Analyse avec RStudio

Henri Makika

April 28, 2019

Listes et Tableaux de données

```
L1 = list(1:6, "abc")
print(L1)
```

```
## [[1]]
## [1] 1 2 3 4 5 6
##
## [[2]]
## [1] "abc"
```

```
length(L1)
```

```
## [1] 2
```

La longueur d'une liste correspond aux nombres d'éléments qu'elle contient et s'obtient avec length.

Comme les vecteurs, une liste peut être nommées et les noms des éléments d'une liste accessibles avec names :

```
L2 <- list(minuscules = letters, majuscules = LETTERS, mois = month.name)
print(L2)
```

```
## $minuscules
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q"
## [18] "r" "s" "t" "u" "v" "w" "x" "y" "z"
##
## $majuscules
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q"
## [18] "R" "S" "T" "U" "V" "W" "X" "Y" "Z"
##
## $mois
##  [1] "January"   "February"  "March"     "April"     "May"
##  [6] "June"      "July"      "August"    "September" "October"
## [11] "November"  "December"
```

```
L = list(L1, L2)
print(L)
```

```
## [[1]]
## [[1]][[1]]
## [1] 1 2 3 4 5 6
##
```

```
## [[1]][[2]]
## [1] "abc"
##
##
## [[2]]
## [[2]]$minuscules
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q"
## [18] "r" "s" "t" "u" "v" "w" "x" "y" "z"
##
## [[2]]$majuscules
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q"
## [18] "R" "S" "T" "U" "V" "W" "X" "Y" "Z"
##
## [[2]]$mois
##  [1] "January"   "February"  "March"     "April"     "May"
##  [6] "June"      "July"      "August"    "September" "October"
## [11] "November"  "December"
```

Cela est plus lisible si l'on fait appel à la fonction str qui permet de visualiser la structure d'un objet.

```
str(L)
```

```
## List of 2
##  $ :List of 2
##   ..$ : int [1:6] 1 2 3 4 5 6
##   ..$ : chr "abc"
##  $ :List of 3
##   ..$ minuscules: chr [1:26] "a" "b" "c" "d" ...
##   ..$ majuscules: chr [1:26] "A" "B" "C" "D" ...
##   ..$ mois      : chr [1:12] "January" "February" "March" "April" ...
```

Tableaux de données

Dans R, les tableaux de données sont tout simplement des listes avec quelques propriétés spéficiques : • les tableaux de données ne peuvent contenir que des vecteurs ; • tous les vecteurs d'un tableau de données ont la même longueur ; • tous les éléments d'un tableau de données sont nommés et ont chacun un nom unique.

On peut créer un tableau de données avec la fonction data.frame :

```
df <- data.frame(sexe = c("fe", "fe", "ho", "ho"), age = c(52, 31, 29, 35),
                 blond = c(FALSE, TRUE, TRUE, FALSE))
print(df)
```

```
##   sexe age blond
## 1   fe  52 FALSE
## 2   fe  31  TRUE
## 3   ho  29  TRUE
## 4   ho  35 FALSE
```

```
str(df)
```

```
## 'data.frame':    4 obs. of  3 variables:
##  $ sexe : Factor w/ 2 levels "fe","ho": 1 1 2 2
##  $ age  : num  52 31 29 35
##  $ blond: logi  FALSE TRUE TRUE FALSE
```

De plus, tout comme les colonnes ont un nom, il est aussi possible de nommer les lignes avec row.names :

```
row.names(df) <- c("Henriette", "Bertine", "Jonas", "Benedict")
print(df)
```

```
##           sexe age blond
## Henriette   fe  52 FALSE
## Bertine     fe  31  TRUE
## Jonas       ho  29  TRUE
## Benedict    ho  35 FALSE
```

Afficher les données

```
mtcars
```

```
##                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4          21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag      21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710         22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive     21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant            18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360         14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D          24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230           22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280           19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C          17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE         16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL         17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC        15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4  8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial  14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128           32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic        30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla     33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona      21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger   15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin        15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28         13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird   19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9          27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2      26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa       30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L     15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino       19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
```

```
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
d <- mtcars
View(d)
```

Les fonctions head et tail , qui marchent également sur les vecteurs, permettent d'afficher seulement les premières (respectivement les dernières) lignes d'un tableau de données :

```
head(d)
```

```
##                    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
```

```
tail(d)
```

```
##                 mpg cyl  disp  hp drat    wt qsec vs am gear carb
## Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.7  0  1    5    2
## Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.9  1  1    5    2
## Ford Pantera L 15.8   8 351.0 264 4.22 3.170 14.5  0  1    5    4
## Ferrari Dino   19.7   6 145.0 175 3.62 2.770 15.5  0  1    5    6
## Maserati Bora  15.0   8 301.0 335 3.54 3.570 14.6  0  1    5    8
## Volvo 142E     21.4   4 121.0 109 4.11 2.780 18.6  1  1    4    2
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
glimpse(d)
```

```
## Observations: 32
## Variables: 11
## $ mpg  <dbl> 21.0, 21.0, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19....
## $ cyl  <dbl> 6, 6, 4, 6, 8, 6, 8, 4, 4, 6, 6, 8, 8, 8, 8, 8, 8, 4, 4, ...
## $ disp <dbl> 160.0, 160.0, 108.0, 258.0, 360.0, 225.0, 360.0, 146.7, 1...
## $ hp   <dbl> 110, 110, 93, 110, 175, 105, 245, 62, 95, 123, 123, 180, ...
```

```
## $ drat <dbl> 3.90, 3.90, 3.85, 3.08, 3.15, 2.76, 3.21, 3.69, 3.92, 3.9...
## $ wt   <dbl> 2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570, 3.190, 3...
## $ qsec <dbl> 16.46, 17.02, 18.61, 19.44, 17.02, 20.22, 15.84, 20.00, 2...
## $ vs   <dbl> 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, ...
## $ am   <dbl> 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, ...
## $ gear <dbl> 4, 4, 4, 3, 3, 3, 3, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 4, 4, ...
## $ carb <dbl> 4, 4, 1, 1, 2, 1, 4, 2, 2, 4, 4, 3, 3, 3, 4, 4, 4, 1, 2, ...
```

La méthode summary qui fonctionne sur tout type d'objet permet d'avoir quelques statistiques de base sur les différentes variables de notre tableau, les statistiques affichées dépendant du type de variable.

```
summary(d)
```

```
##       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
```

On peut également appliquer summary à une variable particulière.

```
summary(d$mpg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   10.40   15.43   19.20   20.09   22.80   33.90
```

Convertir un data.frame en data.table Il suffit d'avoir recours à la fonction as.data.table

```
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
iris2 <- as.data.table(iris)
class(iris2)
```

```
## [1] "data.table" "data.frame"
```

```
class(iris)
```

```
## [1] "data.frame"
```

```
library(dtplyr)
iris_dt <- tbl_dt(iris)
class(iris_dt)
```

```
## [1] "tbl_dt"     "tbl"        "data.table" "data.frame"
```

La syntaxe des crochets change radicalement avec data.table. Elle est de la forme objet[i, j, by] (dans sa forme la plus simple, pour une présentation exhaustive, voir le fichier d'aide de data.table-package).

```
iris2[Sepal.Length < 5]
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
##  1:          4.9         3.0          1.4         0.2     setosa
##  2:          4.7         3.2          1.3         0.2     setosa
##  3:          4.6         3.1          1.5         0.2     setosa
##  4:          4.6         3.4          1.4         0.3     setosa
##  5:          4.4         2.9          1.4         0.2     setosa
##  6:          4.9         3.1          1.5         0.1     setosa
##  7:          4.8         3.4          1.6         0.2     setosa
##  8:          4.8         3.0          1.4         0.1     setosa
##  9:          4.3         3.0          1.1         0.1     setosa
## 10:          4.6         3.6          1.0         0.2     setosa
## 11:          4.8         3.4          1.9         0.2     setosa
## 12:          4.7         3.2          1.6         0.2     setosa
## 13:          4.8         3.1          1.6         0.2     setosa
## 14:          4.9         3.1          1.5         0.2     setosa
## 15:          4.9         3.6          1.4         0.1     setosa
## 16:          4.4         3.0          1.3         0.2     setosa
## 17:          4.5         2.3          1.3         0.3     setosa
## 18:          4.4         3.2          1.3         0.2     setosa
## 19:          4.8         3.0          1.4         0.3     setosa
## 20:          4.6         3.2          1.4         0.2     setosa
## 21:          4.9         2.4          3.3         1.0 versicolor
## 22:          4.9         2.5          4.5         1.7  virginica
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
```

Pour sélectionner une variable, il suffit d'indiquer son nom dans la seconde partie, à savoir j . Noter la virgule qui permets d'indiquer que c'est une condition sur j et non sur i .

```
iris2[, Sepal.Length]
```

```
##   [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4
##  [18] 5.1 5.7 5.1 5.4 5.1 4.6 5.1 4.8 5.0 5.0 5.2 5.2 4.7 4.8 5.4 5.2 5.5
##  [35] 4.9 5.0 5.5 4.9 4.4 5.1 5.0 4.5 4.4 5.0 5.1 4.8 5.1 4.6 5.3 5.0 7.0
##  [52] 6.4 6.9 5.5 6.5 5.7 6.3 4.9 6.6 5.2 5.0 5.9 6.0 6.1 5.6 6.7 5.6 5.8
##  [69] 6.2 5.6 5.9 6.1 6.3 6.1 6.4 6.6 6.8 6.7 6.0 5.7 5.5 5.5 5.8 6.0 5.4
##  [86] 6.0 6.7 6.3 5.6 5.5 5.5 6.1 5.8 5.0 5.6 5.7 5.7 6.2 5.1 5.7 6.3 5.8
## [103] 7.1 6.3 6.5 7.6 4.9 7.3 6.7 7.2 6.5 6.4 6.8 5.7 5.8 6.4 6.5 7.7 7.7
## [120] 6.0 6.9 5.6 7.7 6.3 6.7 7.2 6.2 6.1 6.4 7.2 7.4 7.9 6.4 6.3 6.1 7.7
## [137] 6.3 6.4 6.0 6.9 6.7 6.9 5.8 6.8 6.7 6.7 6.3 6.5 6.2 5.9
```

