Group Assignment Group F

Sandra Alemayehu, Frederico Andrade, Luigi Giulio Grandi, Usama Kham, Joe Mehanna, Marcela Zablah

17/12/2019

Analysis of the weather in Madrid between 2011 and 2016

Importing and transforming the files

The first step to be performed involved importing the csv and the xlsx on R.

The xlsx had to be read with the following `read_excel()` , hence it was necessary to install the xlsx package `library(readxl)` . Below are the packages needed in order to perfrom the stps undertaken in this analysis:

```
library(stringr)
library(readxl)
library(lubridate)
library(tidyverse)
library(ggplot2)
library(corrplot)
library(data.table)
library(corrplot)
library(reshape2)
```

Furthermore, the working directory was set to the folder where all the daily data was saved, by using `setwd()` .

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

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

```
## -- Attaching packages ------------------------------------------------------------------------
----------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v forcats 0.4.0
## v readr   1.3.1
```

```
## -- Conflicts ---------------------------------------------------------------------------------
----------------------------- tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
## corrplot 0.84 loaded
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday,
##     week, yday, year
```

```
##
## Attaching package: 'reshape2'
```

```
## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

When loading a package, if functions in the library share names with any base R functions or any other functions in loaded packages they warn of the conflict and they replace them as the default function that will be called when their names are written. Functions with names filter and lag exist in base R which means they will always conflict with dplyr functions. These warnings do not affetct the script.

Once the packages were ready it was possible to import the required documents, starting from the xlsx. The structures of the data frame was checked, in order to understand if changes in the file were required.

```
weather <- read_excel('C:/Users/Luigi Giulio Grandi/Documents/Luigi Giulio Grandi/IE/MBD/Term I/Mod
ules/R/Assignment/Group Assignment/weather.xlsx')
str(weather)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    2192 obs. of  7 variables:
##  $ date          : POSIXct, format: "2011-01-01" "2011-01-02" ...
##  $ temp_avg      : num  8.3 8.6 4.2 6.5 8.9 12.2 10.9 9.8 8.4 6.9 ...
##  $ temp_max      : num  13 13 9.4 8 10 15 13 13 11 8.3 ...
##  $ temp_min      : num  3 4 -1.6 4.1 6.3 8.9 8.7 7.6 6.6 5.2 ...
##  $ precipitation : num  0 0 0 0 0 ...
##  $ humidity      : num  84 81 86 93 90 87 81 88 92 91 ...
##  $ wind_avg_speed: num  5.2 5.4 3.5 6.3 10.4 15.7 15.6 14.3 6.5 7.4 ...
```

From this step it was possible to notice that the date in this file was not in the desired fomrat, hence `as.Date()` was applied to the date column.

```
weather$date <- as.Date(weather$date)
str(weather)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    2192 obs. of  7 variables:
##  $ date          : Date, format: "2011-01-01" "2011-01-02" ...
##  $ temp_avg      : num  8.3 8.6 4.2 6.5 8.9 12.2 10.9 9.8 8.4 6.9 ...
##  $ temp_max      : num  13 13 9.4 8 10 15 13 13 11 8.3 ...
##  $ temp_min      : num  3 4 -1.6 4.1 6.3 8.9 8.7 7.6 6.6 5.2 ...
##  $ precipitation : num  0 0 0 0 0 ...
##  $ humidity      : num  84 81 86 93 90 87 81 88 92 91 ...
##  $ wind_avg_speed: num  5.2 5.4 3.5 6.3 10.4 15.7 15.6 14.3 6.5 7.4 ...
```

Then the daily data was imported. These documents, unlike weather, were csv files, hence it was sufficient to apply the `read.csv()` command in order to import them. However, as they were all saved in a single folder some more commands steps were applied:

As the working directory had already been set to the correct folder, it was possible to use `lapply()` , after having saved the the files as a list.

```
temp <- list.files()
myfiles <- lapply(temp, read.csv)
str(myfiles)
```

we used an automated substring that returns 'year_month' using the length of each element in 'temp' (list of all csv file names) and removes the last 4 characters '.csv'.

These values where stored under 'ID' which was then used in a loop to paste all corresponding table rows of the list of 72 csv files

To do so, some changes had to be performed, the lentgh of the names of the elements in the list, `str_length()` , and the names of each element of this list had to be recorded. as some months conists of 2 digits and some other 1, a for loop had to be performed, in order to get the correct amount of characters to be read.

```
for (i in seq_along(temp)){
  ID[i] <- substr(temp[i], 13, string_length[i] - 4)
}
```

Furthermore from this date fromat some alterations, such as changing `_` to `-` had to be performed, in order to have a format that could allow to convert the value into a date, by using `gsub('_', '-' , ID)` .

Now it was possible to insert these values into the list of data frames, by performing the following:

```
for (i in seq_along(myfiles)) {
  myfiles[[i]]$Date <- rep(ID[i], nrow(myfiles[[i]]))
}
```

Now that the month and the year of the values have been added to the data frame, it was possible to bind all the data frames together, by `data.table::rbindlist(myfiles, use.names = TRUE)` .

The result from the previous operations resulted in this dataframe with the following strucure:

```
## Classes 'data.table' and 'data.frame':   6471098 obs. of  6 variables:
##  $ day      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ hour     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ station  : int  28079004 28079008 28079017 28079018 28079024 28079035 28079036 28079038 28079
040 28079057 ...
##  $ parameter: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ value    : num  6 12 12 10 7 11 22 13 9 12 ...
##  $ Date     : chr  "11-1" "11-1" "11-1" "11-1" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

All the redundant colums such as ‘day’, ‘Date’ and ‘station’ were removed from the data frame. The latter one was removed because all of the analysis were performed at a city level rather than a station level. Furthermore, a new column ‘date’ was added. This column involved the unification of the day, month and year alltogether. the year was modified, in order to record the ‘20’, for example, 11 became 2011. the values in this colun were saved in a date format. Only 1 command was enough to perform this:
`daily$date <- as.Date(paste0('20',daily$Date, '-', daily$day))`

This was the strucutre of the resulting data frame:

```
## Classes 'data.table' and 'data.frame':   6471098 obs. of  4 variables:
##  $ hour     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ parameter: Factor w/ 12 levels "1","6","7","8",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ value    : num  6 12 12 10 7 11 22 13 9 12 ...
##  $ date     : Date, format: "2011-01-01" "2011-01-01" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

As the data frame parameters were saved by a code, in the original files, they were changed in order to show their formula rather than the code. A data frame matching the code to its fromula was created according to parameters.png, then a left join was performed between the original data frame and these values in order to add the correct formula to the data frame.

