Library loading

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.2.2
```

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

```
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 4.2.2
```

Loading data from the Mauna Loa observatory

```
co2_monthly_full <- read.delim(file = 'co2_mm_mlo.txt', comment.char = '#', header = F, sep = '')
co2_monthly_full <- co2_monthly_full[, c(1, 2, 4)]
names(co2_monthly_full) <- c('year', 'month', 'co2_concentration')
head(co2_monthly_full)
```

```
##   year month co2_concentration
## 1 1958     3            315.70
## 2 1958     4            317.45
## 3 1958     5            317.51
## 4 1958     6            317.24
## 5 1958     7            315.86
## 6 1958     8            314.93
```

Data cleanup and transformations

Let's first check if the dataset contains null values

```
which(is.na(co2_monthly_full))
```

```
## integer(0)
```

Perfect, we don't have any null data. This is explained in the comments of the dataset: > Missing months have been interpolated

It must be noted, however, that readings in 1958 (first year included in the dataset) and 2022 (ongoing year) are incomplete. These readings will be removed so as not to affect the analysis.

```
co2_monthly_full <- co2_monthly_full %>% filter(year != 2022 & year != 1958)
```

Now we can convert the "year" and "month" fields to date type, which should improve the readability of our plots in the future.The "day" field will be set to the start of the month by default.

```
co2_monthly_full$date <- as.Date(paste(co2_monthly_full$year, co2_monthly_full$month, 1, sep = '-'), fo
co2_monthly <- co2_monthly_full[, c('date', 'co2_concentration')]
head(co2_monthly)
```

```
##         date co2_concentration
## 1 1959-01-01            315.58
## 2 1959-02-01            316.48
## 3 1959-03-01            316.65
## 4 1959-04-01            317.72
## 5 1959-05-01            318.29
## 6 1959-06-01            318.15
```

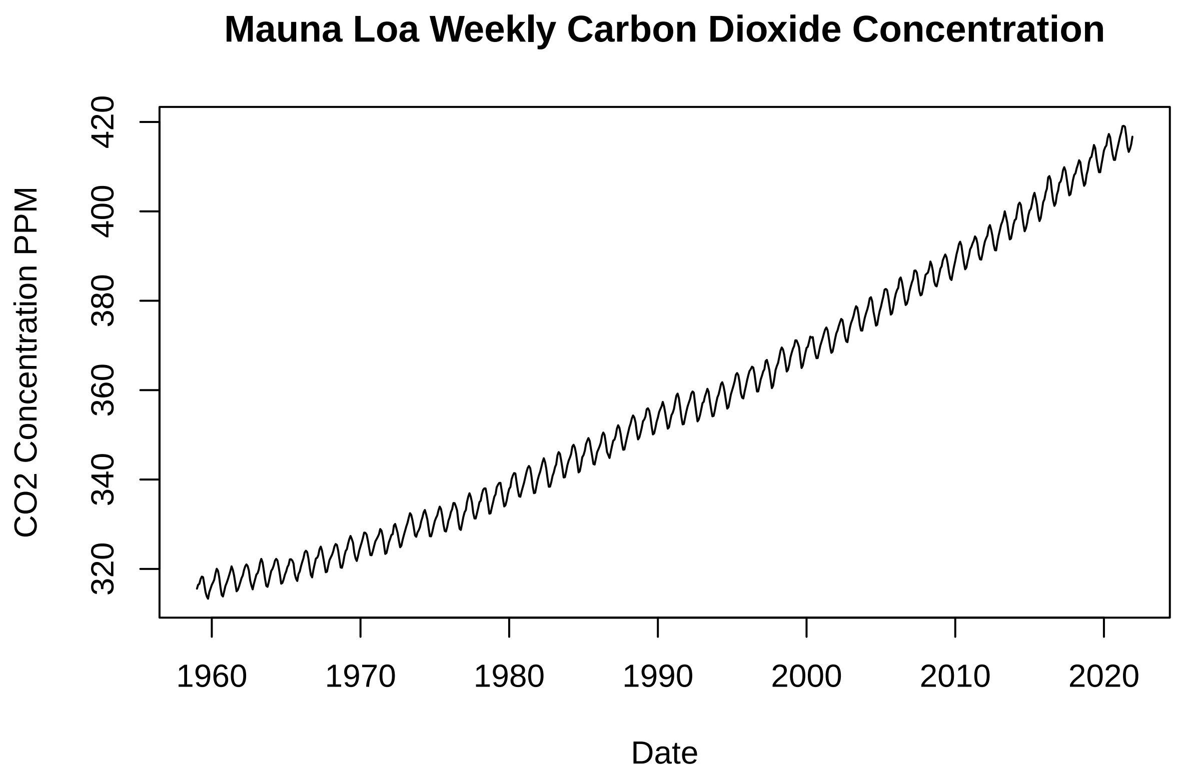
Finally, we'll transform the dataframe into a time series object, which makes it easier to do various plots with the loaded packages.