Pour sélectionner plusieurs variables, on fournira une liste définie avec list (et non un vecteur défini avec c).

```
iris2[, list(Sepal.Length, Sepal.Width)]
```

```
##      Sepal.Length Sepal.Width
##   1:          5.1         3.5
##   2:          4.9         3.0
##   3:          4.7         3.2
##   4:          4.6         3.1
##   5:          5.0         3.6
##  ---
## 146:          6.7         3.0
## 147:          6.3         2.5
## 148:          6.5         3.0
## 149:          6.2         3.4
## 150:          5.9         3.0
```

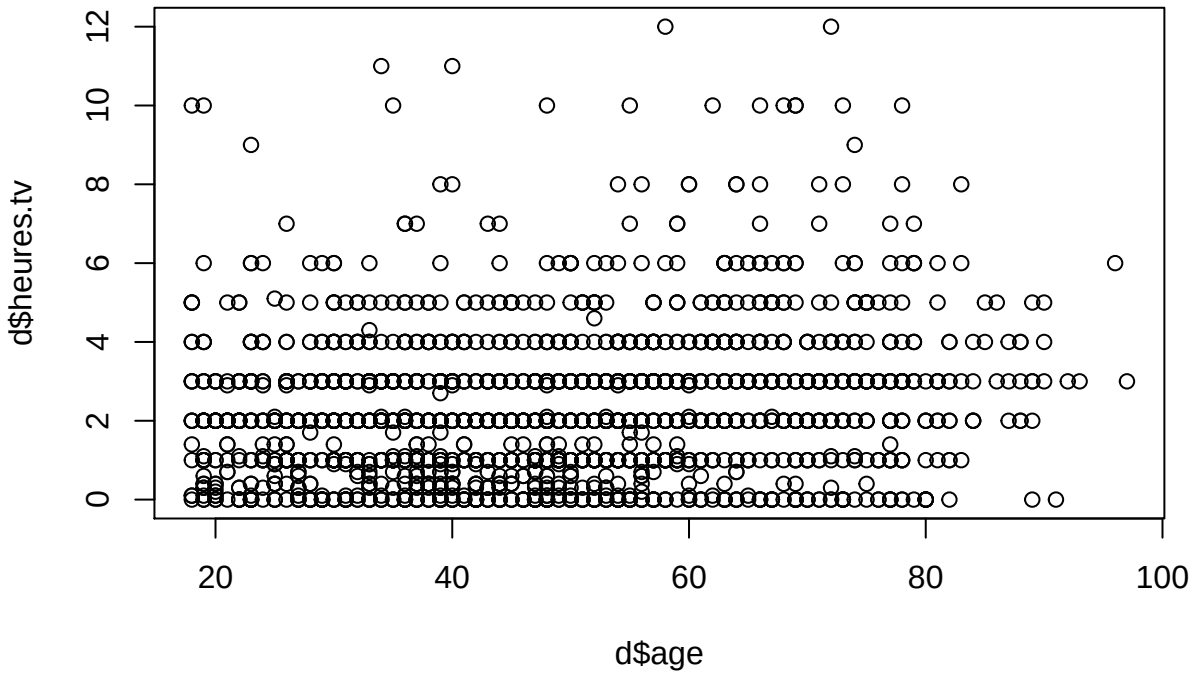
Statistique bivariée

```
library(questionr)
data(hdv2003)
d <- hdv2003
data(rp99)
```

Deux variables quantitatives

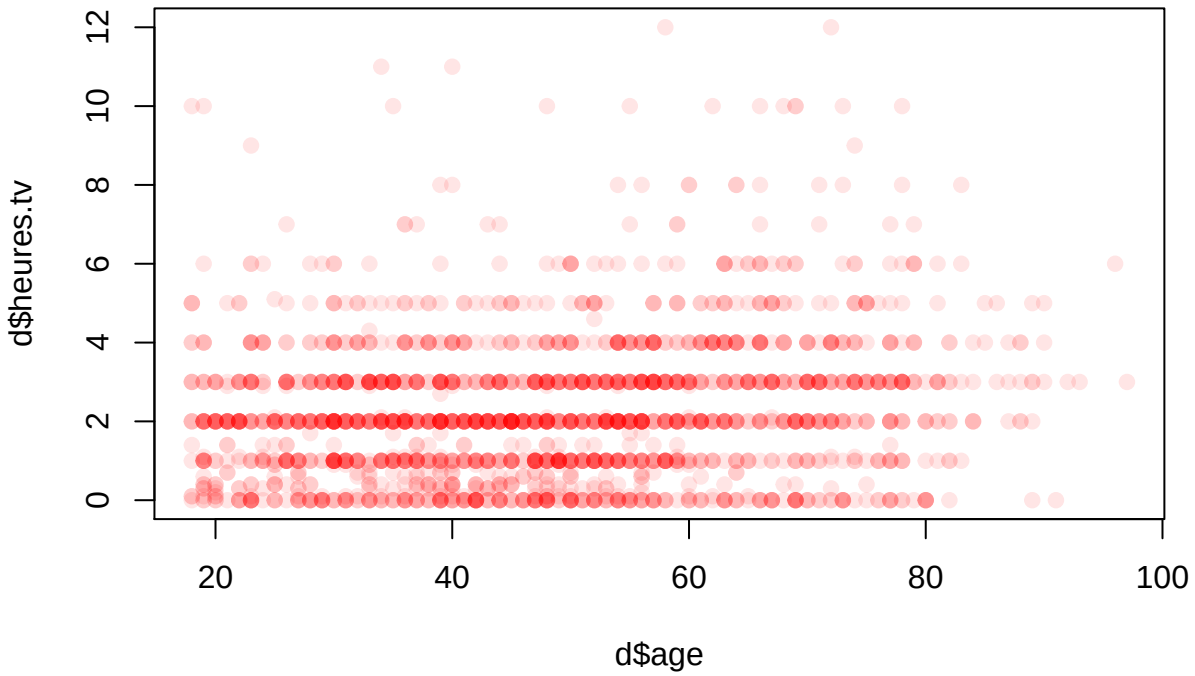
On peut ainsi représenter les valeurs du nombre d'heures passées devant la télévision selon l'âge.

```
plot(d$age, d$heures.tv)
```



Le fait que des points sont superposés ne facilite pas la lecture du graphique. On peut utiliser une représentation avec des points semi-transparents.

```
plot(d$age, d$heures.tv, pch = 19, col = rgb(1, 0, 0, 0.1))
```



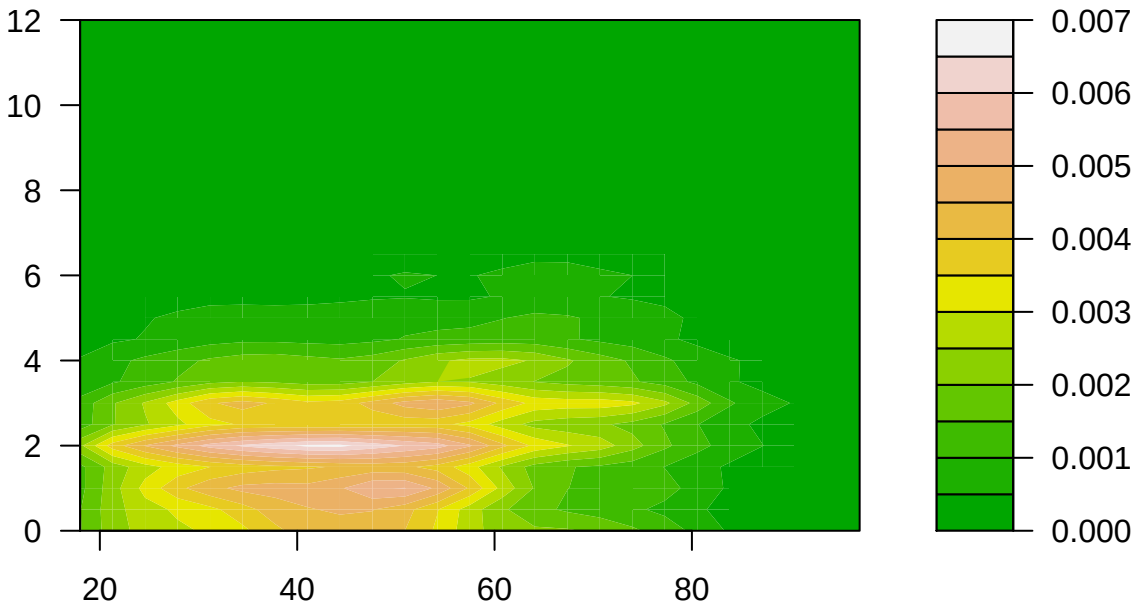
Plus sophistiqué, on peut faire une estimation locale de densité et représenter le résultat sous forme de «carte». Pour cela, le package MASS va nous aider :

```
library(MASS)
```

```
##
```

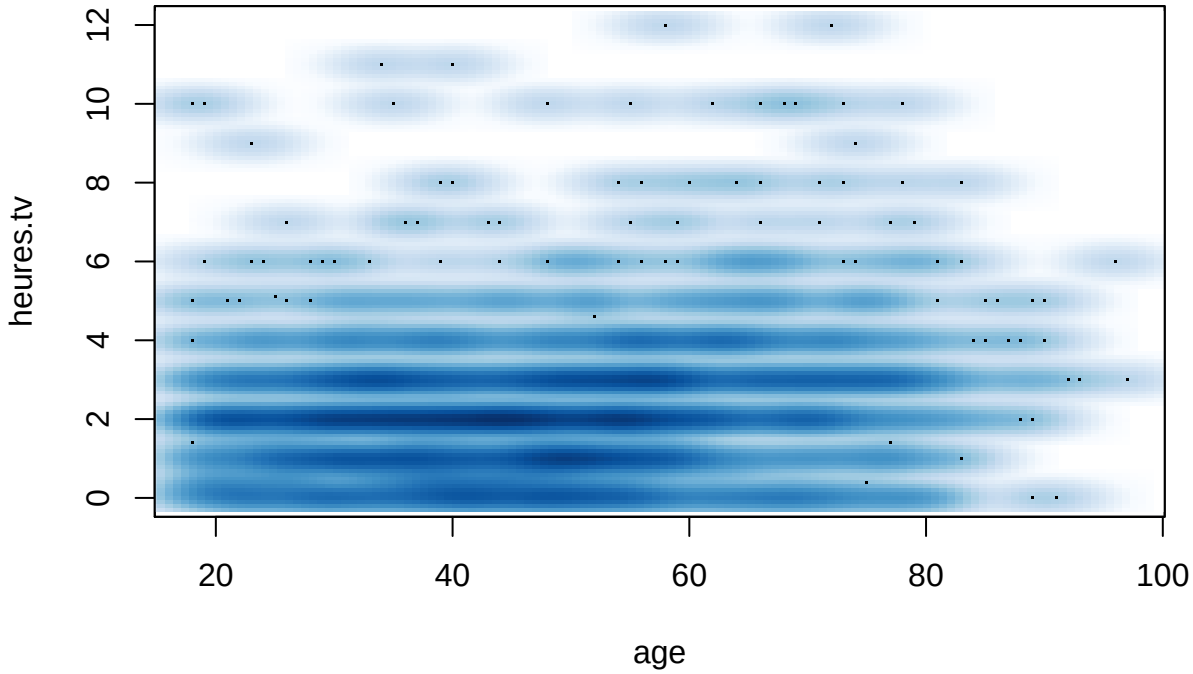
```
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
tmp <- d[, c("age", "heures.tv")]
tmp <- tmp[complete.cases(tmp), ]
filled.contour(kde2d(tmp$age, tmp$heures.tv), color = terrain.colors)
```