```
formula <- c('SO_2', 'CO', 'NO', 'NO_2','PM2.5','PM10', 'NOx', 'O_3', 'TOL','BEN', 'EBE', 'MXY', 'P
XY','OXY', 'TCH', 'CH4', 'NMHC')
parameter <- c('1','6','7','8','9','10','12','14','20','30','35','37','38','39','42','43','44')
parameters_1 <- as.data.frame(t(rbind(parameter,formula)))
str(parameters_1)
```

```
## 'data.frame':    17 obs. of  2 variables:
##  $ parameter: Factor w/ 17 levels "1","10","12",..: 1 14 15 16 17 2 3 4 5 6 ...
##  $ formula  : Factor w/ 17 levels "BEN","CH4","CO",..: 15 3 7 8 13 12 9 10 17 1 ...
```

```
temp_2 <- merge(daily, parameters_1 , by = 'parameter', all.x = T)
temp_2$parameter <- NULL
str(temp_2)
```

```
## Classes 'data.table' and 'data.frame':   6471098 obs. of  4 variables:
##  $ hour   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ value  : num  6 12 12 10 7 11 22 13 9 12 ...
##  $ date   : Date, format: "2011-01-01" "2011-01-01" ...
##  $ formula: Factor w/ 17 levels "BEN","CH4","CO",..: 15 15 15 15 15 15 15 15 15 15 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

Computing a data frame returning the daily average values of each parameter

In order to get a better visulasiation of the data frame, and to grasp a better understanding, the parameters column was transposed in order to get the parameters formula as columns, with values to be recorded as the daily averages. This also allowed to remove the 'hour' column from the data frame.

```
library(tidyverse)
daily_mean <- temp_2 %>%
  group_by (date, formula) %>%
  summarize (averaged_value = mean(value))

daily_mean_parameter <- as.data.frame(tidyr::spread(daily_mean,formula,averaged_value))
str(daily_mean_parameter)
```

```
## 'data.frame':    2192 obs. of  13 variables:
##  $ date : Date, format: "2011-01-01" "2011-01-02" ...
##  $ BEN  : num  0.765 0.869 1.176 0.857 0.856 ...
##  $ CO   : num  0.372 0.453 0.532 0.367 0.395 ...
##  $ EBE  : num  0.595 0.722 1.047 0.788 1.074 ...
##  $ NMHC : num  0.187 0.208 0.246 0.196 0.196 ...
##  $ NO   : num  16.5 28.8 71.1 27.8 36 ...
##  $ NO_2 : num  41.5 48.5 63.6 46.3 51.5 ...
##  $ O_3  : num  20.47 15.56 9.45 13.34 10.88 ...
##  $ PM10 : num  13.8 13.7 21.6 14.4 15.6 ...
##  $ PM2.5: num  9.36 9.08 11.94 9.4 10.51 ...
##  $ SO_2 : num  10.71 11.93 11.91 8.84 9.51 ...
##  $ TCH  : num  1.43 1.62 1.62 1.46 1.52 ...
##  $ TOL  : num  1.66 2.14 3.56 2.4 3.19 ...
```

Merging weather with means & adding the seasons

After performing all the transormation to the hourly data and making it daily, it was time to include the weather data from weather.xlsx, again this was performed by merging the two data frames by the unique 'date' value, and the result was the following:

```
daily_data <- merge(daily_mean_parameter, weather, by = 'date', sort = T)
str(daily_data)
```

```
## 'data.frame':    2192 obs. of  19 variables:
##  $ date          : Date, format: "2011-01-01" "2011-01-02" ...
##  $ BEN           : num  0.765 0.869 1.176 0.857 0.856 ...
##  $ CO            : num  0.372 0.453 0.532 0.367 0.395 ...
##  $ EBE           : num  0.595 0.722 1.047 0.788 1.074 ...
##  $ NMHC          : num  0.187 0.208 0.246 0.196 0.196 ...
##  $ NO            : num  16.5 28.8 71.1 27.8 36 ...
##  $ NO_2          : num  41.5 48.5 63.6 46.3 51.5 ...
##  $ O_3           : num  20.47 15.56 9.45 13.34 10.88 ...
##  $ PM10          : num  13.8 13.7 21.6 14.4 15.6 ...
##  $ PM2.5         : num  9.36 9.08 11.94 9.4 10.51 ...
##  $ SO_2          : num  10.71 11.93 11.91 8.84 9.51 ...
##  $ TCH           : num  1.43 1.62 1.62 1.46 1.52 ...
##  $ TOL           : num  1.66 2.14 3.56 2.4 3.19 ...
##  $ temp_avg      : num  8.3 8.6 4.2 6.5 8.9 12.2 10.9 9.8 8.4 6.9 ...
##  $ temp_max      : num  13 13 9.4 8 10 15 13 13 11 8.3 ...
##  $ temp_min      : num  3 4 -1.6 4.1 6.3 8.9 8.7 7.6 6.6 5.2 ...
##  $ precipitation : num  0 0 0 0 0 ...
##  $ humidity      : num  84 81 86 93 90 87 81 88 92 91 ...
##  $ wind_avg_speed: num  5.2 5.4 3.5 6.3 10.4 15.7 15.6 14.3 6.5 7.4 ...
```

