in-situ CO2 Data

08 juin 2021

In this document we perform an analysis about carbon dioxide in the atmosphere. The goal is to detect a periodic oscillation and a slow but continuous increase in the concentration of carbon dioxide.

Technical information on the computer on which the analysis is run

We will be using the R language using the following libraries :

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.1     v dplyr   1.0.6
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(parsedate)
```

```
##
## Attaching package: 'parsedate'

## The following object is masked from 'package:readr':
##
##     parse_date
```

```
library(anytime)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

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

```r
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
library(fpp2)
```

```
## -- Attaching packages ---------------------------------------------- fpp2 2.4 --
```

```
## v fma       2.4     v expsmooth 2.3
```

```
##
```

```r
#sessionInfo()
```

Here are the available libraries

```r
devtools::session_info()
```

```
## - Session info ---------------------------------------------------------------
##  setting  value
##  version  R version 4.0.3 (2020-10-10)
##  os       Windows 10 x64
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  French_France.1252
##  ctype    French_France.1252
##  tz       Europe/Paris
##  date     2021-06-08
##
## - Packages -------------------------------------------------------------------
##  package     * version  date       lib source
##  anytime     * 0.3.9    2020-08-27 [1] CRAN (R 4.0.4)
##  assertthat    0.2.1    2019-03-21 [1] CRAN (R 4.0.3)
##  backports     1.2.1    2020-12-09 [1] CRAN (R 4.0.3)
##  broom         0.7.6    2021-04-05 [1] CRAN (R 4.0.5)
##  cachem        1.0.5    2021-05-15 [1] CRAN (R 4.0.5)
##  callr         3.7.0    2021-04-20 [1] CRAN (R 4.0.5)
##  cellranger    1.1.0    2016-07-27 [1] CRAN (R 4.0.3)
##  cli           2.5.0    2021-04-26 [1] CRAN (R 4.0.5)
##  colorspace    2.0-1    2021-05-04 [1] CRAN (R 4.0.5)
##  crayon        1.4.1    2021-02-08 [1] CRAN (R 4.0.5)
##  curl          4.3.1    2021-04-30 [1] CRAN (R 4.0.5)
##  DBI           1.1.1    2021-01-15 [1] CRAN (R 4.0.5)
##  dbplyr        2.1.1    2021-04-06 [1] CRAN (R 4.0.5)
##  desc          1.3.0    2021-03-05 [1] CRAN (R 4.0.5)
##  devtools      2.4.1    2021-05-05 [1] CRAN (R 4.0.5)
##  digest        0.6.27   2020-10-24 [1] CRAN (R 4.0.3)
```

```
## dplyr       * 1.0.6    2021-05-05 [1] CRAN (R 4.0.5)
## ellipsis      0.3.2    2021-04-29 [1] CRAN (R 4.0.5)
## evaluate      0.14     2019-05-28 [1] CRAN (R 4.0.3)
## expsmooth   * 2.3      2015-04-09 [1] CRAN (R 4.0.5)
## fansi         0.4.2    2021-01-15 [1] CRAN (R 4.0.5)
## fastmap       1.1.0    2021-01-25 [1] CRAN (R 4.0.5)
## fma         * 2.4      2020-01-14 [1] CRAN (R 4.0.5)
## forcats     * 0.5.1    2021-01-27 [1] CRAN (R 4.0.5)
## forecast    * 8.15     2021-06-01 [1] CRAN (R 4.0.5)
## fpp2        * 2.4      2020-09-09 [1] CRAN (R 4.0.5)
## fracdiff      1.5-1    2020-01-24 [1] CRAN (R 4.0.5)
## fs            1.5.0    2020-07-31 [1] CRAN (R 4.0.3)
## generics      0.1.0    2020-10-31 [1] CRAN (R 4.0.3)
## ggplot2     * 3.3.3    2020-12-30 [1] CRAN (R 4.0.5)
## glue          1.4.2    2020-08-27 [1] CRAN (R 4.0.3)
## gtable        0.3.0    2019-03-25 [1] CRAN (R 4.0.3)
## haven         2.4.1    2021-04-23 [1] CRAN (R 4.0.5)
## hms           1.1.0    2021-05-17 [1] CRAN (R 4.0.5)
## htmltools     0.5.1.1  2021-01-22 [1] CRAN (R 4.0.5)
## httr          1.4.2    2020-07-20 [1] CRAN (R 4.0.3)
## jsonlite      1.7.2    2020-12-09 [1] CRAN (R 4.0.5)
## knitr         1.33     2021-04-24 [1] CRAN (R 4.0.5)
## lattice       0.20-41  2020-04-02 [2] CRAN (R 4.0.3)
## lifecycle     1.0.0    2021-02-15 [1] CRAN (R 4.0.5)
## lmtest        0.9-38   2020-09-09 [1] CRAN (R 4.0.5)
## lubridate   * 1.7.10   2021-02-26 [1] CRAN (R 4.0.5)
## magrittr      2.0.1    2020-11-17 [1] CRAN (R 4.0.5)
## memoise       2.0.0    2021-01-26 [1] CRAN (R 4.0.5)
## modelr        0.1.8    2020-05-19 [1] CRAN (R 4.0.3)
## munsell       0.5.0    2018-06-12 [1] CRAN (R 4.0.3)
## nlme          3.1-149  2020-08-23 [2] CRAN (R 4.0.3)
## nnet          7.3-14   2020-04-26 [2] CRAN (R 4.0.3)
## parsedate   * 1.2.1    2021-04-20 [1] CRAN (R 4.0.5)
## pillar        1.6.1    2021-05-16 [1] CRAN (R 4.0.5)
## pkgbuild      1.2.0    2020-12-15 [1] CRAN (R 4.0.5)
## pkgconfig     2.0.3    2019-09-22 [1] CRAN (R 4.0.3)
## pkgload       1.2.1    2021-04-06 [1] CRAN (R 4.0.5)
## prettyunits   1.1.1    2020-01-24 [1] CRAN (R 4.0.3)
## processx      3.5.2    2021-04-30 [1] CRAN (R 4.0.5)
## ps            1.6.0    2021-02-28 [1] CRAN (R 4.0.5)
## purrr       * 0.3.4    2020-04-17 [1] CRAN (R 4.0.3)
## quadprog      1.5-8    2019-11-20 [1] CRAN (R 4.0.3)
## quantmod      0.4.18   2020-12-09 [1] CRAN (R 4.0.5)
## R6            2.5.0    2020-10-28 [1] CRAN (R 4.0.3)
## Rcpp          1.0.6    2021-01-15 [1] CRAN (R 4.0.5)
## readr       * 1.4.0    2020-10-05 [1] CRAN (R 4.0.3)
## readxl        1.3.1    2019-03-13 [1] CRAN (R 4.0.3)
## remotes       2.3.0    2021-04-01 [1] CRAN (R 4.0.5)
## reprex        2.0.0    2021-04-02 [1] CRAN (R 4.0.5)
## rlang         0.4.11   2021-04-30 [1] CRAN (R 4.0.5)
## rmarkdown     2.8      2021-05-07 [1] CRAN (R 4.0.5)
## rprojroot     2.0.2    2020-11-15 [1] CRAN (R 4.0.3)
## rstudioapi    0.13     2020-11-12 [1] CRAN (R 4.0.3)
## rvest         1.0.0    2021-03-09 [1] CRAN (R 4.0.5)
```

```
##  scales        1.1.1    2020-05-11 [1] CRAN (R 4.0.3)
##  sessioninfo   1.1.1    2018-11-05 [1] CRAN (R 4.0.5)
##  stringi       1.5.3    2020-09-09 [1] CRAN (R 4.0.3)
##  stringr     * 1.4.0    2019-02-10 [1] CRAN (R 4.0.3)
##  testthat      3.0.2    2021-02-14 [1] CRAN (R 4.0.5)
##  tibble      * 3.1.1    2021-04-18 [1] CRAN (R 4.0.5)
##  tidyr       * 1.1.3    2021-03-03 [1] CRAN (R 4.0.5)
##  tidyselect    1.1.1    2021-04-30 [1] CRAN (R 4.0.5)
##  tidyverse   * 1.3.1    2021-04-15 [1] CRAN (R 4.0.5)
##  timeDate      3043.102 2018-02-21 [1] CRAN (R 4.0.5)
##  tseries       0.10-48  2020-12-04 [1] CRAN (R 4.0.5)
##  TTR           0.24.2   2020-09-01 [1] CRAN (R 4.0.5)
##  urca          1.3-0    2016-09-06 [1] CRAN (R 4.0.5)
##  usethis       2.0.1    2021-02-10 [1] CRAN (R 4.0.5)
##  utf8          1.2.1    2021-03-12 [1] CRAN (R 4.0.5)
##  vctrs         0.3.8    2021-04-29 [1] CRAN (R 4.0.5)
##  withr         2.4.2    2021-04-18 [1] CRAN (R 4.0.5)
##  xfun          0.23     2021-05-15 [1] CRAN (R 4.0.5)
##  xml2          1.3.2    2020-04-23 [1] CRAN (R 4.0.3)
##  xts           0.12.1   2020-09-09 [1] CRAN (R 4.0.5)
##  yaml          2.2.1    2020-02-01 [1] CRAN (R 4.0.3)
##  zoo           1.8-9    2021-03-09 [1] CRAN (R 4.0.5)
##
## [1] C:/Users/f.chaillan/Documents/R/win-library/4.0
## [2] C:/Program Files/R/R-4.0.3/library
```