Une représentation alternative de la densité locale peut être obtenue avec la fonction smoothScatter.

```
smoothScatter(d[, c("age", "heures.tv")])
```



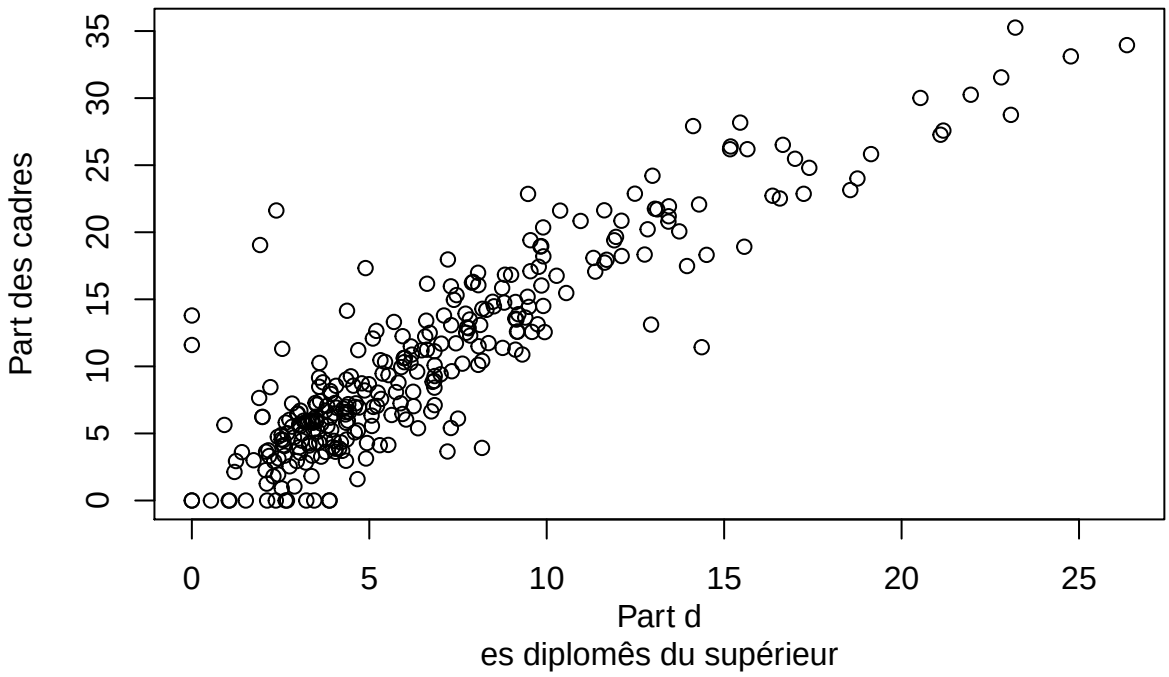
Calculons le coefficient de corrélation de ces deux variables :

```
cor(d$age, d$heures.tv, use = "complete.obs")
```

```
## [1] 0.1776249
```

L'option use permet d'éliminer les observations pour lesquelles l'une des deux valeurs est manquante. Le coefficient de corrélation est très faible.

```
plot(rp99$dipl.sup, rp99$cadres, ylab = "Part des cadres", xlab = "Part d
es diplomês du supérieur")
```



Ça ressemble déjà beaucoup plus à une relation de type linéaire. Calculons le coefficient de corrélation :

```
cor(rp99$dipl.sup, rp99$cadres)
```

```
## [1] 0.8975282
```

Effectuons alors la regression linéaire pour nos deux variables.

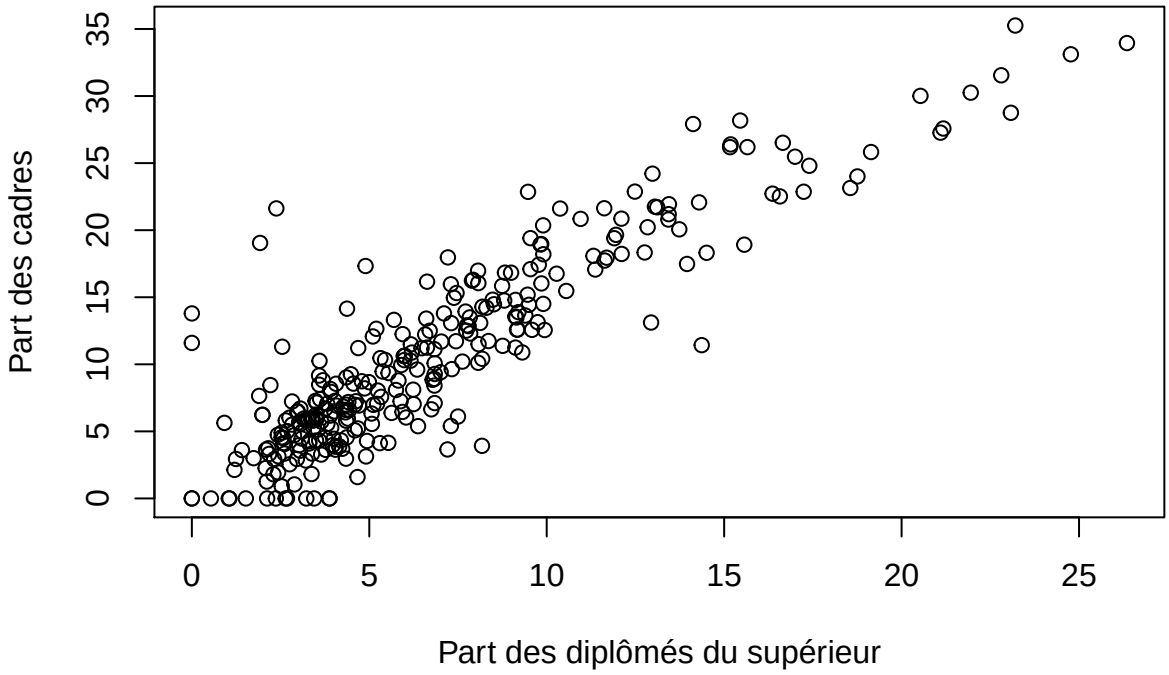
```
reg <- lm(cadres ~ dipl.sup, data = rp99)
summary(reg)
```

```
##
## Call:
## lm(formula = cadres ~ dipl.sup, data = rp99)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -9.6905 -1.9010 -0.1823  1.4913 17.0866
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.24088    0.32988   3.762 0.000203 ***
## dipl.sup     1.38352    0.03931  35.196  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.281 on 299 degrees of freedom
## Multiple R-squared:  0.8056, Adjusted R-squared:  0.8049
## F-statistic:  1239 on 1 and 299 DF,  p-value: < 2.2e-16
```

On remarque que nos variables sont statistiquement significatives. La part de cadres augmente donc avec celle de diplômés du supérieur.

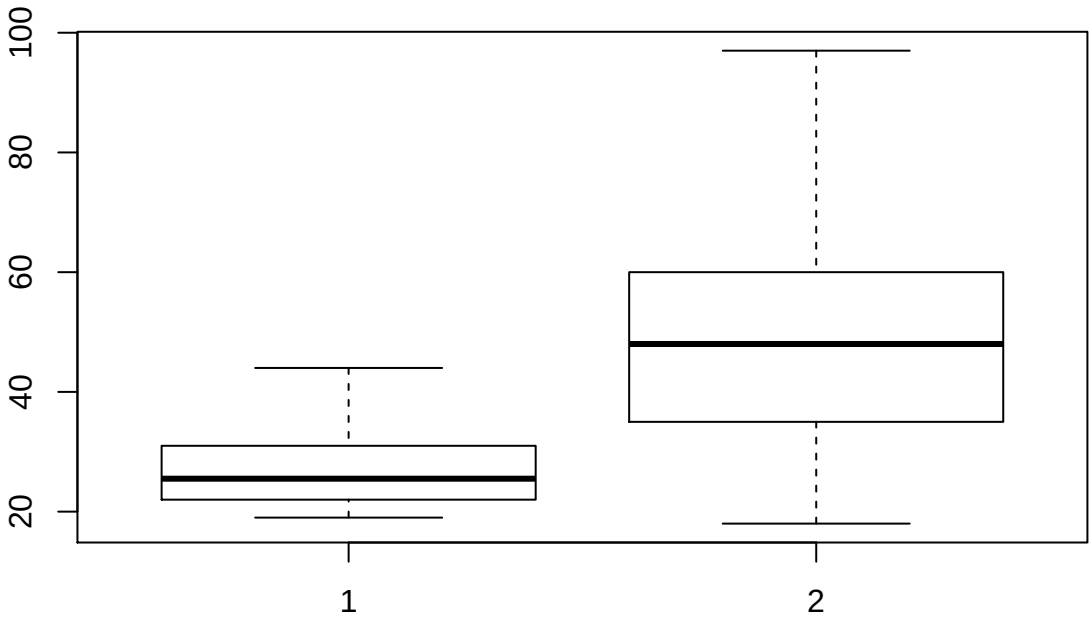
```
plot(rp99$dipl.sup, rp99$cadres, ylab = "Part des cadres", xlab = "Part des diplômés du supérieur")
```



