Sujet 1 : Concentration de CO2 dans l'atmosphère depuis 1958

E. Guiffart

1 Juin 2020

Préparation des données

Les données de *la concentration du CO2 dans l'atmosphère* sont disponibles sur le site web du Scripps CO2 Program, un site très intéressant comme on devrait en voir plus souvent de nos jours, sauf qu'il est américain.

Nous les récupérons sous forme d'un fichier en format CSV dont chaque ligne correspond à une semaine de la période demandée. Nous téléchargeons toujours le jeu de données complet, qui commence en 1991 et se termine avec une semaine récente.

En pratique, nous chargons les données grâce à une copie du fichier enregistrée localement. Un code télécharge directement et enregistre une copie au cas où ce fichier n'exste pas encore en local.

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

```
## Warning: package 'data.table' was built under R version 3.6.3
```

```
library(timeSeries)
```

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

```
## Loading required package: timeDate
```

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

```
library(parsedate)
```

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

```
if (!file.exists("monthly_in_situ_co2_mlo.csv")) download.file("https://scrippsco2.ucsd.edu/assets/data/
```

```
data_url <- fread(file = "monthly_in_situ_co2_mlo.csv",skip=54, sep=",",strip.white=TRUE, header = FALSE
```

Allez on visualise les données:

```
head(data_url)
```

```
##      V1 V2    V3        V4     V5         V6     V7           V8     V9
## 1:   Yr Mn  Date      Date    CO2 seasonally    fit   seasonally    CO2
## 2:                                adjusted        adjusted fit filled
## 3:         Excel            [ppm]      [ppm]  [ppm]        [ppm]  [ppm]
## 4: 1958 01 21200 1958.0411 -99.99     -99.99 -99.99       -99.99 -99.99
## 5: 1958 02 21231 1958.1260 -99.99     -99.99 -99.99       -99.99 -99.99
## 6: 1958 03 21259 1958.2027 315.70     314.44 316.18       314.90 315.70
##               V10
## 1:      seasonally
## 2: adjusted filled
## 3:           [ppm]
## 4:          -99.99
## 5:          -99.99
## 6:          314.44
```

```
tail(data_url)
```

```
##      V1 V2    V3        V4     V5     V6     V7     V8     V9    V10
## 1: 2020 07 44027 2020.5383 -99.99 -99.99 -99.99 -99.99 -99.99 -99.99
## 2: 2020 08 44058 2020.6230 -99.99 -99.99 -99.99 -99.99 -99.99 -99.99
## 3: 2020 09 44089 2020.7077 -99.99 -99.99 -99.99 -99.99 -99.99 -99.99
## 4: 2020 10 44119 2020.7896 -99.99 -99.99 -99.99 -99.99 -99.99 -99.99
## 5: 2020 11 44150 2020.8743 -99.99 -99.99 -99.99 -99.99 -99.99 -99.99
## 6: 2020 12 44180 2020.9563 -99.99 -99.99 -99.99 -99.99 -99.99 -99.99
```

Nettoyage des données

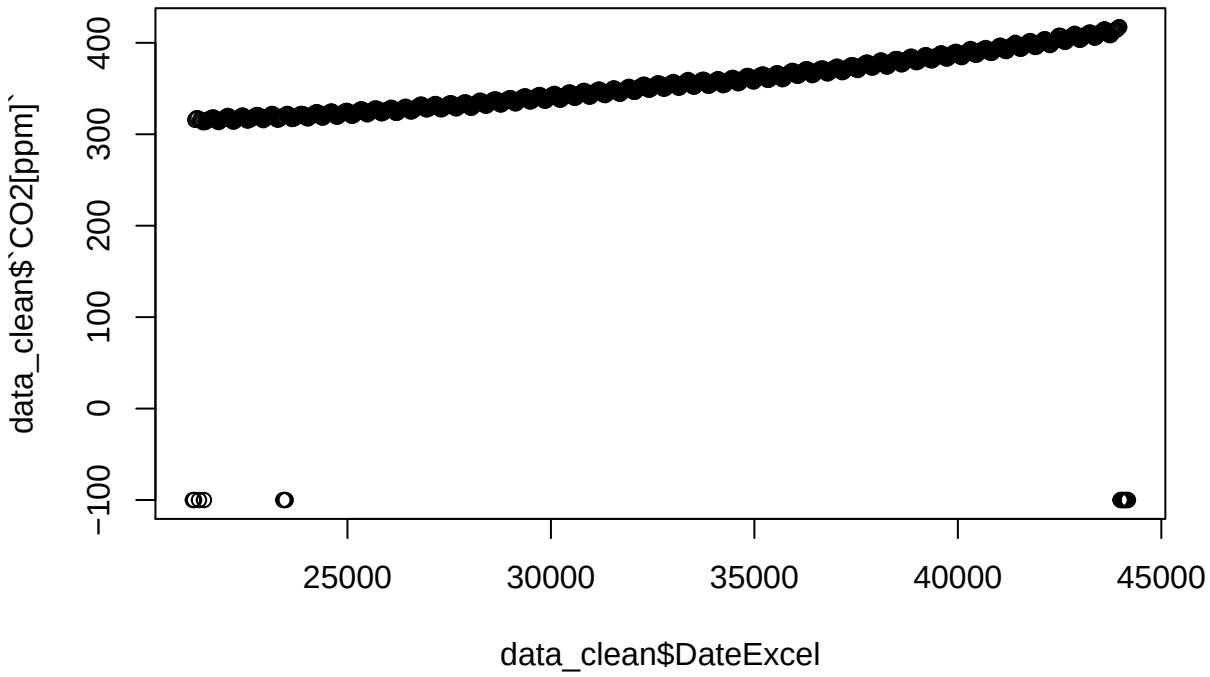
Va falloir nettoyer tout ça, notamment le nom des colonnes:

```
new_names<-paste0(data_url[1,],data_url[2,],data_url[3,])
setnames(data_url,names(data_url),new_names)
data_clean<-data_url[-c(1,2,3),]
head(data_clean)
```

```
##      Yr Mn DateExcel      Date CO2[ppm] seasonallyadjusted[ppm] fit[ppm]
## 1: 1958 01     21200 1958.0411   -99.99                 -99.99   -99.99
## 2: 1958 02     21231 1958.1260   -99.99                 -99.99   -99.99
## 3: 1958 03     21259 1958.2027   315.70                 314.44   316.18
## 4: 1958 04     21290 1958.2877   317.45                 315.16   317.29
## 5: 1958 05     21320 1958.3699   317.51                 314.71   317.86
## 6: 1958 06     21351 1958.4548   -99.99                 -99.99   317.24
##    seasonallyadjusted fit[ppm] CO2filled[ppm] seasonallyadjusted filled[ppm]
## 1:                     -99.99         -99.99                         -99.99
## 2:                     -99.99         -99.99                         -99.99
## 3:                     314.90         315.70                         314.44
## 4:                     314.98         317.45                         315.16
## 5:                     315.06         317.51                         314.71
## 6:                     315.14         317.24                         315.14
```

Bon ok c'est un peu mieux, moins confusant c'est déjà ça. Les 2 premières lignes sont suspectes. Voyons s'il ya en a d'autres. Pour ça, faisons un petit graph :

```
plot(data_clean$DateExcel,data_clean$`CO2[ppm]`)
```