Atmospheric CO2 data

Data are available at (Scrips)[https://scrippsco2.ucsd.edu/data/atmospheric_co2/primary_mlo_co2_record.html]

C. D. Keeling, S. C. Piper, R. B. Bacastow, M. Wahlen, T. P. Whorf, M. Heimann, and H. A. Meijer, Exchanges of atmospheric CO2 and 13CO2 with the terrestrial biosphere and oceans from 1978 to 2000. I. Global aspects, SIO Reference Series, No. 01-06, Scripps Institution of Oceanography, San Diego, 88 pages, 2001.

Description of the dataset

The data file below contains **10 columns**.
- Columns 1-4 give the dates in several redundant formats. - Column 5 below gives monthly Mauna Loa CO2 concentrations in micro-mol CO2 per
mole (ppm), reported on the 2012 SIO manometric mole fraction scale. This is the standard version of the data most often sought. *The monthly values have been adjusted to 24:00 hours on the 15th of each month.*
- Column 6 gives the same data after a seasonal adjustment to remove the quasi-regular seasonal cycle. The adjustment involves subtracting from the data a 4-harmonic fit with a linear gain factor.
- Column 7 is a smoothed version of the data generated from a stiff cubic spline function plus 4-harmonic functions with linear gain.
- Column 8 is the same smoothed version with the seasonal cycle removed.
- Column 9 is identical to Column 5 except that the missing values from
- Column 5 have been filled with values from Column 7.
- Column 10 is identical to Column 6 except missing values have been filled with values from Column 8.

