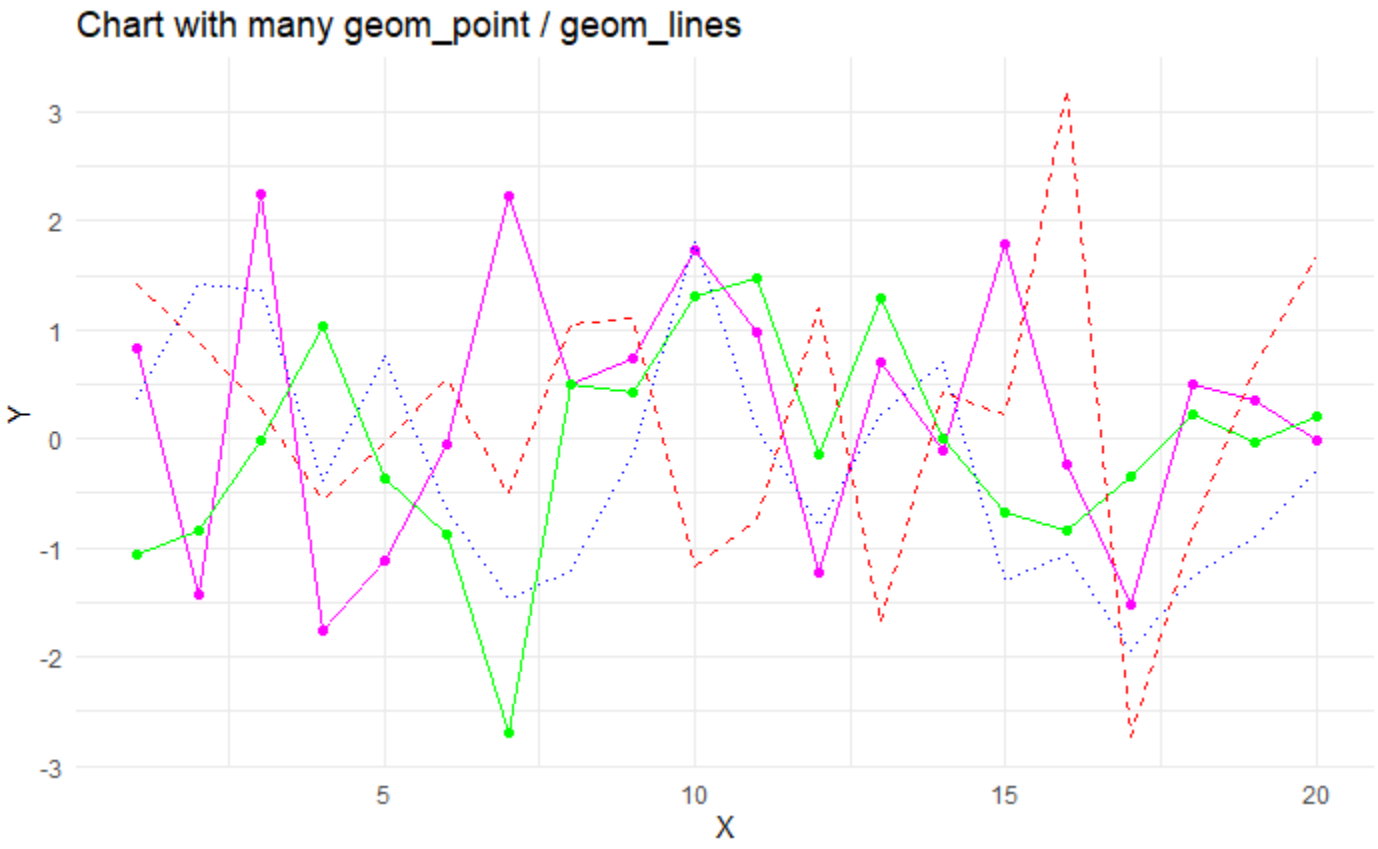
Graph n°2:

```
ggplot(data = data_plus, aes(x = Age, y = Revenus, color = Genre, shape = Education)) +
  geom_point(size = 3, alpha = 0.8) +
  facet_wrap(~Dep, scales = "free") +
  theme_bw() +
  scale_colour_manual(values = c("blue","red"))+
  scale_shape_manual(values = c(16,15,17,18))+
  labs(title = "Income depending the age and the place of living",
       x = "Age",
       y = "Income",
       color = "Gender",
       shape = "Education")
```

Graph n°3:

"Simple" Version :

```
ggplot(data, aes(x = x)) +
  geom_point(aes(y = y3), color = "magenta") +
  geom_line(aes(y = y3), color = "magenta") +
  geom_line(aes(y = y1), color = "blue", linetype = "dotted") +
  geom_point(aes(y = y4), color = "green") +
  geom_line(aes(y = y4), color = "green") +
  geom_line(aes(y = y2), color = "red", linetype = "dashed") +
  labs(title = "Chart with many geom_point / geom_lines",
       x = "X",
       y = "Y") +
  theme_minimal()
```



“Complexe” Version :

```
data_long  <- data %>%
  pivot_longer(cols = start_with("y"), names_to = "Variable", values_to = "Va
lue")


ggplot(data, aes(x = x, y = Value, color = Variable, linetype = Variable)) +
  geom_point()+
  geom_line() +
  labs(title = "Chart with a line for each variable",
       x = "X",
       y = "Y",
       color = "Variable",
       linetype = "Variable") +
  theme_minimal()
```