

Exo4 module 2

Marc

07/04/2020

L'étude

Cette étude consiste à évaluer mon usage du téléphone depuis le 25 février 2020 pour établir un lien entre la mise en place du confinement le 17 mars 2020 et mon usage téléphonique.

Le fichier de données

Le fichier de données ci-après est une table représentant différents paramètres chaque jour depuis le 25/02/2020 :

- Le nombre d'appels émis : Appels_emis
- Le nombre d'appels reçus : Appels_recus (Comprends aussi les appels manqués)
- La durée totale des appels de la journée en seconde : Duree_appels
- Le nombre de messages reçus : Messages_recus
- Le nombre de messages envoyés : Messages_envoyes

Le fichier peut être importé comme ceci :

```
df<-read.csv("C:/Users/Marc/Desktop/M00C/mooc-rr/module2/exo4/Book1.csv", sep = ";")
head(df)
```

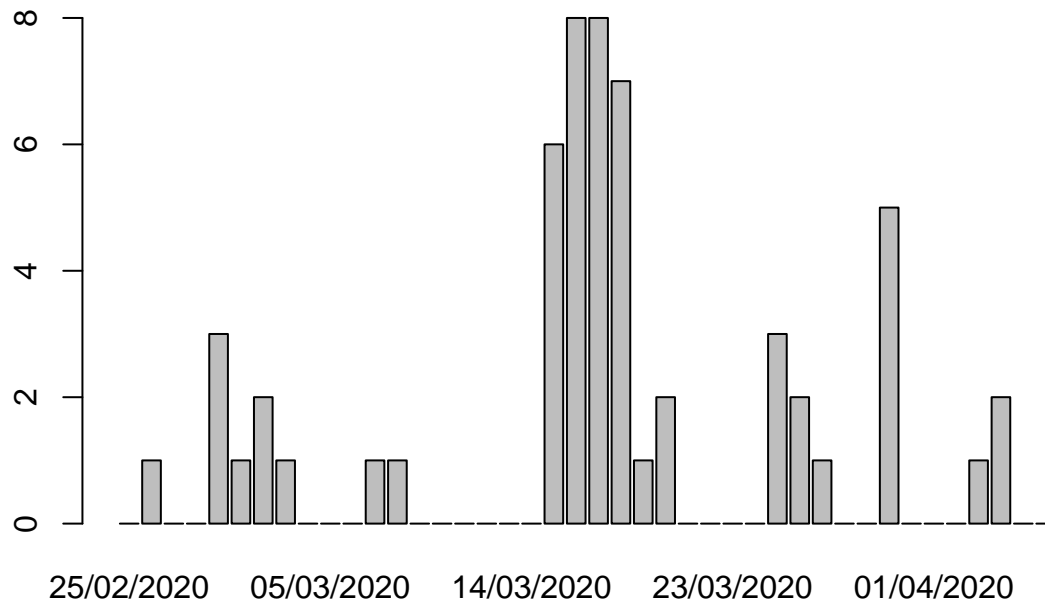
```
##      i..Date Appels_emis Appels_recus Duree_appel Messages_recus
## 1 25/02/2020          0           0          0             6
## 2 26/02/2020          1           1          5             3
## 3 27/02/2020          0           0          0             3
## 4 28/02/2020          0           0          0             0
## 5 29/02/2020          3           1        3470             1
## 6 01/03/2020          1           1        141            15
##      Messages_envoyes
## 1                    4
## 2                    4
## 3                    2
## 4                    0
## 5                    0
## 6                    7
```

L'analyse de l'usage téléphonique

Tout d'abord on peut plotter les différents paramètres au cours du temps pour se donner un aperçu de mon usage :

1. Les appels émis

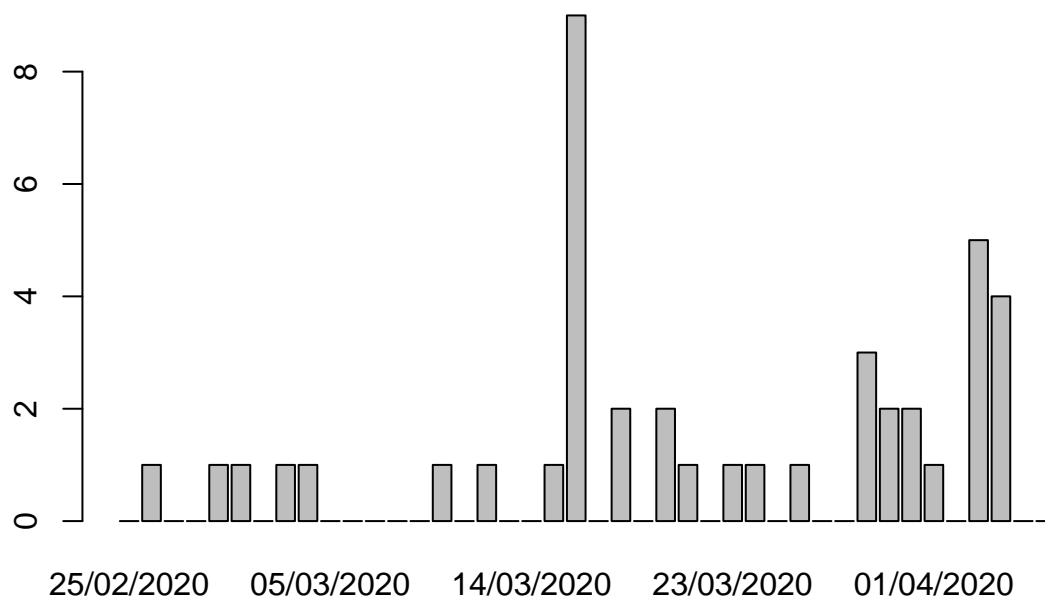
```
barplot(df$Appels_emis, names.arg = df$i..Date)
```



Les dates sont mal positionnées mais c'est pas grave. Il semble que j'ai beaucoup appelé autour de la date du 17 mars.

