ping - sujet 4 module 3 du MOOC RR

Bertrand Muller

Ayant galéré à rendre les données de base dans un format manipulable, j'ai simplifié le jeu de données dans Excel en transformant les données de date en prenant t=0 pour la première mesure, retirant toutes les colonnes inutiles et ne laissant que les deux colonnes utiles et crée un csv avec 3 colonnes : time, size, duration (of the transfer).

```
data_path = "C:/Users/muller/Desktop/mooc/rr/mooc-rr/module3/liglab2.csv"
```

Téléchargement

```
data = read.csv2(data_path, sep=(';'), dec = ",", header = TRUE)
head(data)
```

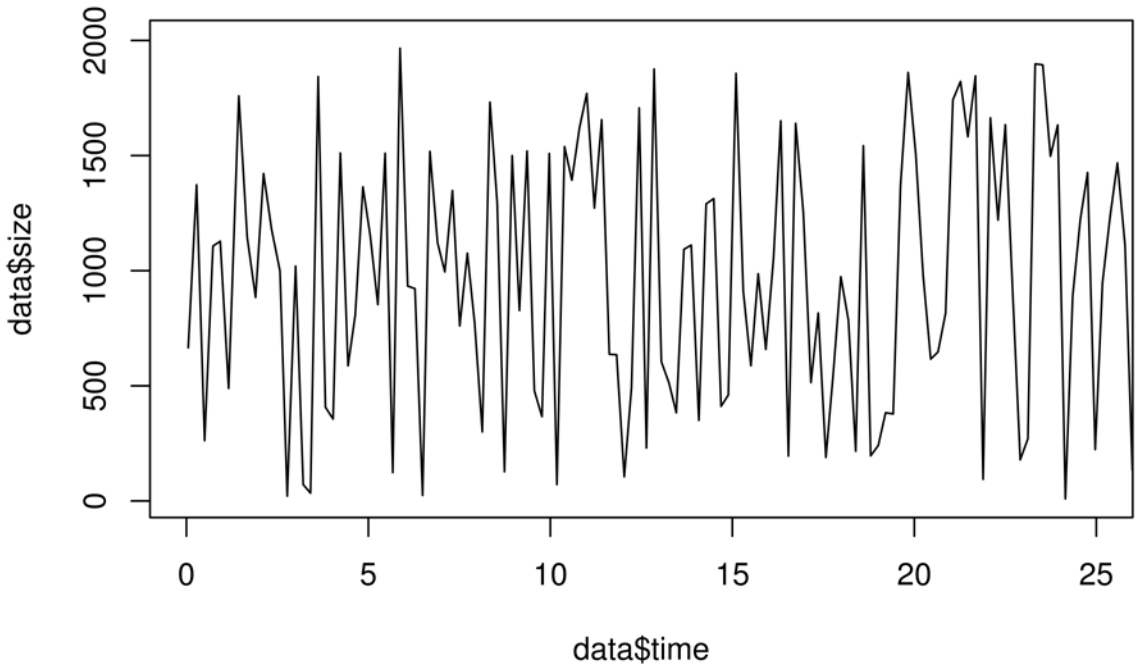
```
##   time size duration
## 1 0.05  665    22.50
## 2 0.28 1373    21.20
## 3 0.50  262    21.20
## 4 0.73 1107    23.30
## 5 0.93 1128     1.41
## 6 1.16  489    21.90
```

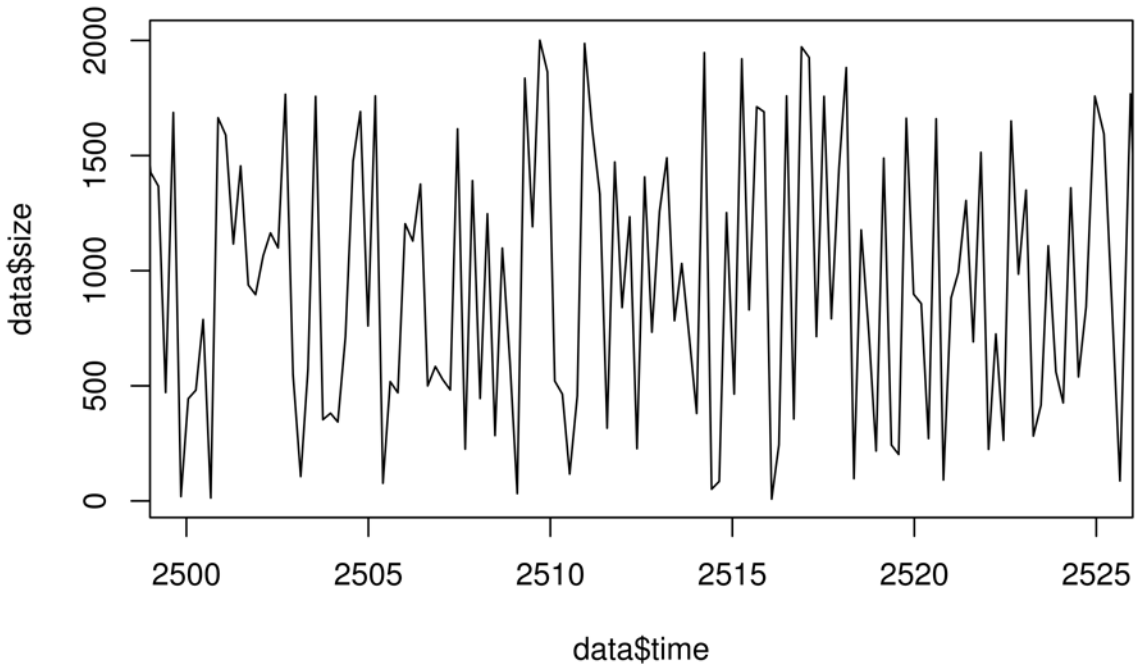
exploration des données sur un échantillon de graphes (4) représentant 1% du jeu de données et à peu près lisibles :