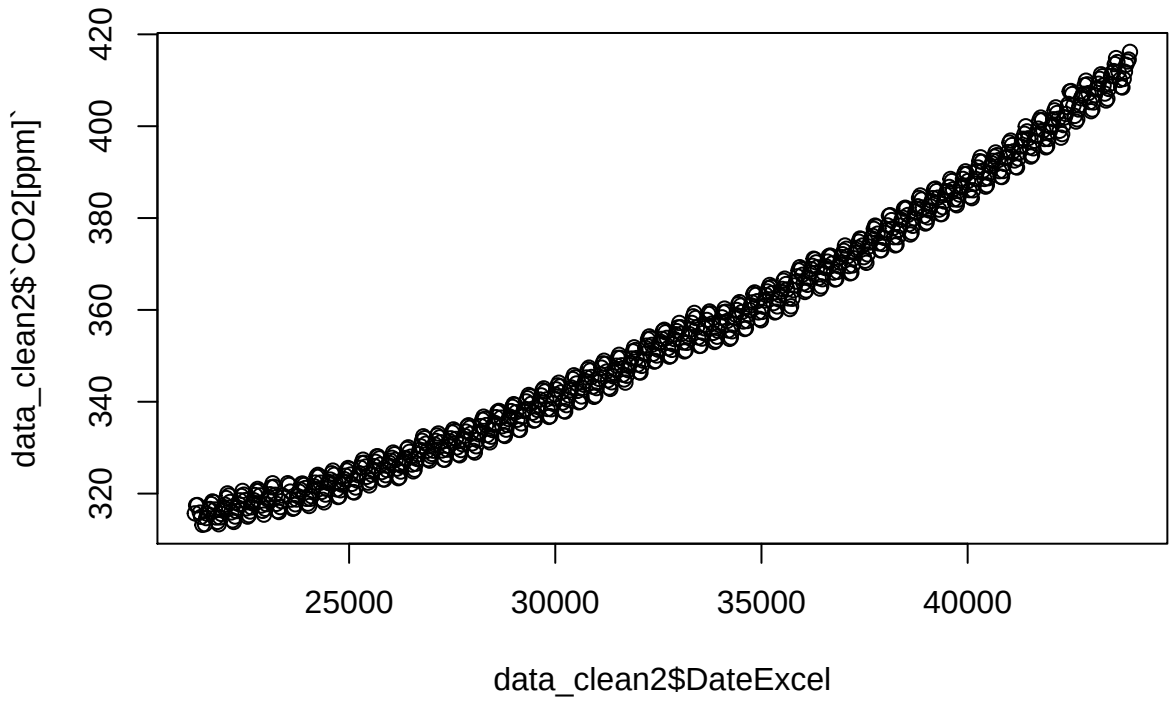
Bon on va les détecter et les modifier en les supprimant.

```
na_records = apply(data_clean, 1, function (x) any(is.na(x)+(x<0)))
data_clean[na_records,]
```

```
##       Yr Mn DateExcel      Date CO2[ppm] seasonallyadjusted[ppm] fit[ppm]
##  1: 1958 01     21200 1958.0411   -99.99                  -99.99   -99.99
##  2: 1958 02     21231 1958.1260   -99.99                  -99.99   -99.99
##  3: 1958 06     21351 1958.4548   -99.99                  -99.99   317.24
##  4: 1958 10     21473 1958.7890   -99.99                  -99.99   312.43
##  5: 1964 02     23422 1964.1257   -99.99                  -99.99   320.00
##  6: 1964 03     23451 1964.2049   -99.99                  -99.99   320.73
##  7: 1964 04     23482 1964.2896   -99.99                  -99.99   321.83
##  8: 2020 05     43966 2020.3716   417.16                  413.75   -99.99
##  9: 2020 06     43997 2020.4563   -99.99                  -99.99   -99.99
## 10: 2020 07     44027 2020.5383   -99.99                  -99.99   -99.99
## 11: 2020 08     44058 2020.6230   -99.99                  -99.99   -99.99
## 12: 2020 09     44089 2020.7077   -99.99                  -99.99   -99.99
## 13: 2020 10     44119 2020.7896   -99.99                  -99.99   -99.99
## 14: 2020 11     44150 2020.8743   -99.99                  -99.99   -99.99
## 15: 2020 12     44180 2020.9563   -99.99                  -99.99   -99.99
##     seasonallyadjusted fit[ppm] CO2filled[ppm] seasonallyadjusted filled[ppm]
##  1:                     -99.99         -99.99                       -99.99
```

```
##  2:                    -99.99         -99.99                    -99.99
##  3:                    315.14         317.24                    315.14
##  4:                    315.40         312.43                    315.40
##  5:                    319.36         320.00                    319.36
##  6:                    319.40         320.73                    319.40
##  7:                    319.45         321.83                    319.45
##  8:                    -99.99         417.16                    413.75
##  9:                    -99.99         -99.99                    -99.99
## 10:                    -99.99         -99.99                    -99.99
## 11:                    -99.99         -99.99                    -99.99
## 12:                    -99.99         -99.99                    -99.99
## 13:                    -99.99         -99.99                    -99.99
## 14:                    -99.99         -99.99                    -99.99
## 15:                    -99.99         -99.99                    -99.99
```

```
data_clean2<-data_clean[!na_records,]
plot(data_clean2$DateExcel,data_clean2$`CO2[ppm]`)
```