At this point the data frame was complete of all the information needed to perform the analysis needed. However, for better understanding, and more insightful analysis, the corresponding seasons were added in a new column.

```r
seasons = function(x){
  if(x %in% 3:5) return("Spring")
  if(x %in% 6:8) return("Summer")
  if(x %in% 9:11) return("Fall")
  if(x %in% c(12,1,2)) return("Winter")

}

daily_data$season <-sapply(month(daily_data$date), seasons)
str(daily_data)
```

```
## 'data.frame':    2192 obs. of  20 variables:
##  $ date          : Date, format: "2011-01-01" "2011-01-02" ...
##  $ BEN           : num  0.765 0.869 1.176 0.857 0.856 ...
##  $ CO            : num  0.372 0.453 0.532 0.367 0.395 ...
##  $ EBE           : num  0.595 0.722 1.047 0.788 1.074 ...
##  $ NMHC          : num  0.187 0.208 0.246 0.196 0.196 ...
##  $ NO            : num  16.5 28.8 71.1 27.8 36 ...
##  $ NO_2          : num  41.5 48.5 63.6 46.3 51.5 ...
##  $ O_3           : num  20.47 15.56 9.45 13.34 10.88 ...
##  $ PM10          : num  13.8 13.7 21.6 14.4 15.6 ...
##  $ PM2.5         : num  9.36 9.08 11.94 9.4 10.51 ...
##  $ SO_2          : num  10.71 11.93 11.91 8.84 9.51 ...
##  $ TCH           : num  1.43 1.62 1.62 1.46 1.52 ...
##  $ TOL           : num  1.66 2.14 3.56 2.4 3.19 ...
##  $ temp_avg      : num  8.3 8.6 4.2 6.5 8.9 12.2 10.9 9.8 8.4 6.9 ...
##  $ temp_max      : num  13 13 9.4 8 10 15 13 13 11 8.3 ...
##  $ temp_min      : num  3 4 -1.6 4.1 6.3 8.9 8.7 7.6 6.6 5.2 ...
##  $ precipitation : num  0 0 0 0 0 ...
##  $ humidity      : num  84 81 86 93 90 87 81 88 92 91 ...
##  $ wind_avg_speed: num  5.2 5.4 3.5 6.3 10.4 15.7 15.6 14.3 6.5 7.4 ...
##  $ season        : chr  "Winter" "Winter" "Winter" "Winter" ...
```

Before plotting the required graphs, all the NAs in the data set had to be changed to a numeric value, and the correlation analysis was performed to better understand which values are correlated. At this stage only the parameters requested by the assignment were considered.

First were identified the columns that contained NAs,

```
requested_parameters <- c('NO_2', 'SO_2', 'O_3', 'PM2.5', 'temp_avg','precipitation','wind_avg_spee
d')
daily_final <- daily_data[ ,c('date',requested_parameters,'season')]

colnames(daily_final)[colSums(is.na(daily_final)) > 0]
```

```
## [1] "SO_2"  "PM2.5"
```

Then, the NAs in these 2 columns where replaced by the median of each columns.

```
total_NAs<-apply(daily_final,2, function(var) { return(sum(is.na(var)))})
total_NAs
```

```
##          date          NO_2          SO_2           O_3         PM2.5 
##             0             0             2             0            20 
##      temp_avg  precipitation wind_avg_speed        season 
##             0             0             0             0
```

```
for (i in seq_along(daily_final)){
  Sys.sleep(0.1)
  print(colnames(daily_data[i]))
  for (j in 1:nrow(daily_final)) {
    if (is.na(daily_final[j,i])){
      daily_final[j,i] <- median(daily_final[,i], na.rm = T)
    }
  }
}
```

```
## [1] "date"
## [1] "BEN"
## [1] "CO"
## [1] "EBE"
## [1] "NMHC"
## [1] "NO"
## [1] "NO_2"
## [1] "O_3"
## [1] "PM10"
```

```
sum(is.na(daily_final))
```

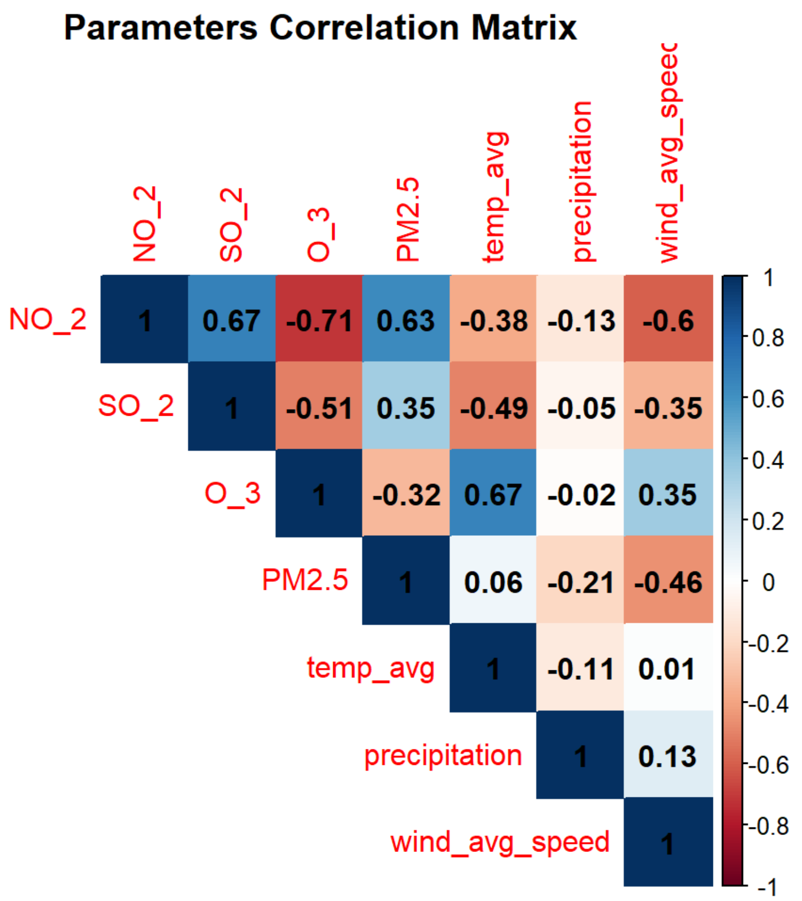
```
## [1] 0
```

Plotting

As all the data has been cleaned, it is now possible to run a correlation analysis between the different parameters analysed.

```
param <- subset(daily_final, select = c(2:8))

cor_matrix <- cor(param)
cor_plot <- corrplot(cor_matrix,method="color", type = "upper", title = "Parameters Correlation Mat
rix", addCoef.col = "black", mar=c(0,0,1,0))
```