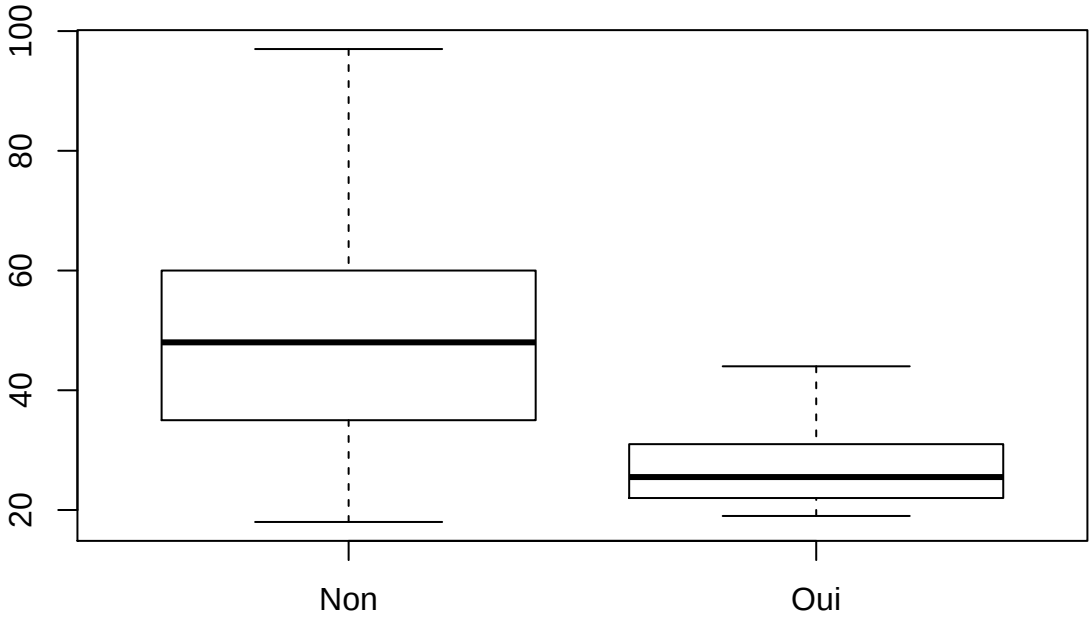
Une variable quantitative et une variable qualitative

Représentations graphiques

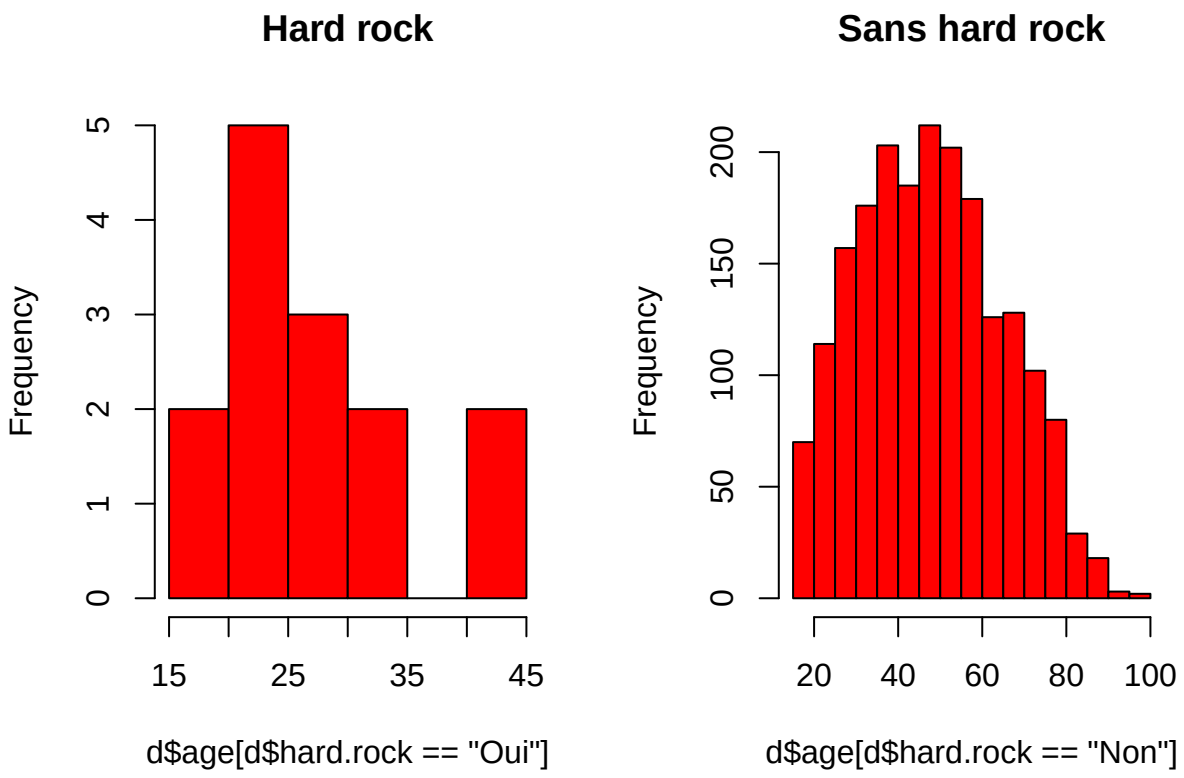
```
d.hard <- subset(d, hard.rock == "Oui")
d.non.hard <- subset(d, hard.rock == "Non")
boxplot(d.hard$age, d.non.hard$age)
```




```
boxplot(age ~ hard.rock, data = d)
```




```
par(mfrow = c(1, 2))
hist(d$age[d$hard.rock == "Oui"], main = "Hard rock", col = "red")
hist(d$age[d$hard.rock == "Non"], main = "Sans hard rock", col = "red")
```



Introduction à ggplot2, la grammaire des graphiques

R possède un puissant moteur graphique interne, qui permet de «dessiner» dans un graphique en y rajoutant des segments, des points, du texte, ou toutes sortes d'autres symboles. Toutefois, pour produire un graphique complet avec les fonctions basiques de R, il faut un peu bricoler : d'abord, ouvrir une fenêtre ; puis rajouter des points ; puis rajouter des lignes ; tout en configurant les couleurs au fur-et-à-mesure ; puis finir par fermer la fenêtre graphique.

Exemple

En 2010, les chercheurs Carmen M. Reinhart et Kenneth S. Rogoff publiaient un article intitulé *Growth in a Time of Debt*, dans lequel ils faisaient la démonstration qu'un niveau élevé de dette publique nuisait à la croissance économique. Plus exactement, les deux chercheurs y défendaient l'idée que, lorsque la dette publique dépasse 90 % du produit intérieur brut, ce produit cesse de croître.

Cette conclusion, proche du discours porté par des institutions comme le Fonds Monétaire International, *a alimenté plusieurs argumentaires politiques*. Des parlementaires américains s'en ainsi sont servi pour exiger une diminution du budget fédéral, et surtout, la Commission européenne s'est appuyée sur cet argumentaire pour exiger que des pays comme la Grèce, durement frappés par la crise financière globale de 2008, adoptent des plans d'austérité drastiques.

Or, en tentant de reproduire les résultats de Reinhart et Rogoff, les chercheurs Thomas Herndon, Michael Ash et Robert Pollin y *ont trouvé de nombreuses erreurs*, ainsi qu'une bête erreur de calcul due à une utilisation peu attentive du logiciel Microsoft Excel. La révélation de ces erreurs donna lieu à un débat très vif entre adversaires et partisans des politiques économiques d'austérité, débat toujours autant d'actualité aujourd'hui.

```r
# charger l'extension lisant le format CSV
library(readr)
library(dplyr)
library(ggplot2)
```

Les données de Reinhart et Rogoff contiennent, pour un échantillon de 20 pays occidentaux membres de la zone OCDE, la croissance de leur produit intérieur brut (PIB), et le ratio entre leur dette publique et ce produit, exprimé sous la forme d'un pourcentage «Dette/PIB». Les données vont du milieu des années 1940 à la fin des années 2000. La première colonne du jeu de données ne contenant que les numéros des lignes, on va la supprimer d'entrée de jeu :

```r
# suppression de la première colonne
debt <- read_csv("~/Videos/Unicamp_IE 2019/HO:012A Economia Matemática/debt.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   Country = col_character(),
##   Year = col_double(),
##   growth = col_double(),
##   ratio = col_double()
## )
```

```r
debt <- debt[, -1]
print(debt)
```

```
## # A tibble: 1,171 x 4
##    Country    Year growth ratio
##    <chr>     <dbl>  <dbl> <dbl>
##  1 Australia  1946 -3.56  190.
##  2 Australia  1947  2.46  177.
##  3 Australia  1948  6.44  149.
##  4 Australia  1949  6.61  126.
##  5 Australia  1950  6.92  110.
##  6 Australia  1951  4.27   87.1
##  7 Australia  1952  0.905  86.1
##  8 Australia  1953  3.12   79.9
##  9 Australia  1954  6.22   76.8
## 10 Australia  1955  5.46   75.0
## # ... with 1,161 more rows
```

Il faut aussi noter d'emblée que certaines mesures sont manquantes : pour certains pays, on ne dispose pas d'une mesure fiable du PIB et/ou de la dette publique. En conséquence, le nombre d'observations par pays est différent, et va de 40 observations «pays-année» pour la Grèce à 64 observations «pays-année» pour plusieurs pays comme l'Australie ou les États-Unis :

```r
table(debt$Country)
```

```
##
##   Australia     Austria     Belgium      Canada     Denmark     Finland
##          64          59          63          64          56          64
##      France     Germany      Greece     Ireland       Italy       Japan
##          54          59          40          63          59          54
## Netherlands New Zealand      Norway    Portugal       Spain      Sweden
##          53          64          64          58          42          64
##          UK          US
##          63          64
```

Recodage d'une variable

Dernière manipulation préalable avant l'analyse : on va calculer la décennie de chaque observation, en divisant l'année de mesure par 10, et en multipliant la partie entière de ce résultat par 10. Cette manipulation très simple donne «1940» pour les mesures des années 1940 à 1949, «1950» pour les années 1950-1959, et ainsi de suite.

```
debt$Decade <- factor(10 * debt$Year%/%10)
```

Voici comment représente nos données:

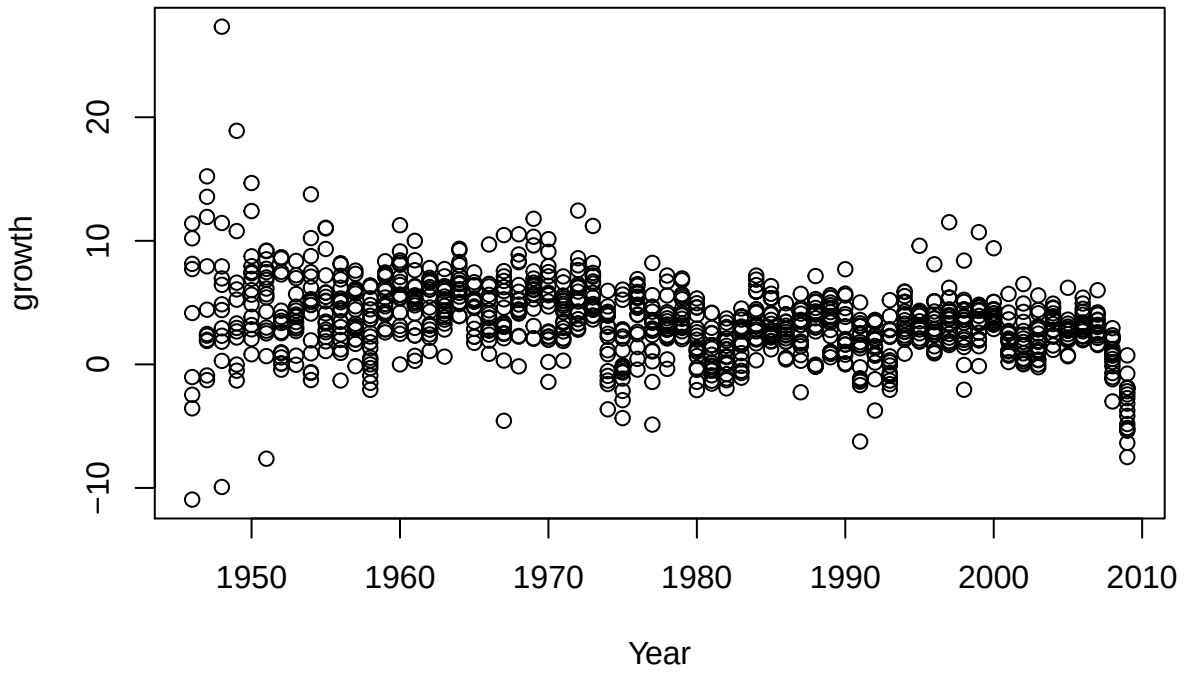
```
head(debt)
```

```
## # A tibble: 6 x 5
##   Country    Year growth ratio Decade
##   <chr>     <dbl>  <dbl> <dbl> <fct>
## 1 Australia  1946  -3.56 190.  1940
## 2 Australia  1947   2.46 177.  1940
## 3 Australia  1948   6.44 149.  1940
## 4 Australia  1949   6.61 126.  1940
## 5 Australia  1950   6.92 110.  1950
## 6 Australia  1951   4.27  87.1 1950
```

Visualisation des données

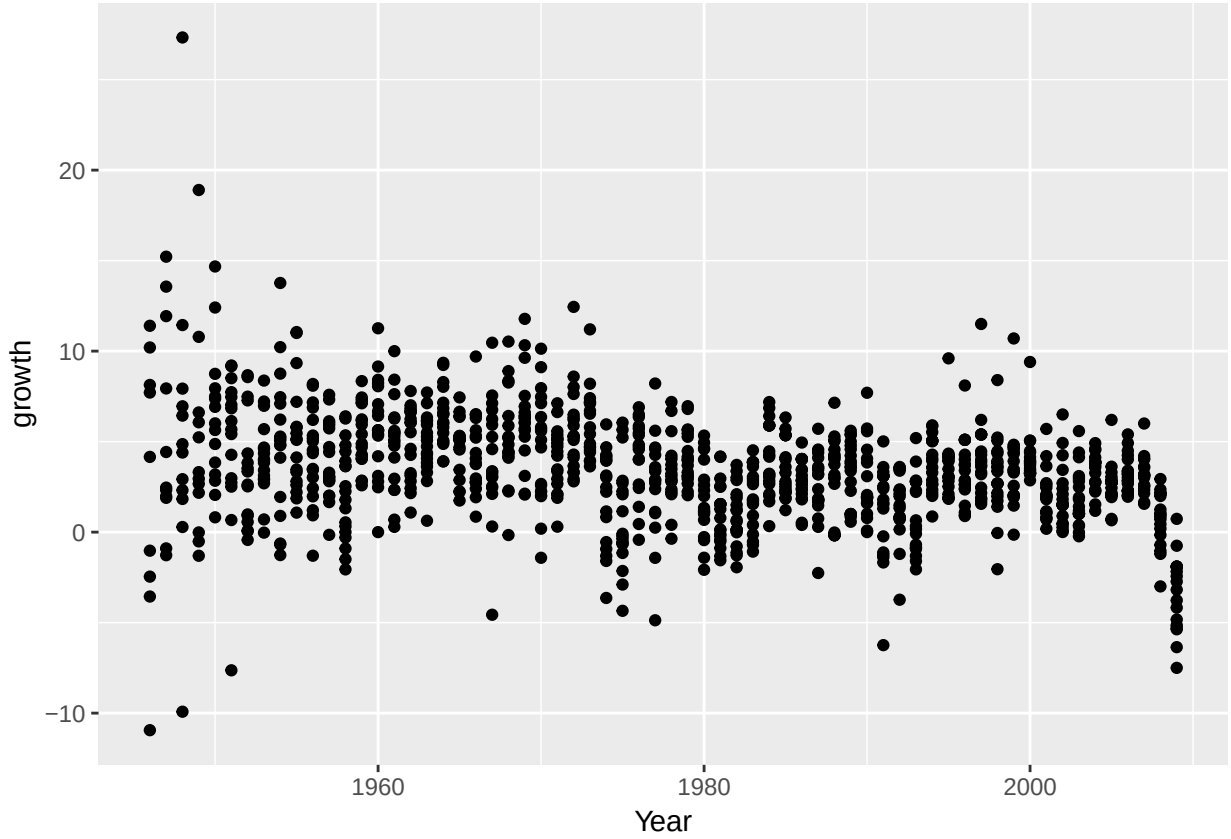
Procédons désormais à quelques visualisations très simples de ces données. On dispose de trois variables continues : l'année, le taux de croissance du PIB, et le ratio «Dette publique/PIB».

```
with(debt, plot(Year, growth))
```

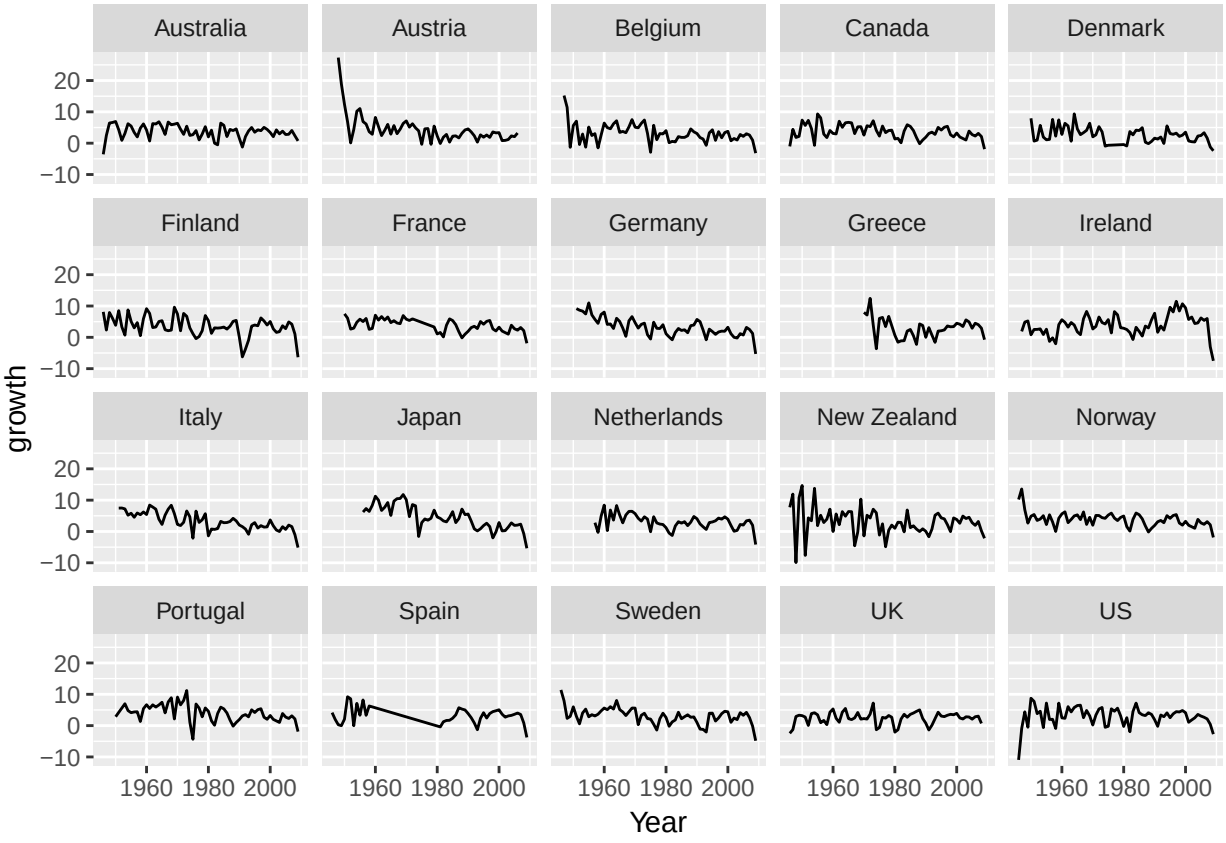


Le même graphique s'écrit de la manière suivante avec l'extension ggplot2 :

```
with(debt, qplot(Year, growth))
```



