GGPLOT2

Maxime & Siva

23/11/2020

I. PRESENTATION

ggplot2 est un package R spécialement conçu pour la visualisation de données et pour fournir la meilleure analyse exploratoire des données. Il fournit de belles parcelles sans tracas qui prennent soin des détails infimes comme dessiner des légendes et les représenter. Les tracés peuvent être créés de manière itérative et modifiés ultérieurement. Ce package est conçu pour fonctionner en couches, en commençant par une couche montrant les données brutes collectées lors de l'analyse exploratoire des données avec R puis en ajoutant des couches d'annotations et de résumés statistiques.

Ce paquet fonctionne sous une grammaire profonde appelée «Grammaire des graphiques» qui est composée d'un ensemble de composants indépendants qui peuvent être créés de plusieurs façons. La «grammaire des graphiques» est la seule raison qui rend ggplot2 très puissant car le développeur R n'est pas limité à un ensemble de graphiques pré-spécifiés qui sont utilisés dans d'autres packages. La grammaire comprend un ensemble simple de règles et de principes fondamentaux.

En 2005, Wilkinson a créé ou plutôt créé le concept de grammaire des graphiques pour décrire les caractéristiques profondes qui sont incluses entre tous les graphiques statistiques. Il se concentre sur les couches primaires qui incluent l'adaptation des fonctionnalités intégrées avec R.

II. INSTALLATION DE GGPLOT2

Pour installer le package ggplot2, on utilise la syntaxe suivante sous R : install.packages(ggplot2)

Pour charger le package, on utilise la syntaxe suivante :

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.6.2
```

Pour comprendre les besoins du package et les fonctionnalités de base, R fournit une aide integrée et la syntaxe pour retrouver l'aide concernant le package ggplot2 est la suivante :

```
help(ggplot2)
```

```
## starting httpd help server ... done
```

Installation et chargement du Package pour grouper les images sur une page

```
#install.packages("gridExtra")
library(gridExtra)
```

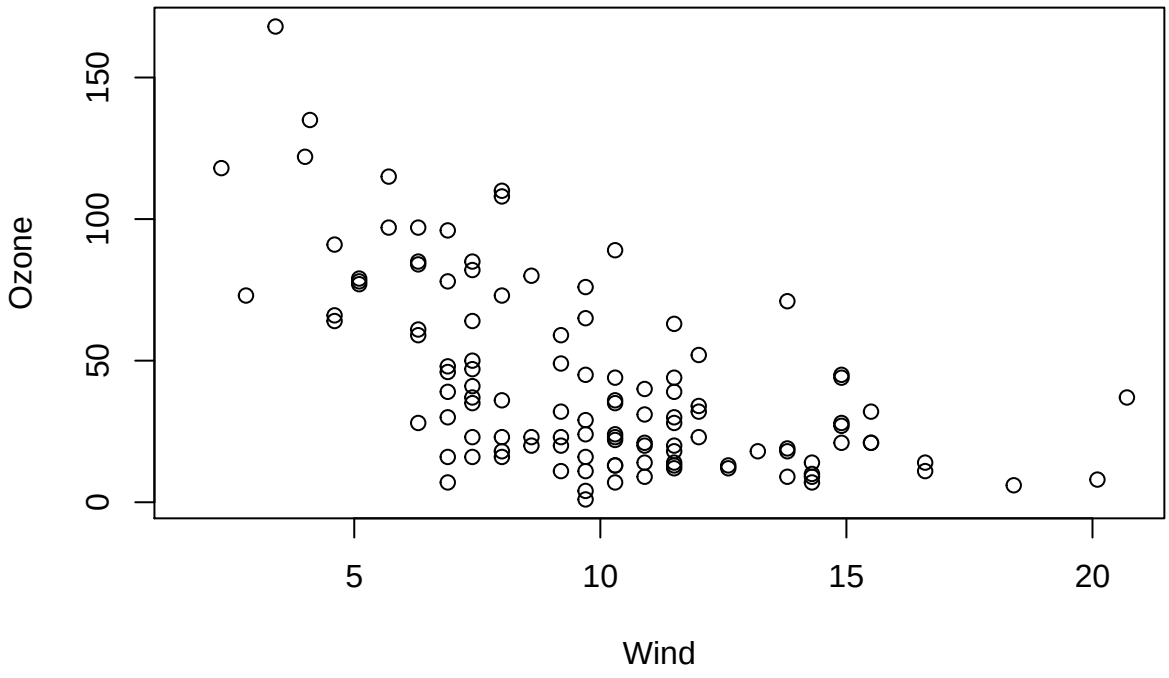
```
## Warning: package 'gridExtra' was built under R version 3.6.3
```

III. LES GRAPHIQUES CONVENTIONNELLES Nous allons proceder dans cette partie à l'affichage des graphiques conventionnelles en utilisant des jeux de données compris incorporés dans R.

```
data("airquality")#chargement du jeu de données airquality
head(airquality)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
## 6    28      NA 14.9   66     5   6
```

```
plot(Ozone~Wind, data = airquality)#affiche le graphique reliant les deux variables 'vent' et 'ozone'
```



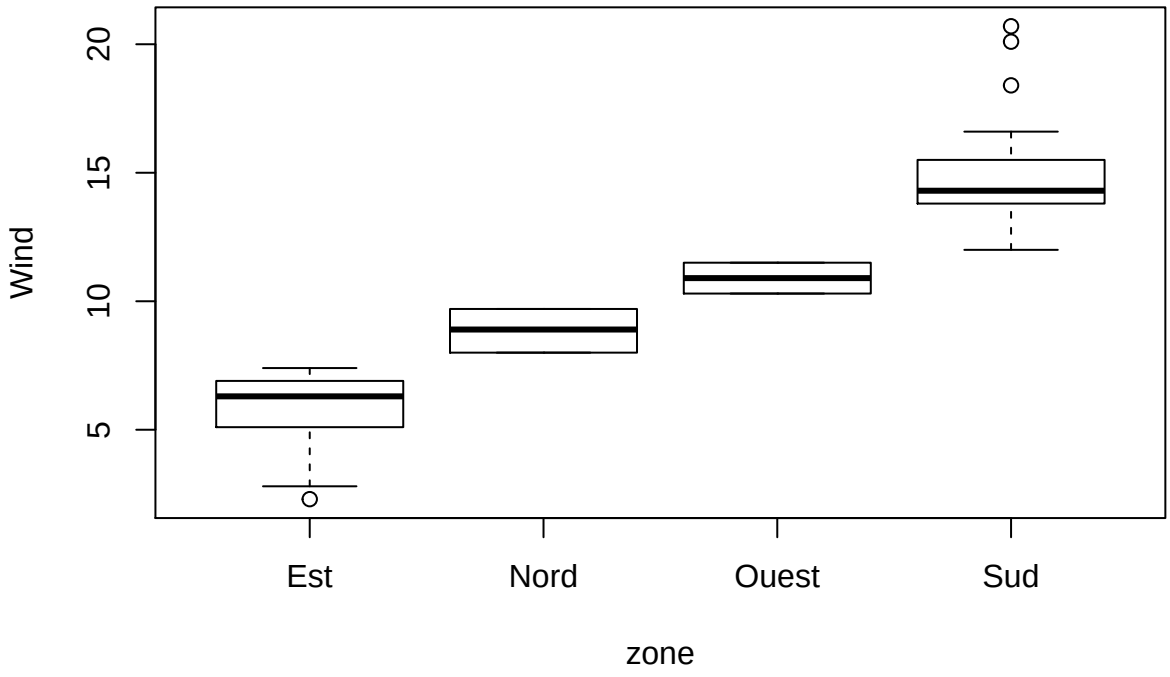
Il s'agit d'un graphique conventionnel utilisé pour relier deux variables afin de faire des analyses.

Nous allons ci-dessous creer un graphique permettant de mettre en relief la quantité de vent dans chaque zone créée (Nord, Sud, Est, Ouest.

```
#Creer une nouvelle variable dans un dataframe
airquality$Max <- NA
airquality$zone <- cut(airquality$Wind, breaks = quantile(airquality$Wind))
levels(airquality$zone) <- c("Est","Nord","Ouest","Sud")
table(airquality$zone, airquality$Wind) #Nous permet d'avoir une idée sur la quantité de vent dans chaq
```

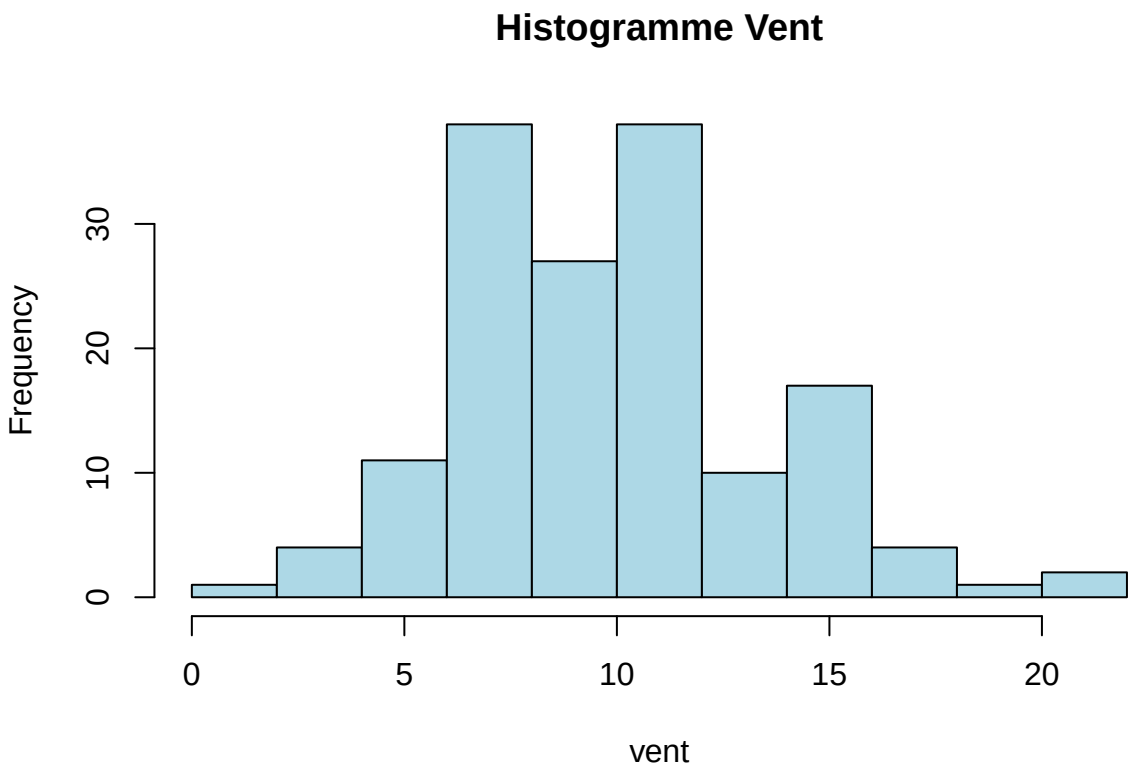
```
##
##         1.7 2.3 2.8 3.4  4 4.1 4.6 5.1 5.7 6.3 6.9 7.4  8 8.6 9.2 9.7 10.3 10.9
##   Est     0   1   1   1  1   1   4   3   3   8   9  10  0   0   0   0    0    0
##   Nord    0   0   0   0  0   0   0   0   0   0   0   0 11   8   8  11    0    0
##   Ouest   0   0   0   0  0   0   0   0   0   0   0   0  0   0   0   0   11    8
##   Sud     0   0   0   0  0   0   0   0   0   0   0   0  0   0   0   0    0    0
##
##         11.5 12 12.6 13.2 13.8 14.3 14.9 15.5 16.1 16.6 18.4 20.1 20.7
##   Est      0  0    0    0    0    0    0    0    0    0    0    0    0
##   Nord     0  0    0    0    0    0    0    0    0    0    0    0    0
##   Ouest   15  0    0    0    0    0    0    0    0    0    0    0    0
##   Sud      0  4    3    2    5    6    8    3    1    3    1    1    1
```