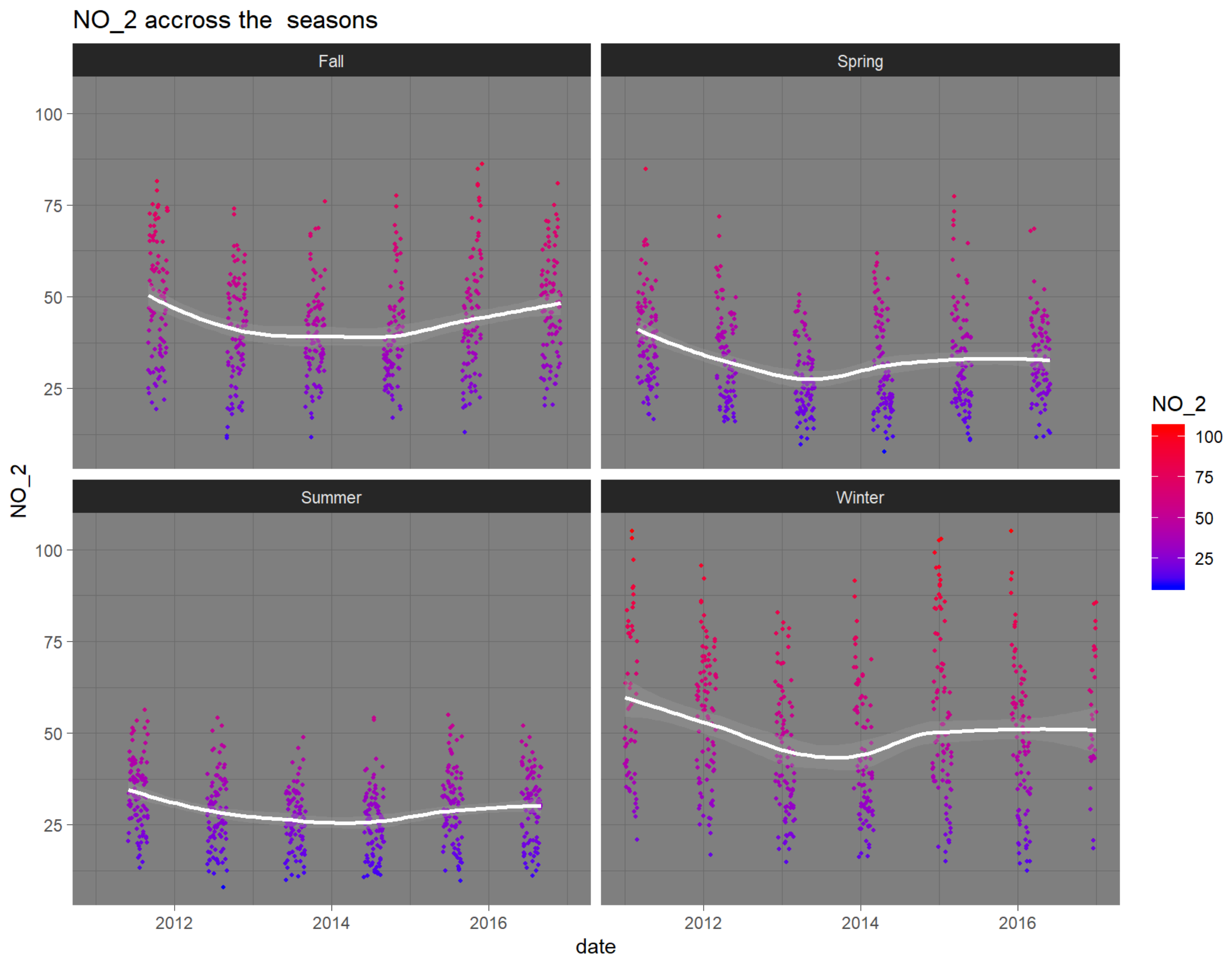


The correlation matrix allowed to quickly notice that the strongest correlation was among NO_2 and O_3, where the correlation is close to negative 1. However, NO_2 also has strong relationships with other parameters, namely, SO_2, PM2.5, while O_3 is also highly correlated to the average daily temperature.

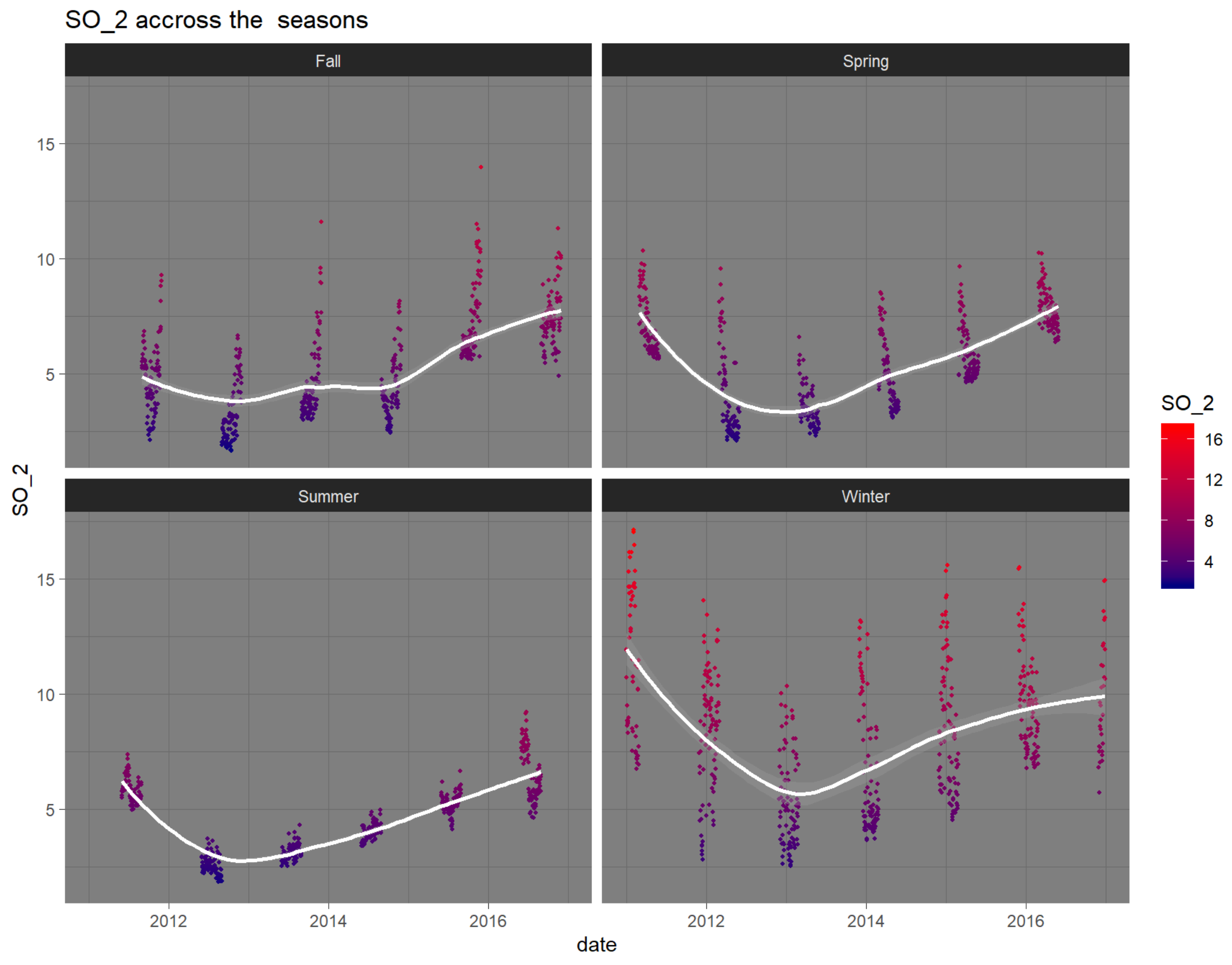
Then it was possible to proceed to plotting. The following ggplot was applied to the different parameters:

```
print(ggplot(daily_final, aes(date,<parameter>,colour = <parameter>))+
  geom_point(size = 0.75)+
  scale_color_gradient(low = "blue", high = "red") +
  theme_dark()+
  facet_wrap(vars(season))+
  labs(title = '<parameter> accross the  seasons')+
  geom_smooth(colour = 'white', method = loess, formula = y ~ x)
```