2. Appels reçus

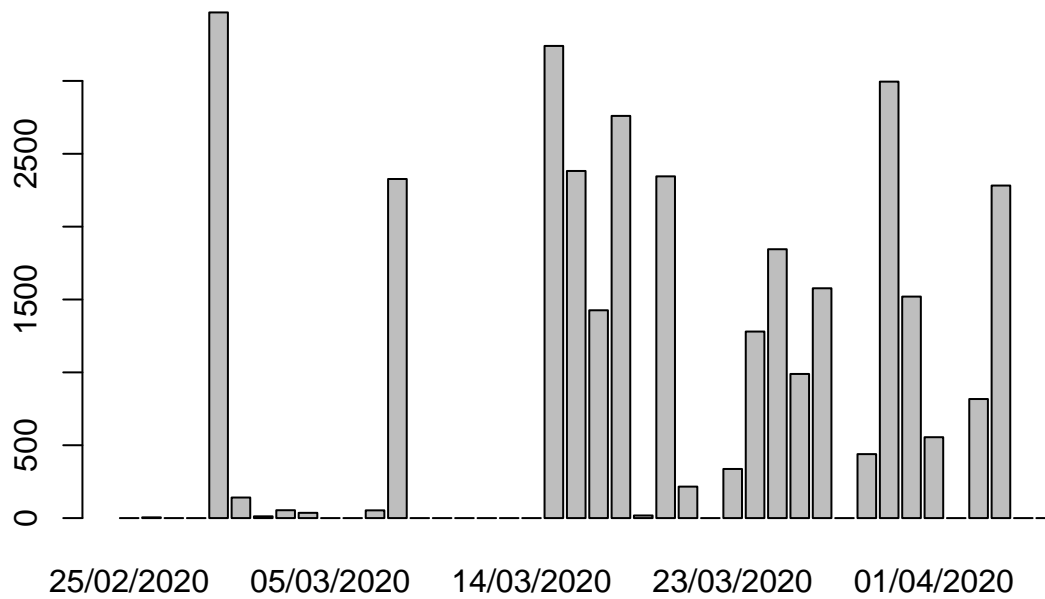
```
barplot(df$Appels_recus, names.arg = df$i..Date)
```



Il semble que j'ai reçu beaucoup d'appels la veille du 17 mars puis que j'ai reçu + d'appels en général après cette période qu'avant.

3. Durée appel

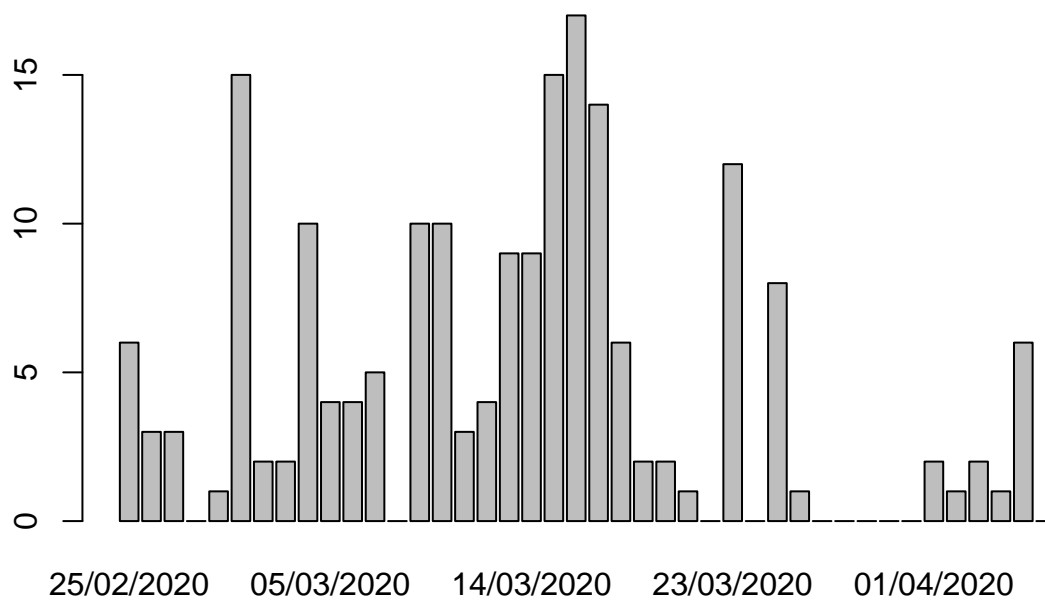
```
barplot(df$Duree_appel, names.arg = df$i..Date)
```



Là c'est très voyant. J'ai passé beaucoup de temps au téléphone après le 16 mars comparé à avant (sauf 2 fois).

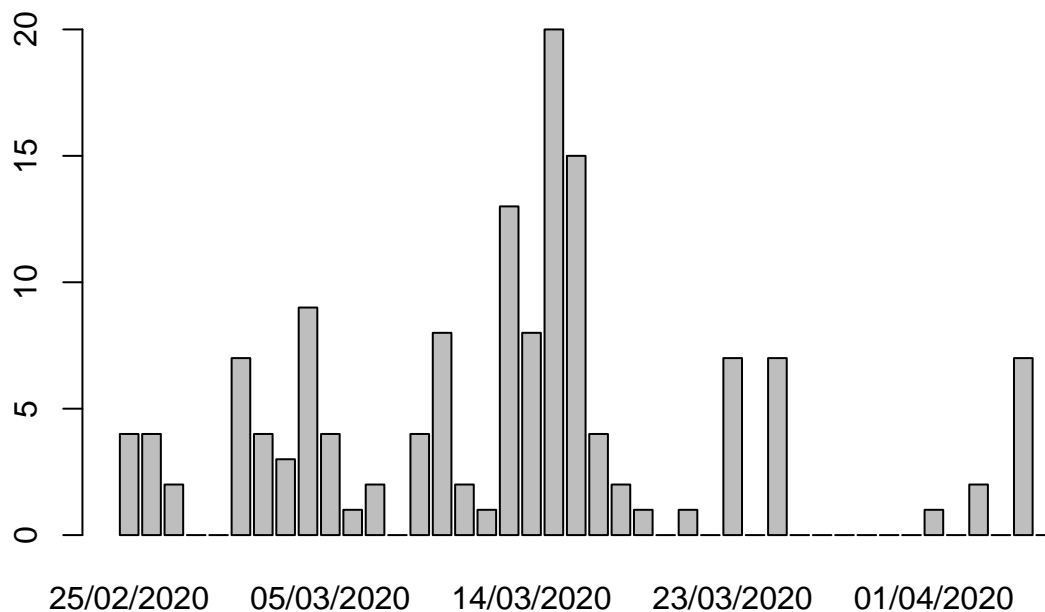
4. Messages reçus

```
barplot(df$Messages_recus, names.arg = df$i..Date)
```



5. Messages envoyés

```
barplot(df$Messages_envoyes, names.arg = df$i..Date)
```



Pour les messages, la tendance est inverse aux appels : je reçois et envoie - de sms depuis le 17 mars comparé à avant.