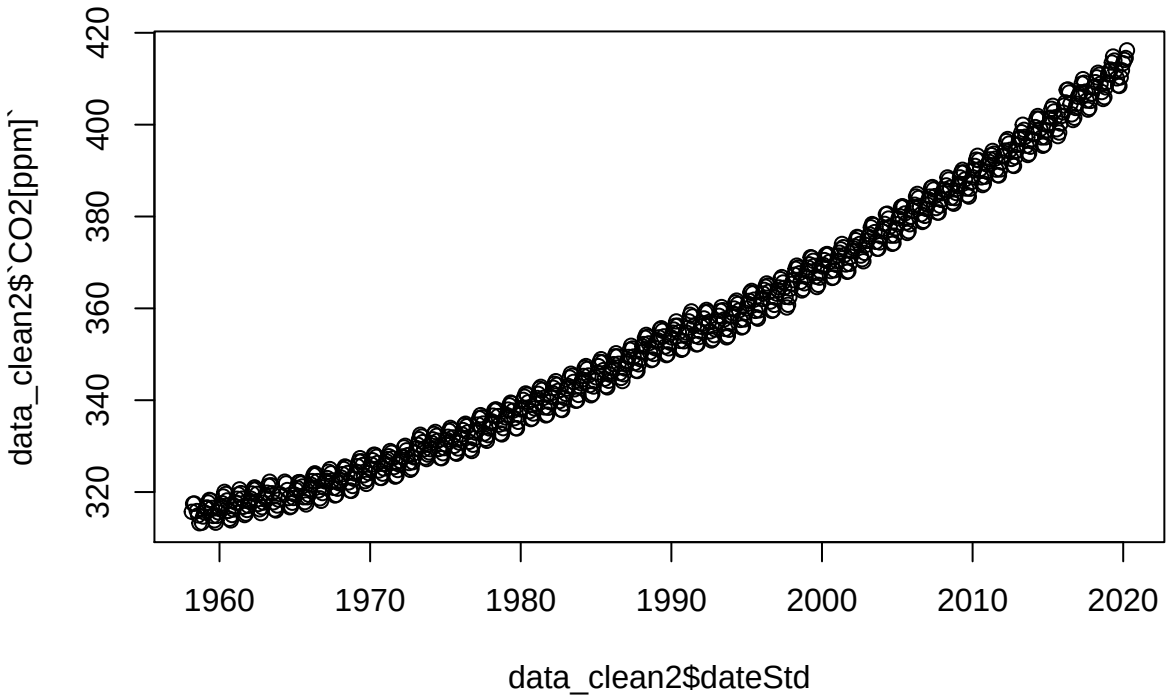
ok on est sur la bonne voie, on va s'en sortir. Reste les dates à mettre en meilleur format:

```
convert_iso_date = function(y,m) {
    iso = paste0(y, "-", m, "-","01")
    as.character(parse_iso_8601(iso))
}
```

```r
data_clean2$dateStd<-as.Date(convert_iso_date(data_clean2$Yr,data_clean2$Mn))
head(data_clean2)
```

```
##      Yr Mn DateExcel      Date CO2[ppm] seasonallyadjusted[ppm] fit[ppm]
## 1: 1958 03     21259 1958.2027   315.70                  314.44   316.18
## 2: 1958 04     21290 1958.2877   317.45                  315.16   317.29
## 3: 1958 05     21320 1958.3699   317.51                  314.71   317.86
## 4: 1958 07     21381 1958.5370   315.86                  315.19   315.86
## 5: 1958 08     21412 1958.6219   314.93                  316.19   313.99
## 6: 1958 09     21443 1958.7068   313.21                  316.08   312.45
##    seasonallyadjusted fit[ppm] CO2filled[ppm] seasonallyadjusted filled[ppm]
## 1:                     314.90         315.70                         314.44
## 2:                     314.98         317.45                         315.16
## 3:                     315.06         317.51                         314.71
## 4:                     315.21         315.86                         315.19
## 5:                     315.28         314.93                         316.19
## 6:                     315.35         313.21                         316.08
##       dateStd
## 1: 1958-03-01
## 2: 1958-04-01
## 3: 1958-05-01
## 4: 1958-07-01
## 5: 1958-08-01
## 6: 1958-09-01
```

```r
plot(data_clean2$dateStd,data_clean2$`CO2[ppm]`)
```

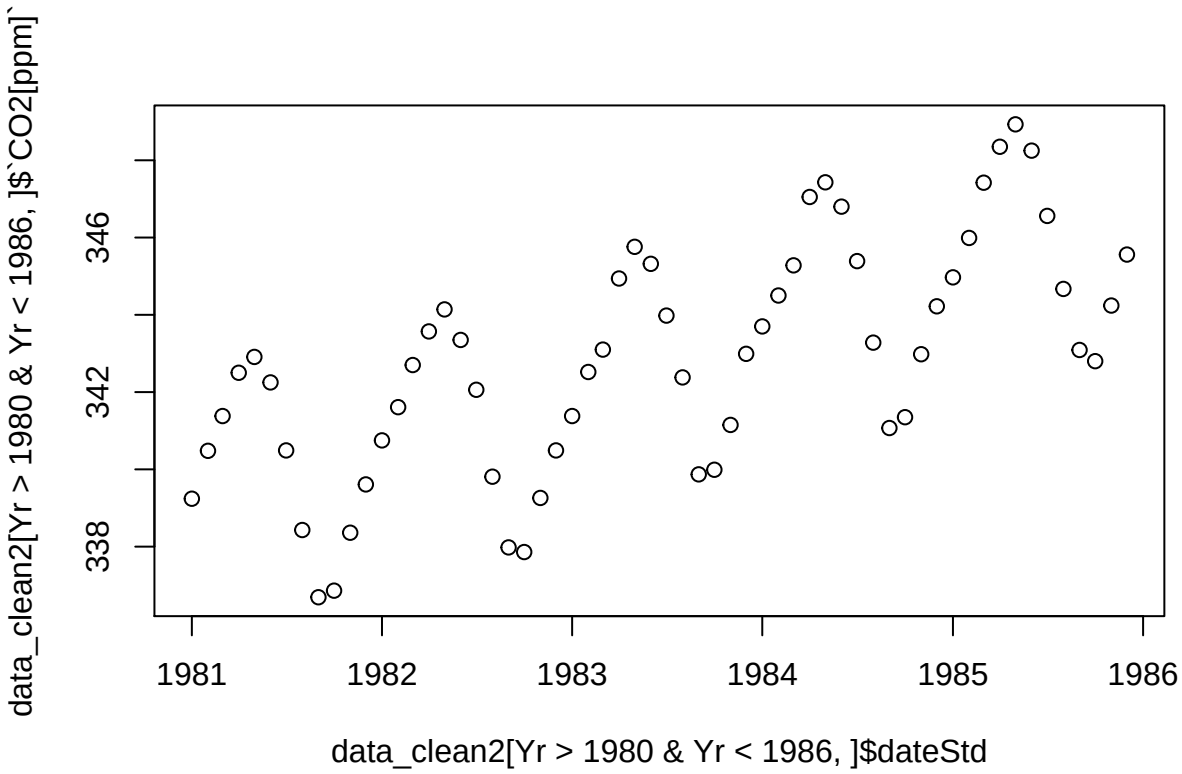


On va pouvoir passer aux stats maintenant. Cette étude est vraiment pleine de suspens...

Visualisation des oscillations périodiques et tendance

Regardons de plus près l'évolution de la série en focalisant sur des années, par exemple de 1980 à 1985:

```
plot(data_clean2[Yr>1980 & Yr<1986,]$dateStd,data_clean2[Yr>1980 & Yr<1986,]$`CO2[ppm]`)
```