```
boxplot(Wind~zone, data = airquality)#Visualisation du graphique avec les valeurs aberantes
```

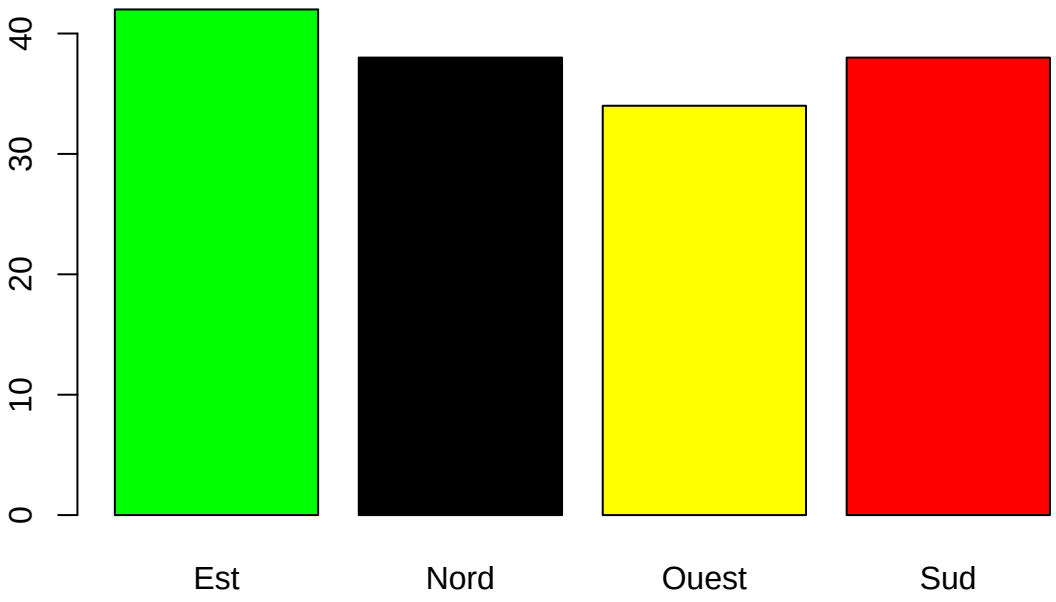


1. Histogramme conventionnel

```
#Histogramme
hist(airquality$Wind, main="Histogramme Vent", probability = FALSE, xlab = "vent", col = "lightblue")
```




```
#Graphique Barplot
plot(airquality$zone, col= c("green", "black","yellow","red")) #On peut preciser les couleurs de chaque
```



IV. TRACE PAR DEFAUT

Nous allons dans cette section, essayer de creer un graphique simple dit trace par defaut sous ggplot2, et pour cela nous allons utiliser le jeu de données iris inclut dans R.

"Les iris de Fisher" sont des données proposées en 1933 par le statisticien Ronald Aylmer Fisher comme données de référence pour l'analyse discriminante et la classification. Les données correspondent à 3 espèces de fleurs (Iris setosa, Iris virginica, Iris versicolor). Les variables mesurées sont la longueur et la largeur des sépales, la longueur et la largeur des pétales. Toutes ces variables sont exprimées en millimètres.

```
#chargement du jeu de données
data(iris)
head(iris)#affichage des 6 premieres lignes du jeu de données
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

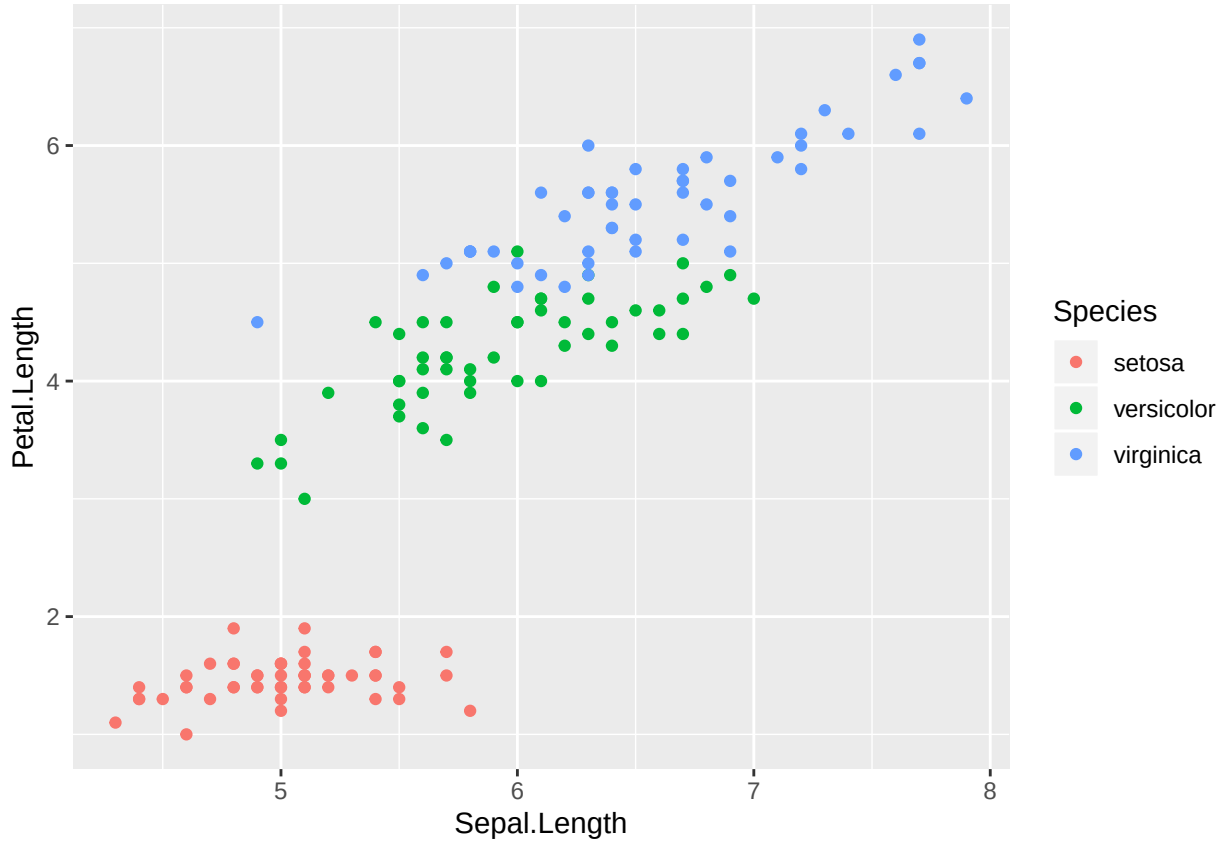
```
summary(iris)#resumé statistiques des données
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
```

```
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

Syntaxe de traçage simple du jeu de données iris :

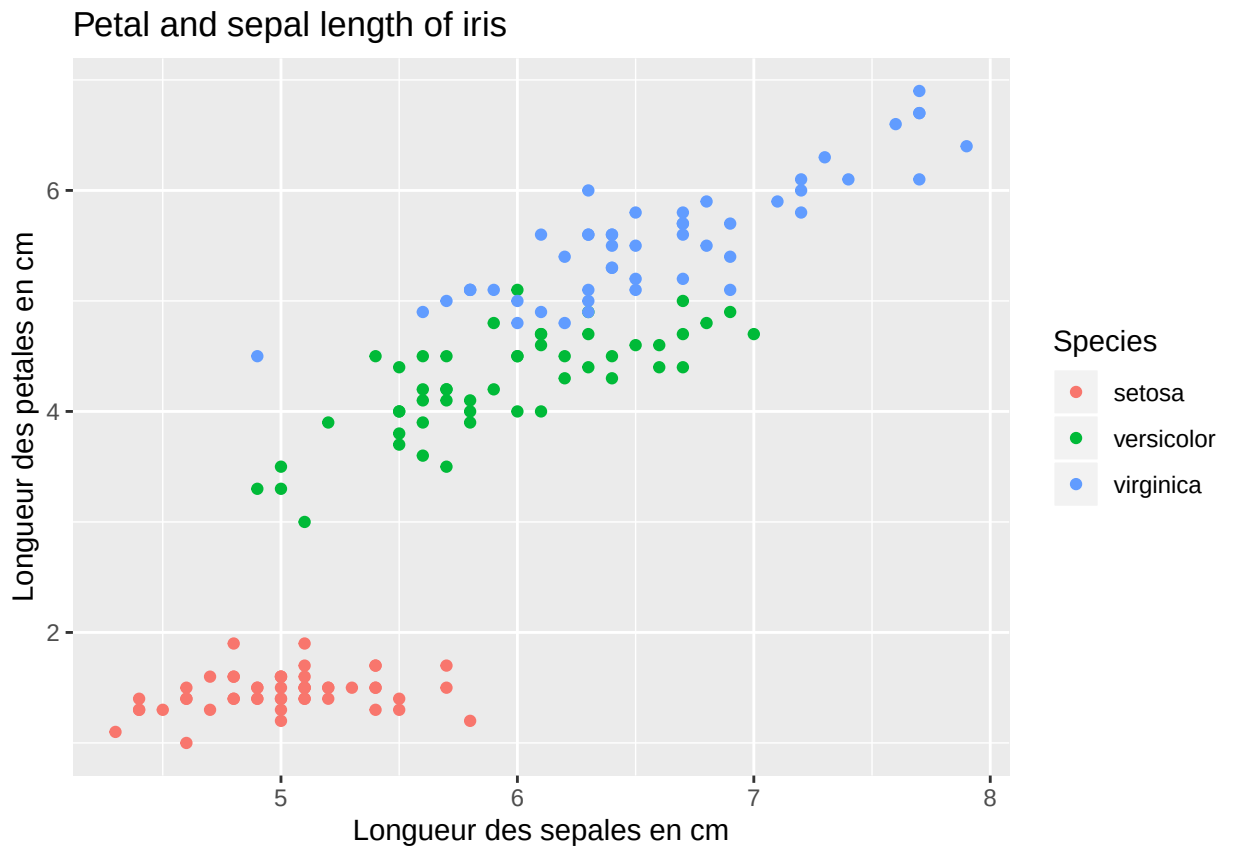
```
# Plot
graph_simple <- ggplot(iris, aes(Sepal.Length, Petal.Length, colour=Species))+geom_point()
print(graph_simple)
```



Le premier parametre defini prend en compte l'ensemble de données en entrée, le second parametre mentionne les attricbuts qui doivent etre tracés dans le graphique et geom_point implique un diagramme dispersé.

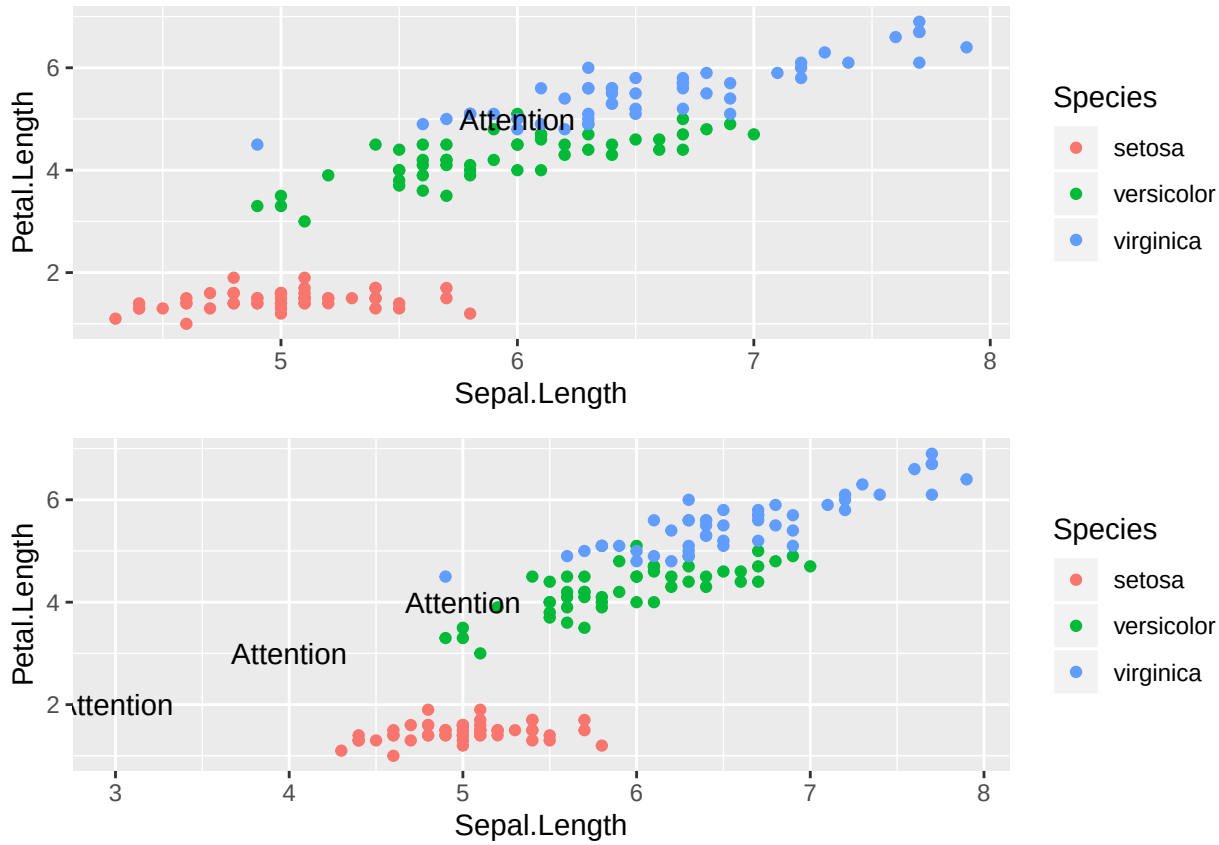
1. Modification des axes Nous pouvons apporter quelques petites modifications basique à notre graphique d'ou les axes X et Y avec la syntaxe suivante :