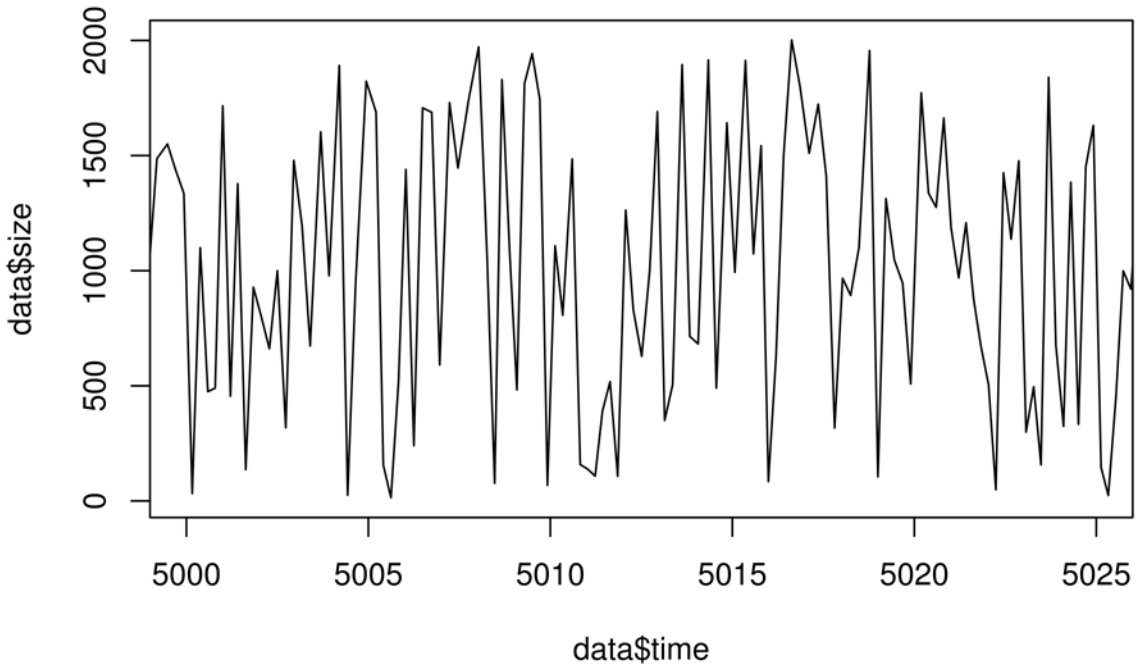
```
max(data$time)
```

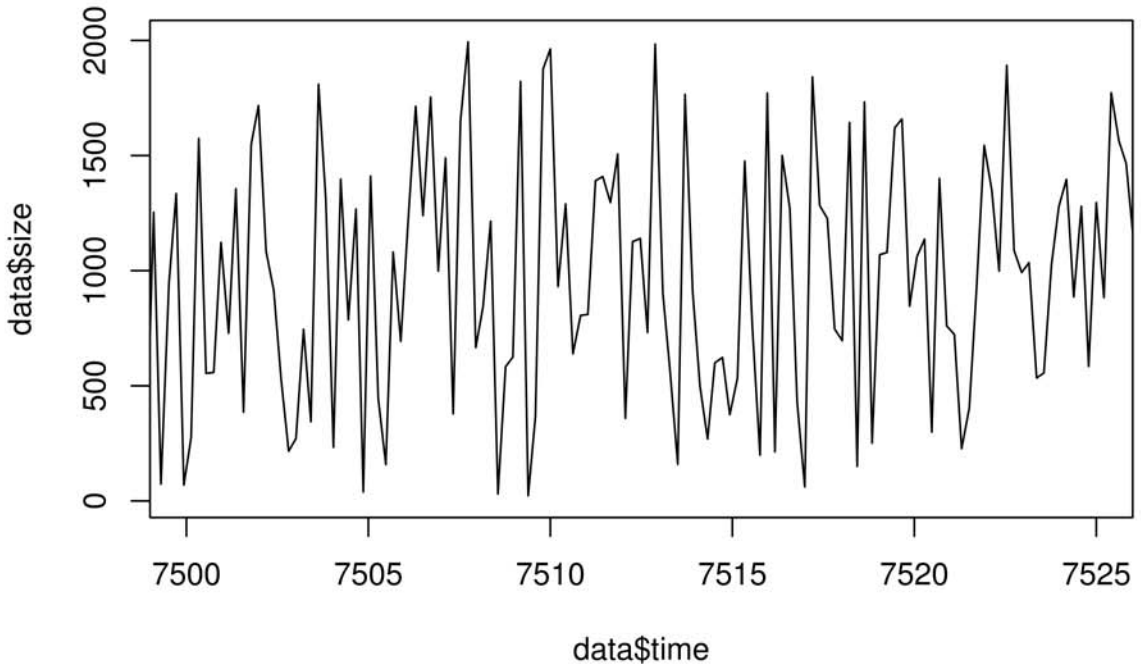
```
## [1] 9504.84
```

```
interval_size <- 25
number_plot <- round(max(data$time)/interval_size, digit=0)/100
for (i in 0:number_plot) {
plot(data$time, data$size, type="l", xlim = c(interval_size*i*100,interval_size*(i*100+1)))
}
```