J'ai eput être changé mon usage de l'un à l'autre.

Sauf autour du 16-17 mars où j'ai beaucoup communiqué.

Représentation des moyennes avant et après le 17 mars

Je vais rajouté une colonne pour indiqué avant ou après le 17 mars.

```
add<-c(rep("avant",21), rep("après",21))
df$add<-add
```

Oui je sais c'est vraiment nul mais en gros je sais qu'il y a 42 lignes dans mon tableau (je peux le vérifier avec `length(df$Appels_emis)` par exemple) et que le 17 mars est la 22ème ligne.

J'ai donc ajouté 21 fois "avant" et 21 fois "après" sur une colonne dans mon data frame df.

Maintenant on va pouvoir calculer les moyennes des paramètres avant et après (inclus) le 17 mars 2020.

```
m_appels_emis<-c(mean(df$Appels_emis[df$add=="avant"]), mean(df$Appels_emis[df$add=="après"]))
m_appels_recus<-c(mean(df$Appels_recus[df$add=="avant"]), mean(df$Appels_recus[df$add=="après"]))
m_duree_appel<-c(mean(df$Duree_appel[df$add=="avant"]), mean(df$Duree_appel[df$add=="après"]))
m_messages_recus<-c(mean(df$Messages_recus[df$add=="avant"]), mean(df$Messages_recus[df$add=="après"]))
m_messages_envoyes<-c(mean(df$Messages_envoyes[df$add=="avant"]), mean(df$Messages_envoyes[df$add=="après"]))
m_appels_emis
```

```
## [1] 1.142857 1.523810
```

```
m_appels_recus
```

```
## [1] 0.8095238 1.1904762
```

```
m_messages_recus
```

```
## [1] 6.285714 2.761905
```

```
m_messages_envoyes
```

```
## [1] 5.285714 1.523810
```

Et les écarts-types :

```
sd_appels_emis<-c(sd(df$Appels_emis[df$add=="avant"]), sd(df$Appels_emis[df$add=="après"]))
sd_appels_recus<-c(sd(df$Appels_recus[df$add=="avant"]), sd(df$Appels_recus[df$add=="après"]))
sd_duree_appel<-c(sd(df$Duree_appel[df$add=="avant"]), sd(df$Duree_appel[df$add=="après"]))
sd_messages_recus<-c(sd(df$Messages_recus[df$add=="avant"]), sd(df$Messages_recus[df$add=="après"]))
sd_messages_envoyes<-c(sd(df$Messages_envoyes[df$add=="avant"]), sd(df$Messages_envoyes[df$add=="après"])
```

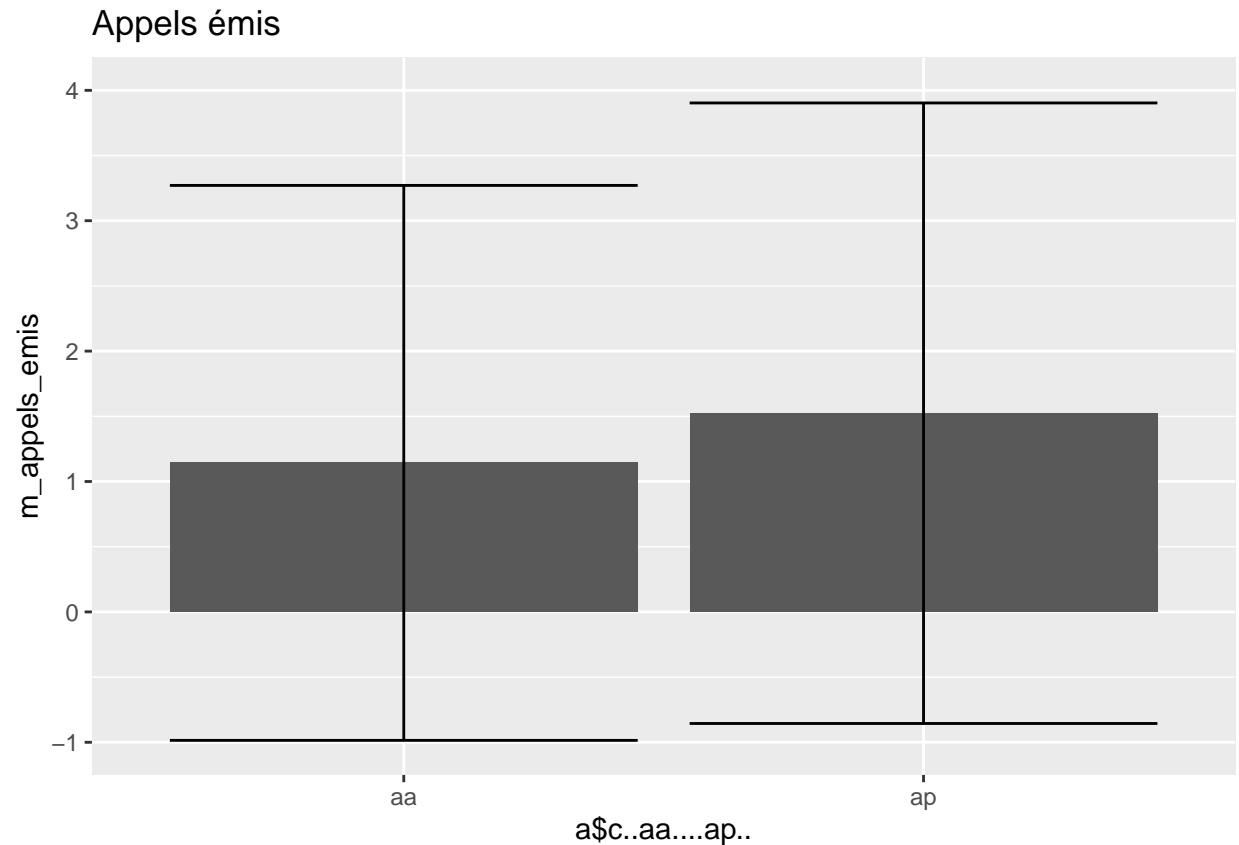
Maintenant on va pouvoir plotter les moyennes de tous les paramètres avant et après (inclus) le 17 mars 2020.