```
print(graph_simple + labs(y="Longueur des petales en cm", x = "Longueur des sepales en cm")
   + ggtitle("Petal and sepal length of iris"))
```



Il arrive souvent que lors du tracé d'un graphique, on a besoin d'insister sur un point precis du graphique en y ajoutant du texte pour indiquer l'importance du point par rapport à nos analyses,pour ce faire nous allons essayer de travailler sur le graphique en inserant un texte selon des coordonnées des axes. **2. Insertion d'un texte**

```
modif <- ggplot(iris,aes(Sepal.Length, Petal.Length, colour=Species))+geom_point()
X <- modif + annotate("text", x = 6, y = 5, label = "Attention")#Insertion de "Attention" avec pour pos
Y <- modif + annotate("text", x =3:5, y = 2:4, label = "Attention") #2eme graphique : Repetition des po
grid.arrange(X,Y)#Combinaison des deux premiers graphiques
```



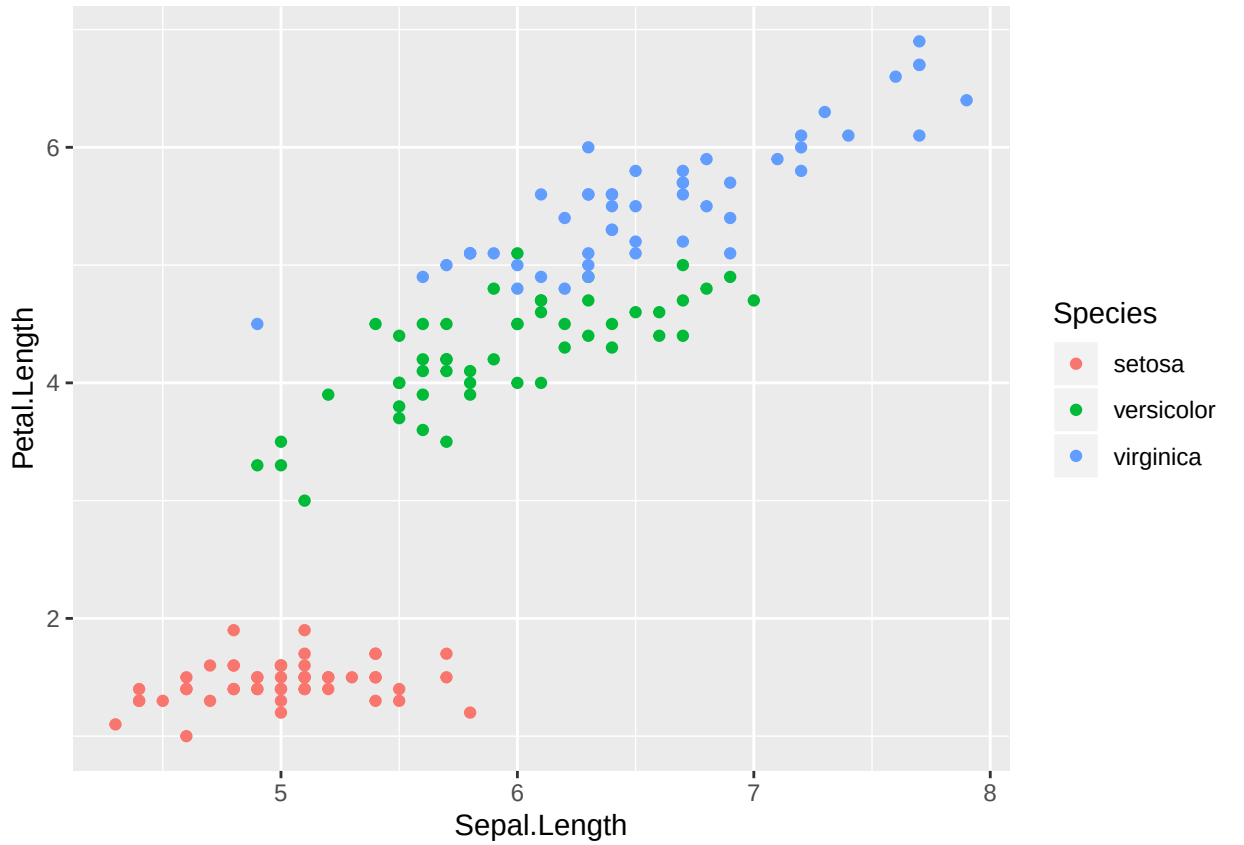

```
modif + annotate("rect", xmin = 5, xmax = 7, ymin = 4, ymax = 6, alpha = .5)#3eme graphique : On essaie
```

V. MODIFICATION DES LEGENDES

les légendes sont collectivement appelées comme guides. Ils nous permettent de lire les observations du graphique et de les cartographier par rapport aux valeurs d'origine. Les clés de légende et les étiquettes de graduation sont toutes deux déterminées par les sauts d'échelle. Les légendes et les axes sont produits automatiquement en fonction des échelles et géomètres respectifs nécessaires au tracé.

Les étapes suivantes seront mises en œuvre pour comprendre le fonctionnement des légendes dans ggplot2

```
modif
```



Les legendes sont créées a gauche : Species : setosa, versicolor, virginica Notre legende comprend une diversité d'especes de l'ensemble de notre jeu de données iris, nous allons donc proceder aux differentes modifications qu'on peut apporter avec les syntaxes suivantes :