data$size
0
500
1000
1500
2000
0
5
10
15
20
25
data$time

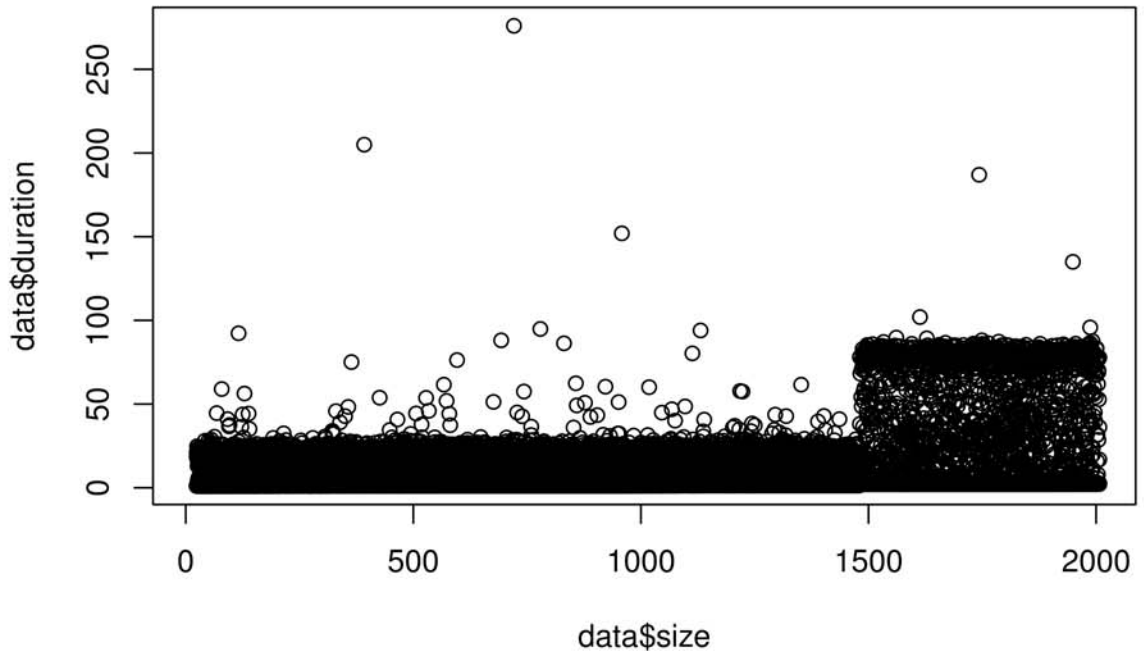

data$size
0
500
1000
1500
2000
2500
2505
2510
2515
2520
2525
data$time


data$size
0
500
1000
1500
2000
5000
5005
5010
5015
5020
5025
data$time



relation entre taille et durée du transfert..

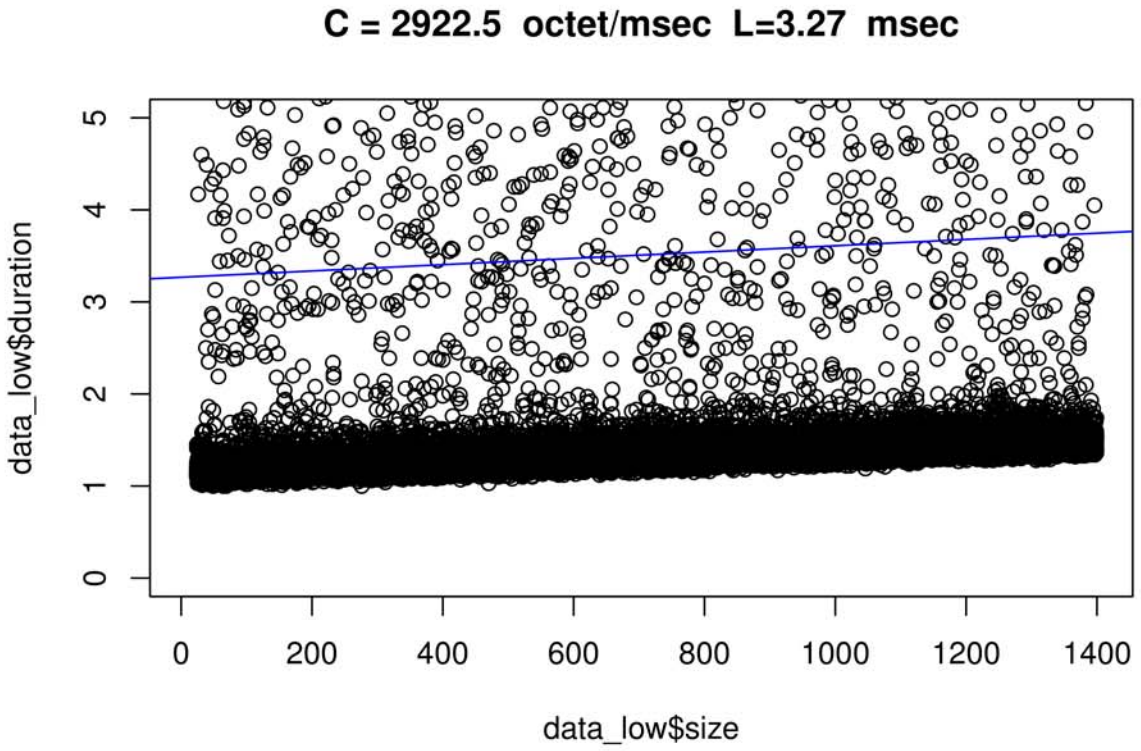
```
plot (data$size, data$duration, type = "p")
```