Le mieux c'est d'utiliser ggplot.

```
#install.packages("ggplot2")
library(ggplot2)
a<-data.frame(m_appels_emis, sd_appels_emis, c("aa", "ap"))
ggplot(a, aes(x = a$c..aa....ap.., y = m_appels_emis))+
  geom_bar(stat = "identity")+
  geom_errorbar(ymin = a$m_appels_emis-a$sd_appels_emis, ymax = a$m_appels_emis+a$sd_appels_emis)+
  ylim(-1,4)+
  labs(title="Appels émis")
```

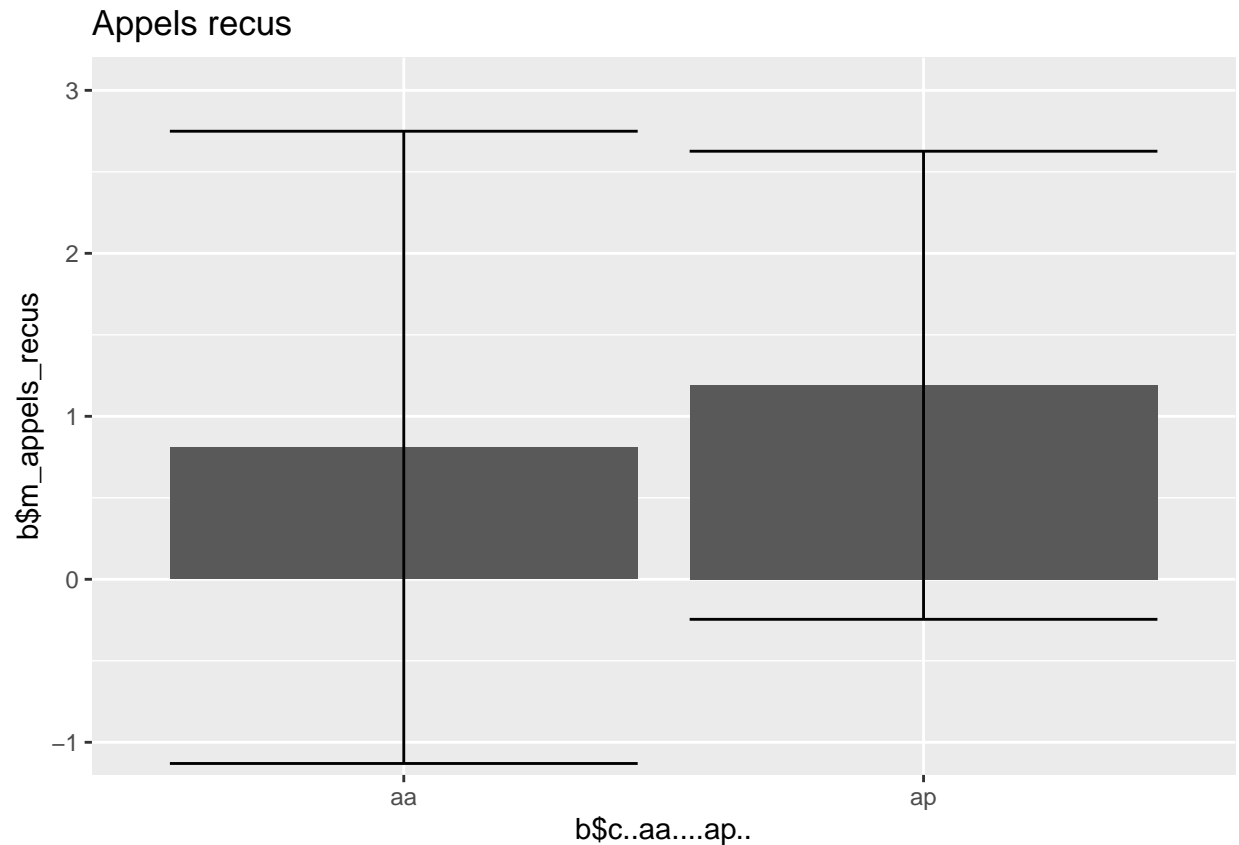
```
## Warning: Use of `a$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
```

```
## Warning: Use of `a$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
```