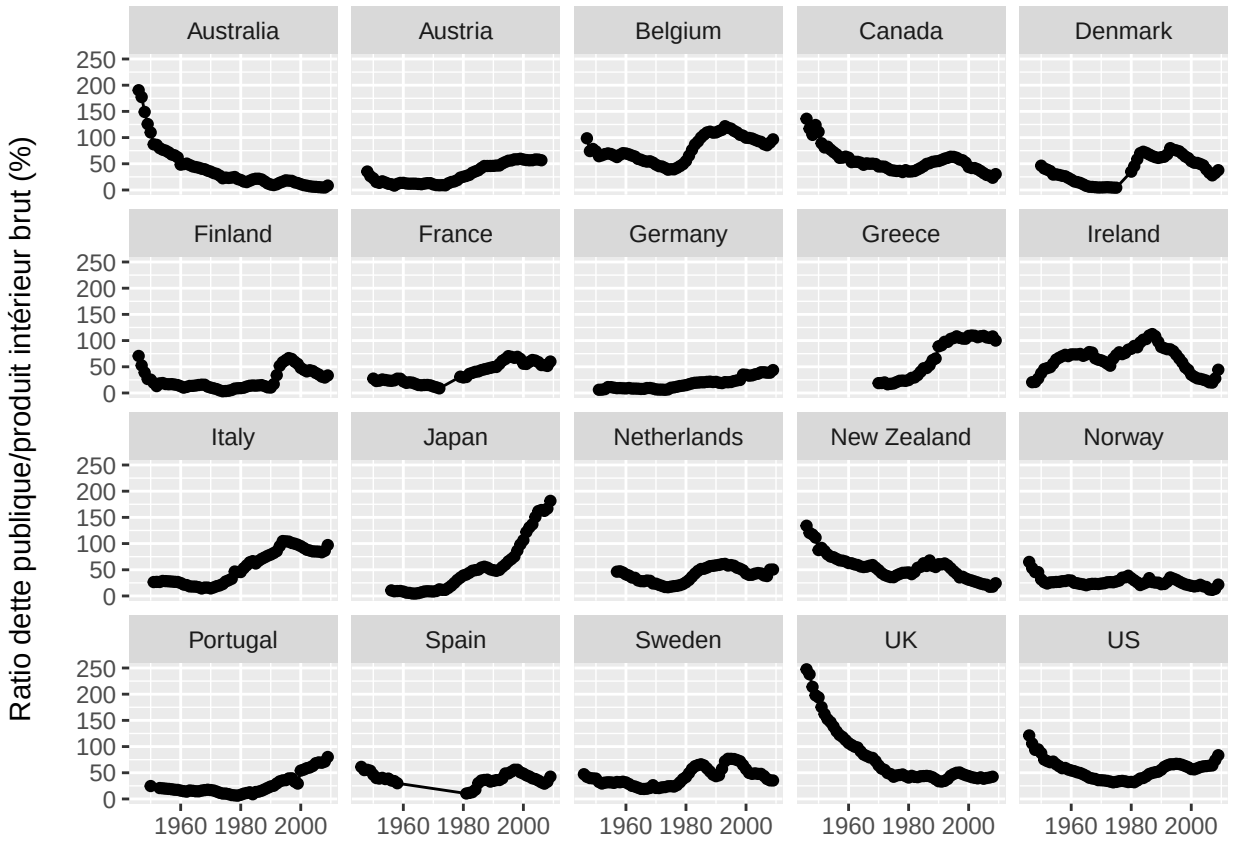
```
qplot(Year, growth, data = debt, geom = "line" ) +
  facet_wrap(~ Country)
```



Combinaisons d'éléments graphiques

On n'a pas encore visualisé le ratio «Dette publique / PIB», l'autre variable du raisonnement de Reinhart et Rogoff. C'est l'occasion de voir comment rajouter des titres aux axes des graphiques, et d'utiliser les lignes en même temps que des points, toujours grâce à l'argument geom , qui peut prendre plusieurs valeurs (ici, "point" produit les points et "line" produit les lignes) :

```
qplot(data = debt, y = ratio, x = Year, geom = c("line", "point")) + facet_wrap(~ Country) +
labs(x = NULL, y = "Ratio dette publique/produit intérieur brut (%)\n")
```



Composition graphique avec ggplot2

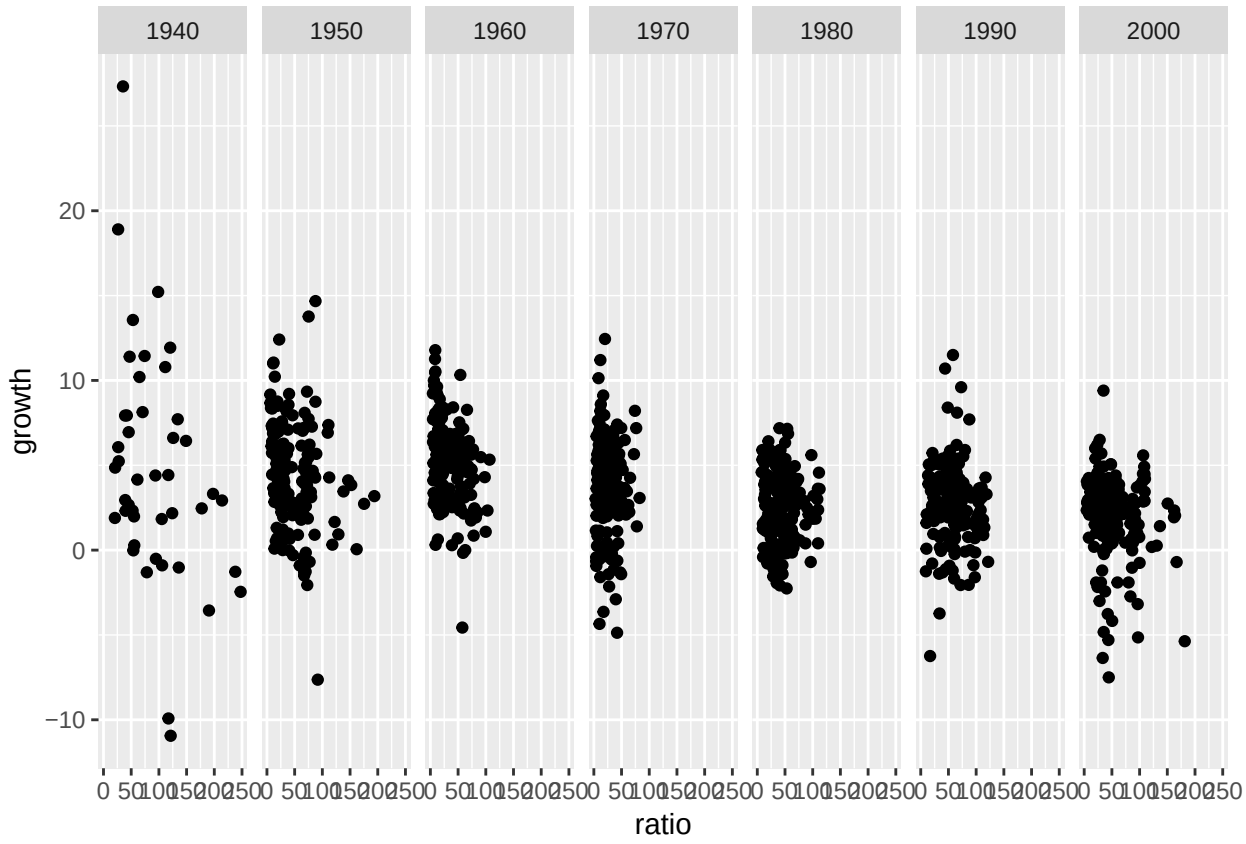
La section précédente a montré comment utiliser la fonction qplot (quick plot). La syntaxe complète de l'extension ggplot2 passe par une autre fonction, ggplot , qui permet de mieux comprendre les différents éléments de sa grammaire graphique. Dans cette section, on va détailler cette syntaxe pour en tirer un graphique plus complexe que les précédents.

```
p <- ggplot(data = debt, aes(y = growth, x = ratio))
```

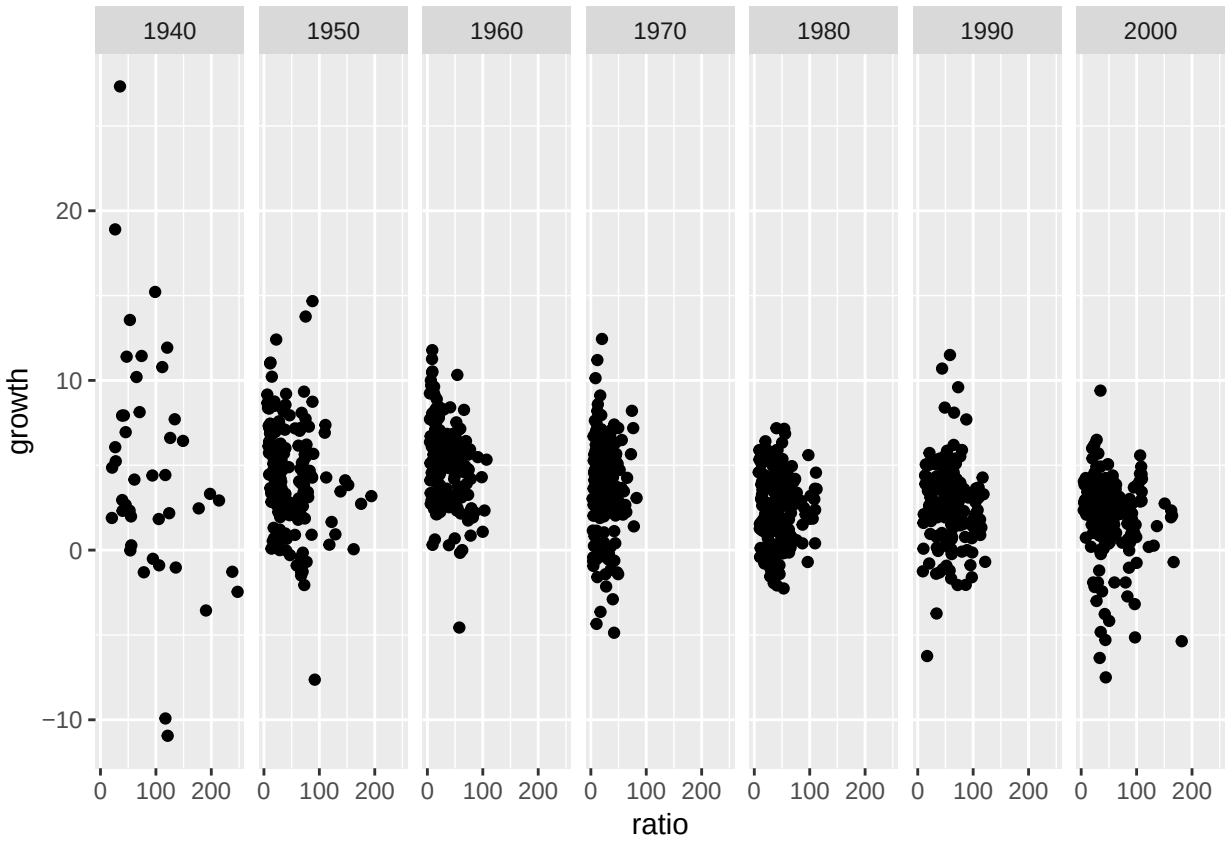
Aucun graphique ne s'affiche ici : en effet, ce que l'on a stocké, dans l'objet p , n'est pas un graphique complet, mais une base de travail. Cette base définit les coordonnées x et y du graphique dans l'argument aes (aesthetics). Ici, on a choisi de mettre la variable dépendante de Reinhart et Rogoff, growth (le taux de croissance du PIB), sur l'axe y , et la variable indépendante ratio (le ratio «Dette publique / PIB») sur l'axe x .