```
co2_ts <- ts(co2_monthly$co2_concentration, start = c(1959, 1), frequency = 12)
```

1. Exploratory analysis

Let's plot the data for the first time:

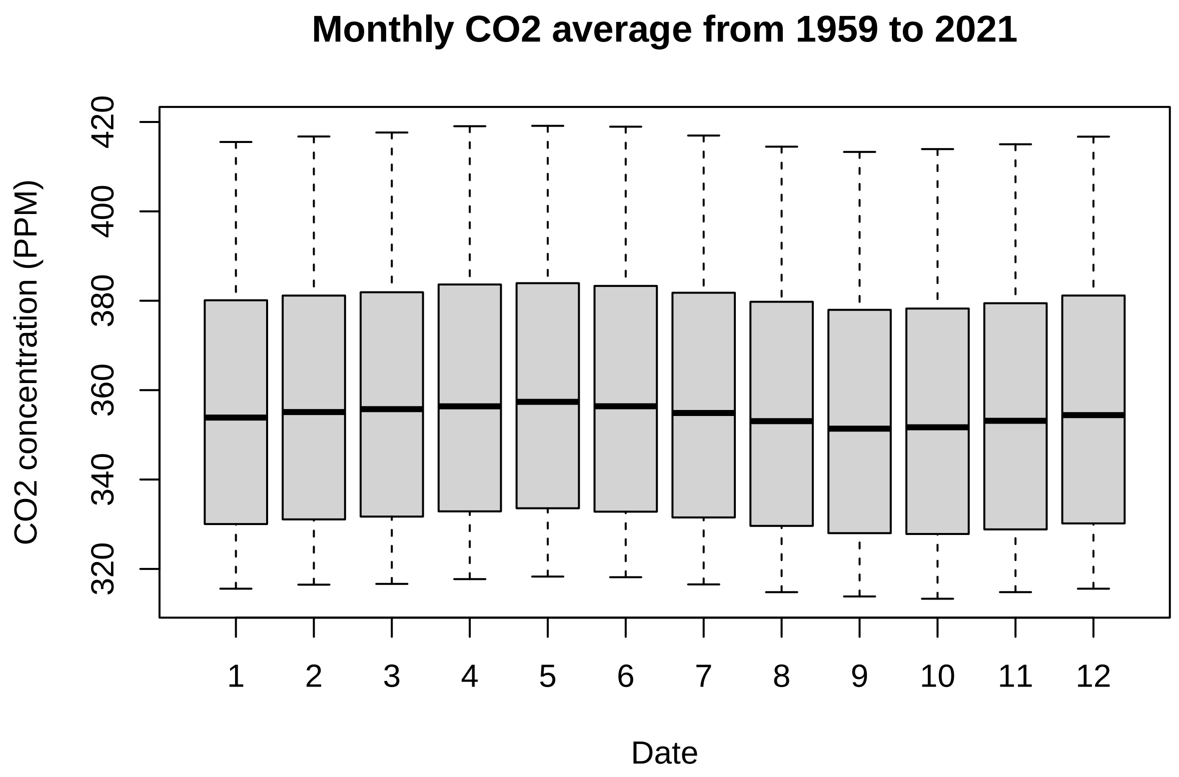
```
plot(
  co2_monthly$date,
  co2_monthly$co2_concentration,
  type = 'l',
  xlab = 'Date',
  ylab = 'CO2 Concentration PPM',
  main = 'Mauna Loa Weekly Carbon Dioxide Concentration'
)
```



From a purely visual analysis, it seems like the data presents both seasonality in its cycle and an upward trend. Let's first take a look at the seasonality:

Examining seasonality

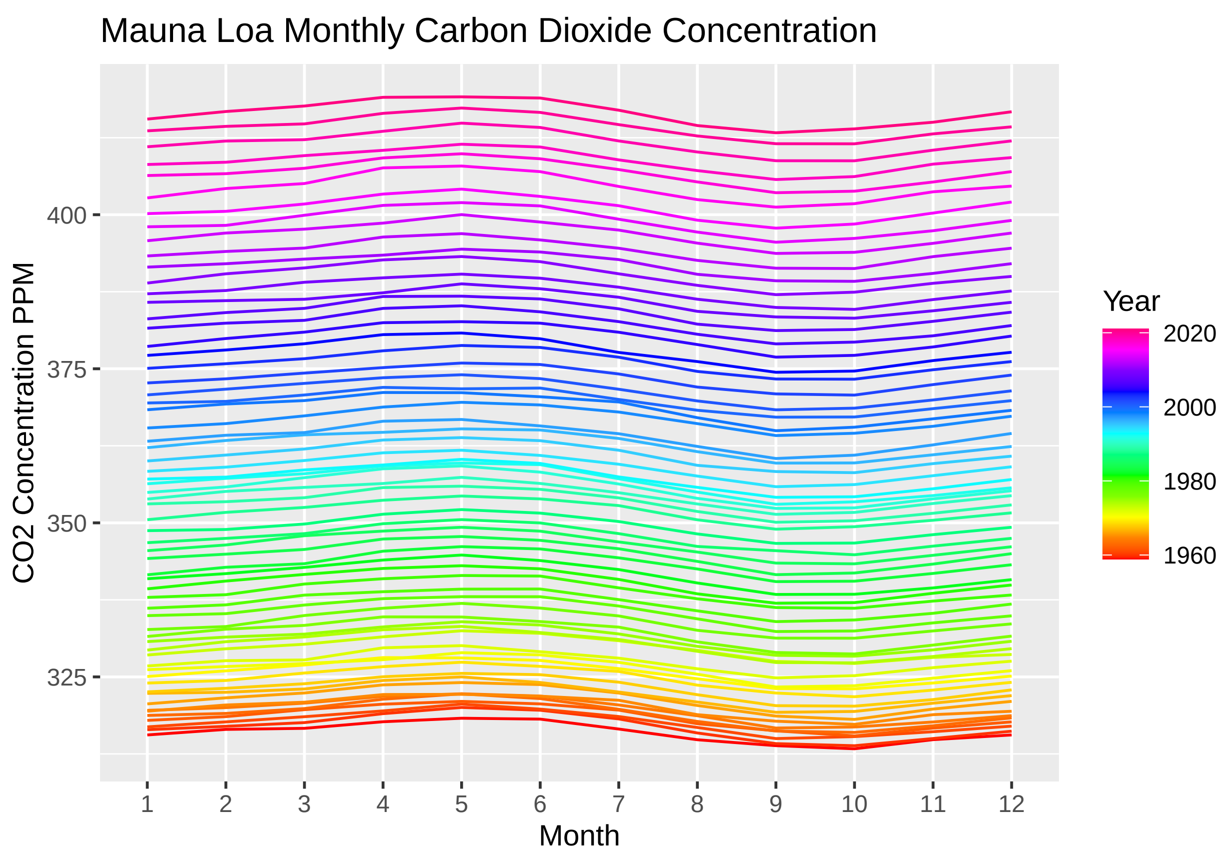
```
boxplot(co2_ts~cycle(co2_ts),xlab="Date", ylab = "CO2 concentration (PPM)",main ="Monthly CO2 average fr
```



Here we can see that the monthly concentrations show seasonality, with a high point around the months 4-5 and a low point around the months 9-10.

Let's take a look at the seasonality over the years:

```
ggplot(data = co2_monthly_full, aes(factor(month), co2_concentration, colour = year, group = year)) +
  geom_line() +
  xlab('Month') +
  ylab('CO2 Concentration PPM') +
  ggtitle('Mauna Loa Monthly Carbon Dioxide Concentration') +
  scale_color_gradientn('Year', colors = rainbow(length(unique(co2_monthly_full$month))))
```



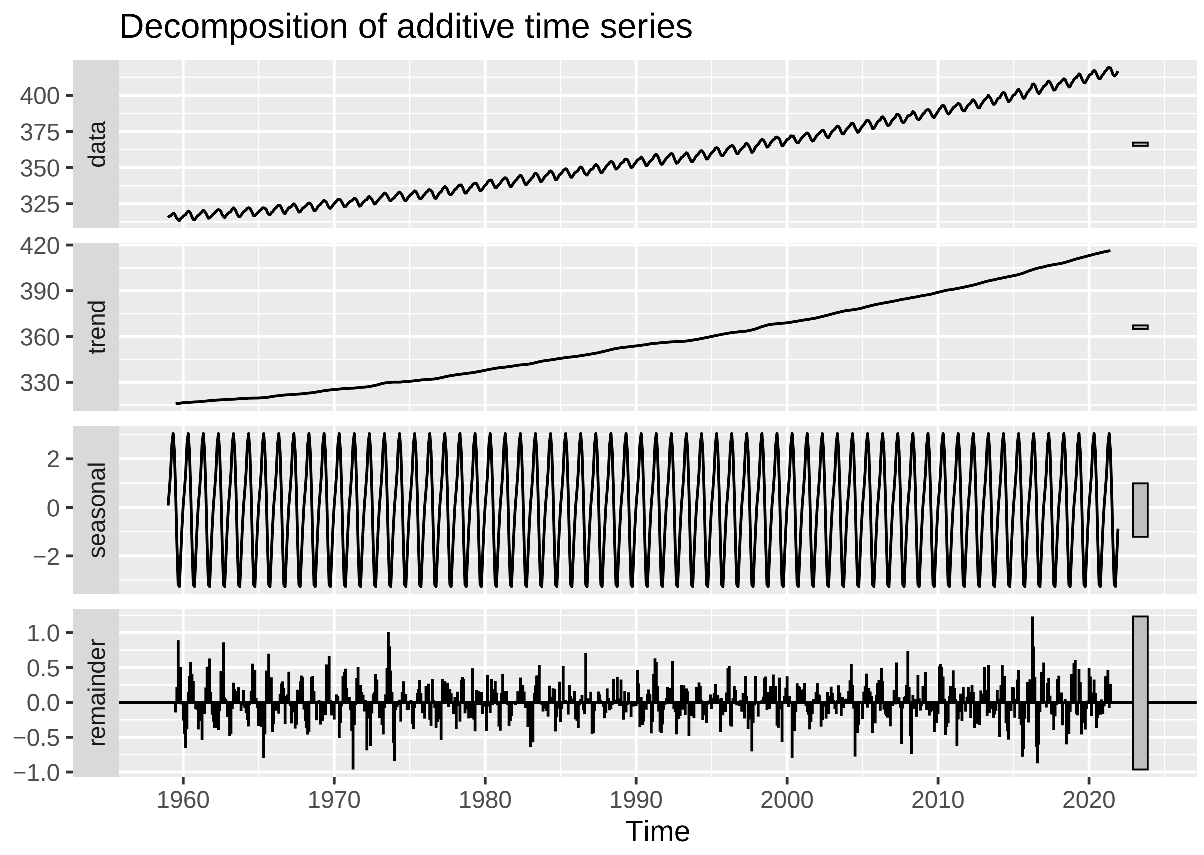
The results seem to be consistent over the years, with the seasonality being maintained through the upward trend in CO2 concentrations.

From these plots we could gather that: - The CO2 concentration shows a clear upward linear trend over the years - The monthly concentrations show seasonality, with a peak around the months 4-5 and a valley around the months 9-10.

Series decomposition

Let's take a better look at the time series decomposition:

```r
decomposeCO2 <- decompose(co2_ts,"additive")
autoplot(decomposeCO2)
```



Removing the trend seen above can be useful for further analysis. Let's check if differencing would be enough to make the time series stationary with adf.test():

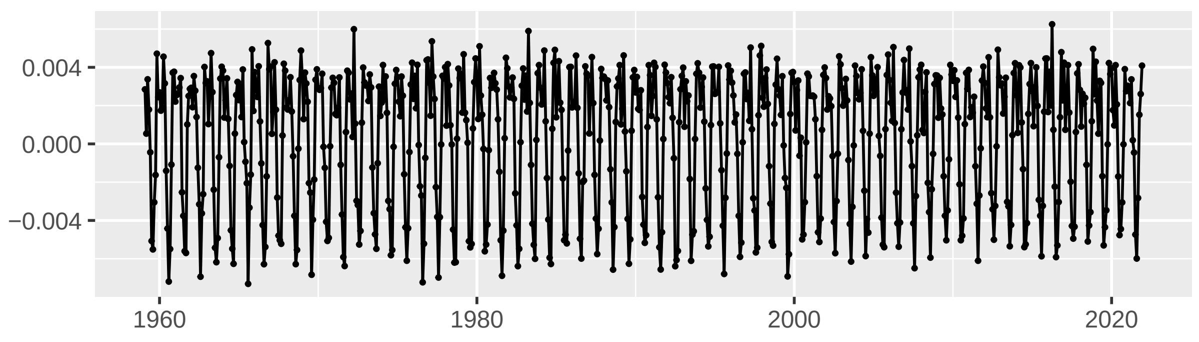
```
adf.test(diff(log(co2_ts)), alternative="stationary", k=0)
```

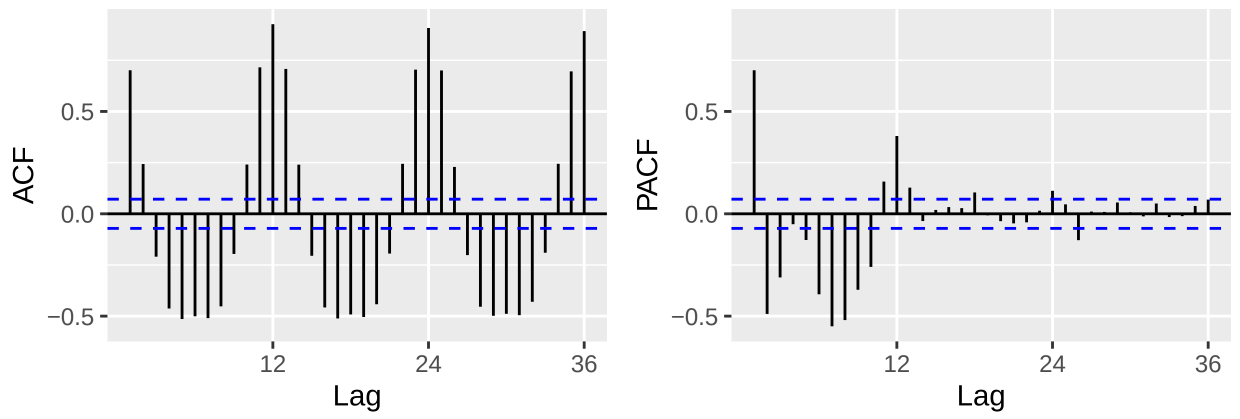
```
## Warning in adf.test(diff(log(co2_ts)), alternative = "stationary", k = 0): p-
## value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(log(co2_ts))
## Dickey-Fuller = -11.458, Lag order = 0, p-value = 0.01
## alternative hypothesis: stationary
```

This low P value confirms the alternative hypothesis (that is, the time series has become stationary after one stage of differencing); now let's go ahead and plot ACF:

```
diff(log(co2_ts)) %>% ggtsdisplay()
```





Making predictions

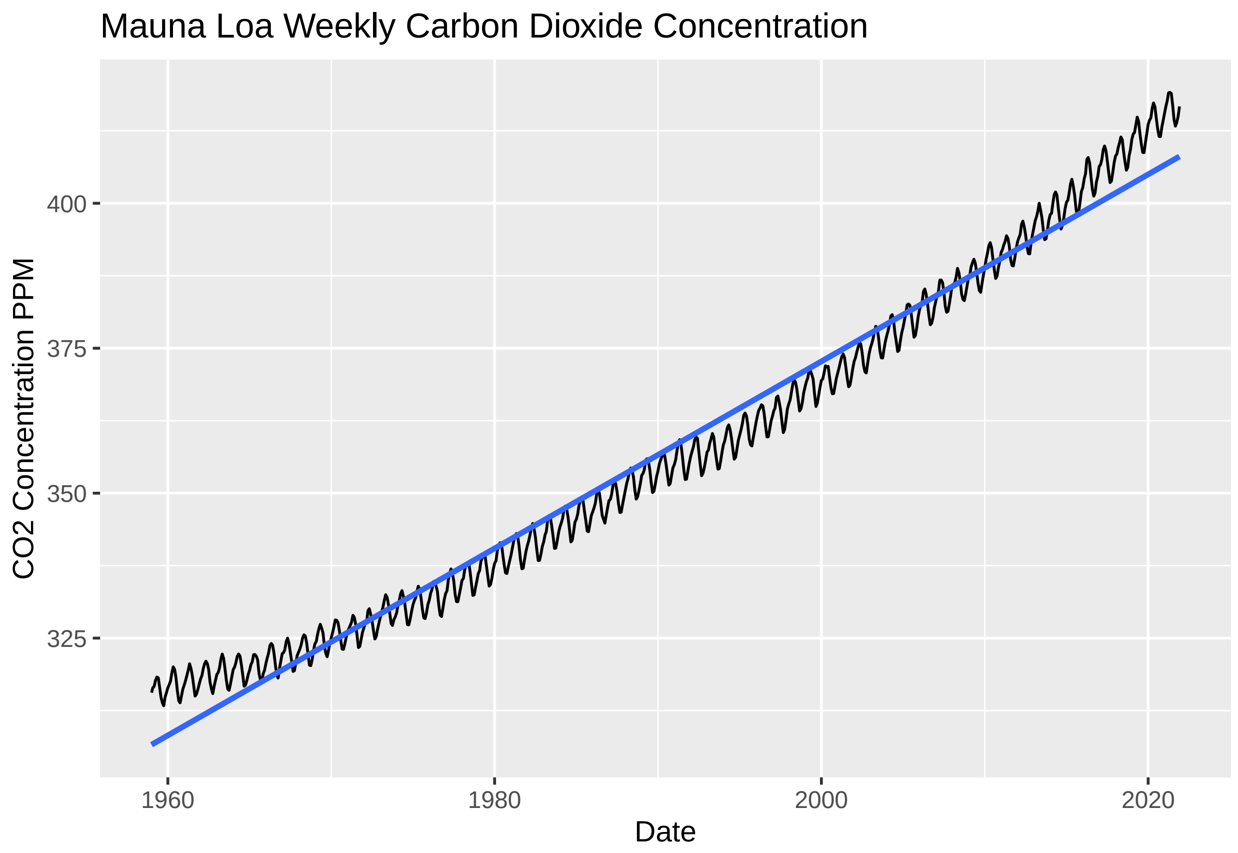
First attempt:

Let's first take a look at the P value for the trend, using the date as the independent variable.