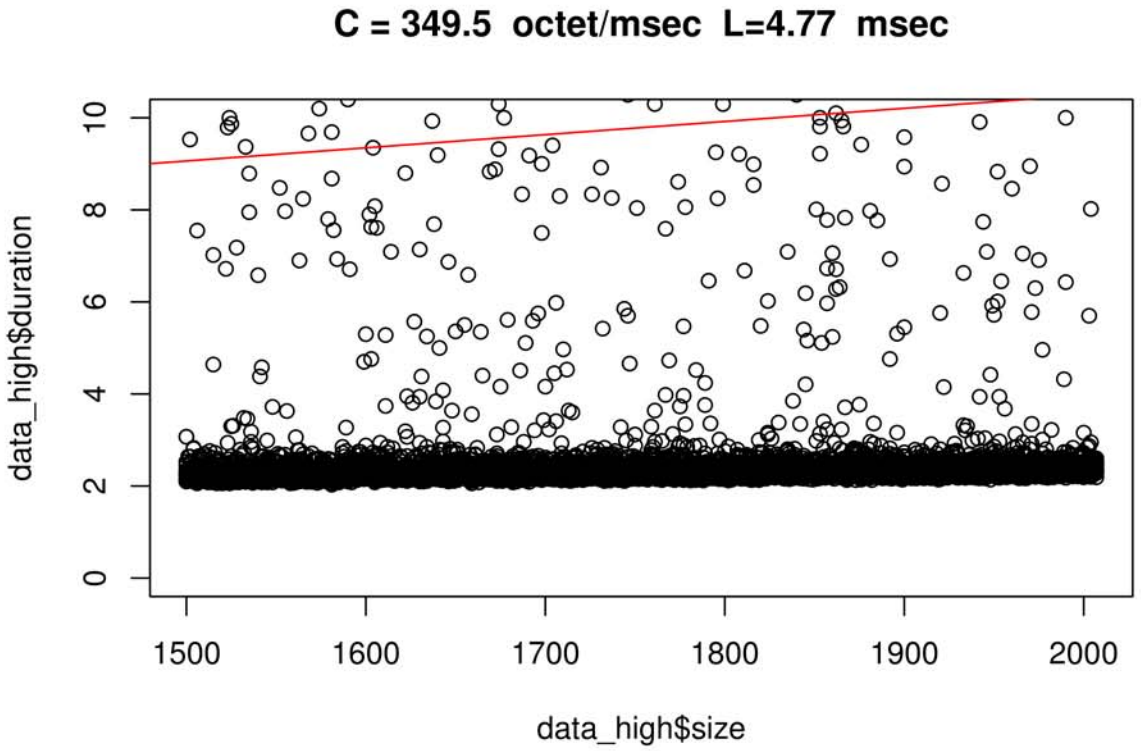
Une rupture apparait un peu en dessous de 1500 octet. On coupe les données en deux.On laisse volontairement un vide entre 1400 et 1500 pour ne pas empiéter sur l'intervalle voisin

```
data_low <- subset(data, data$size<1400)
data_high <- subset(data, data$size>=1500)


reg <- lm(duration ~ size, data = data_low)
coeff=coefficients(reg)
# Equation de la droite de regression :
eq = paste0("C = ", round(1/coeff[2],1), "  octet/msec  ", "L=", round(coeff[1],2), "  msec")
# Graph
plot(data_low$size, data_low$duration, type = "p", main = eq, ylim=c(0,5))
abline(reg=reg, col="blue")
```




```
reg <- lm(duration ~ size, data = data_high)
coeff=coefficients(reg)
# Equation de la droite de regression :
eq = paste0("C = ", round(1/coeff[2],1), "  octet/msec  ", "L=", round(coeff[1],2), "  msec")
plot(data_high$size, data_high$duration, type = "p", main = eq, ylim=c(0,10))
abline(reg=reg, col="red")
```