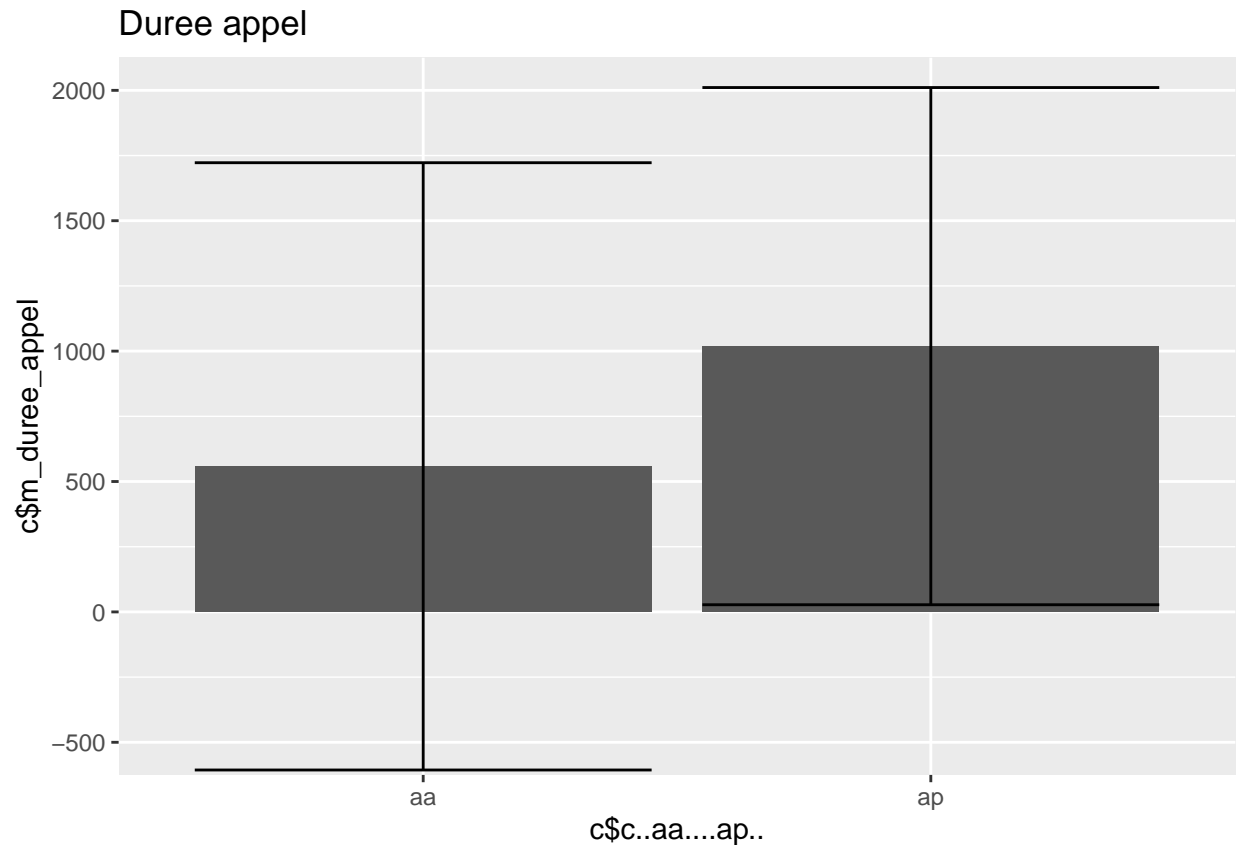
```
b<-data.frame(m_appels_recus, sd_appels_recus, c("aa", "ap"))
ggplot(b, aes(x = b$c..aa....ap.., y = b$m_appels_recus))+
  geom_bar(stat = "identity")+
  geom_errorbar(ymin = b$m_appels_recus-b$sd_appels_recus, ymax = b$m_appels_recus+b$sd_appels_recus)+
  ylim(-1,3)+
  labs(title="Appels recus")
```

```
## Warning: Use of `b$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
## Warning: Use of `b$m_appels_recus` is discouraged. Use `m_appels_recus` instead.
## Warning: Use of `b$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
## Warning: Use of `b$m_appels_recus` is discouraged. Use `m_appels_recus` instead.
```



```
c<-data.frame(m_duree_appel, sd_duree_appel, c("aa", "ap"))
ggplot(c, aes(x = c$c..aa....ap.., y = c$m_duree_appel))+
  geom_bar(stat = "identity")+
  geom_errorbar(ymin = c$m_duree_appel-c$sd_duree_appel, ymax = c$m_duree_appel+c$sd_duree_appel)+
  ylim(-500,2000)+
  labs(title="Duree appel")
```

```
## Warning: Use of `c$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
## Warning: Use of `c$m_duree_appel` is discouraged. Use `m_duree_appel` instead.
## Warning: Use of `c$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
## Warning: Use of `c$m_duree_appel` is discouraged. Use `m_duree_appel` instead.
```



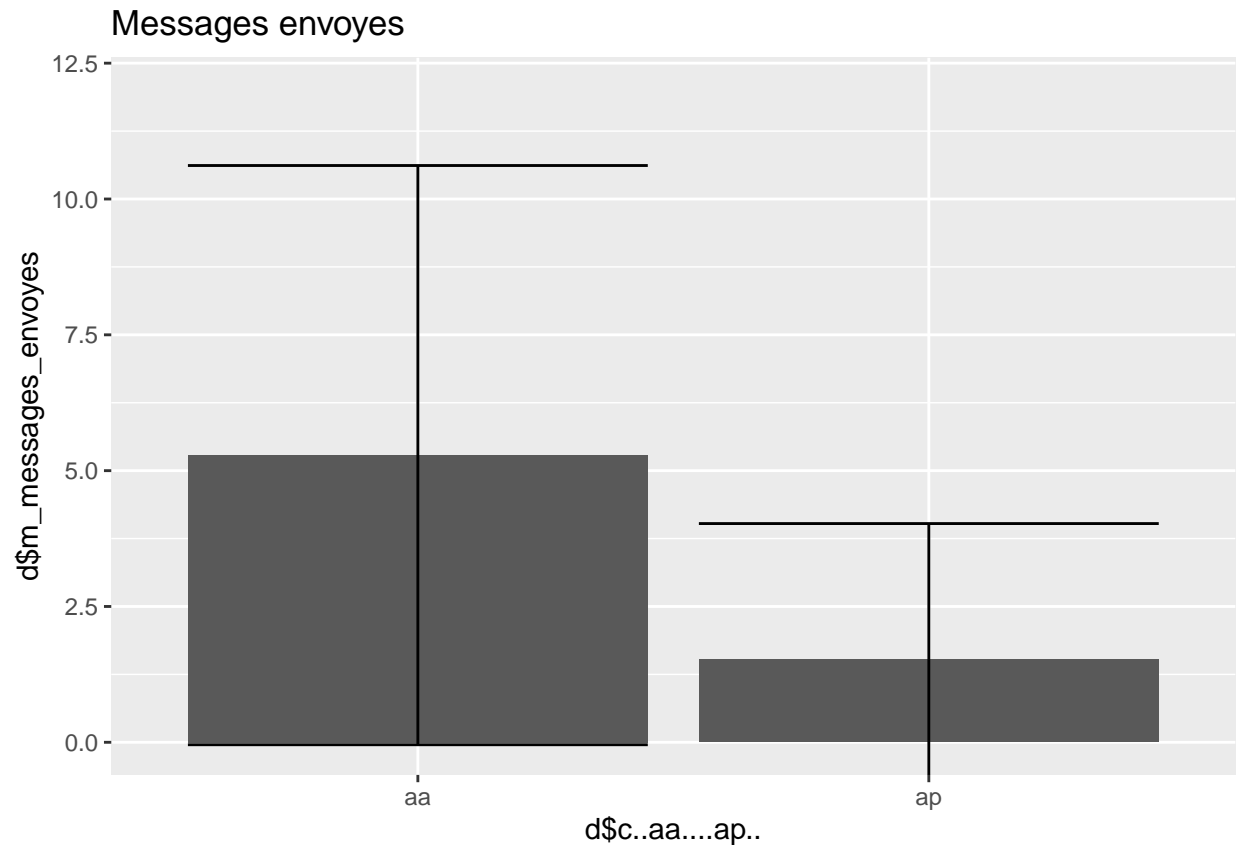
```
d<-data.frame(m_messages_envoyes, sd_messages_envoyes, c("aa", "ap"))
ggplot(d, aes(x = d$c..aa....ap.., y = d$m_messages_envoyes))+
  geom_bar(stat = "identity")+
  geom_errorbar(ymin = d$m_messages_envoyes-d$sd_messages_envoyes, ymax = d$m_messages_envoyes+d$sd_m
  ylim(0,12)+
  labs(title="Messages envoyes")
```

```
## Warning: Use of `d$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
```

```
## Warning: Use of `d$m_messages_envoyes` is discouraged. Use `m_messages_envoyes`
## instead.
```

```
## Warning: Use of `d$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
```

```
## Warning: Use of `d$m_messages_envoyes` is discouraged. Use `m_messages_envoyes`
## instead.
```