```
"1er graph : on supprime ici la legende"
```

```
## [1] "1er graph : on supprime ici la legende"
```

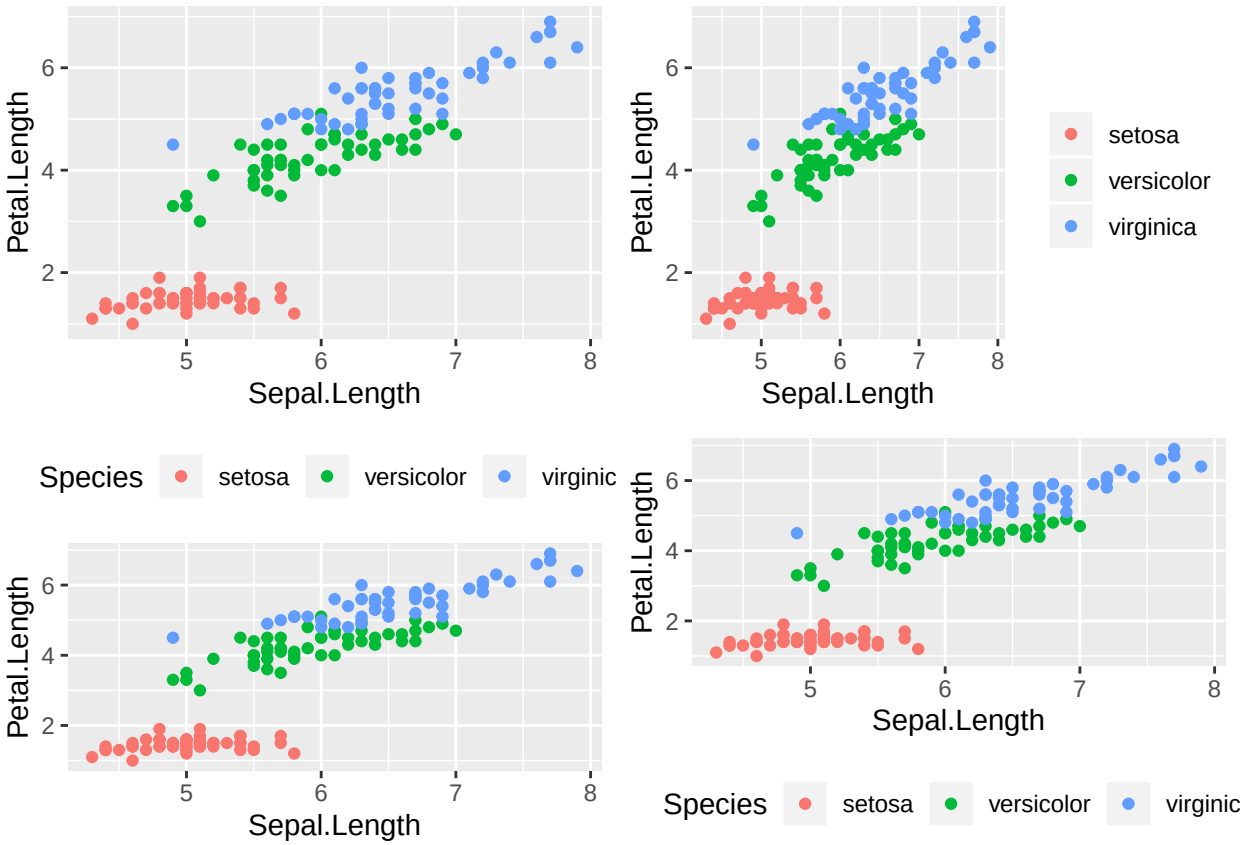
```
A <- modif + theme(legend.position="none")
"2eme graph : on masque le titre de la legende"
```

```
## [1] "2eme graph : on masque le titre de la legende"
```

```
B <- modif + theme(legend.title=element_blank())
"3eme graph : on deplace la position de la legende vers le haut et vers le bas avec la syntaxe suivante
```

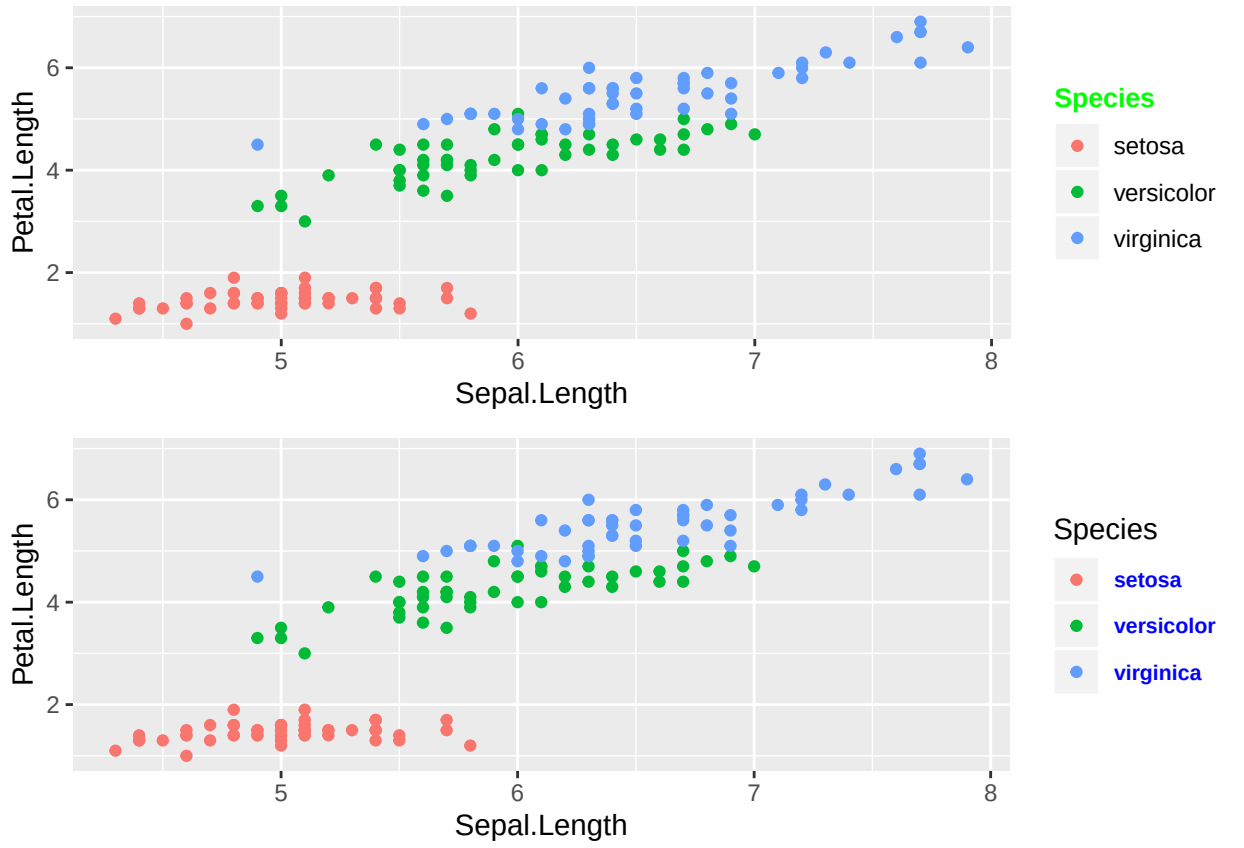
```
## [1] "3eme graph : on deplace la position de la legende vers le haut et vers le bas avec la syntaxe su
```