On utilise la régression quantile pour donner moins de poids aux outlier. Le principal effet concerne l'abscisse à l'origine (et pas la pente)

```
library(quantreg)
```

```
## Warning: package 'quantreg' was built under R version 3.5.3
```

```
## Loading required package: SparseM
```

```
## Warning: package 'SparseM' was built under R version 3.5.2
```

```
##
## Attaching package: 'SparseM'
```

```
## The following object is masked from 'package:base':
##
##     backsolve
```

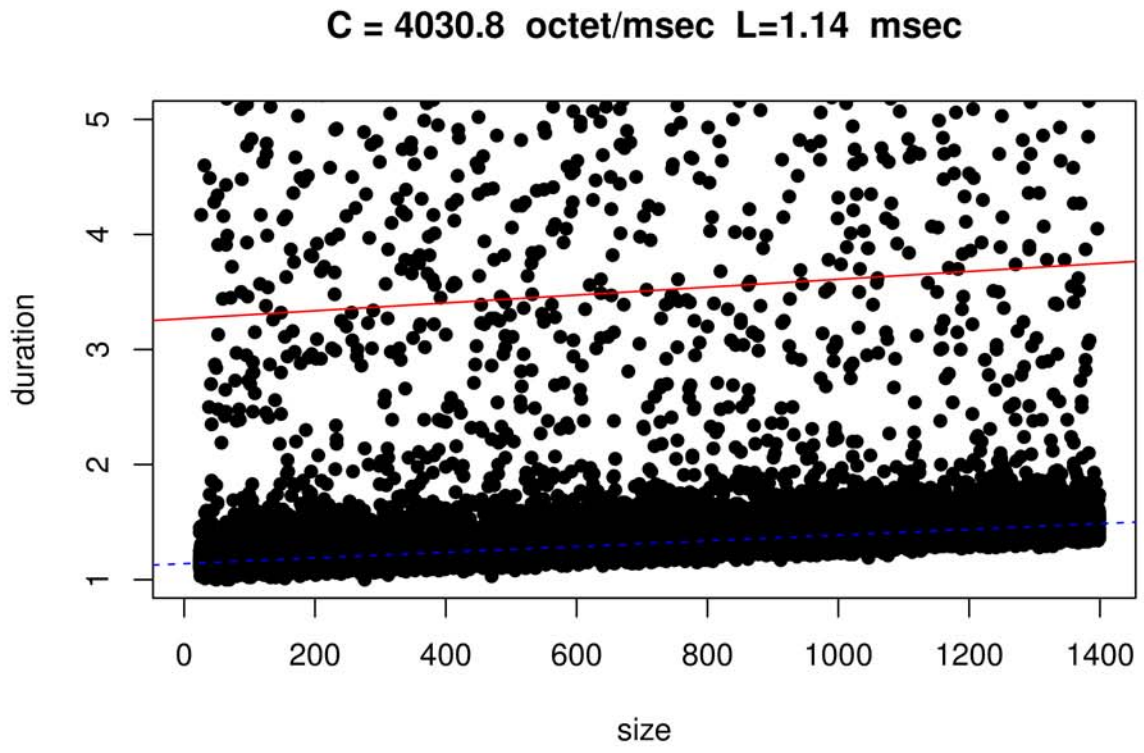
```
# regression against 50% quantile in the lower part
################
rqfit <- rq(duration ~ size, data = data_low, tau = 0.5)
coeff=coefficients(rqfit)

# Equation de la droite de regression quantile :
```

```r
eq = paste0("C = ", round(1/coeff[2],1), "  octet/msec  ", "L=", round(coeff[1],2), "  msec")

plot(duration ~ size, data = data_low, pch = 16, main = eq, ylim=c(1,5))
abline(rq(duration ~ size, data = data_low), col = "blue", lty = 2, ylim=c(1,5))

#plotting again the regression on the same graph
reg <- lm(duration ~ size, data = data_low)
abline(reg, col="red", lty = 1, ylim=c(1,5))
```