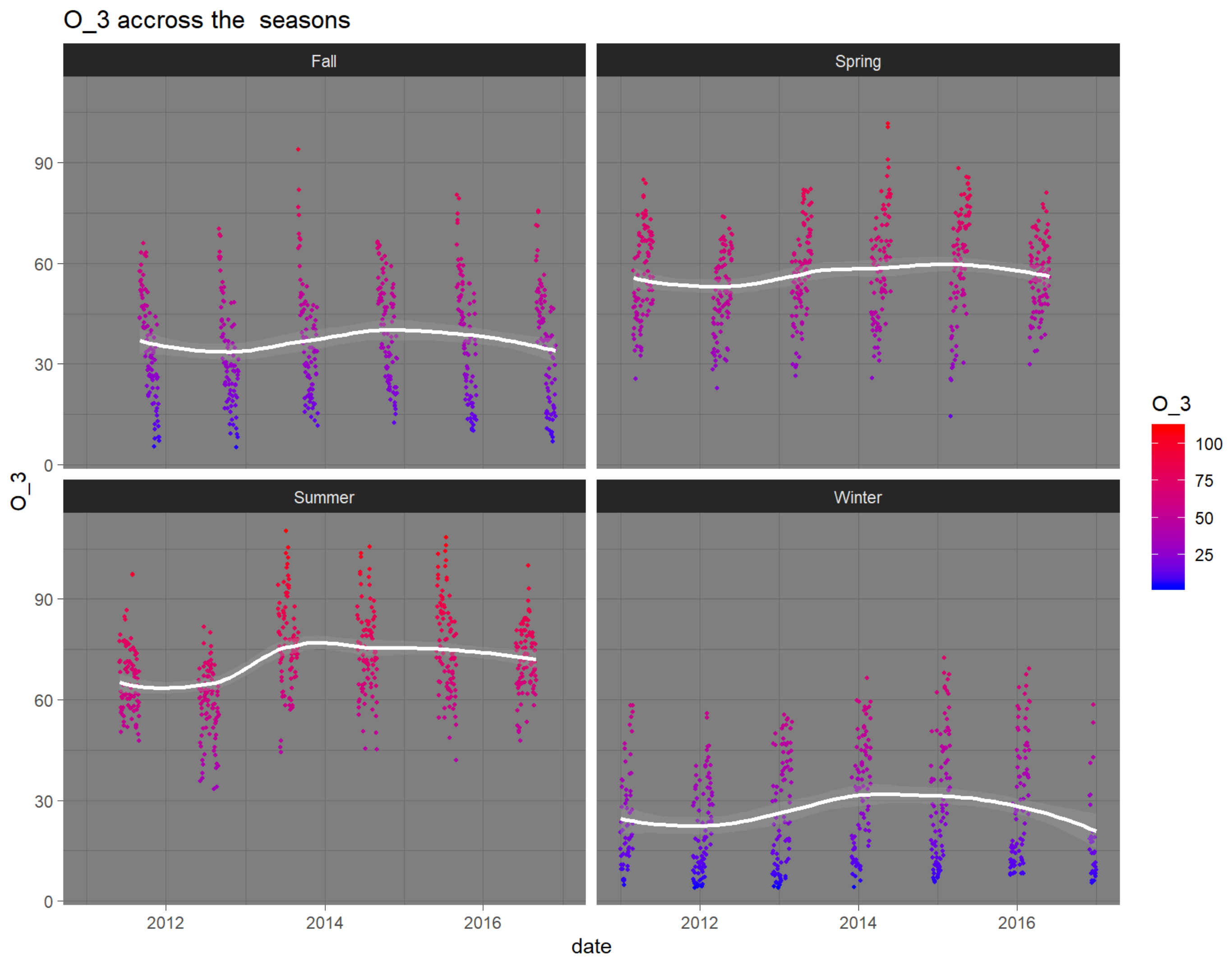
When looking at the differnet plots it was possible to draw some conclusions for all the different parameters analysed. For the following analysis only the requested values were considered. Furthermore, all the values in the plots are daily average vales, claculated previously.