- Missing values are denoted by -99.99
CO2 concentrations are measured on the '12' calibration scale

Loading data

The data start row 54 with the header. For this reason I skipped the 53 first rows when reading data. To proceed to analysis on the same dataset. I downloaded the file only if it not already on my computer.

```
data_url="https://scrippsco2.ucsd.edu/assets/data/atmospheric/stations/in_situ_co2/monthly/monthly_in_s
data_file="monthly_in_situ_co2_mlo.csv"

if (!file.exists(data_file)){download.file(data_url,data_file,method="auto")}
data=read.csv("monthly_in_situ_co2_mlo.csv", header= T,skip=54)
head(data)
```

```
##     Yr Mn     Date   Date.1       CO2 seasonally         fit seasonally.1
## 1   NA NA                NA             adjusted             adjusted fit
## 2   NA NA    Excel       NA    [ppm]      [ppm]        [ppm]        [ppm]
## 3 1958  1    21200 1958.041   -99.99     -99.99       -99.99       -99.99
## 4 1958  2    21231 1958.126   -99.99     -99.99       -99.99       -99.99
## 5 1958  3    21259 1958.203   315.70     314.44       316.19       314.91
## 6 1958  4    21290 1958.288   317.45     315.16       317.30       314.99
##          CO2.1    seasonally.2
## 1       filled adjusted filled
## 2        [ppm]           [ppm]
## 3       -99.99          -99.99
## 4       -99.99          -99.99
## 5       315.70          314.44
## 6       317.45          315.16
```

```
tail(data)
```

```
##       Yr Mn     Date   Date.1       CO2 seasonally         fit seasonally.1
## 765 2021  7    44392 2021.537   -99.99     -99.99       -99.99       -99.99
## 766 2021  8    44423 2021.622   -99.99     -99.99       -99.99       -99.99
## 767 2021  9    44454 2021.707   -99.99     -99.99       -99.99       -99.99
## 768 2021 10    44484 2021.789   -99.99     -99.99       -99.99       -99.99
## 769 2021 11    44515 2021.874   -99.99     -99.99       -99.99       -99.99
## 770 2021 12    44545 2021.956   -99.99     -99.99       -99.99       -99.99
##           CO2.1 seasonally.2
## 765      -99.99       -99.99
## 766      -99.99       -99.99
## 767      -99.99       -99.99
## 768      -99.99       -99.99
## 769      -99.99       -99.99
## 770      -99.99       -99.99
```

Insecpecting data

The 2 first lines of the dataset are containing information about the data. I removed them from the dataset.

```
data2=data[-c(1,2),]
```

Is there some missing value in the data set:

```
na_records= apply(data2,1, function (x) any(is.na(x)))
data2[na_records,]
```

```
##  [1] Yr            Mn            Date          Date.1        CO2
##  [6] seasonally    fit           seasonally.1 CO2.1         seasonally.2
## <0 rows> (or 0-length row.names)
```

Every rows are filled with values. Nevertheless some missing value are denoted by **-99,99**. See paragraph above describing data. Before dealing with the missing value, the type "Character" must be converted into numeric in order to look for 99.99 value.

```
i<-c(5:10)
data2[,i] =apply(data2[,i],2,function(x) as.numeric(as.character(x)))
```

As I need to proceed to time series analysis which involve lag shifting I need to verify that every year will contain 12 months. So instead of deleting data which may lead to incoherent analysis I decided to replace the CO2 concentration by the data generated from a *stiff cubic spline function plus 4-harmonics (the column 7 of the original dataset)*

```
data2<-data2%>%mutate(CO2_clean=case_when(CO2==-99.99 ~ CO2.1, TRUE ~CO2))
data2<-data2%>%
  mutate(seasonally_clean=case_when(seasonally==-99.99 ~ seasonally.1, TRUE ~seasonally))
```

Now, any data still containing missing value **-99.99**, are removed to allow time series analysis.

```
mv_records= sapply(data2$CO2_clean, function (x) ((x==-99.99)))

data2[mv_records,]
```

```
##       Yr Mn     Date   Date.1    CO2 seasonally    fit seasonally.1  CO2.1
## 3   1958  1    21200 1958.041 -99.99     -99.99 -99.99       -99.99 -99.99
## 4   1958  2    21231 1958.126 -99.99     -99.99 -99.99       -99.99 -99.99
## 763 2021  5    44331 2021.370 -99.99     -99.99 -99.99       -99.99 -99.99
## 764 2021  6    44362 2021.455 -99.99     -99.99 -99.99       -99.99 -99.99
## 765 2021  7    44392 2021.537 -99.99     -99.99 -99.99       -99.99 -99.99
## 766 2021  8    44423 2021.622 -99.99     -99.99 -99.99       -99.99 -99.99
## 767 2021  9    44454 2021.707 -99.99     -99.99 -99.99       -99.99 -99.99
## 768 2021 10    44484 2021.789 -99.99     -99.99 -99.99       -99.99 -99.99
## 769 2021 11    44515 2021.874 -99.99     -99.99 -99.99       -99.99 -99.99
## 770 2021 12    44545 2021.956 -99.99     -99.99 -99.99       -99.99 -99.99
##     seasonally.2 CO2_clean seasonally_clean
## 3         -99.99    -99.99           -99.99
## 4         -99.99    -99.99           -99.99
## 763       -99.99    -99.99           -99.99
## 764       -99.99    -99.99           -99.99
## 765       -99.99    -99.99           -99.99
## 766       -99.99    -99.99           -99.99
```

```
## 767       -99.99    -99.99           -99.99
## 768       -99.99    -99.99           -99.99
## 769       -99.99    -99.99           -99.99
## 770       -99.99    -99.99           -99.99
```

```
data4=data2[!mv_records,]

data4%>%group_by(Yr)%>%summarise(nb=n())%>%filter(nb<12)
```

```
## # A tibble: 2 x 2
##      Yr    nb
##   <int> <int>
## 1  1958    10
## 2  2021     4
```

The first year *1958* and the last year *2021* don't show data for the 12 months. From my understanding so far this will not impair the further analysis so I decided to maintain those years in the dataset.