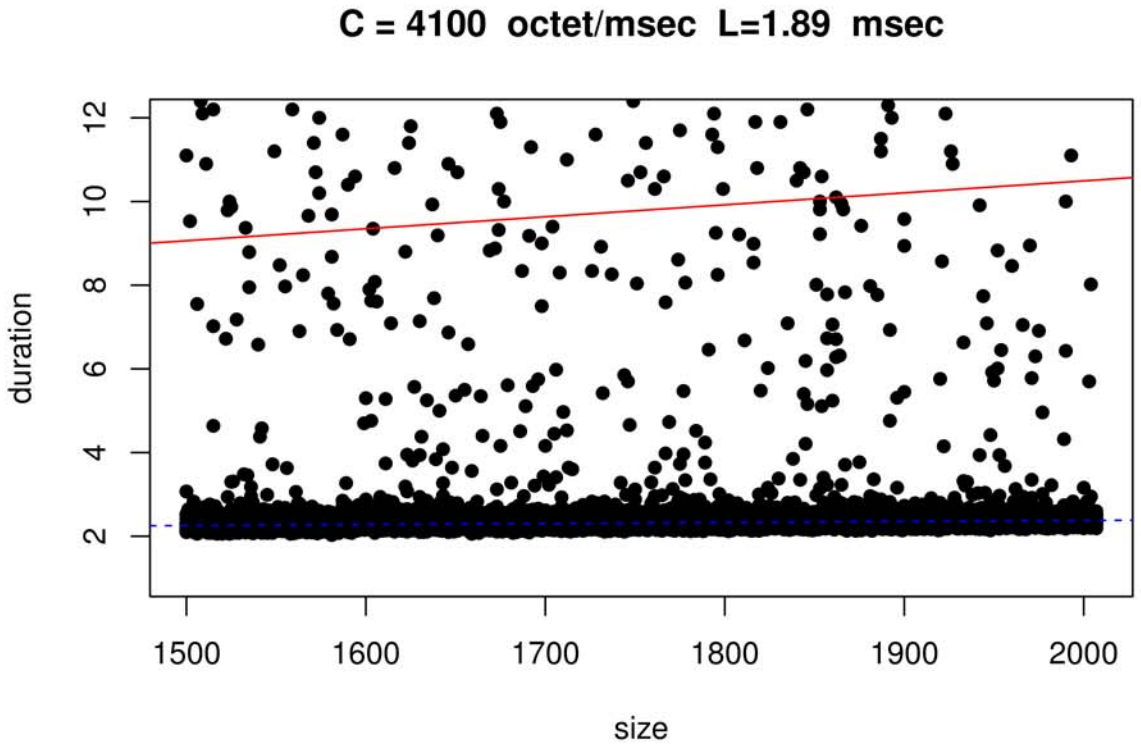



```r
## regression against 50% quantile in the higher part
################
rqfit <- rq(duration ~ size, data = data_high, tau = 0.5)
coeff=coefficients(rqfit)

# Equation de la droite de regression quantile :
eq = paste0("C = ", round(1/coeff[2],1), "  octet/msec  ", "L=", round(coeff[1],2), "  msec")

plot(duration ~ size, data = data_high, pch = 16, main = eq, ylim=c(1,12))
abline(rq(duration ~ size, data = data_high), col = "blue", lty = 2, ylim=c(1,12))

#plotting again the regression on the same graph
reg <- lm(duration ~ size, data = data_high)
abline(reg, col="red", lty = 1, ylim=c(1,12))
```

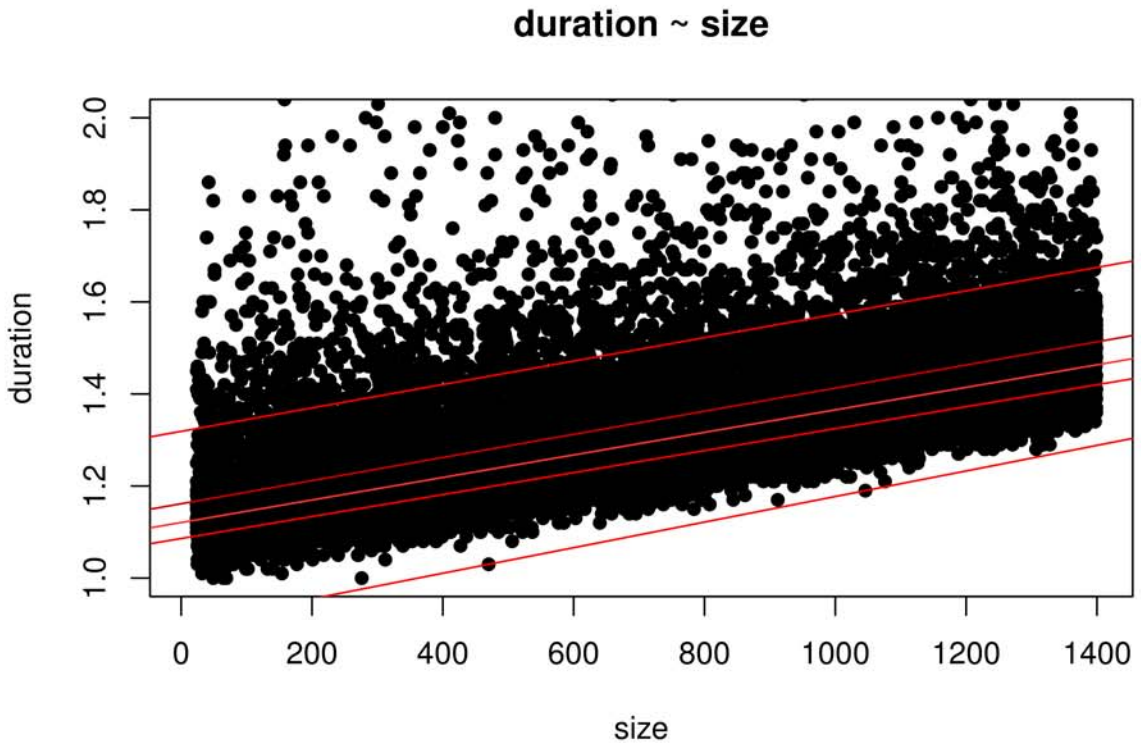


On voit que les vitesses sont très proches dans les deux jeux de données

Par curiosité, on peut observer les régressions sur les différents quantiles.