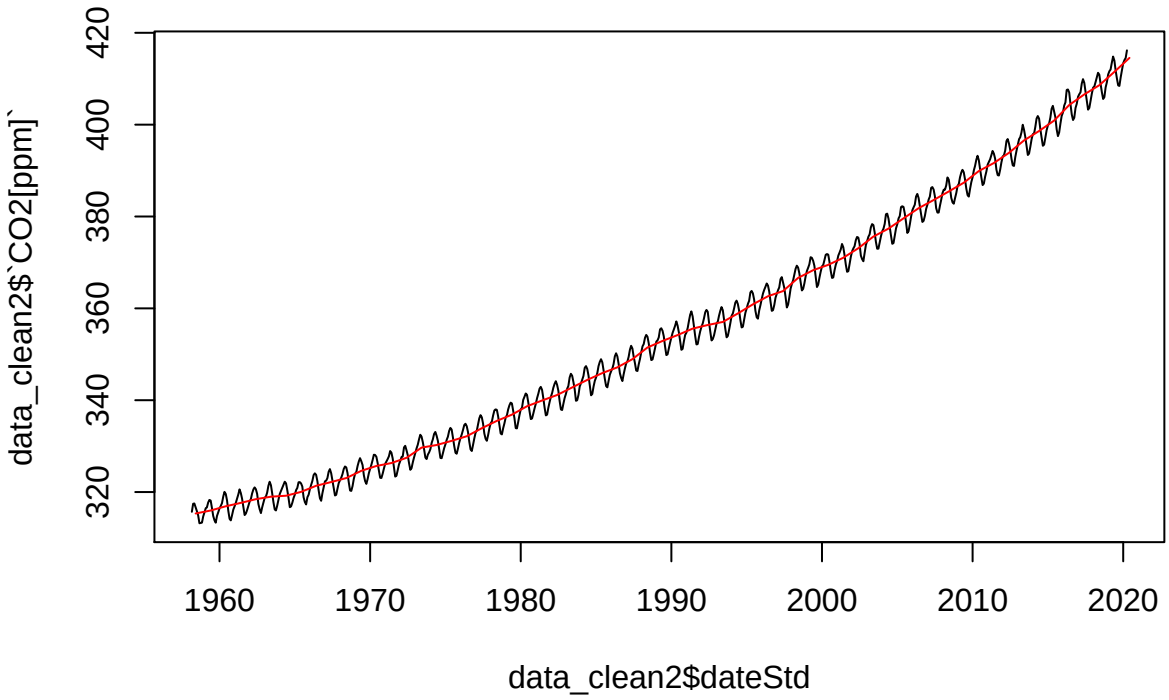


Nous observons bien une saisonalité de période annuelle. On ne va que s'intéresser dans la suite de cette étude aux données brutes c'est à dire le champ "*CO2(pm)*" car je ne sais pas trop à quoi correpond le reste des champs. Pour isoler la partie tendance, nous allons utiliser une technique simple de moyenne mobile. C'est un estimateur non-paramétrique de la tendance, au sens ou nous ne supposons pas de structure a-priori de cette tendance (par ex. linéaire ou polynomiale). On aurait aussi pu utiliser un lissage par noyaux mais bon restons sur des choses basiques.

```
MA<-data_clean2[, .(MA=mean(as.numeric(`CO2[ppm]`))),by=Yr]
MA[,Yr_date:=as.Date(convert_iso_date(Yr,"06"))]
MA$Yr<-as.numeric(MA$Yr)
tail(MA)
```

```
##      Yr       MA    Yr_date
## 1: 2015 400.8850 2015-06-01
## 2: 2016 404.2775 2016-06-01
## 3: 2017 406.5842 2017-06-01
## 4: 2018 408.5875 2018-06-01
## 5: 2019 411.4950 2019-06-01
## 6: 2020 414.5375 2020-06-01
```

```
plot(data_clean2$dateStd,data_clean2$`CO2[ppm]`,type='l')
lines(MA$Yr_date,MA$MA,col='red')
```

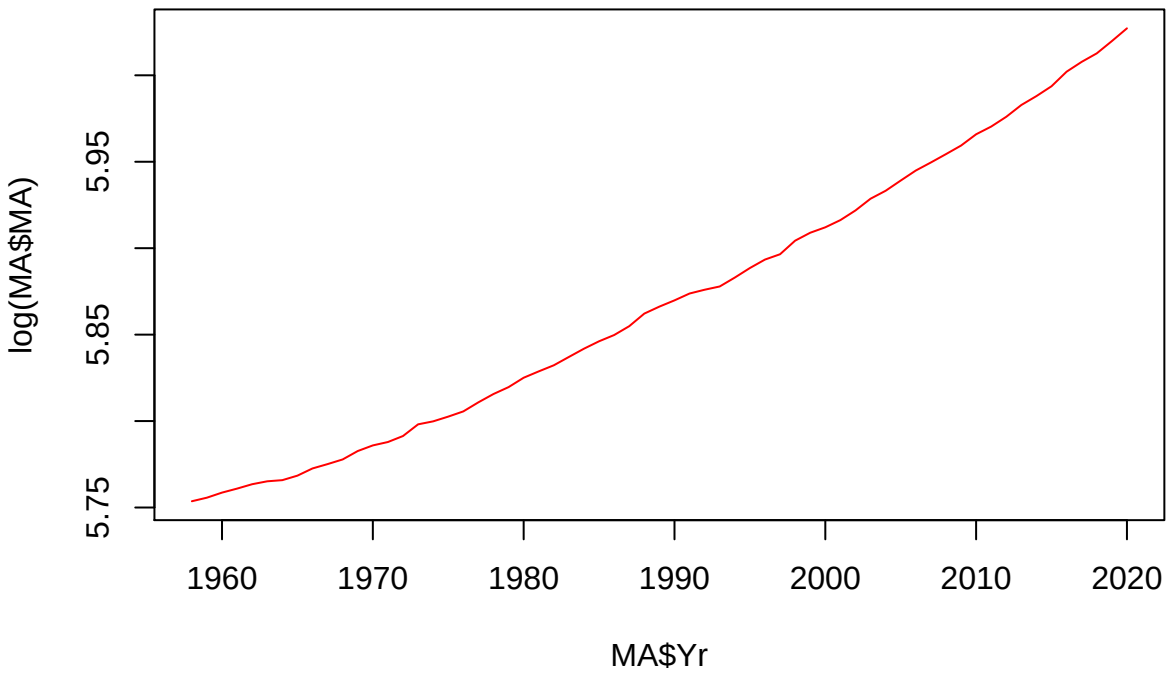


Nous obtenons la tendance de la série corrigée des effets de saisonalité

Modèle statistique et calibrage de la tendance (contribution lente)

Nous allons construire un modèle simple de la tendance qui servira à prédire le niveau de CO2 dans l'atmosphère pour les années futures. Faisons une régression sur la tendance. On pourrait enlever les premiers et derniers points car ce ne sont pas des donnés complètes.

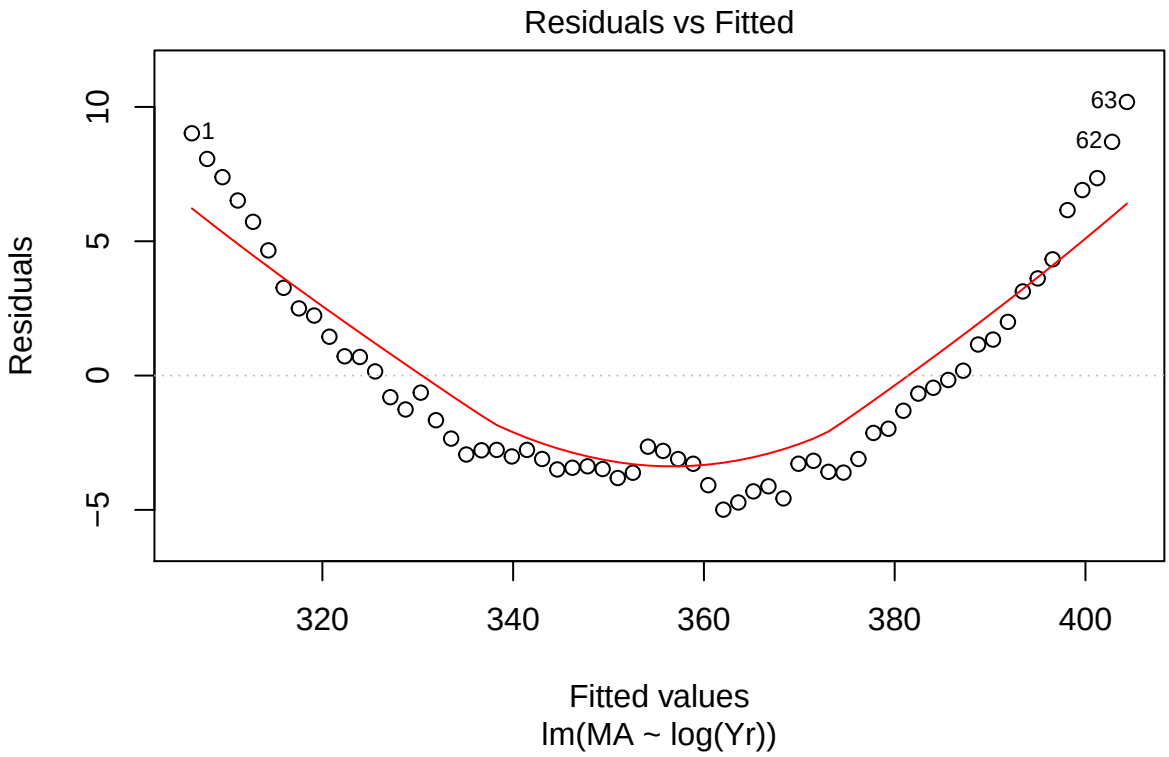
```
plot(MA$Yr,log(MA$MA),col='red',type='l')
```