I will perform the analysis from March 1958 to April 2021.

Date management :

Creating a column date inheriting from year and month as it seems impossible to convert correctly other format. It would have been beneficial to know which format has been used in column Date. I used the *lubridate package* and assume that day 15 is a good candidate for the conversion.

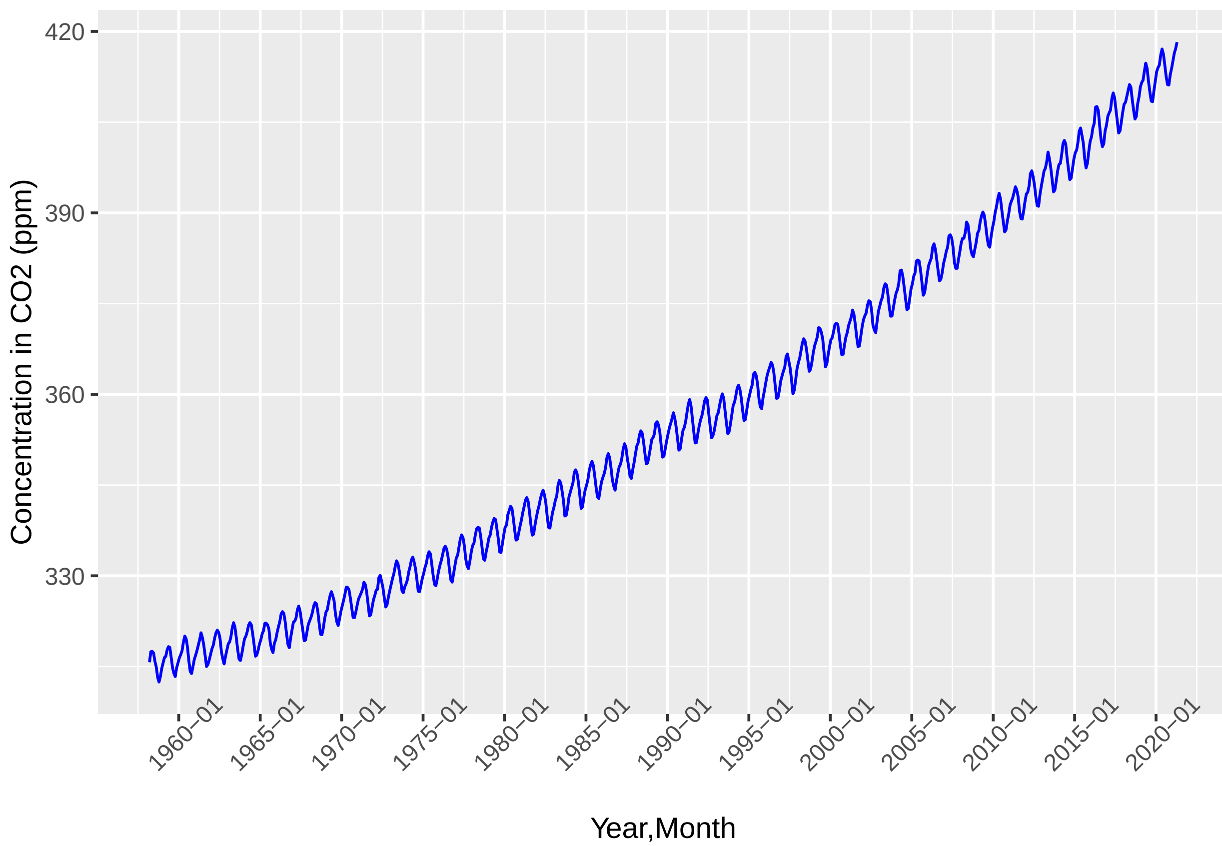
```
data4$builtdate<-ymd(paste0(data4$Yr," ",data4$Mn," ","15"))
```

Data Visualisation

Raw data viz

I used the package ggplot2 to proceed to the first visualisation of the data set.