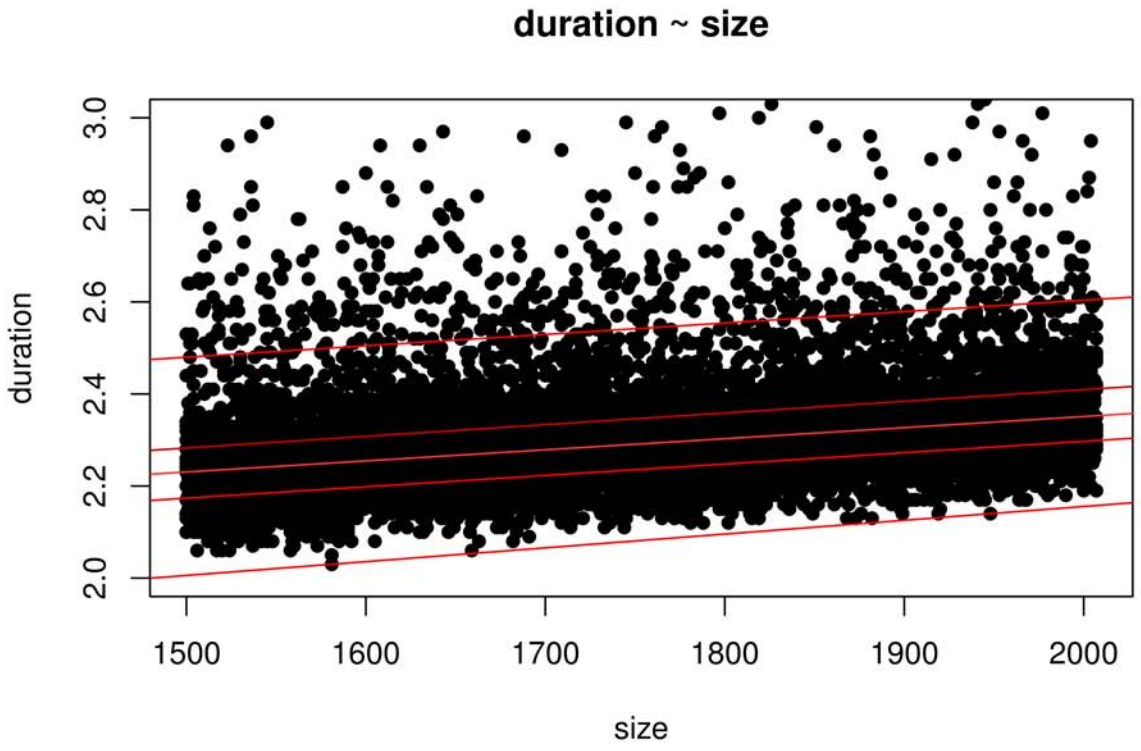
```
multi_rqfit <- rq(duration ~ size, data = data_low, tau = seq(0, 1, by = 0.2))

# plotting different quantiles
colors <- c("red", "red","#ff3333", "#cc0000", "red")
plot(duration ~ size, data = data_low, pch = 16, main = "duration ~ size", ylim=c(1,2))
for (j in 1:ncol(multi_rqfit$coefficients)) {
    abline(coef(multi_rqfit)[, j], col = colors[j])
}
```




```
multi_rqfit <- rq(duration ~ size, data = data_high, tau = seq(0, 1, by = 0.2))

# plotting different quantiles
colors <- c("red", "red","#ff3333", "#cc0000", "red")
plot(duration ~ size, data = data_high, pch = 16, main = "duration ~ size", ylim=c(2,3))
for (j in 1:ncol(multi_rqfit$coefficients)) {
    abline(coef(multi_rqfit)[, j], col = colors[j])
}
```