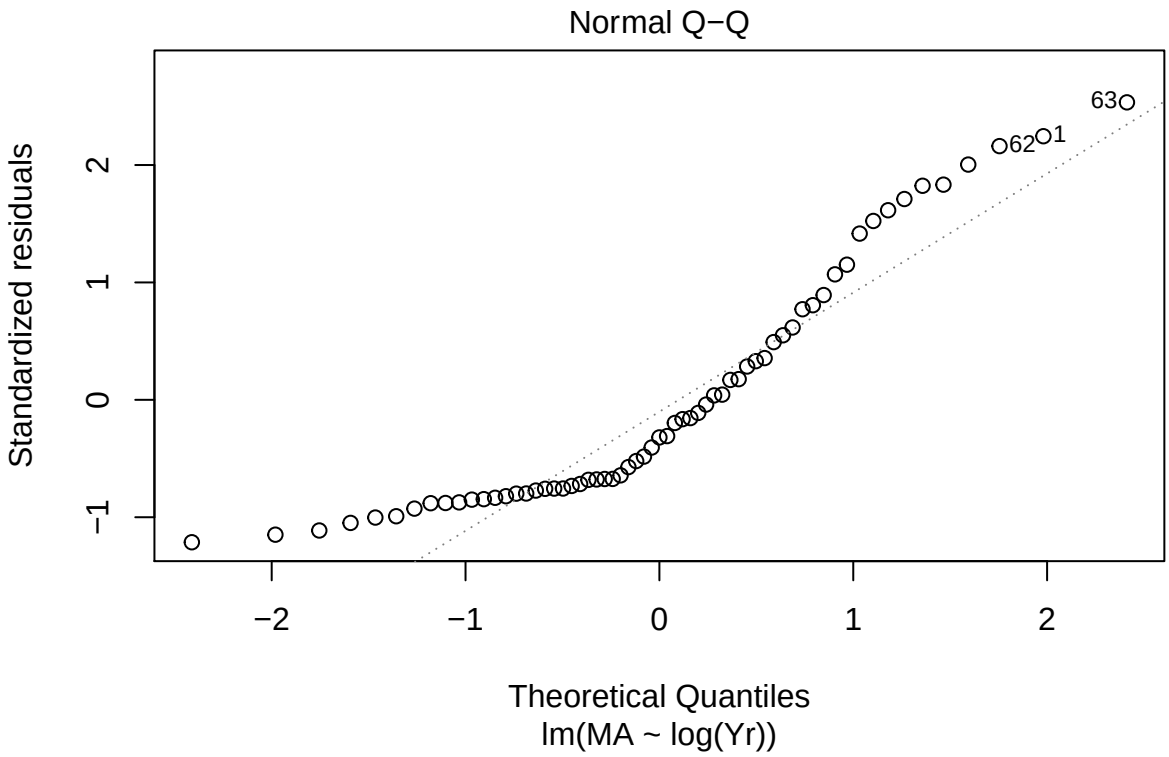


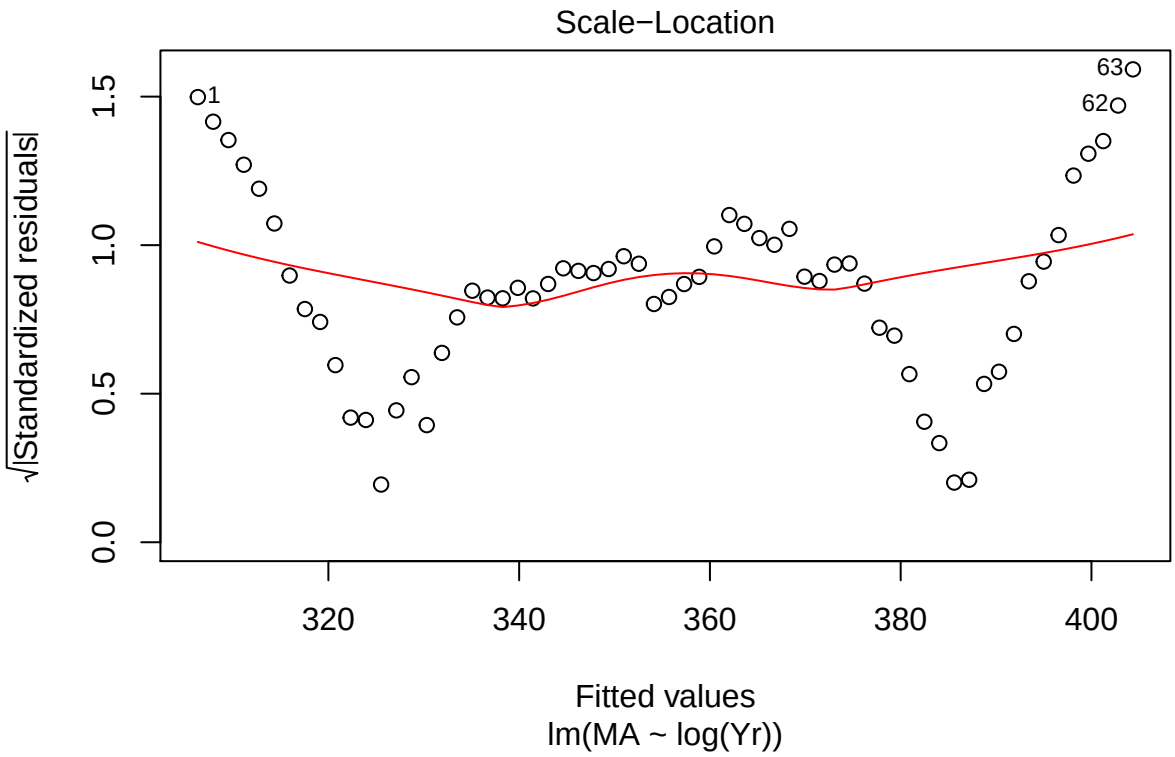

```
reg_lin_log<-lm(MA~log(Yr),data=MA)
summary(reg_lin_log)
```