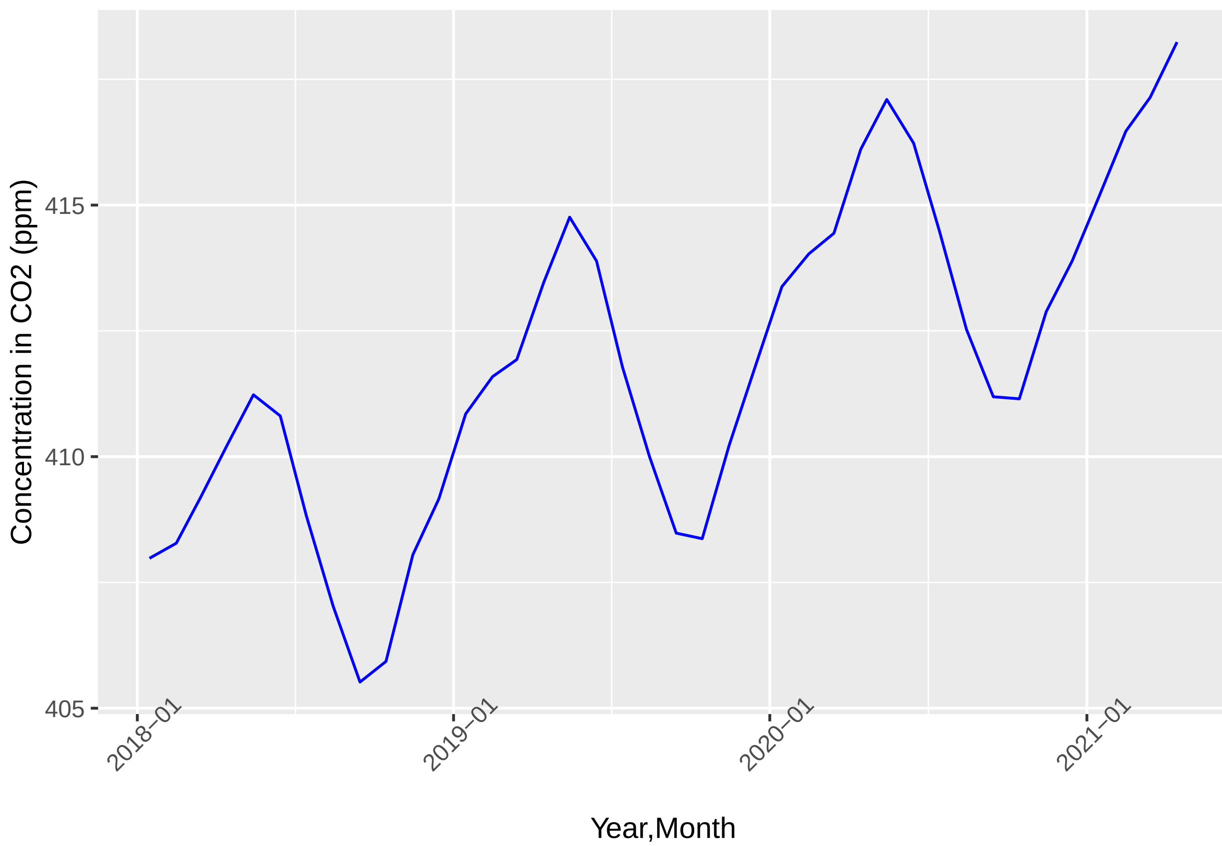
```
ggplot(data4,aes(builtdate,CO2_clean))+
  geom_line(color='blue')+
  xlab("Year,Month")+
  ylab("Concentration in CO2 (ppm)")+
  scale_x_date(date_labels = "%Y-%m",date_breaks = "5 year")+
  theme(axis.text.x =element_text(angle=45) )
```



This figure shows both the intra-annual variation of co2 concentration and the inter-annual increase.

In order to focus on the the intra-annual variation, a zoom is provided on the 40 last entries of the dataset.

```
data5=tail(data4,40)
ggplot(data5,aes(builtdate,CO2_clean))+
  geom_line(color='blue')+
  xlab("Year,Month")+
  ylab("Concentration in CO2 (ppm)")+
  scale_x_date(date_labels = "%Y-%m")+
  theme(axis.text.x =element_text(angle=45))
```



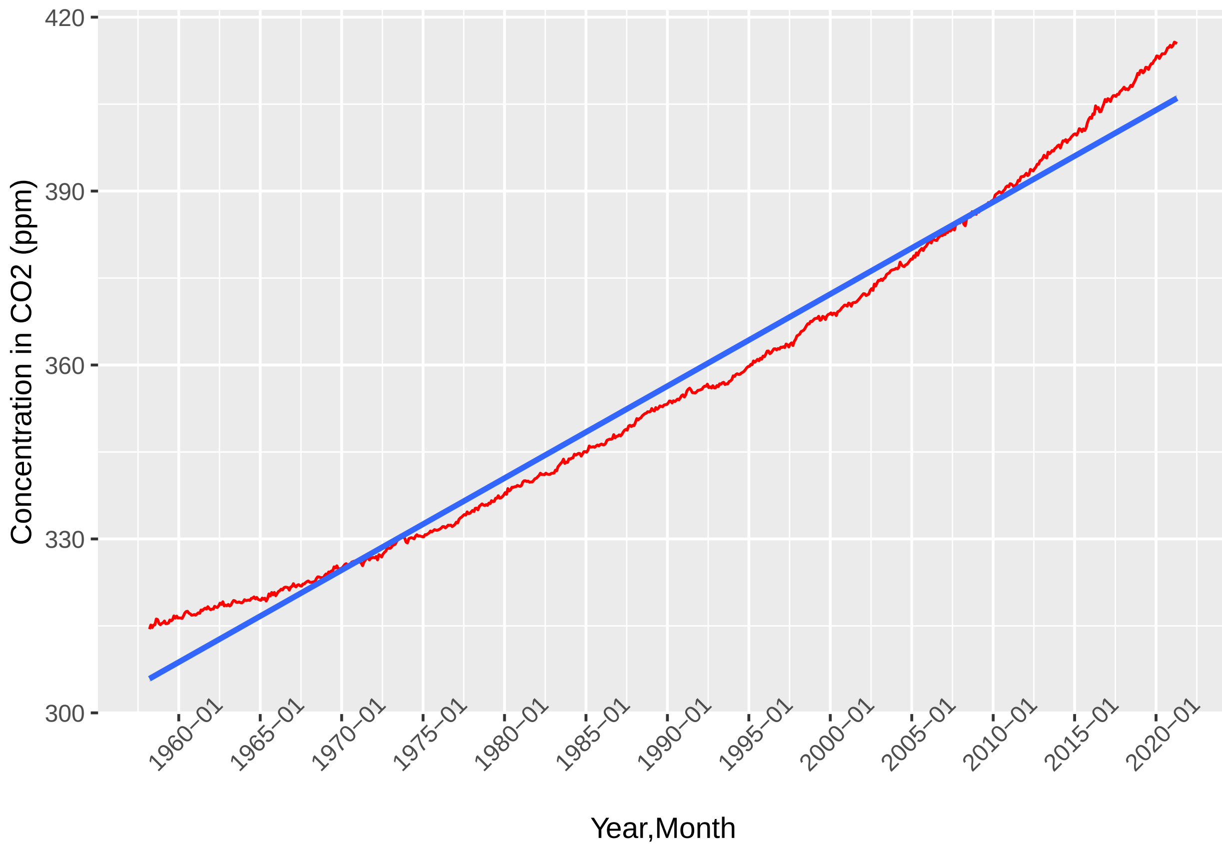
Considering that the data are measured in the North Hemisphere, we understand that the maximum concentration is reached during the summer and the minimum during the winter.