```
C <- modif + theme(legend.position="top")#vers le haut
D <- modif+ theme(legend.position="bottom")#vers le bas
grid.arrange(A,B,C,D, ncol=2, nrow = 2)
```



1. Modification du style de police de la legende

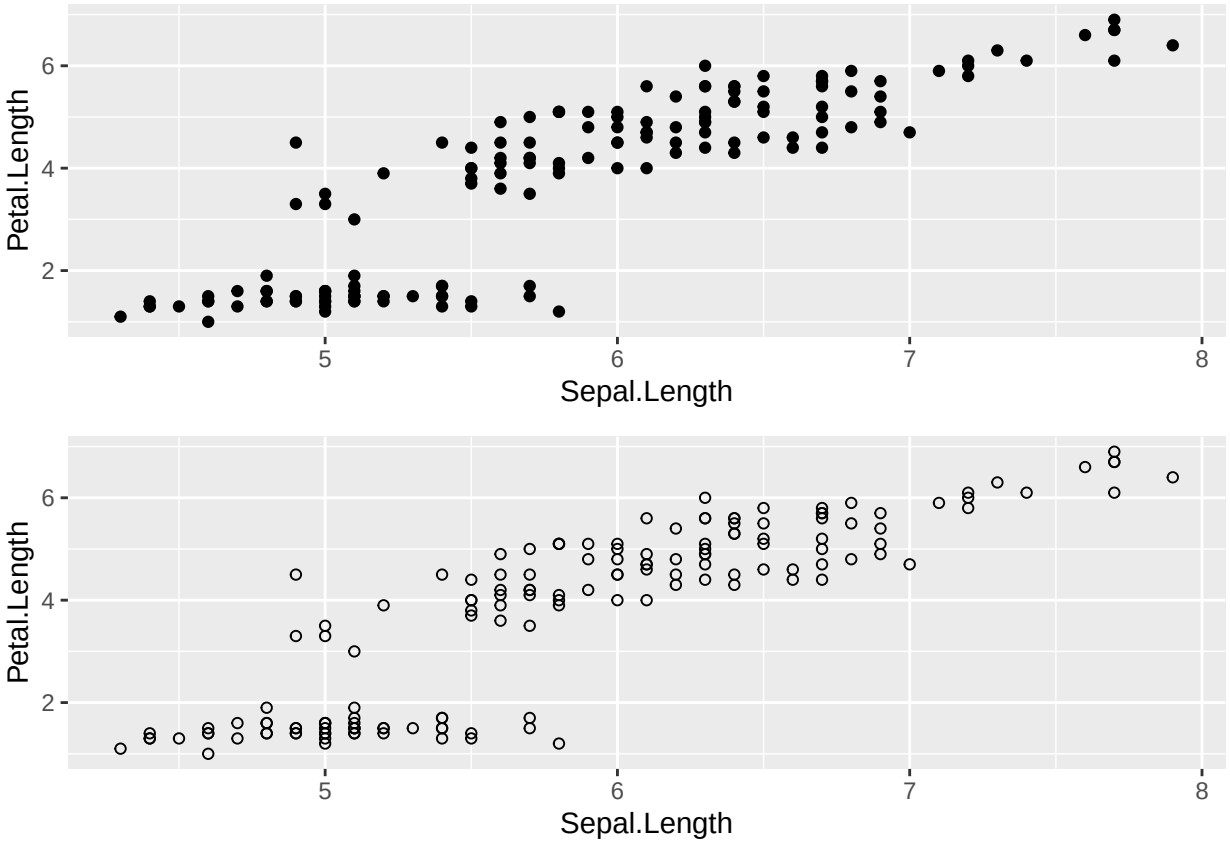
```
# Titre de la legende
E <- modif + theme(legend.title = element_text(colour = "green", size = 10, face = "bold"))
# Variables contenus dans la legende dits labels
G <- modif + theme(legend.text = element_text(colour = "blue", size = 8, face = "bold"))
grid.arrange(E,G)
```



VI. NUAGE DE POINTS ET TRACES DE GIGUE

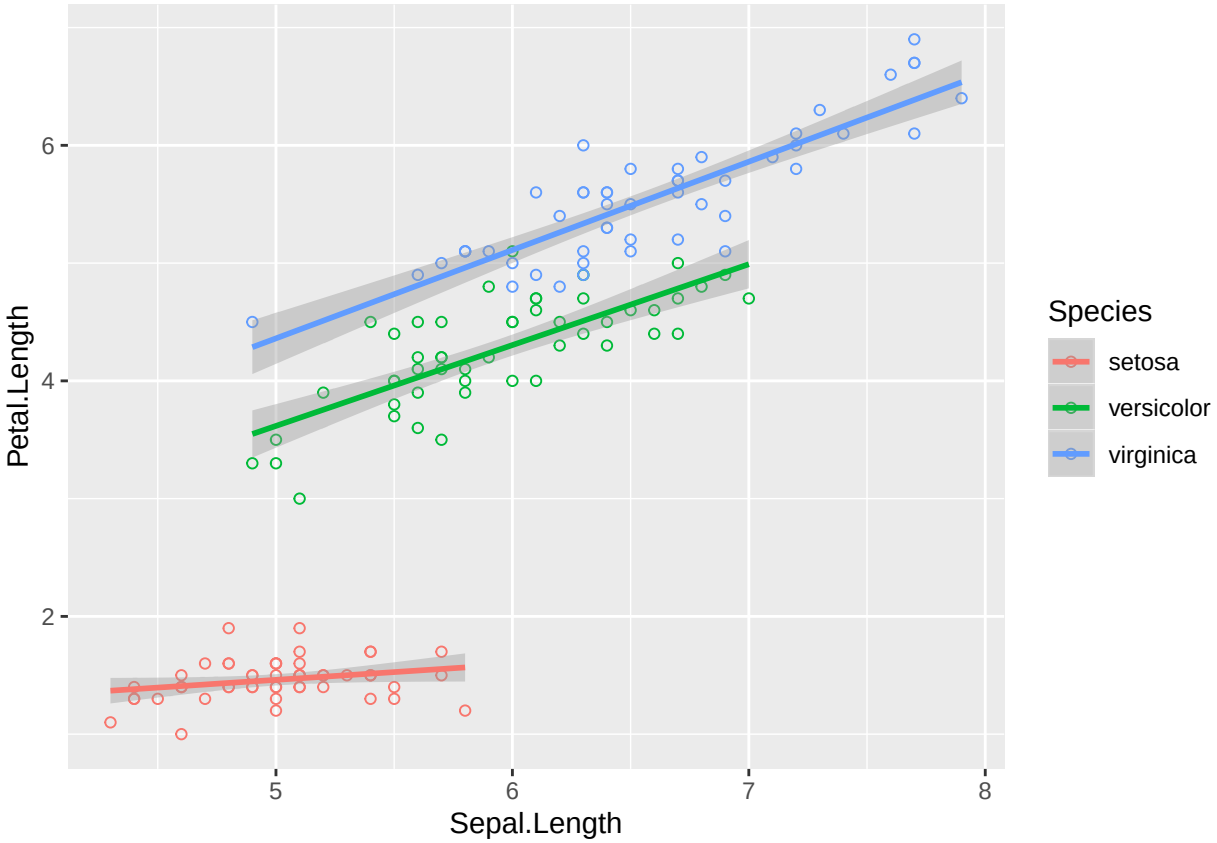
1. Nuage de points Les nuages de points sont similaires aux graphiques linéaires qui sont généralement utilisés pour le traçage. Les nuages de points montrent à quel point une variable est liée à une autre. La relation entre les variables est appelée corrélation qui est généralement utilisée dans les méthodes statistiques

```
#1er graph : Nuage de point
H <- ggplot(iris, aes(Sepal.Length, Petal.Length)) + geom_point()
#2eme graph: ajout des attributs sur les points tracés
J <- ggplot(iris, aes(Sepal.Length, Petal.Length)) + geom_point(shape=1)
#Pour rajouter les couleurs, on deinit l'argument "colour=Species" dans notre syntaxe aes.
grid.arrange(H,J)
```



2. Nuage de points avec droite de regression Nous allons passer aux choses plus interessantes, l'etablissement de relations entre les differentes variables contenus dans notre jeu de données.

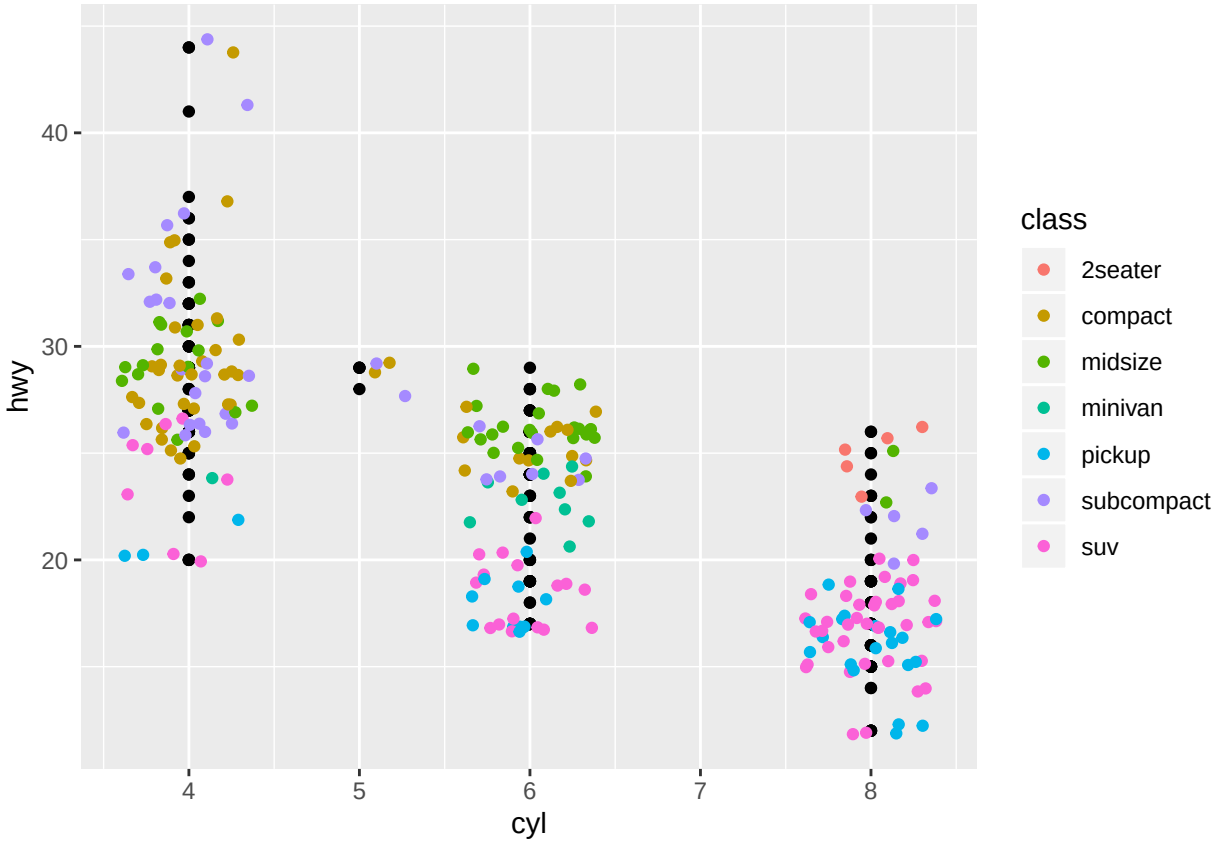
```
ggplot(iris, aes(Sepal.Length, Petal.Length, colour=Species))+ geom_point(shape=1)+ geom_smooth(method=l
```



La fonction geom_smooth permet de creer le motif des variables requises, l'attribut 'lm' permet de creer la droite de regression

3. Trace de Gigue Les graphiques de gigue incluent des effets spéciaux avec lesquels des graphiques dispersés peuvent être représentés. La gigue n'est rien d'autre qu'une valeur aléatoire attribuée aux points pour les séparer.