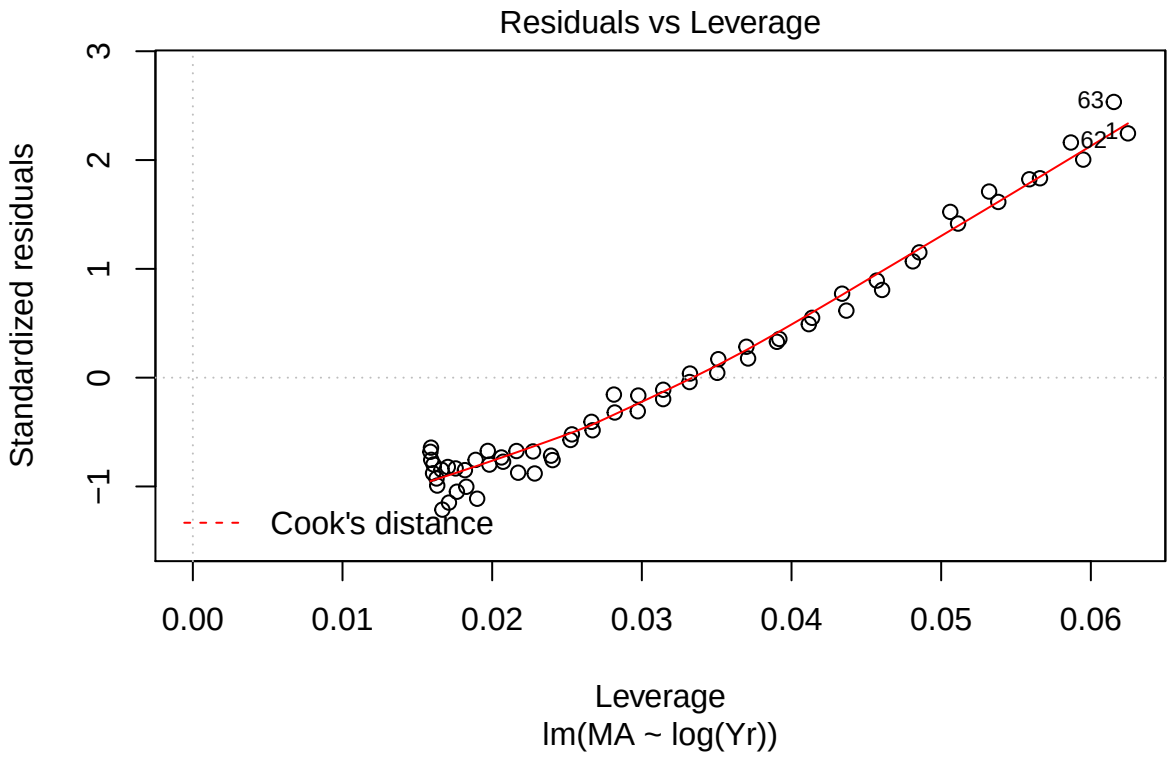
```
##
## Call:
## lm(formula = MA ~ log(Yr), data = MA)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.991 -3.228 -1.310  2.367 10.185
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -23531.28     434.28  -54.19   <2e-16 ***
## log(Yr)       3144.93      57.18   55.00   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.149 on 61 degrees of freedom
## Multiple R-squared:  0.9802, Adjusted R-squared:  0.9799
## F-statistic:  3025 on 1 and 61 DF,  p-value: < 2.2e-16
```

```
plot(reg_lin_log)
```


Residuals vs Fitted
Residuals
10
5
0
−5
1
62
63
320
340
360
380
400
Fitted values
lm(MA ~ log(Yr))


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(MA ~ log(Yr))
63
62
1
−2
−1
0
1
2

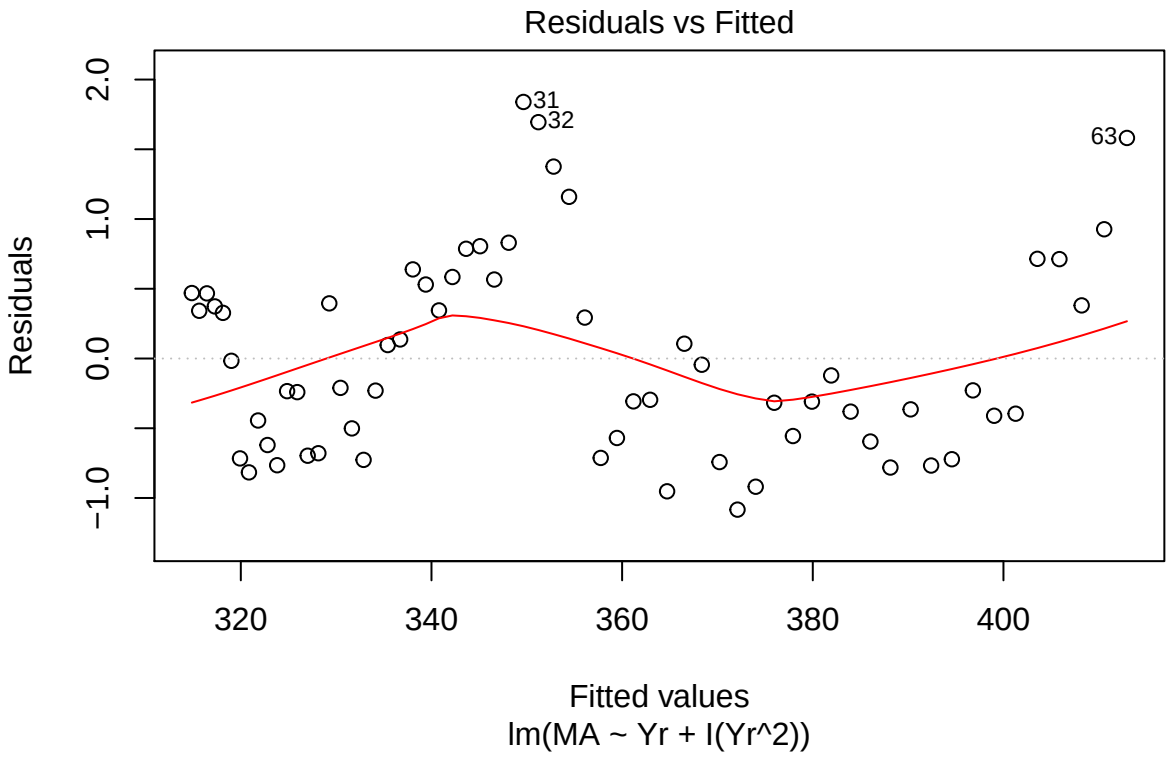

Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
1
62
63
320
340
360
380
400
Fitted values
lm(MA ~ log(Yr))