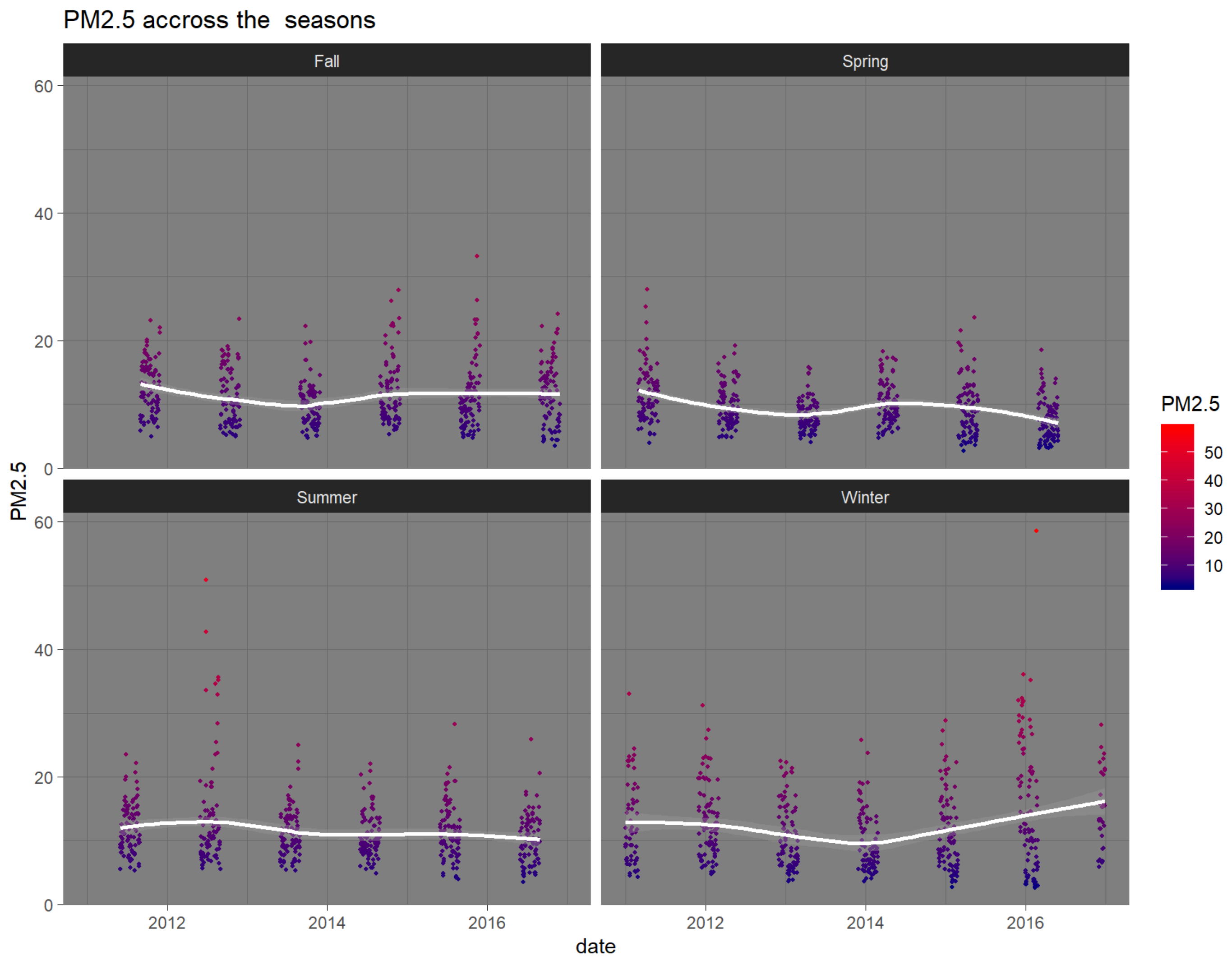
NO_2 tend to be higher during colder seasons, the higher the temperature the lower the values of NO_2, so on average, NO_2 is higher during Winter and fall. In this timeline, it is possible to notice how, in the years, the average value of NO_2 has been decreasing in the first half of the plot, followed by an increase in the second half.



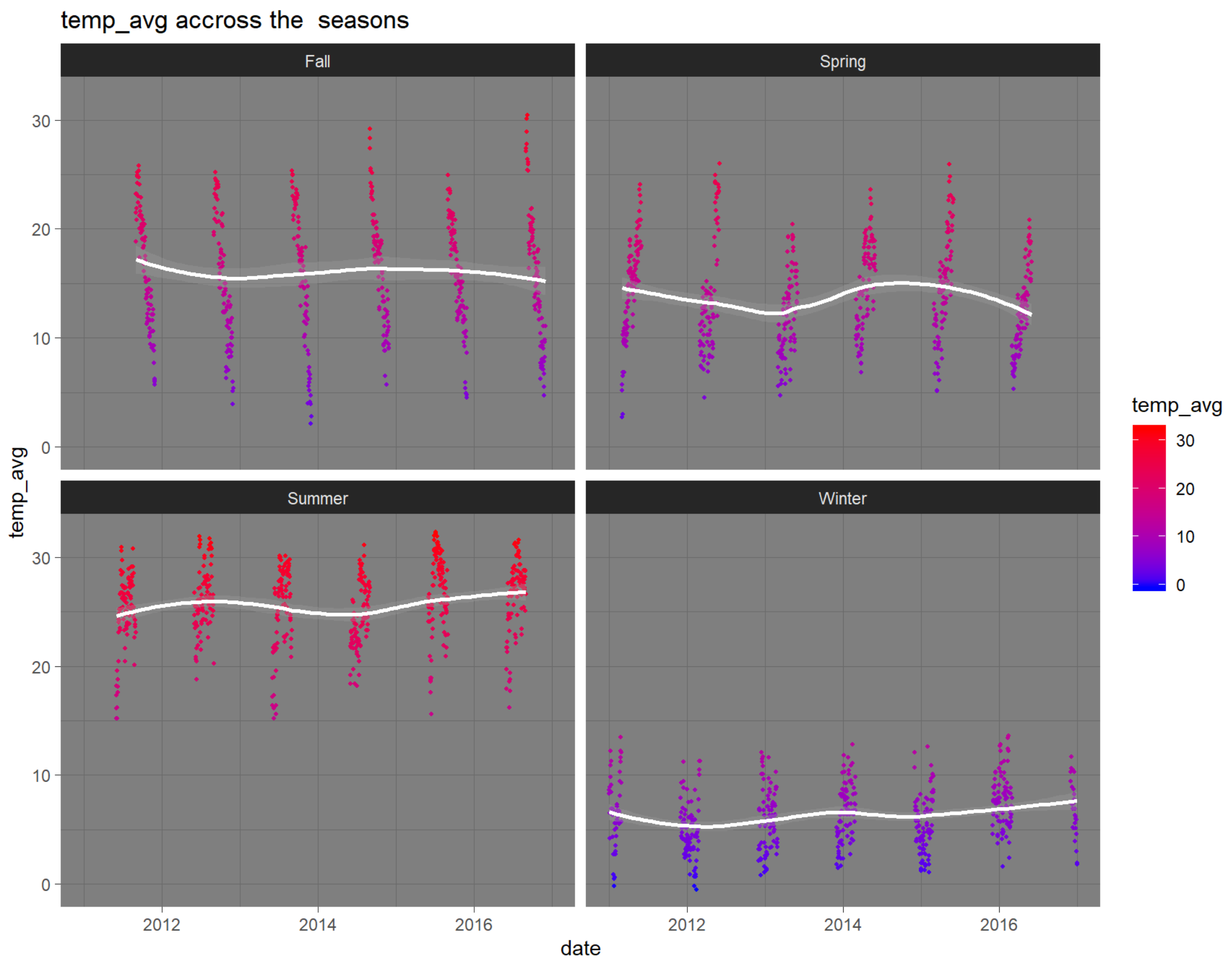
This is also applicable to **SO_2**, however, from this plot it is also possible to notice that the dispersion of the average values odf SO_2 is lower during hot seasons. In the first 2 years anlysed, the values of SO_2 recorded, decreased. However, as of 2013, this average has been increasing in all the different seasons.