```
simple_reg_co2 <- lm(co2_concentration ~ date, data = co2_monthly)
```

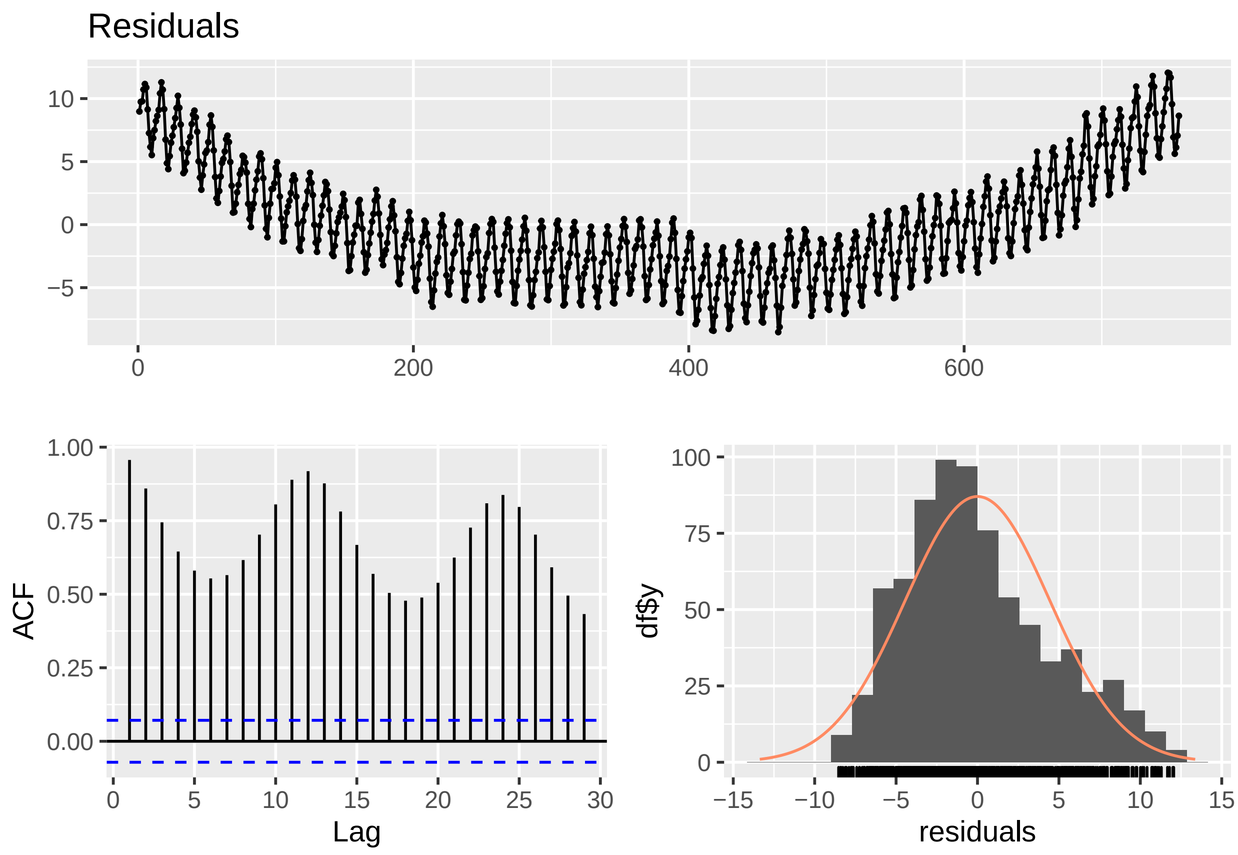
```
ggplot(data = co2_monthly, aes(x = date, y = co2_concentration)) +
  geom_line() +
  xlab('Date') +
  ylab('CO2 Concentration PPM') +
  ggtitle('Mauna Loa Weekly Carbon Dioxide Concentration') +
  stat_smooth(method = "lm", se = FALSE)
```

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



Visually, we can see that this model might not be perfect as it cannot follow the seasonality. I suspect the errors will have a high autocorrelation because of that:

```
checkresiduals(simple_reg_co2)
```



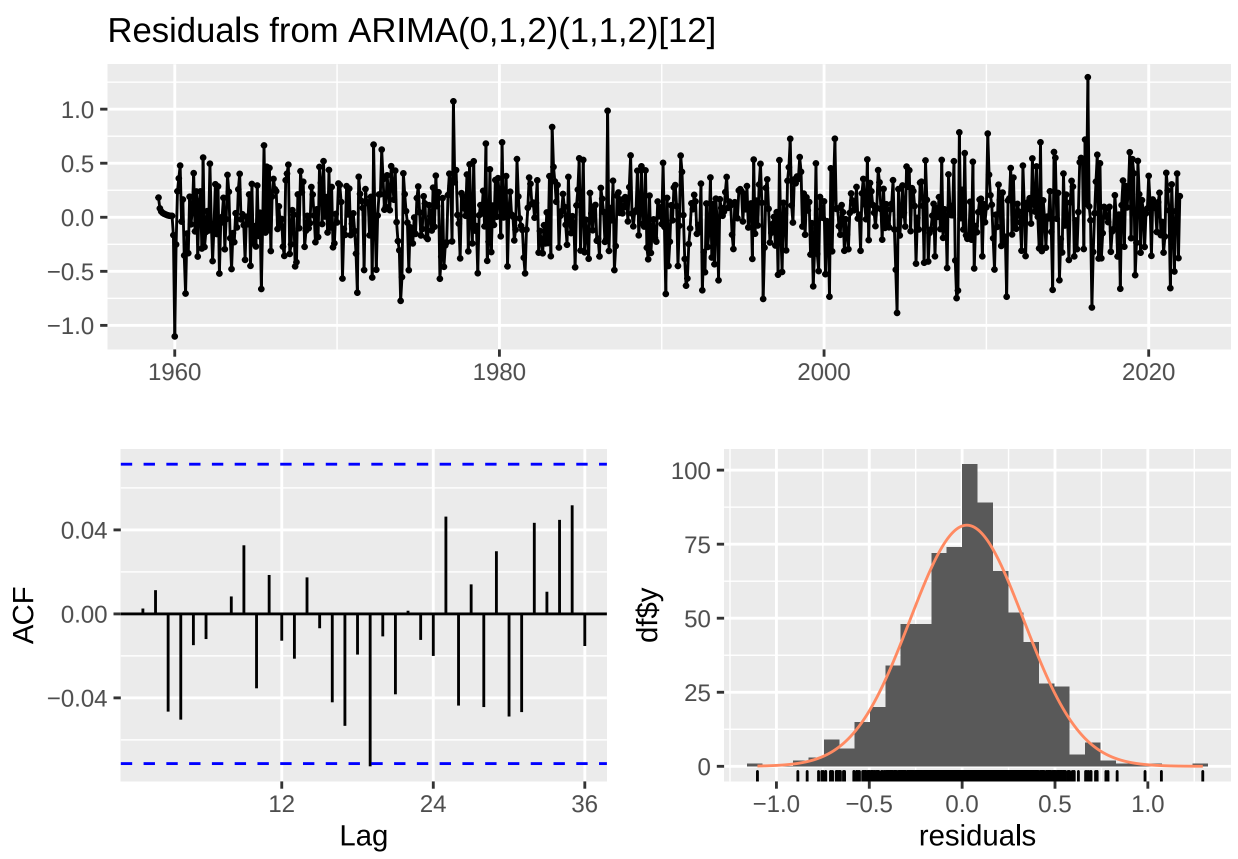
```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 742.63, df = 10, p-value < 2.2e-16
```

As expected, the autocorrelation of errors is pretty high, which is not what we want from a forecasting model. Let's try another one.

Second attempt: ARIMA