```
ggplot(mpg, aes(cyl, hwy)) + geom_point()+ geom_jitter(aes(colour = class))
```



VII. DIAGRAMME A BARRES ET HISTOGRAMMES Un diagramme à barres (ou en barres), également appelé diagramme à bâtons (ou en bâtons), est un graphique qui présente des variables catégorielles avec des barres rectangulaires avec des hauteurs ou des longueurs proportionnelles aux valeurs qu'elles représentent. Les barres peuvent être tracées verticalement ou horizontalement.

Un diagramme à barres montre des comparaisons entre des catégories discrètes. Un axe du diagramme montre les catégories spécifiques comparées et l'autre axe représente une valeur mesurée. Certains diagrammes à barres présentent des barres regroupées, indiquant les valeurs de plusieurs variables mesurées.

En statistique, un histogramme est une représentation graphique permettant de représenter la répartition d'une variable continue en la représentant avec des colonnes verticales. **1. Diagramme a barres**

On va utiliser le jeu de données mtcars inclut dans R qui presente des informations sur 32 voitures decrites sur 11 variables (cyl=cylindre, mpg=perforamance energetique en consommation, wt=poids, am=automatique ou manuelle etc...)

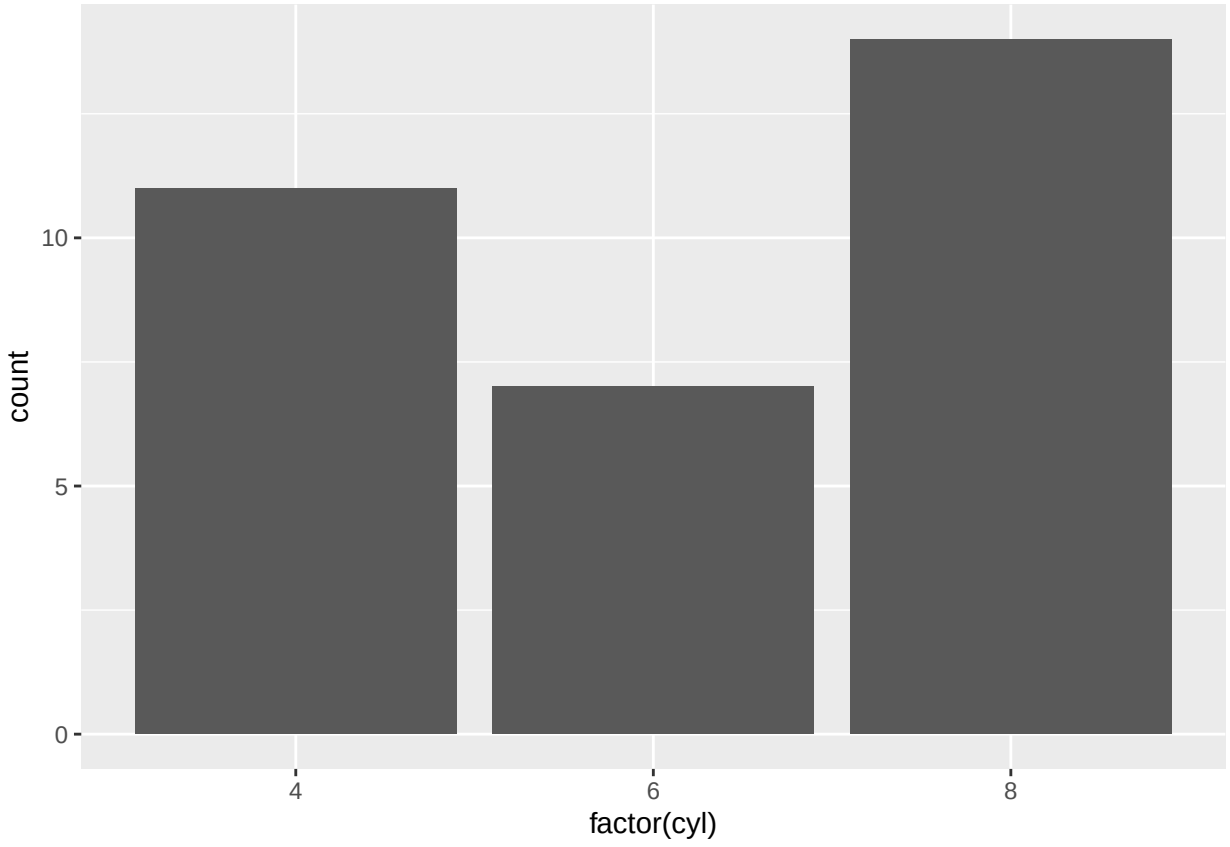
```
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

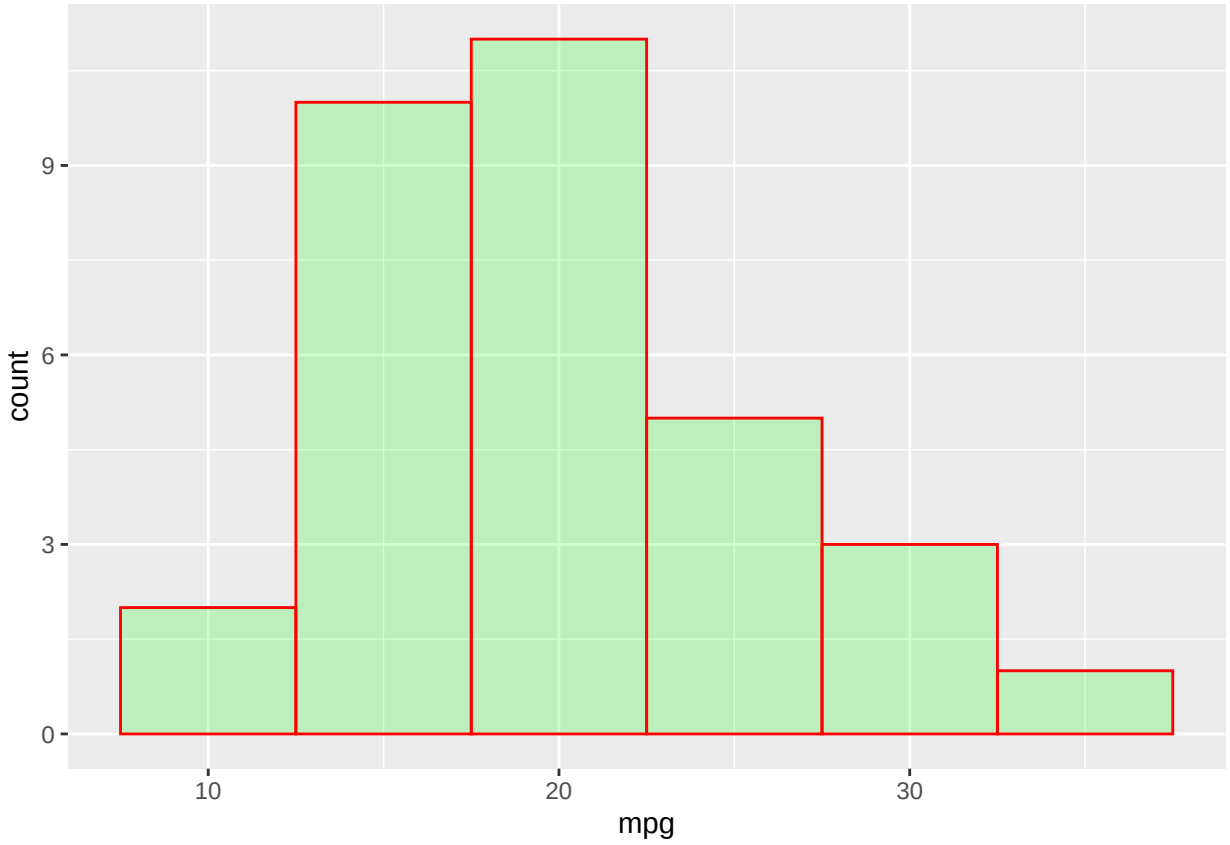
```r
summary(mtcars)
```

```
##       mpg             cyl             disp             hp
##  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0
##  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5
##  Median :19.20   Median :6.000   Median :196.3   Median :123.0
##  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7
##  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0
##  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0
##       drat             wt             qsec             vs
##  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000
##  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000
##  Median :3.695   Median :3.325   Median :17.71   Median :0.0000
##  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375
##  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000
##  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000
##        am              gear            carb
##  Min.   :0.0000   Min.   :3.000   Min.   :1.000
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000
##  Median :0.0000   Median :4.000   Median :2.000
##  Mean   :0.4062   Mean   :3.688   Mean   :2.812
##  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000
##  Max.   :1.0000   Max.   :5.000   Max.   :8.000
```

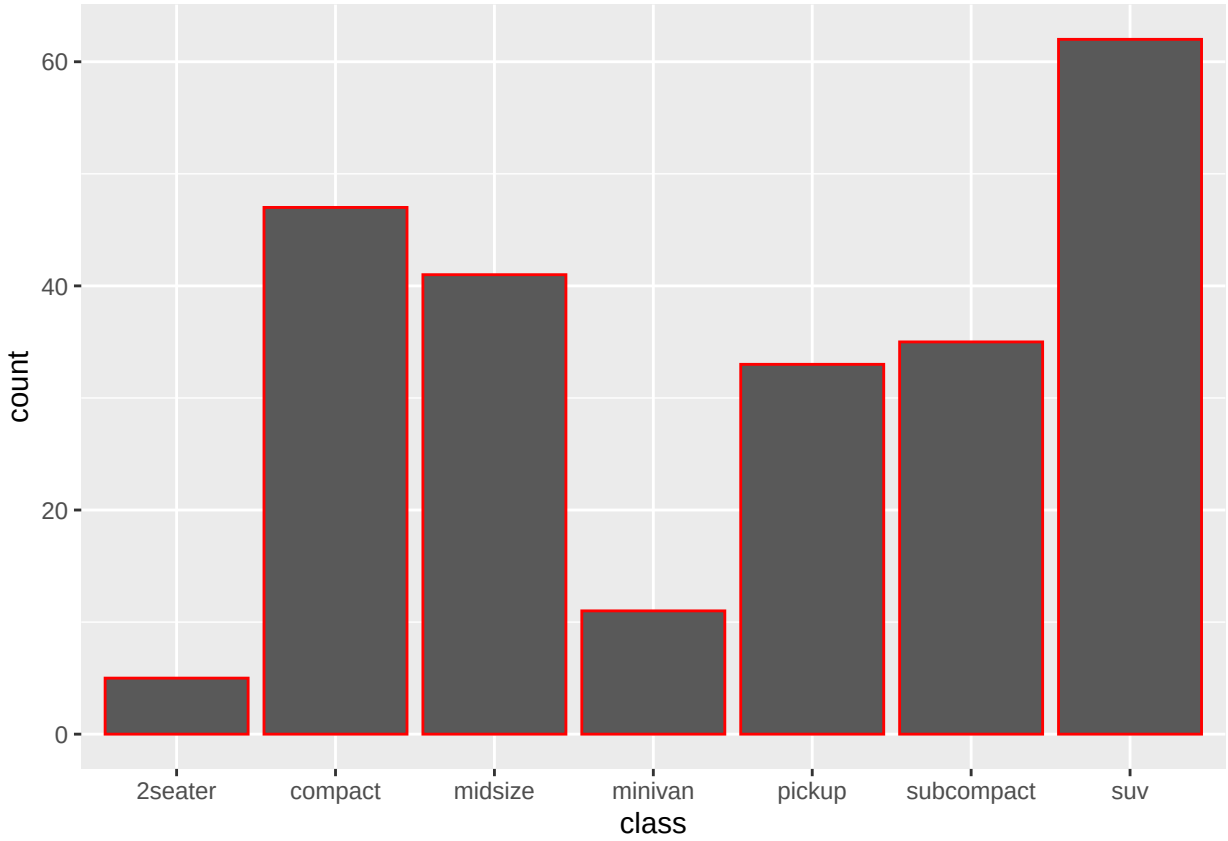
```r
ggplot(mtcars, aes(x=factor(cyl)))+ geom_bar(stat="count")#geom_bar est la fonction qui permet de trace
```



```r
ggplot(data=mtcars, aes(x=mpg)) + geom_histogram(col="red",fill="green",alpha = .2,binwidth = 5)
```




```r
p <- ggplot(mpg, aes(x=factor(cyl)))+geom_bar(stat="count") #Creation de la fonction tracé de barre
p <- ggplot(mpg, aes(class))
p + geom_bar(colour="red")
```