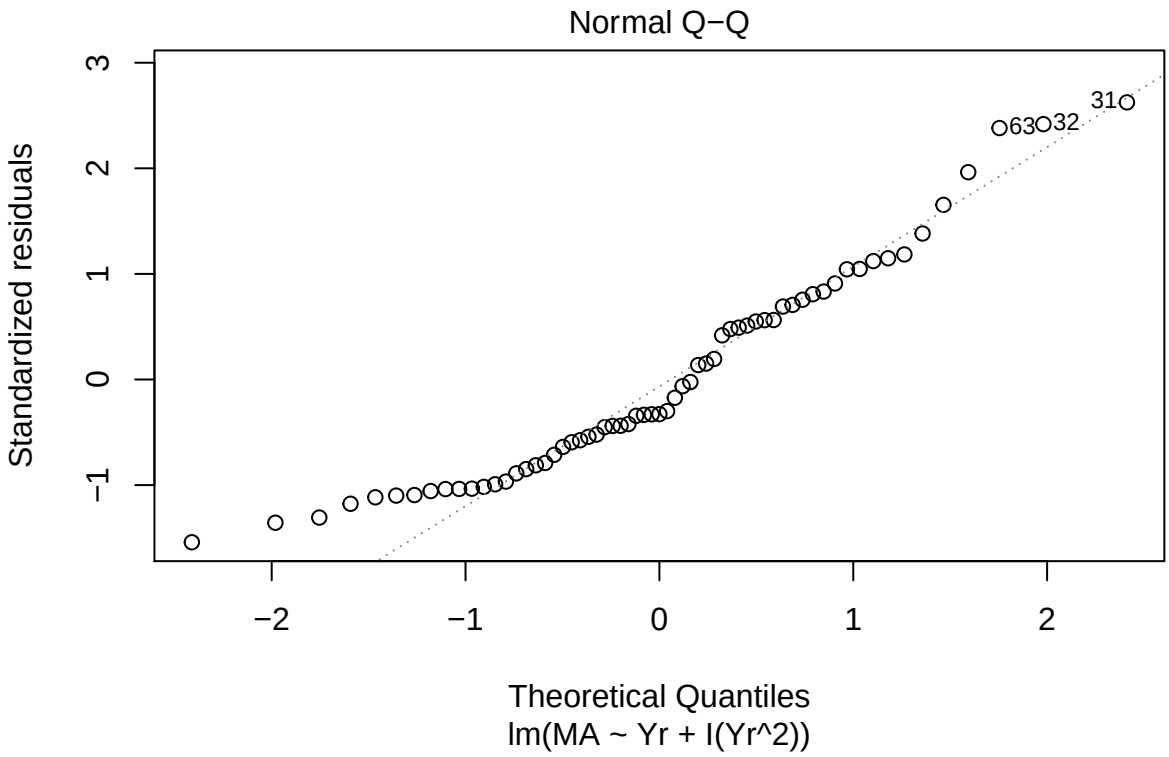


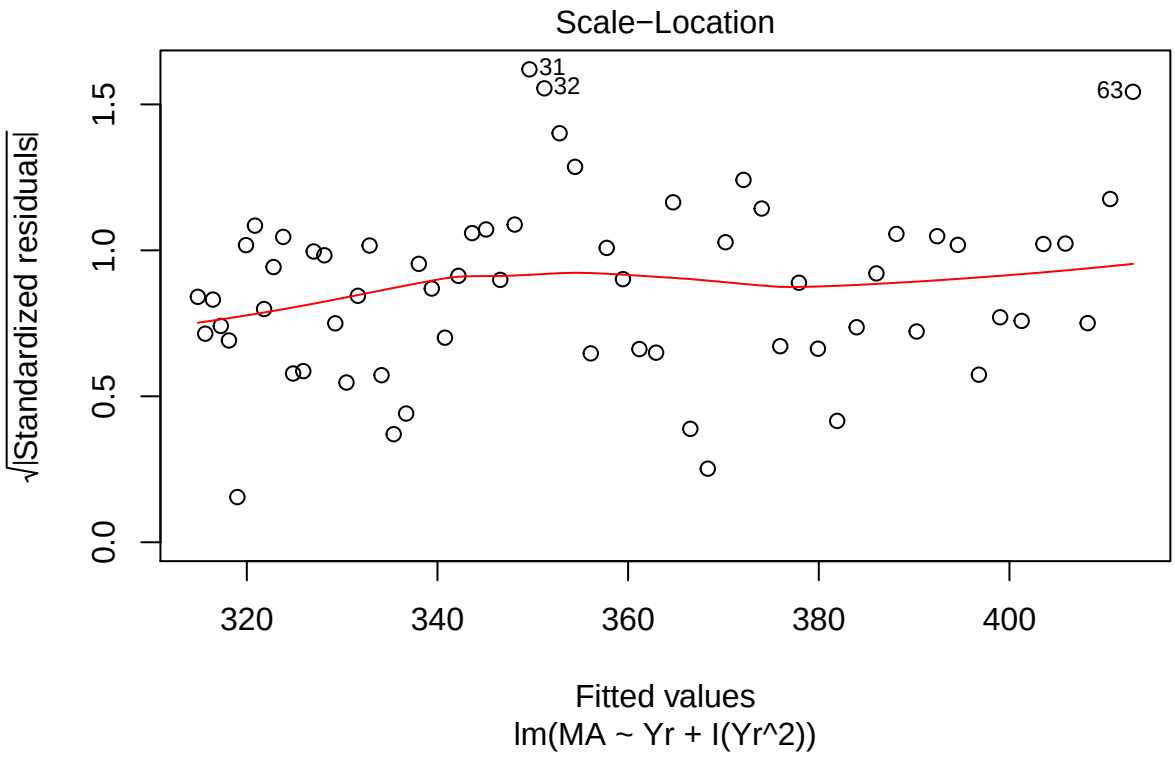

```
reg_lin<-lm(MA~Yr+I(Yr^2),data=MA)
summary(reg_lin)
```

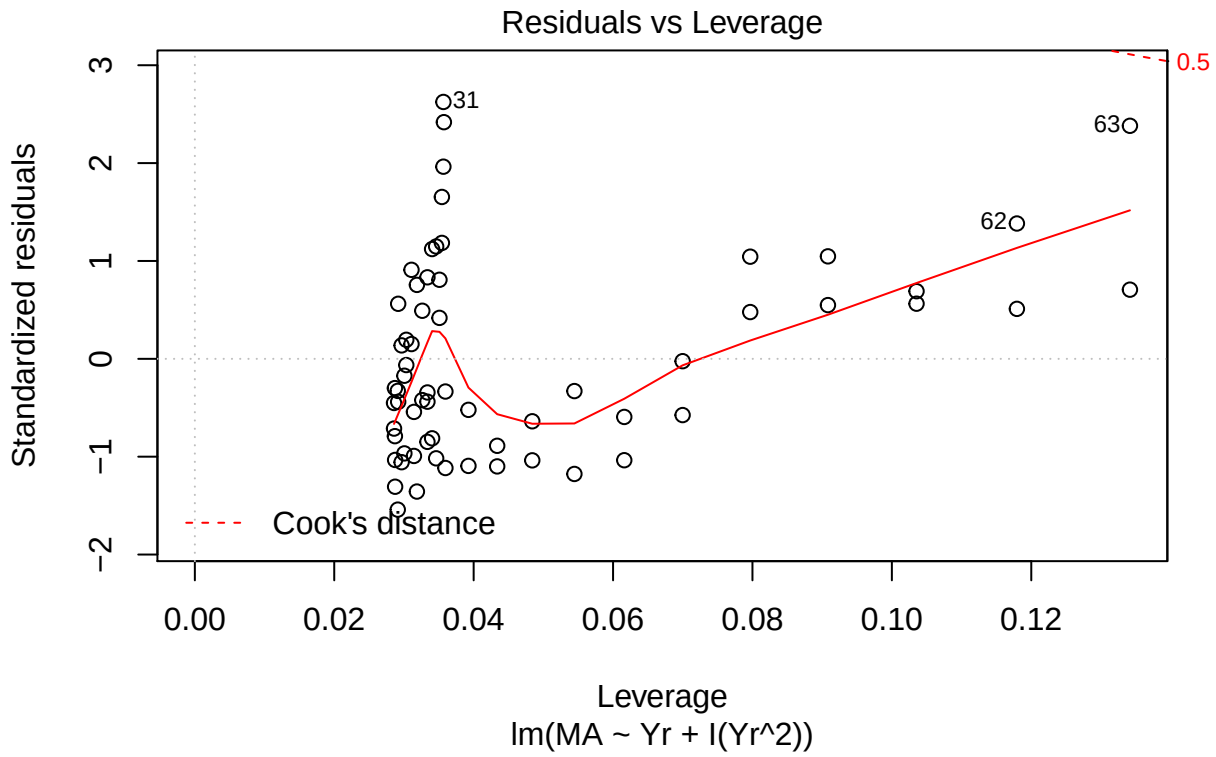
```
##
## Call:
## lm(formula = MA ~ Yr + I(Yr^2), data = MA)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0835 -0.5825 -0.2286  0.4681  1.8393
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.947e+04  1.203e+03   41.12   <2e-16 ***
## Yr          -5.097e+01  1.210e+00  -42.13   <2e-16 ***
## I(Yr^2)      1.321e-02  3.041e-04   43.44   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7136 on 60 degrees of freedom
## Multiple R-squared:  0.9994, Adjusted R-squared:  0.9994
## F-statistic: 5.215e+04 on 2 and 60 DF,  p-value: < 2.2e-16
```

```
plot(reg_lin)
```