Regression model

The dataset provides in column 6 and 10 the same data after a seasonal adjustment to remove the quasi-regular seasonal cycle. The adjustment involves subtracting from the data a 4-harmonic fit with a linear gain factor.
The following plot shows the inter-annual variation with a model of linear regression.

```
ggplot(data4,aes(x=builtdate,y=seasonally_clean))+
  geom_line(color='red')+
  xlab("Year,Month")+
  ylab("Concentration in CO2 (ppm)")+
  scale_x_date(date_labels = "%Y-%m",date_breaks = "5 year")+
  theme(axis.text.x =element_text(angle=45) )+
geom_smooth(method ="lm")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



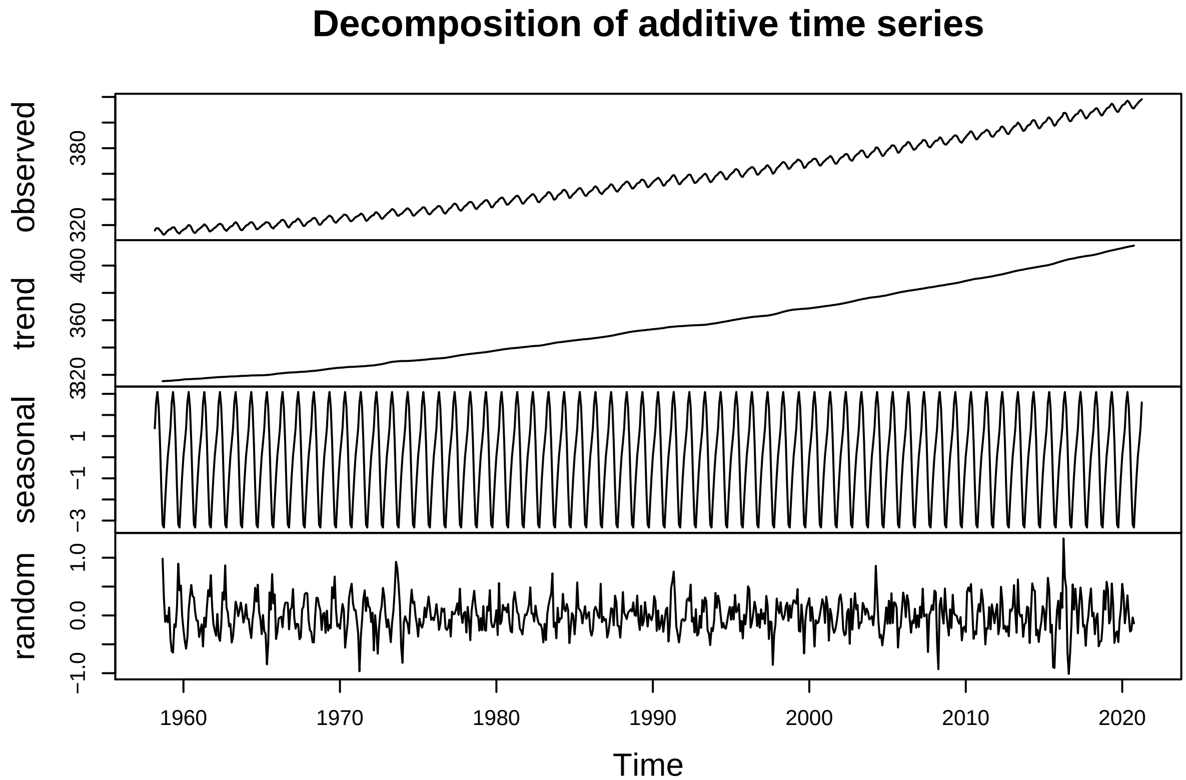
The linear model gives a good match with data. Nevertheless, for the tail and head of the data, the linear model seems to be less accurate.

For this reason I decided to dig into time series analysis with R in order to find a suitable answer to the questions raised.

Differeciating the trend and the cycle

Using the time series tools of R *ts*