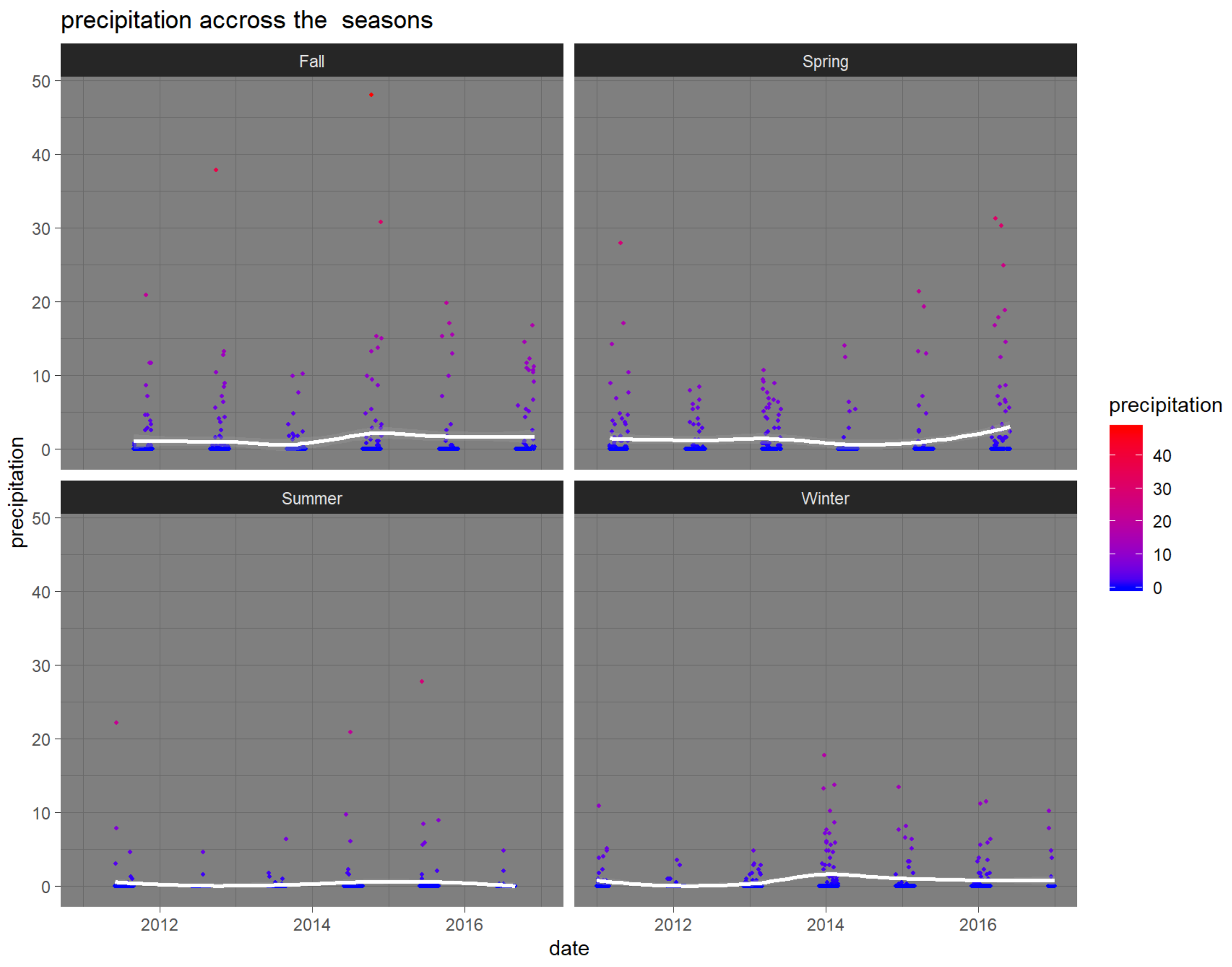
As observed in the correlation matrix, **O_3** is inversely proportional to the 2 previosly analysed parameters. In this case it is higher during hot seasons, summer and spring, compared to the others. Some fluctuations were recorded in the different years, however, the averages have been constant in the period in question.