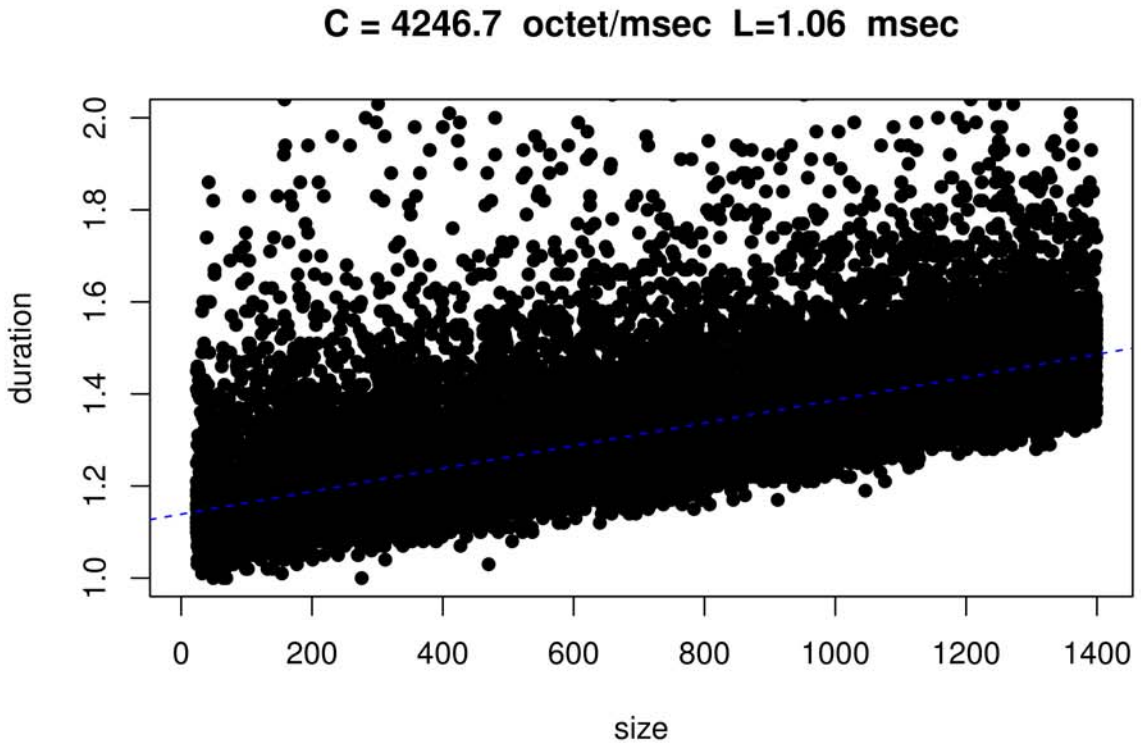


Pour finir, on peut s'intéresser à la régression sur les 10% de messages les plus rapides pour une taille données (10ème centile)

```
library(quantreg)
# regression against 10% quantile in the lower part
################
rqfit <- rq(duration ~ size, data = data_low, tau = 0.1)
coeff=coefficients(rqfit)

# Equation de la droite de regression quantile :
eq = paste0("C = ", round(1/coeff[2],1), "  octet/msec  ", "L=", round(coeff[1],2), "  msec")

plot(duration ~ size, data = data_low, pch = 16, main = eq, ylim=c(1,2))
abline(rq(duration ~ size, data = data_low), col = "blue", lty = 2, ylim=c(1,2))
```

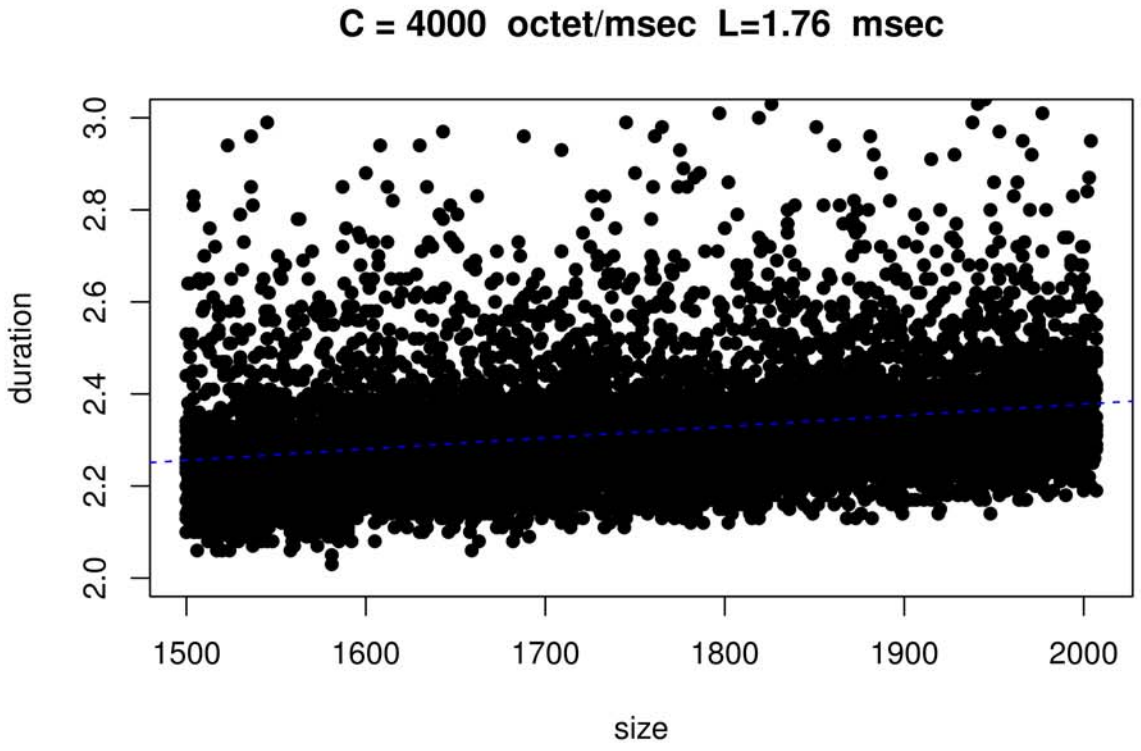


Les résultats sont pour la partie basse de la taille (<1400 octet) C = 4246.7 octet/msec L = 1.06 msec

```
## regression against 10% quantile in the higher part
################
rqfit <- rq(duration ~ size, data = data_high, tau = 0.1)
coeff=coefficients(rqfit)

# Equation de la droite de regression quantile :
eq = paste0("C = ", round(1/coeff[2],1), "  octet/msec  ", "L=", round(coeff[1],2), "  msec")

plot(duration ~ size, data = data_high, pch = 16, main = eq, ylim=c(2,3))
abline(rq(duration ~ size, data = data_high), col = "blue", lty = 2, ylim=c(2,3))
```



Les résultats sont pour la partie haute de la taille (>1500 octet) C = 4000 octet/msec L = 1.76 msec