```
CO2_main<-ts(data4$CO2_clean,start=c(1958,3),frequency = 12)
decomposesignal<-decompose(CO2_main,type="additive")
plot(decomposesignal)
```

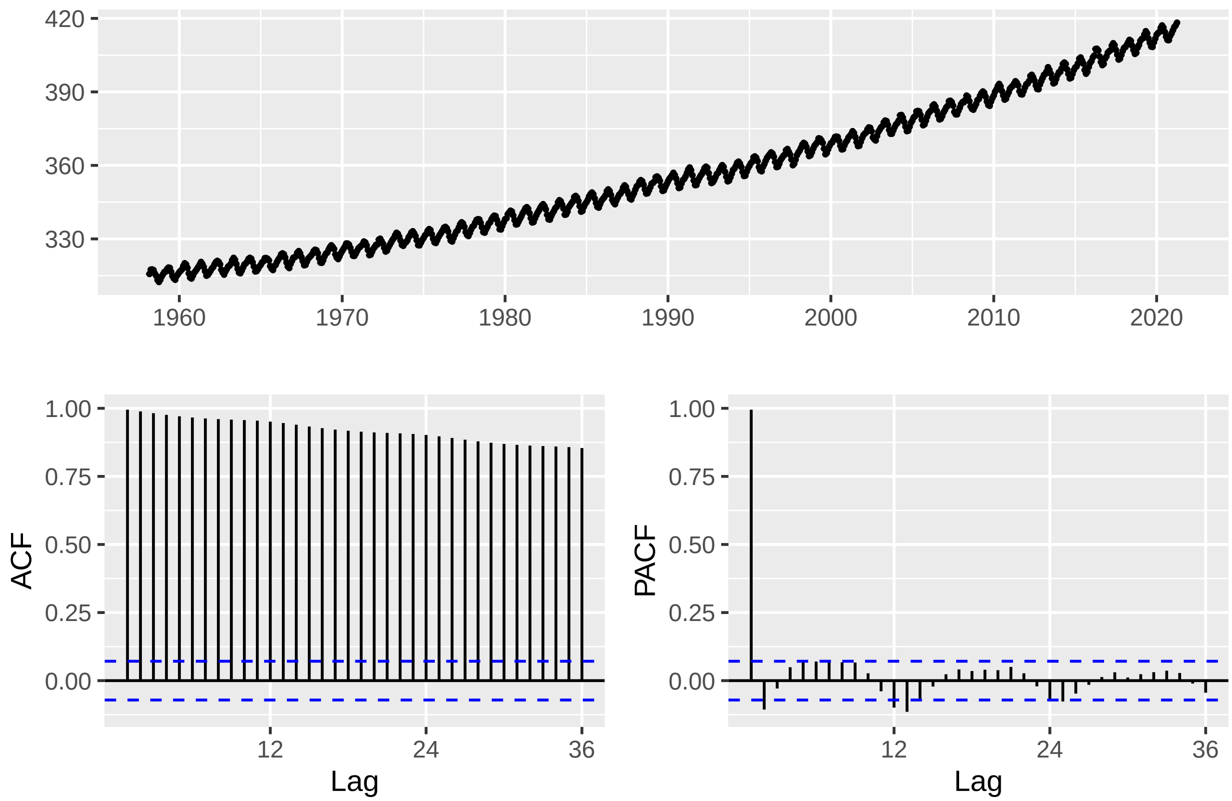


We have solved the first question graphically. I used the *additive type* to decompose the signal. The **trend** plot depicts the dataset when the annual variation is removed. The **seasonal** plot depicts the seasonal cycle observed (max in summer, min in winter ref. Northern hemisphere).

Modeling the data

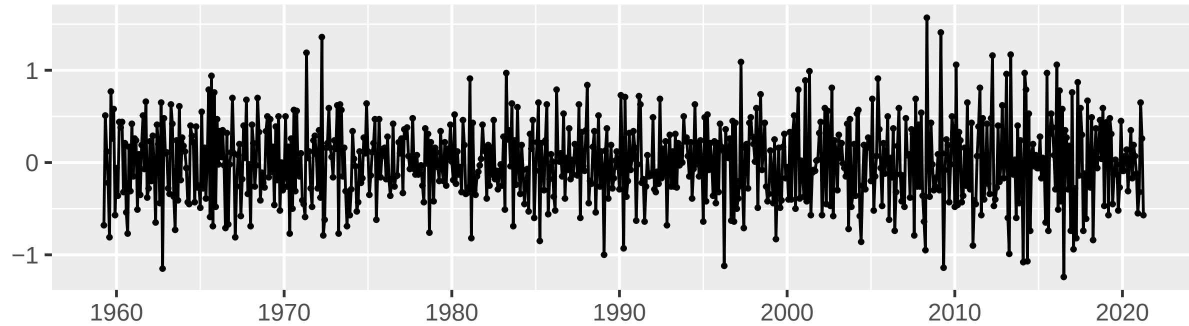
The data is not stationary because the mean of the data is time depending.

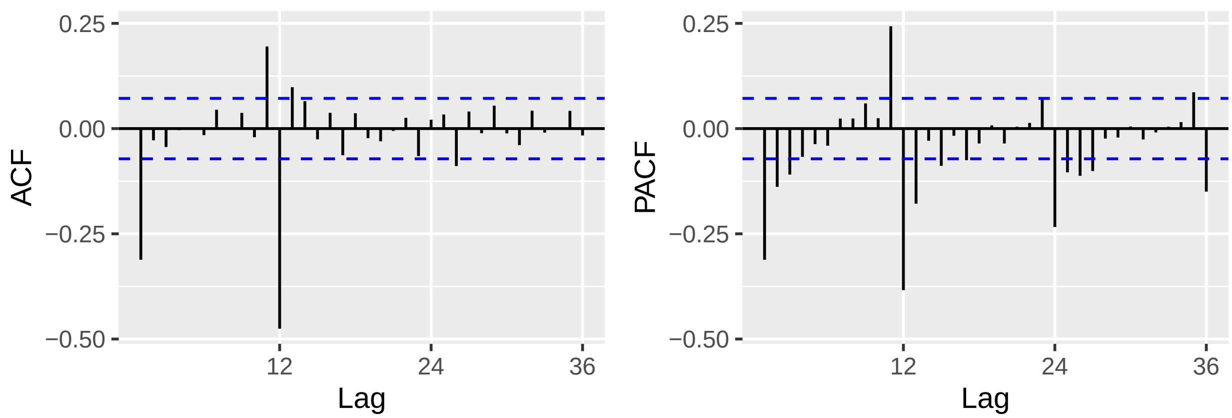
```
ggtsdisplay(CO2_main)
```



The data shows a seasonality. The ACF is not going down to zero.