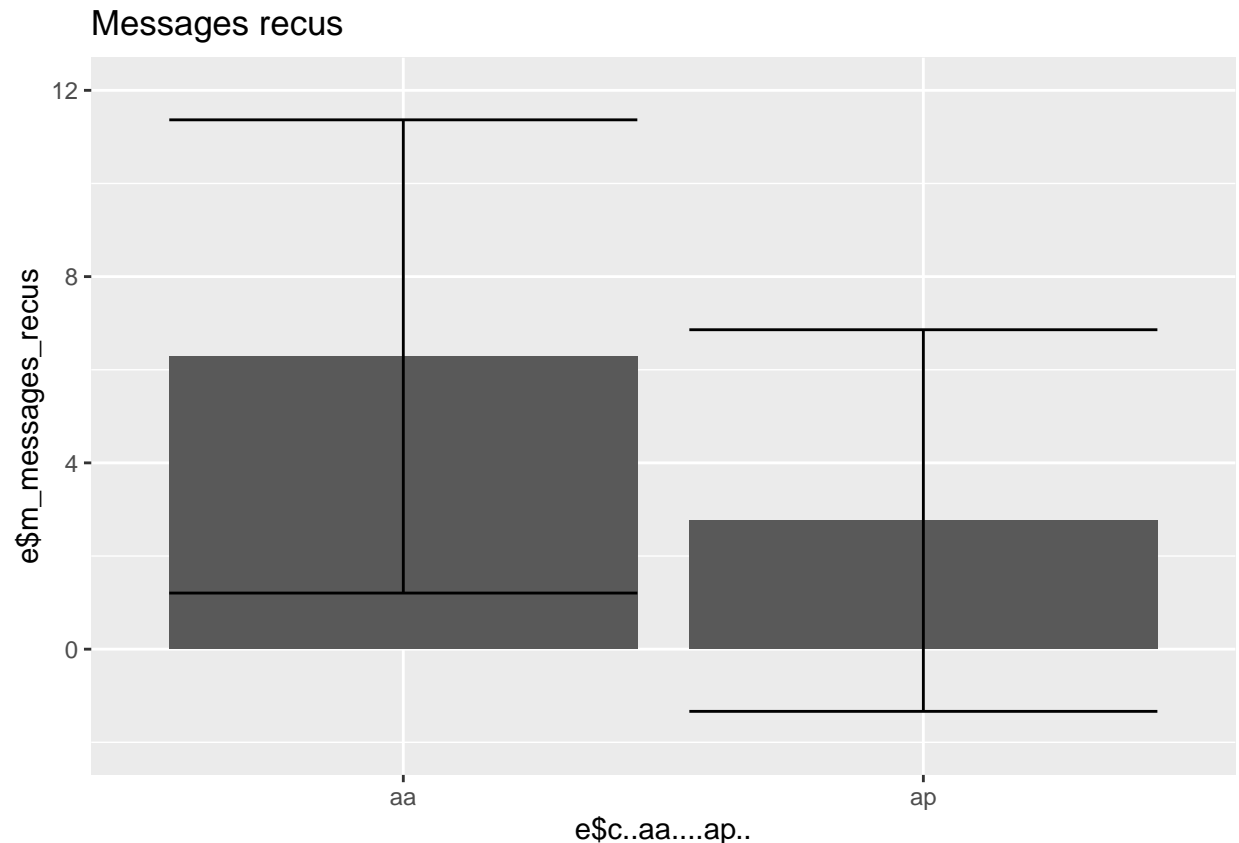
```
e<-data.frame(m_messages_recus, sd_messages_recus, c("aa", "ap"))
ggplot(e, aes(x = e$c..aa....ap.., y = e$m_messages_recus))+
  geom_bar(stat = "identity")+
  geom_errorbar(ymin = e$m_messages_recus-e$sd_messages_recus, ymax = e$m_messages_recus+e$sd_messages_recus)+
  ylim(-2,12)+
  labs(title="Messages recus")
```

```
## Warning: Use of `e$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
```

```
## Warning: Use of `e$m_messages_recus` is discouraged. Use `m_messages_recus` instead.
```

```
## Warning: Use of `e$c..aa....ap..` is discouraged. Use `c..aa....ap..` instead.
```

```
## Warning: Use of `e$m_messages_recus` is discouraged. Use `m_messages_recus` instead.
```



Bilan : On voit des augmentations dans les appels et une diminution dans les messages mais les écarts types sont énormes

A mon avis, rien n'est significatif mais on peut s'entraîner sur un cas.

Comme il n'y a que 21 valeurs dans chaque groupe, je ne peux pas appliquer le théorème central limite. Je vais donc vérifier la distribution normale de chaque groupe ainsi que l'égalité des variances pour voir quel test statistique appliqué.

Prenons comme exemple la durée de l'appel.

```
shapiro.test(df$Duree_appel[df$add=="avant"])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df$Duree_appel[df$add == "avant"]
## W = 0.53087, p-value = 3.968e-07
```

```
shapiro.test(df$Duree_appel[df$add=="après"])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df$Duree_appel[df$add == "après"]
## W = 0.88753, p-value = 0.0202
```

Les tests de Shapiro-Wilk sont significatifs donc les distributions ne sont pas normales.

Utilisation de tests non paramétriques type Mann-Whitney :

```
wilcox.test(df$Duree_appel[df$add=="avant"],df$Duree_appel[df$add=="après"])

## Warning in wilcox.test.default(df$Duree_appel[df$add == "avant"],
## df$Duree_appel[df$add == : cannot compute exact p-value with ties
##
## Wilcoxon rank sum test with continuity correction
##
## data: df$Duree_appel[df$add == "avant"] and df$Duree_appel[df$add == "après"]
## W = 140.5, p-value = 0.03965
## alternative hypothesis: true location shift is not equal to 0
```

Ah ben en fait la durée des appels a significativement augmentée après le 17 mars 2020. Je ne suis pas super fort en statistiques donc j'espère que c'est correct. Aussi, 21 échantillons par groupe c'est pas mal pour un test non paramétrique.