Rajoutons désormais un objet géométrique, geom_point , qui va projeter, sur le graphique, des points aux coordonnées précédemment définies, et divisons le graphique par un «petit multiple», en projetant les points de chaque décennie dans une facette différente du graphique. Ce graphique propose une décomposition temporelle de la relation étudiée par Reinhart et Rogoff :

```
p + geom_point() + facet_grid(. ~ Decade)
```




```
p + geom_point() + facet_grid(. ~ Decade) +
  scale_x_continuous(breaks = seq(0, 200, by = 100))
```



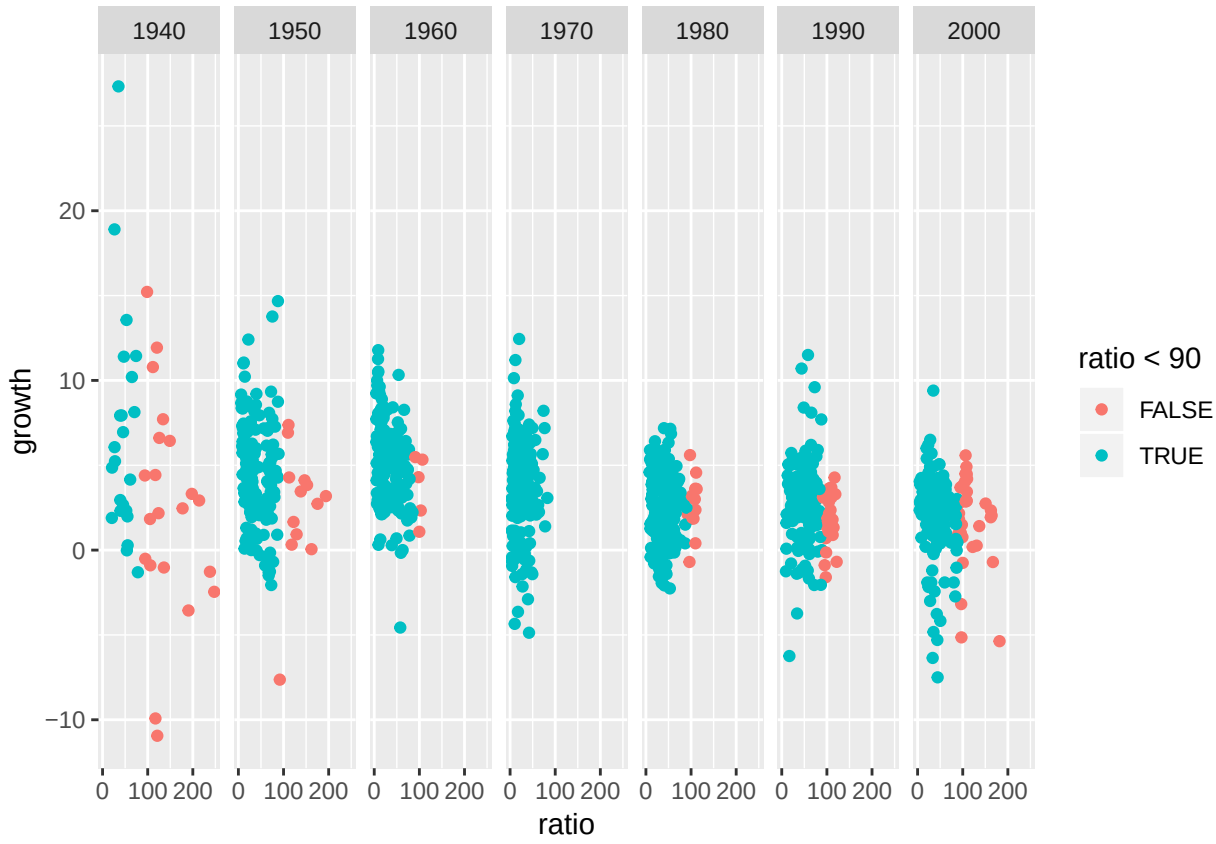
On va donc les sauvegarder dans l'objet p , de manière à continuer de construire notre graphique en incluant ces différents éléments.

```
p <- p + geom_point() + facet_grid(. ~ Decade) +
scale_x_continuous(breaks = seq(0, 200, by = 100))
```

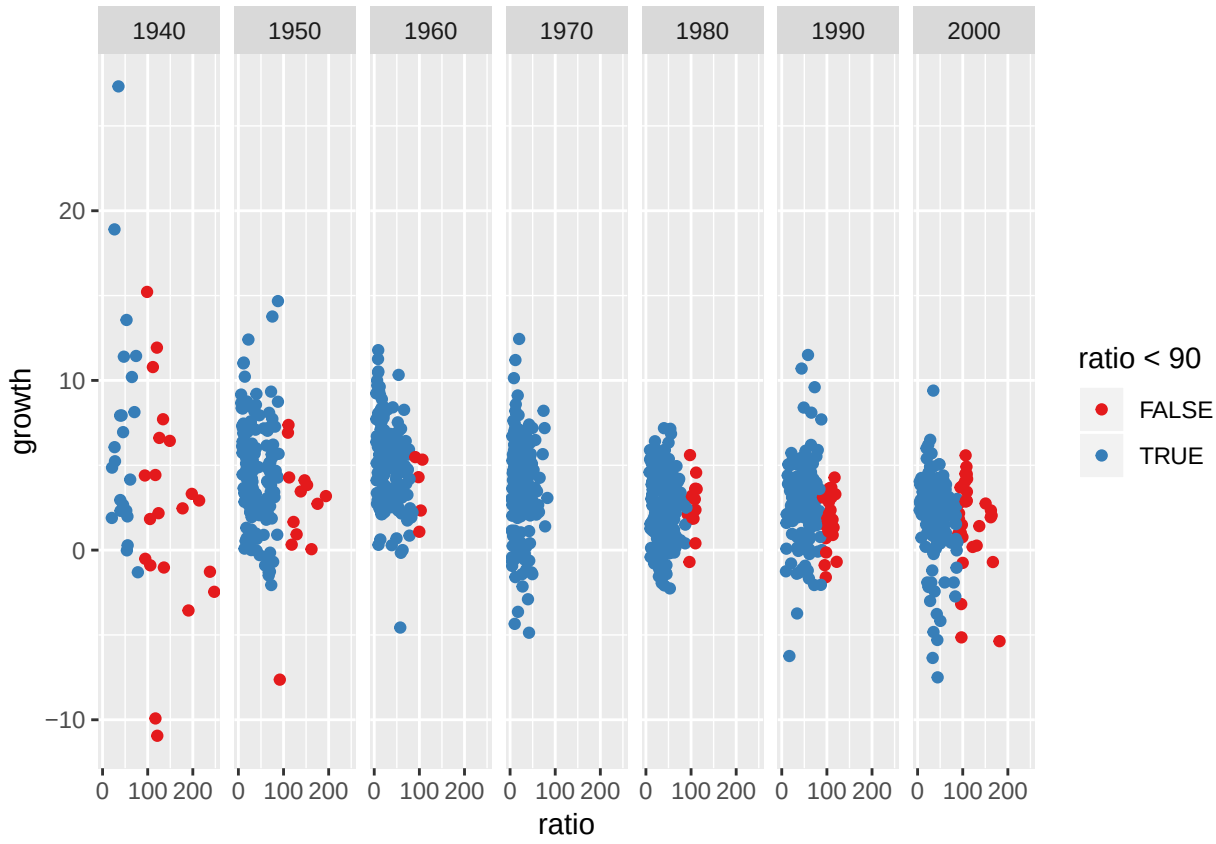
Couleurs et échelles

Abordons désormais un élément-clé de ggplot2 : la manipulation des paramètres esthétiques. Précédemment, on n'a montré que deux de ces paramètres : x et y , les coordonnées du graphique. Mais ces paramètres peuvent aussi influencer la couleur des points de notre graphique comme le montre l'exemple suivant :

```
p + aes(color = ratio < 90)
```



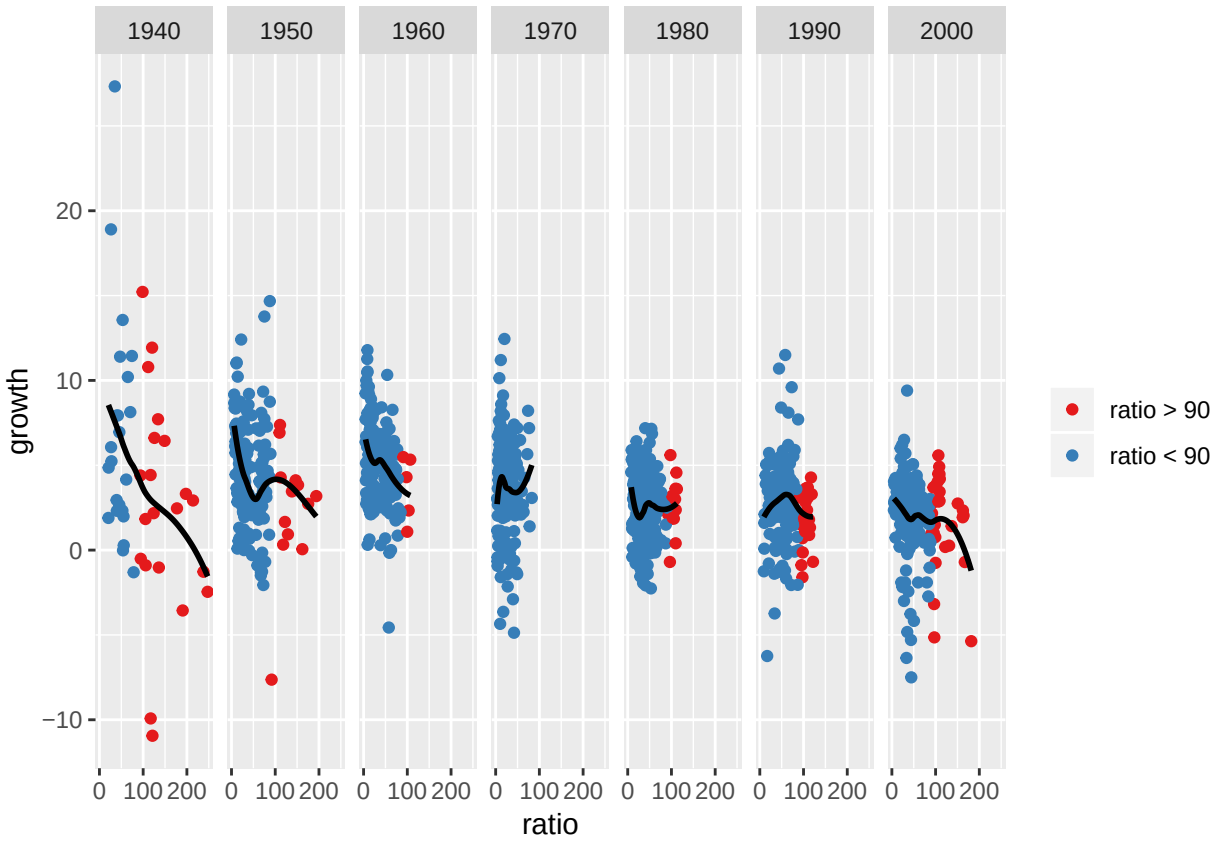

```
p + aes(color = ratio < 90) + scale_colour_brewer(palette = "Set1")
```




```
p <- p + aes(color = ratio < 90) + scale_color_brewer("", palette = "Set1",
labels = c("ratio > 90", "ratio < 90"))
```