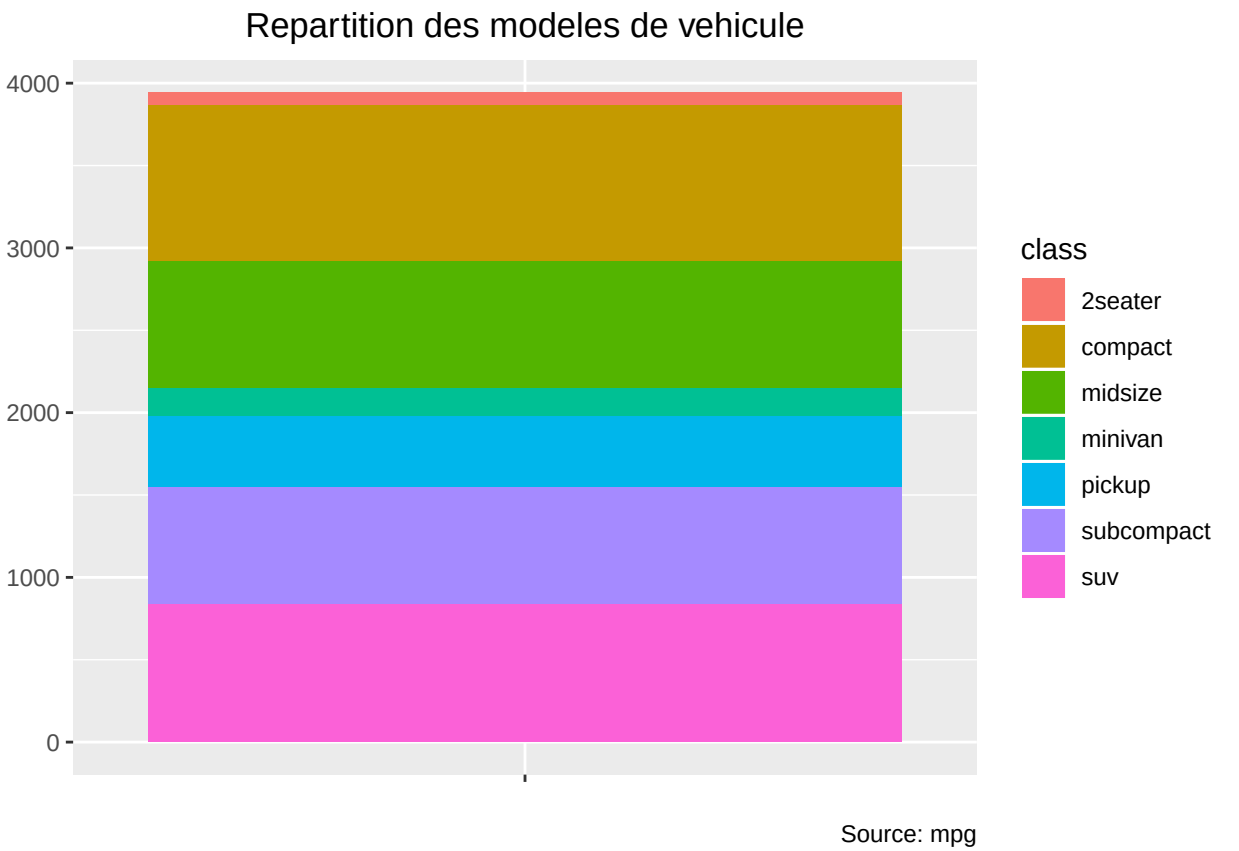
VIII. GRAPHIQUE A SECTEURS

On va essayer de tracer ci-dessous la classe et la frequence de notre variable mpg de notre jeu de données mpcars. **1. Diagramme circulaire ou de camembert**

```
str(mpg)
```

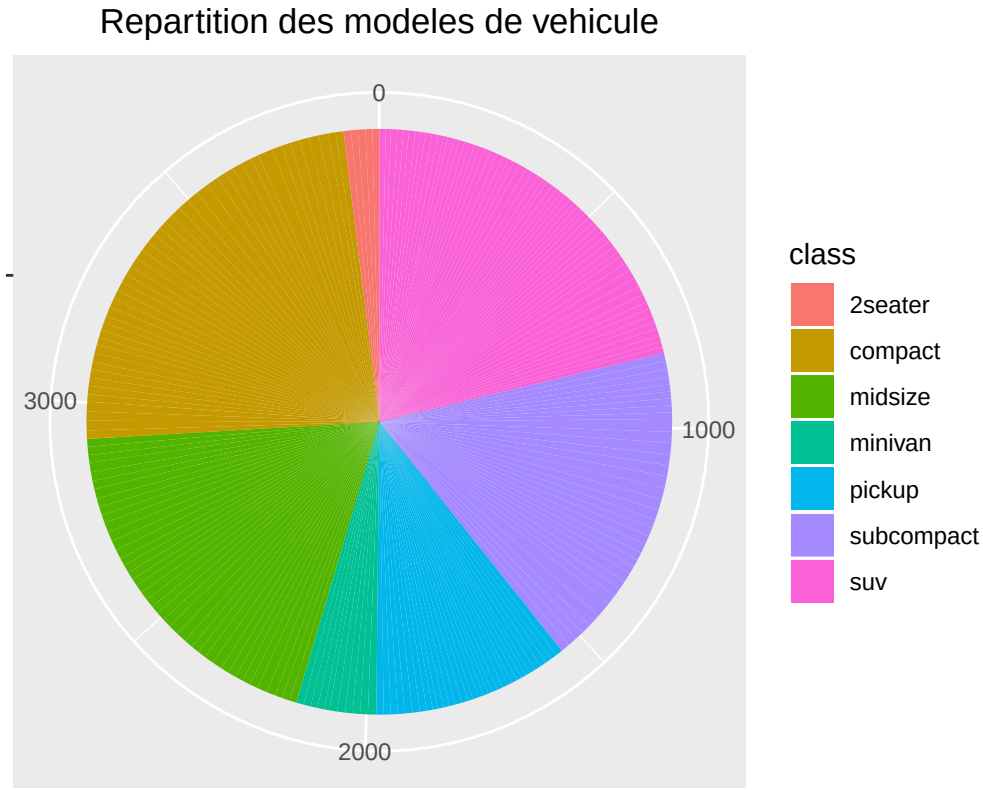
```
## Classes 'tbl_df', 'tbl' and 'data.frame':    234 obs. of  11 variables:
##  $ manufacturer: chr  "audi" "audi" "audi" "audi" ...
##  $ model       : chr  "a4" "a4" "a4" "a4" ...
##  $ displ       : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int  1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
##  $ cyl         : int  4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr  "auto(15)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr  "f" "f" "f" "f" ...
##  $ cty         : int  18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int  29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : chr  "p" "p" "p" "p" ...
##  $ class       : chr  "compact" "compact" "compact" "compact" ...
```

```
diag_cir <- ggplot(mpg, aes(x = "", y=cty, fill =factor(class)))+ geom_bar(width = 1, stat = "identity")
print(diag_cir)
```



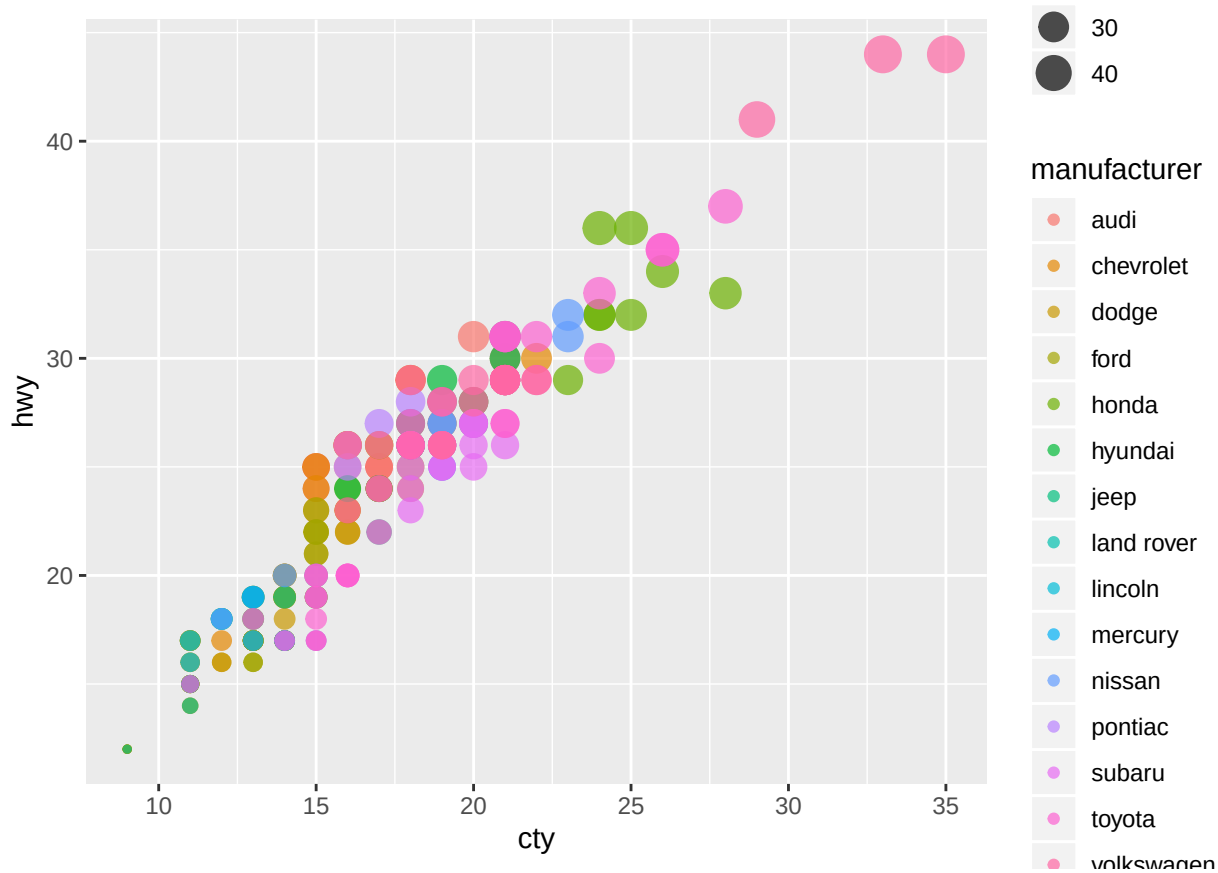
On remarque que le resultat obtenu est totalement different du resultat attendu vu que le diagramme n'est pas circulaire, et pour obtenir un diagramme circulaire, on rajoute la syntaxe suivante :

```
diag_cir + coord_polar(theta = "y", start=0)
```



IX. GRAPHIQUE A BULLES Les graphiques à bulles ne sont rien d'autre que des graphiques à bulles qui sont essentiellement un nuage de points avec une troisième variable numérique utilisée pour la taille du cercle. Créons maintenant le diagramme à bulles le plus élémentaire avec les attributs requis pour augmenter la dimension des points mentionnés dans le diagramme dispersé.

```
ggplot(mpg, aes(x=cty, y=hwy, col=manufacturer, size=hwy)) +geom_point(alpha=0.7)
```



Le graphique décrit la nature des fabricants qui est inclus dans le format de légende. Les valeurs représentées incluent diverses dimensions de l'attribut «hwy».