Du coup on va essayer de mettre une étoile sur le plot (ce qui n'est pas compris dans les fonctions ggplot). Pour cela on va donc tester le code d'un ami disponible sur Github.

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

## Loading required package: usethis
#Besoin de Rtools 3.5
#install_github("EvenStar69/significativity.bar/significativity.bar")
library(significativity.bar)
```

Bon j'ai dû un peu changer sa fonction parce qu'elle n'est plus compatible avec ggplot 3.3.

J'ai changé la manière de retrouver les coordonnées y : The position of ymax in ggplot_build(plot)\$data[[1]] changed from column 6 to 7.

Et j'ai changé la manière de retrouver l'échelle en y : panel_ranges in ggplot_build(plot)\$layout does not exist anymore ... use panel_scale_y instead.

J'ai aussi dû mettre à jour R vers la version 3.6.3.

Voici le code modifié de sa fonction :

```
significativity_bar <- function(plot, groups, text = "*", text_height = 0.0275, size_bar = 1, color_bar

if (!require("ggplot2", character.only=T, quietly=T)){ # use library ggplot

  install.packages("ggplot2")

  library(ggplot2, character.only=T)
}

if (class(plot)[1] != "gg"){

  stop("Your input plot is not a ggplot")
}

if (length(groups) != 2){
```

```

    stop("Please select only 2 groups between which you want the error bar")
}

if (!is.vector(groups)){
    stop("Please input your 2 selected groups in a vector")
}

if (!is.character(text)) {
    stop("Please input the text above the bar as character")
}

if (!is.numeric(text_height) | length(text_height) > 1){
    stop("Please input one numeric value for the text height")
}

if (!is.numeric(size_bar) | length(size_bar) > 1){
    stop("Please input one numeric value for the bar size")
}

if (!is.character(color_bar)){
    stop("Please input an existing R color, as a character, for the color of the bar")
}

if (!is.numeric(size_text) | length(size_text) > 1){
    stop("Please input one numeric value for the text size")
}

if (!is.numeric(font_face) | length(font_face) > 1){
    stop("Please input one numeric value for the font face")
}

if (!is.character(color_text)){
    stop("Please input an existing R color, as a character, for the color of the text")
}

if (!is.character(font_style)){

```

```

    stop("Please input an existing font family, as a character, for the color of the bar")
}

if (!is.character(line_type)){
    stop("Please input an existing line style, as a character, for the color of the bar")
}

if (text_height >=1){
    warning("text_height should be between 0 and 1, default value for * and around 0.04 for text are ad
}

if (class(as.list.environment(plot$layers[[1]])$geom)[1] == "GeomPoint"){ # if the ggplot is a dotplo
    coords = ggplot_build(plot)$data[[1]] # get the coordinates of the points
    xcoords = c()
    ycoords = c()

    for (i in groups){ # get the x coordinates of all coordinates in a vector, for the 2 selected group
        xcoord_temp = unique(coords$x)[i]
        xcoords = append(xcoords, xcoord_temp)
    }

    for (i in c(1,2)){
        ycoord_temp = max(coords[coords$x == xcoords[i],]$y) # get the y coordinate of the upper point of
        ycoords = append(ycoords, ycoord_temp)
    }

    y_range = ggplot_build(plot)$layout$panel_scales_y[[1]]$limits # get the total height of the y scal
    # panel_ranges in ggplot_build(plot)$layout does not exist anymore ... use panel_scale_y instead

    y_sum = sum(abs(y_range))

```

```

y_scale = (7.5/100)*y_sum # starting position of the vertical bar (determined % of the total y scale)
bar_height = y_scale + ((5/100)*y_sum) # final position of the vertical bar (determined % of the total y scale)

ycoord_top = max(ycoords) # the bar should take the highest of the two groups as a reference
coord_bar = data.frame(x = c(xcoords[1], xcoords[1], xcoords[2], xcoords[2]), y = c(ycoord_top + y_scale, ycoord_top + y_scale, ycoord_top + y_scale, ycoord_top + y_scale))

star_x = mean(xcoords) # x coordinate of the text above the bar (in the middle of the two groups)
star_y = ycoord_top + bar_height + ((2.75/100)*y_sum) # y coordinate of the text above the bar (above the bar)
coord_star = c(star_x, star_y) # x,y coordinates of the text above the bar

plot = plot + geom_path(data = coord_bar, aes(x=x, y=y), size = size_bar, color = color_bar, linetype = "solid")
print(plot)

} else if (class(as.list.environment(plot$layers[[1]])$geom)[1] == "GeomBar") { # if the ggplot is a bar plot
  coords = ggplot_build(plot)$data[[1]]
  xcoords = c()
  ycoords = c()
  for (i in groups){ # get the x and y coordinates of the two groups
    xcoord_temp = mean(c(coords[i,]$xmin, coords[i,]$xmax))
    xcoords = append(xcoords, xcoord_temp)
    ycoord_temp = coords[i,7] # The position of ymax in ggplot_build(plot)$data[[1]] changed from col to row
    ycoords = append(ycoords, ycoord_temp)
  }

  y_range = ggplot_build(plot)$layout$panel_scales_y[[1]]$limits # get the total height of the y scale
  y_sum = sum(abs(y_range))
  y_scale = (7.5/100)*y_sum # starting position of the vertical bar (determined % of the total y scale)

```

```

bar_height = y_scale + ((5/100)*y_sum) # final position of the vertical bar (determined % of the to

ycoord_top = max(ycoords) # the bar should take the heighest of the two groups as a reference

coord_bar = data.frame(x = c(xcoords[1], xcoords[1], xcoords[2], xcoords[2]), y = c(ycoord_top + y_s

star_x = mean(xcoords) # x coordinate of the text above the bar (in the middle of the two groups)

star_y = ycoord_top + bar_height + (text_height*y_sum) # y coordinate of the text above the bar (ab