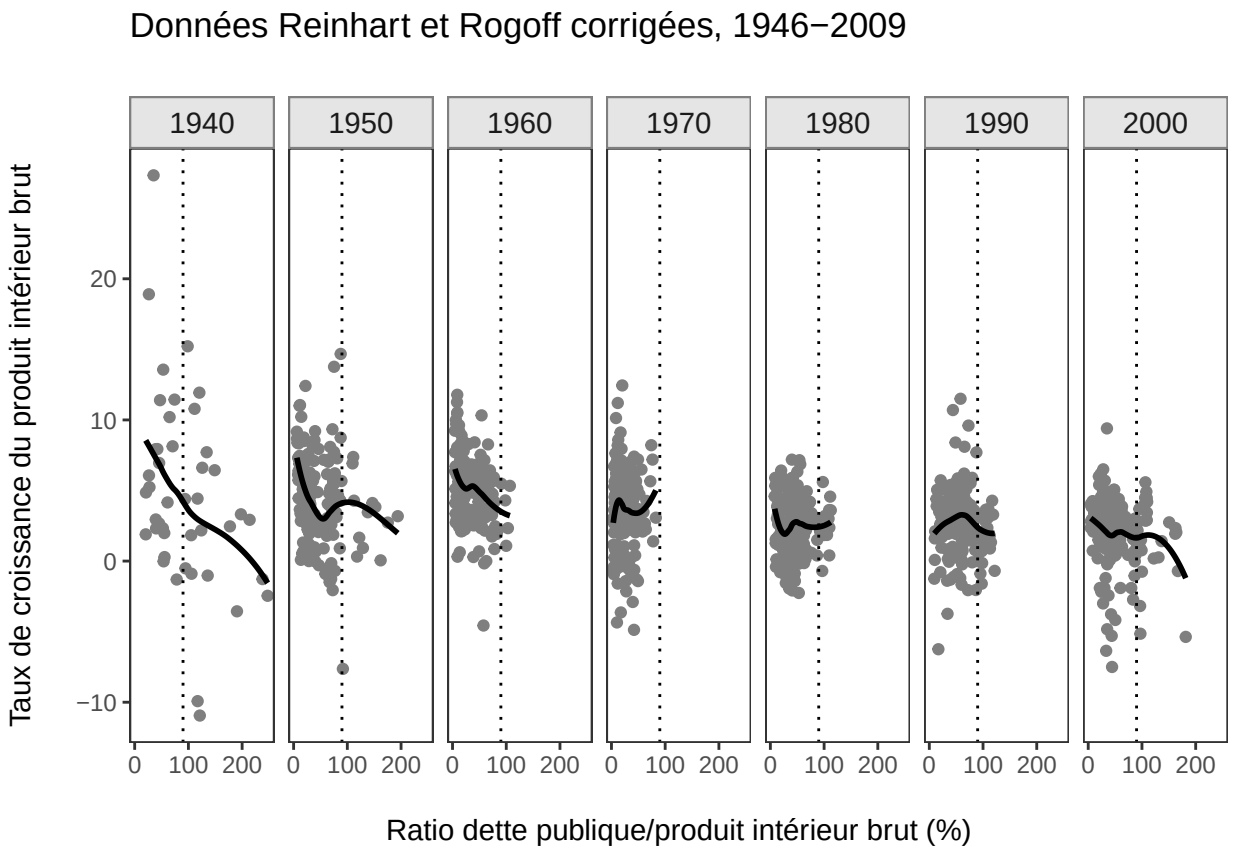
Dans le bloc de code ci-dessus, on a stocké l'ensemble de nos modifications dans l'objet p , sans l'afficher; en effet, on souhaite encore procéder à une dernière modification, en rajoutant une régression locale à travers les points de chaque facette.

```
p + geom_smooth(method = "loess", se = FALSE, size = 1, color = "black")
```



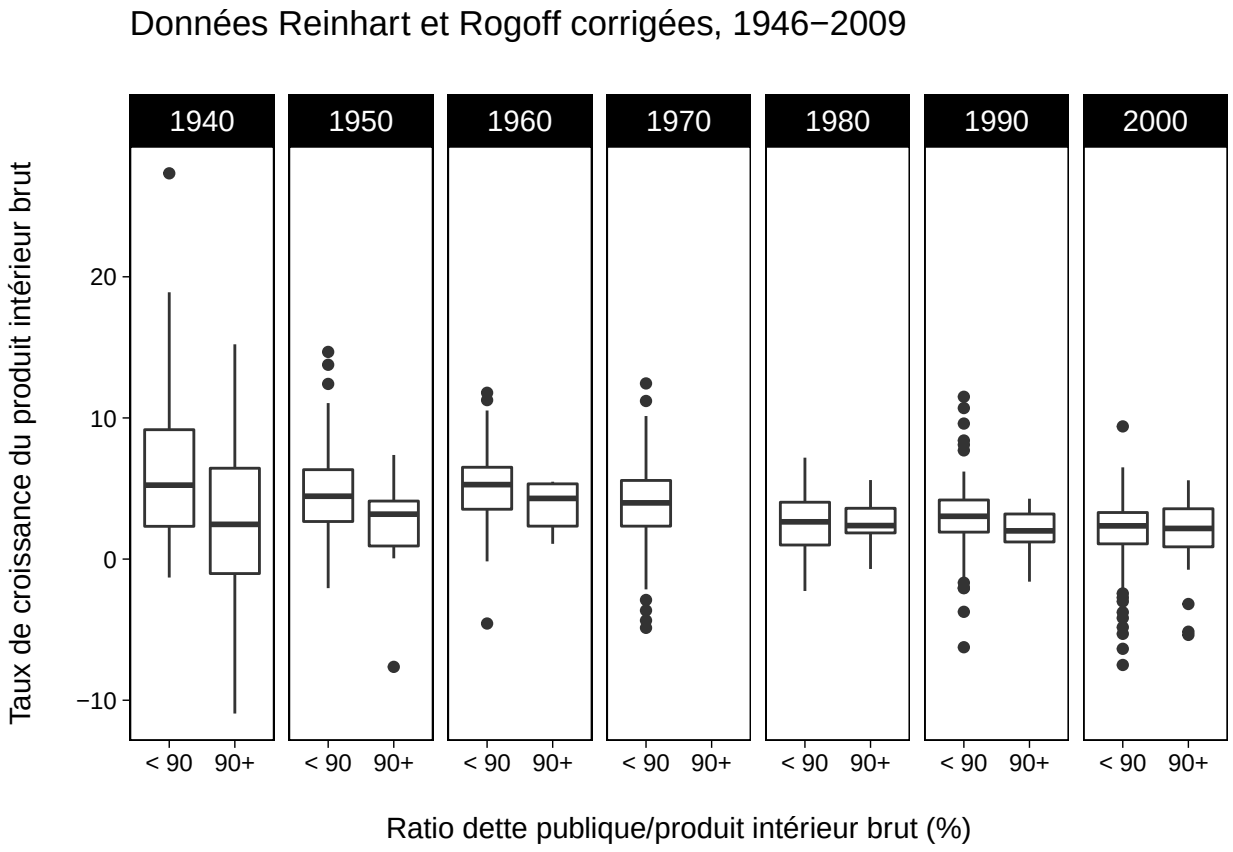
Le graphique permet d'évaluer de manière encore un peu plus précise l'argument de Reinhart et Rogoff, et en particulier la nature pas si «fatidique» du seuil de 90% du ratio "Dette publique/PIB", qui sans être une bonne nouvelle pour l'économie, ne détermine pas «fatidiquement» la direction du taux de croissance : si c'était le cas, toutes les courbes du graphique ressembleraient à celles des années 2000. Autrement dit, l'argumentaire de Reinhart et Rogoff laisse clairement à désirer.

```
ggplot(data = debt, aes(y = growth, x = ratio)) + geom_point(color = "grey50") +
geom_vline(xintercept = 90, lty = "dotted") + geom_smooth(method = "loess", size = 1,
                                                    color = "black", se = FALSE) +
scale_x_continuous(breaks = seq(0, 200, by = 100)) + facet_grid(. ~ Decade) +
labs(y = "Taux de croissance du produit intérieur brut\n",
x = "\nRatio dette publique/produit intérieur brut (%)",
title = "Données Reinhart et Rogoff corrigées, 1946-2009\n") + theme_bw() +
theme(strip.background = element_rect(fill = "grey90", color = "grey50"),
strip.text = element_text(size = rel(1)), panel.grid = element_blank())
```



L'extension de *ggplot* permet de produire plusieurs types des graphiques dès que l'on connaît les principaux éléments de sa syntaxe, par exemple, **boxplot**.

```
ggplot(data = debt, aes(x = ratio > 90, y = growth)) + geom_boxplot() +
scale_x_discrete(labels = c("< 90", "90+")) + facet_grid(. ~ Decade) +
labs(y = "Taux de croissance du produit intérieur brut\n",
x = "\nRatio dette publique/produit intérieur brut (%)",
title = "Données Reinhart et Rogoff corrigées, 1946-2009\n") + theme_linedraw() +
theme(strip.text = element_text(size = rel(1)), panel.grid = element_blank())
```



Graphiques univariés et bivariés avec ggplot2