X. GRAPHIQUE DIVERGENTS Nous allons maintenant nous concentrer sur la variation des mêmes graphiques à barres divergents, des graphiques à sucettes et bien d'autres. Pour commencer, nous allons commencer par créer des histogrammes divergents et les étapes à suivre sont les suivantes :

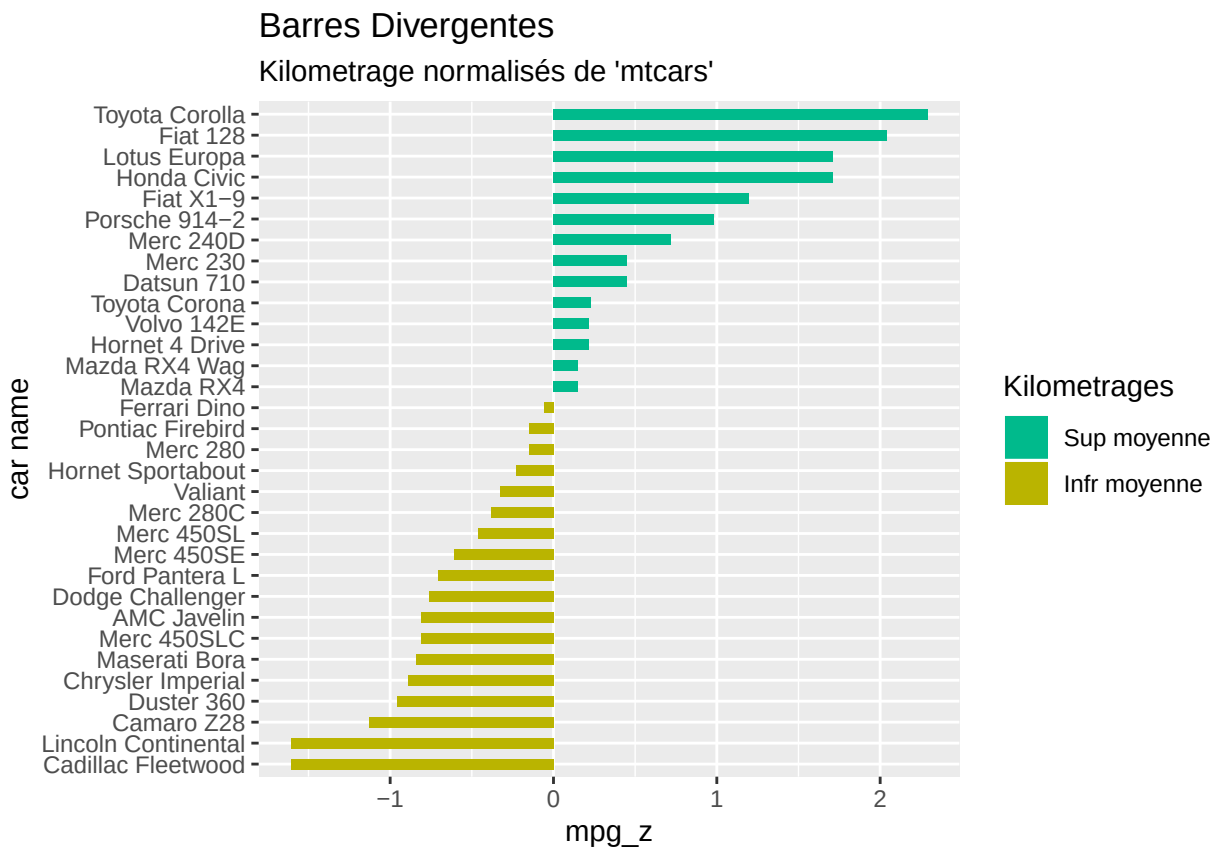
```
#On crée une nouvelle colonne denommé car name
mtcars$`car name` <- rownames(mtcars)
# calcul des mpg normalisés (km)
mtcars$mpg_z <- round((mtcars$mpg - mean(mtcars$mpg))/sd(mtcars$mpg), 2)
# au dessus/ au dessous de zero
mtcars$mpg_type <- ifelse(mtcars$mpg_z < 0, "below", "above")
#Tri
mtcars <- mtcars[order(mtcars$mpg_z), ]
```

Le calcul ci-dessus implique la création d'une nouvelle colonne pour les noms de voitures, le calcul de l'ensemble de données normalisé à l'aide de la fonction round. Nous pouvons également utiliser l'indicateur au-dessus et au-dessous de avg pour obtenir les valeurs de la fonctionnalité «type». Plus tard, nous trions les valeurs pour créer l'ensemble de données requis.

```
#Conertissons les valeurs de la nouvelle colonne en facteur poir le traitement et le tracé
mtcars$`car name` <- factor(mtcars$`car name`, levels = mtcars$`car name`)
```

1. Graphique a barres divergentes

```
ggplot(mtcars, aes(x=‘car name‘, y=mpg_z, label=mpg_z))+geom_bar(stat=’identity’,aes(fill=mpg_type),wid
```



2. Diagrame de Sucette divergentes On crée un diagramme de sucettes divergentes avec les mêmes attributs et coordonnées avec seulement le changement de fonction à utiliser, c'est-à-dire geom_segment () qui aide à créer les graphiques de sucettes.

```
ggplot(mtcars, aes(x=‘car name‘, y=mpg_z, label=mpg_z)) +geom_point(stat=’identity’, fill="black", size=
```

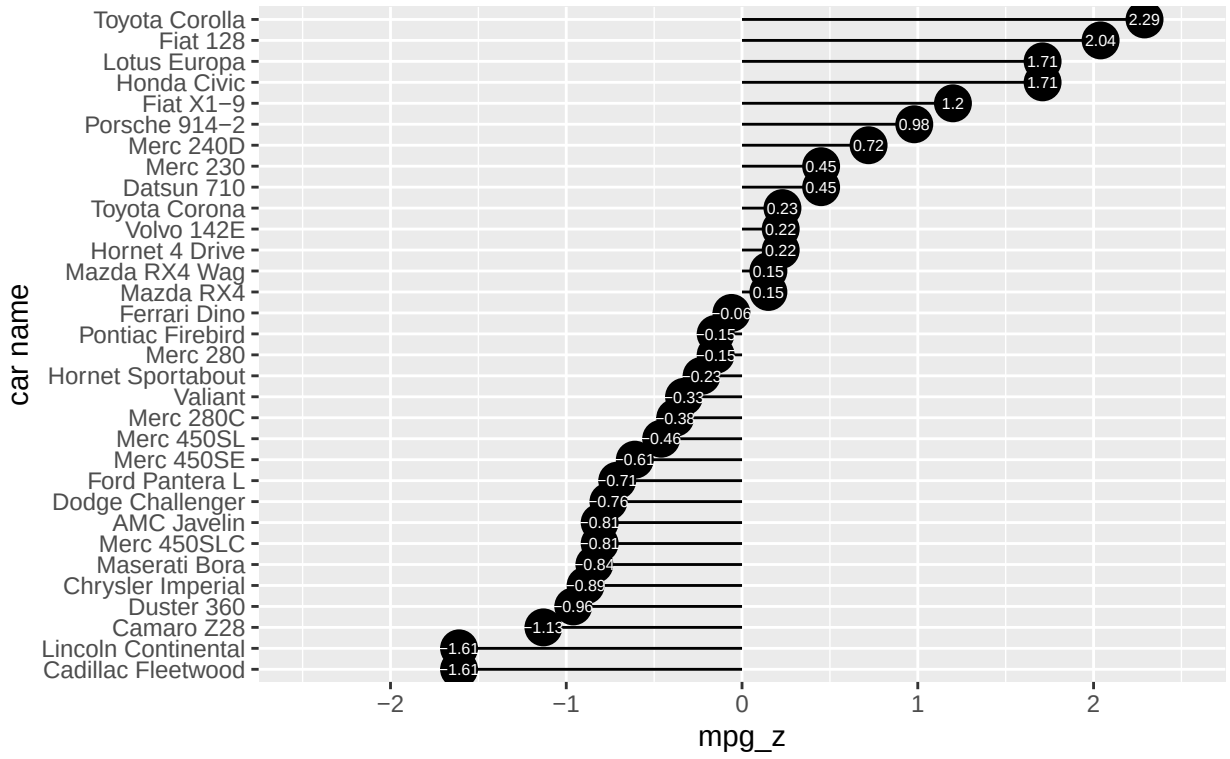


Diagramme de sucettes divergentes
Kilometrages normalisés de 'mtcars': Sucette
car name
Toyota Corolla
Fiat 128
Lotus Europa
Honda Civic
Fiat X1−9
Porsche 914−2
Merc 240D
Merc 230
Datsun 710
Toyota Corona
Volvo 142E
Hornet 4 Drive
Mazda RX4 Wag
Mazda RX4
Ferrari Dino
Pontiac Firebird
Merc 280
Hornet Sportabout
Valiant
Merc 280C
Merc 450SL
Merc 450SE
Ford Pantera L
Dodge Challenger
AMC Javelin
Merc 450SLC
Maserati Bora
Chrysler Imperial
Duster 360
Camaro Z28
Lincoln Continental
Cadillac Fleetwood
2.29
2.04
1.71
1.71
1.2
0.98
0.72
0.45
0.45
0.23
0.22
0.22
0.15
0.15
0.06
−0.15
−0.15
−0.23
−0.33
−0.38
−0.46
−0.61
−0.71
−0.76
−0.81
−0.81
−0.84
−0.89
−0.96
−1.13
−1.61
−1.61
−2
−1
0
1
2
mpg_z