```
CO2_main%>%diff(lag=12)%>%diff()%>%ggtsdisplay()
```





Adding a lag of 12 shows an ACF decreasing to 0.

Considering the seasonality, it is worth using seasonal differencing to model the data

Forecasting

I split the dataset in *training data as Keepling_ts* and *test data as data_test_ts*.

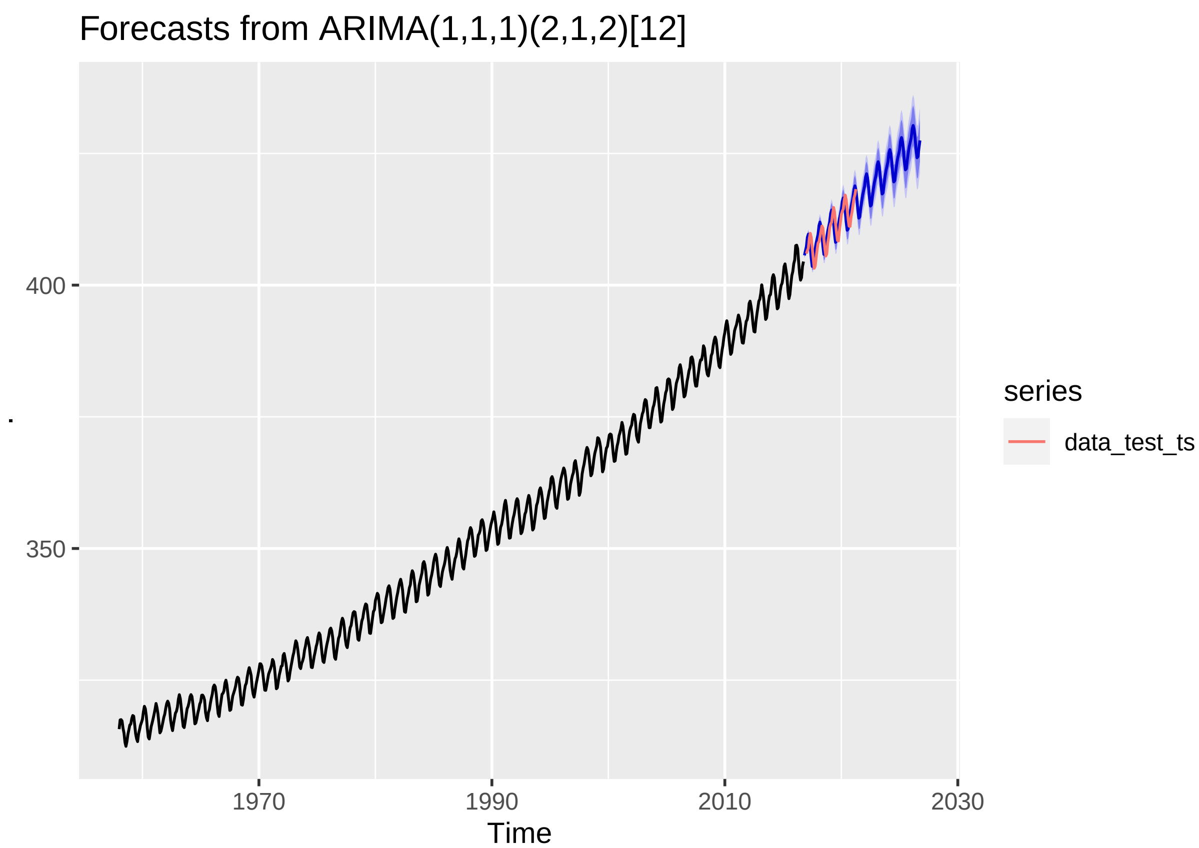
```
data7<-data4%>%select(builtdate,CO2_clean,Yr)%>%
  filter(Yr<2017)%>%select(-Yr)
keepling_ts<-ts(data7%>%
                  dplyr::select(-builtdate),start=c(data7$builtdate[1] %>% year(),1),frequency=12)

data_test<-data4%>%select(builtdate,CO2_clean,Yr)%>%
  filter(Yr>2016)%>%select(-Yr)
data_test_ts<-ts(data_test%>%
                   dplyr::select(-builtdate),start=c(data4$builtdate[711]%>% year(),1),frequency=12)
```

ARIMA model

The parameter are those defined by R in the automatic mode

```
co2_arima<-keepling_ts%>%auto.arima()
co2_arima%>%forecast(h=12*10)%>%autoplot()+
autolayer(data_test_ts)
```




```
summary(co2_arima)
```

```
## Series: .
## ARIMA(1,1,1)(2,1,2)[12]
##
## Coefficients:
##          ar1      ma1     sar1     sar2     sma1     sma2
##       0.2384  -0.5957  -0.3073  -0.0015  -0.5408  -0.2704
## s.e.  0.0898   0.0741   1.1068   0.0456   1.1070   0.9489
##
## sigma^2 estimated as 0.0978:  log likelihood=-176.03
## AIC=366.05   AICc=366.21   BIC=397.84
##
## Training set error measures:
##                      ME      RMSE       MAE         MPE       MAPE      MASE
## Training set 0.02538115 0.3084853 0.2415861 0.006905657 0.06861436 0.1570195
##                      ACF1
## Training set -0.003259808
```

```
Prediction1<-forecast(co2_arima,h=12*10)
```

Giving prediction for 2025 from January to December answered the second question.

```r
window(Prediction1$mean, start = 2025, end=2025.999 )
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2025 425.6857 427.3919 428.0267 427.2647 425.5625 423.4158 421.8844 422.1132
##           Sep      Oct      Nov      Dec
## 2025 423.7810 425.1783 426.3125 427.1372
```