coord_star = c(star_x, star_y) # x,y coordinates of the text above the bar

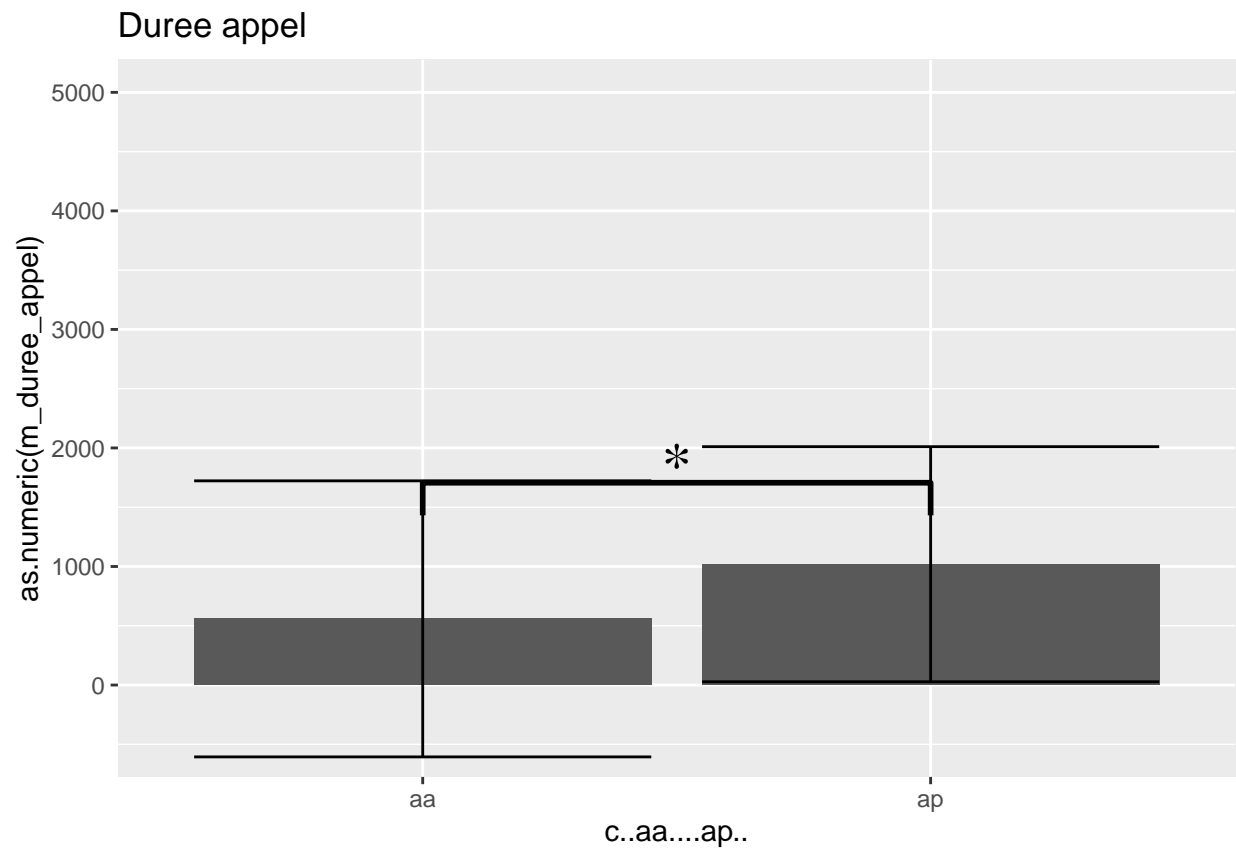
plot = plot + geom_path(data = coord_bar, aes(x=x, y=y), size = size_bar, color = color_bar, linety

print(plot)
}

}

gg<- ggplot(c, aes(x = c..aa....ap.., y = as.numeric(m_duree_appel)))+
  geom_bar(stat = "identity")+
  geom_errorbar(ymin = m_duree_appel-sd_duree_appel, ymax = m_duree_appel+sd_duree_appel)+
  ylim(-500,5000)+
  labs(title="Duree appel")
significativity_bar(gg, groups = c(1,2))

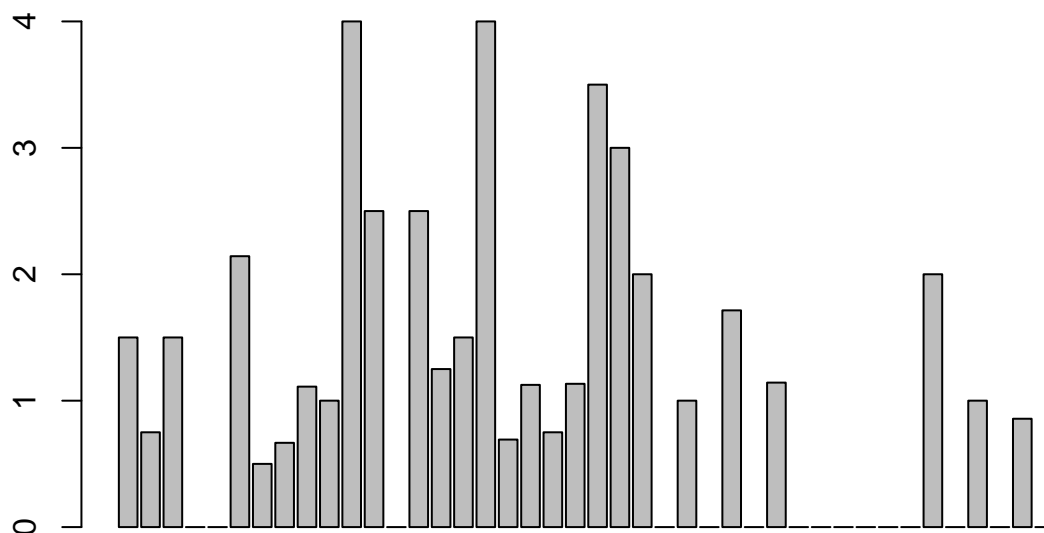
```



Taux de réponses

Cette partie c'est juste pour voir si je répond autant aux messages qu'on m'en envoie.

```
Ratio<-df$Messages_recus/df$Messages_envoyes
#Remplacement des NaN et inf (division par 0) en 0.
Ratio[is.na(Ratio)]<-0
Ratio[is.infinite(Ratio)]<-0
barplot(Ratio)
```



```
mean(Ratio)
```

```
## [1] 1.067513
```

En moyenne c'est assez équilibré : je réponds autant de fois qu'on m'envoie un message.

Conclusion

J'ai passé pas mal de temps à faire ça mais ça m'a permis de bien prendre en main l'outil. Je conçois que mon étude est assez sale et que les manières de plotter ne sont vraiment pas optimisées mais ce n'était pas vraiment le but de l'exercice.

Après avoir compilé, je n'ai pas pu le faire en pdf (des erreurs de polices qui ne passent pas avec LaTeX on dirait ...).

Aussi, j'ai remarqué qu'il fallait laissé un retour chariot après la visualisation du plot sinon le texte se met à côté dans le rendu.