Residuals vs Fitted
Residuals
2.0
1.0
0.0
−1.0
31
32
63
320
340
360
380
400
Fitted values
lm(MA ~ Yr + I(Yr^2))


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(MA ~ Yr + I(Yr^2))
31
63
32
−2
−1
0
1
2
−1
0
1
2
3

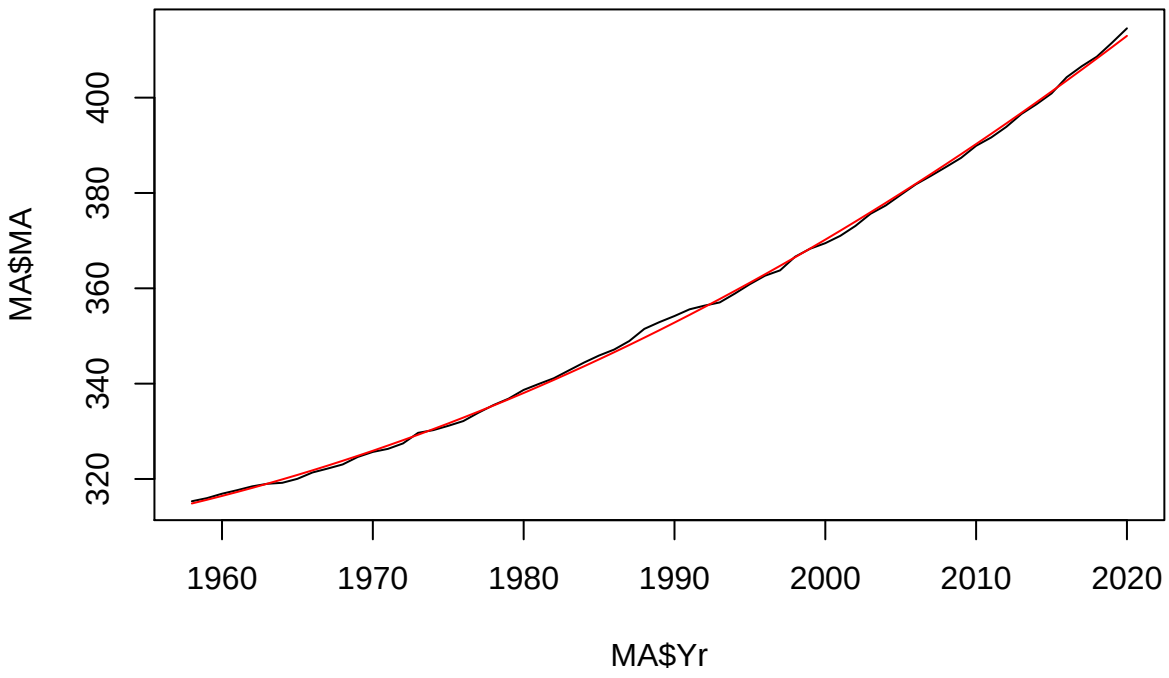

Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
31
32
63
320
340
360
380
400
Fitted values
lm(MA ~ Yr + I(Yr^2))




```
tail(predict(reg_lin))
```

```
##       58       59       60       61       62       63
## 401.2806 403.5629 405.8716 408.2068 410.5683 412.9563
```

```
plot(MA$Yr,MA$MA,type='l')
lines(MA$Yr,predict(reg_lin),col='red')
```



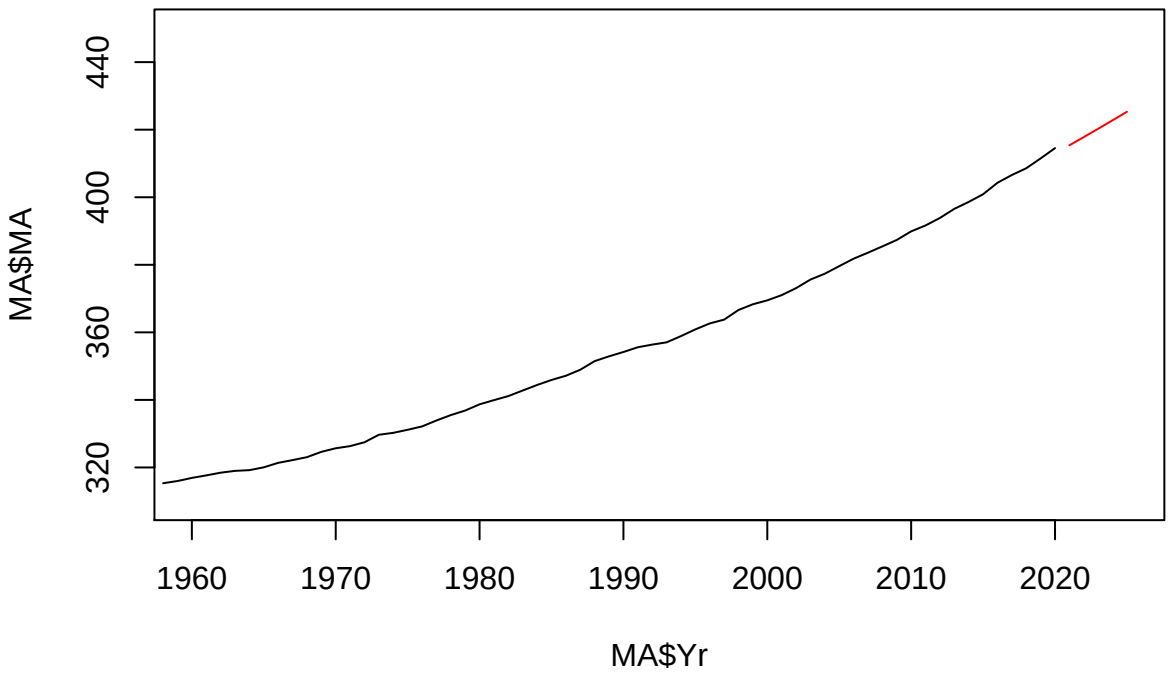
Le modèle de régression est donc créé.

Prédiction / Extrapolation

Il est temps de passser à l'extrapolation:

```
dates_futur<-as.Date(convert_iso_date(seq(2021,2025),"06"))
futur<-data.table(Yr=seq(2021,2025))

pred<-predict(reg_lin,futur,interval = "prediction")
plot(MA$Yr,MA$MA,type='l',xlim=c(1960,2025),ylim=c(310,450))
lines(seq(2021,2025),pred[,1],col='red')
```



La courbe en rouge représente les valeurs futures jusqu'à 2025.