```
arima_co2 <- auto.arima(co2_ts)
```

```
checkresiduals(arima_co2)
```



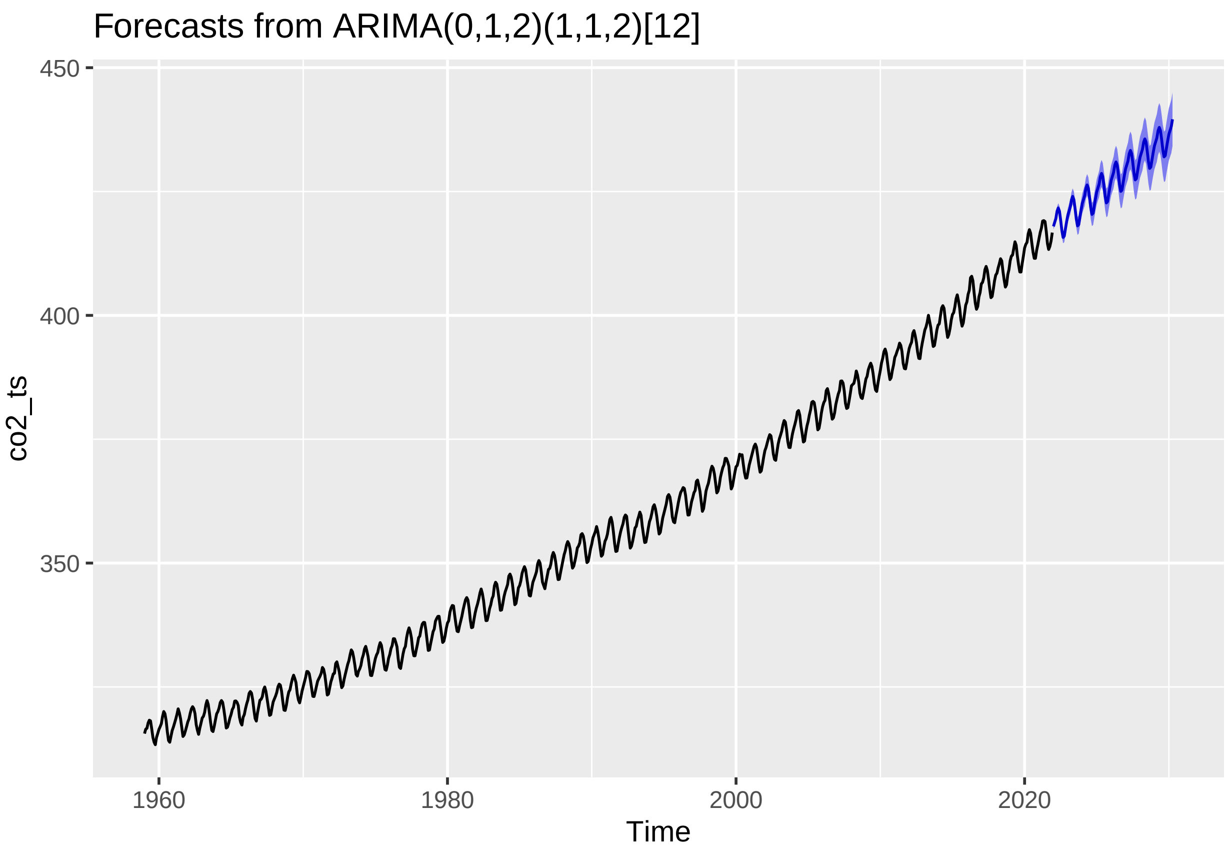
```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,2)(1,1,2)[12]
## Q* = 16.439, df = 19, p-value = 0.6278
##
## Model df: 5.   Total lags used: 24
```

Much better. ARIMA was able to deal with the correlation much more; now we can see that the errors follow a noise pattern, which is preferable for a model.

Sample forecast:

Now we can use the forecast function to plot a forecast on the horizon of 100 months, with a confidence interval of 95%.

```
arima_forecast_co2 <- forecast(arima_co2, level = c(95), h = 100)
autoplot(arima_forecast_co2)
```


Forecasts from ARIMA(0,1,2)(1,1,2)[12]
450
400
350
co2_ts
1960
1980
2000
2020
Time