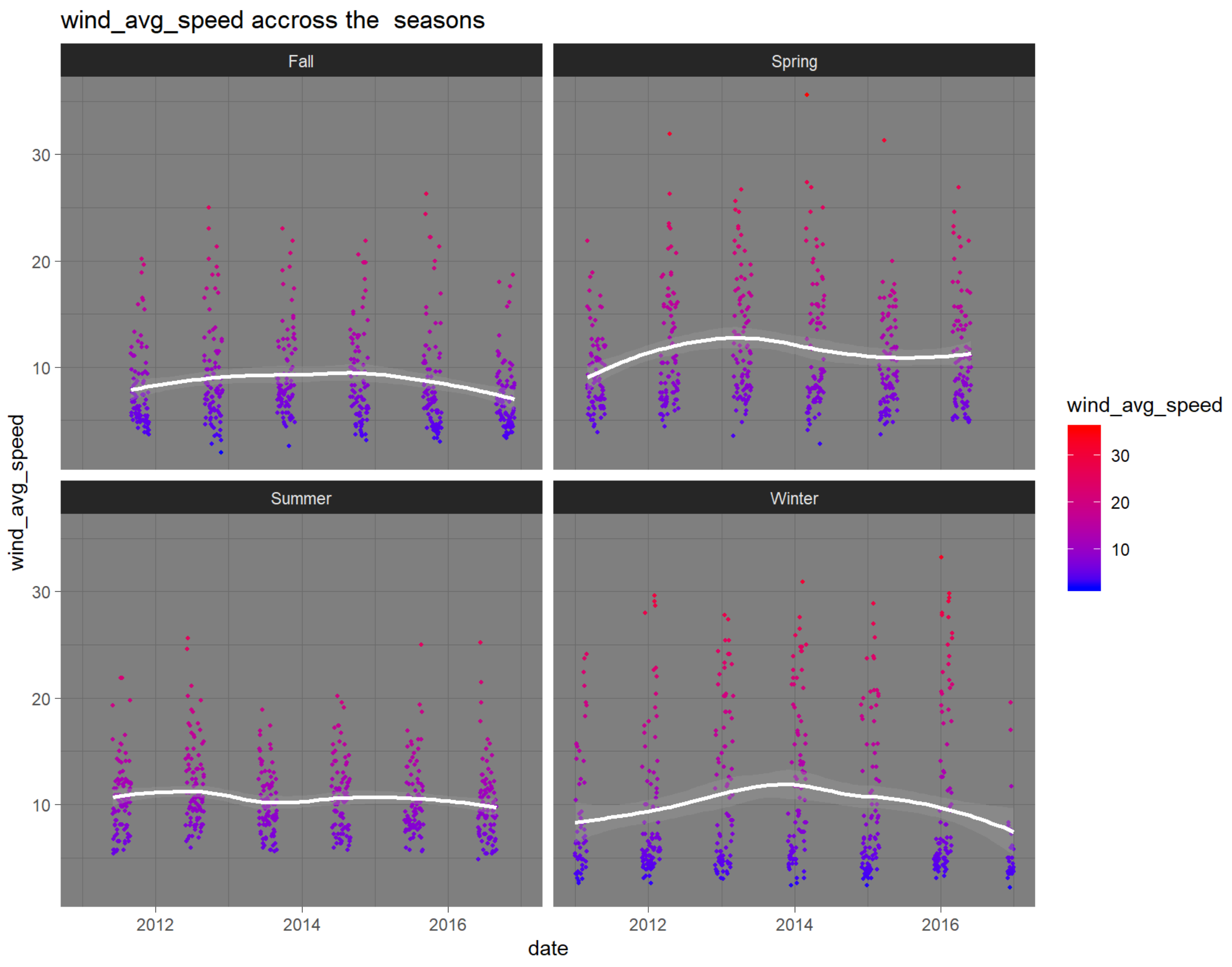
The average values for **PM2.5** are usually more constant accross the 4 seasons, with few spikes recorded in the summer of 2012 and the winter of 2016. However, the valus of PM2.5 have been persistent accross the 6 years with only a slight increase has been recorded in winter and a little decrease recorded in spring.



Average temperatures, as expected, are higher during summer. Winter and summer are shoiwng that the average temperature have increased throughout the years, while spring temperatures have witnessed fluctuations within approximately the same range. Fall has had average temperatures constant accross the 6 years despite the peaks experineced during the seasons first days.



Precipitations are tendentially low in Madrid, few outliers were recorded, mainly in fall and spring, but it is noticeable that the average precipitation accross the 4 seasons is low, below 5 per day.



Winds were stronger over winter and spring, especially with higher highs compared to the other seasons. However, this metric was almost constants accross the different years and seasons studied.

Linear regression of NO_2

```
daily_final <- as.data.table(daily_final)

NO_2_model <- lm(NO_2~., data = daily_final[ ,-'date', with=F])
summary(NO_2_model)
```

```
##
## Call:
## lm(formula = NO_2 ~ ., data = daily_final[, -"date", with = F])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -44.541  -4.880   0.012   4.756  25.792
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    35.46875    1.10320  32.151  < 2e-16 ***
## SO_2            1.99963    0.08360  23.918  < 2e-16 ***
## O_3            -0.26752    0.01374 -19.472  < 2e-16 ***
## PM2.5           0.95820    0.03963  24.179  < 2e-16 ***
## temp_avg        0.14524    0.04528   3.208  0.00136 **
## precipitation  -0.15179    0.05228  -2.904  0.00373 **
## wind_avg_speed -0.69041    0.03790 -18.218  < 2e-16 ***
## seasonSpring   -1.48847    0.54848  -2.714  0.00670 **
## seasonSummer   -4.18817    0.65345  -6.409 1.79e-10 ***
## seasonWinter    0.09605    0.63305   0.152  0.87941
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.633 on 2182 degrees of freedom
## Multiple R-squared:  0.8026, Adjusted R-squared:  0.8018
## F-statistic: 985.6 on 9 and 2182 DF,  p-value: < 2.2e-16
```

From the statistics summary,the p value stands at less than 0.05, stating its significance. This means that the null hypothesis, the association of the variables to the beta equals 0, can be rejected.

The R squared, can be considered acceptable, with its value of 80%.

Annex

Here it is possible to find the entire code used for the assignment.

The working directory should be set to the folder containing exclusivley the 72 csv files.

To access the xlsx file, its distinctive individual path must be used.

```
getwd()
setwd('C:/Users/Luigi Giulio Grandi/Documents/Luigi Giulio Grandi/IE/MBD/Term I/Modules/R/Assignmen
t/Group Assignment/workgroup data')

# Packages to be loaded ----
library(stringr)
library(readxl)
library(lubridate)
library(tidyverse)
library(ggplot2)
library(corrplot)
library(data.table)
library(corrplot)
library(reshape2)

# Importing weather.xlsx ----
weather <- read_excel('C:/Users/Luigi Giulio Grandi/Documents/Luigi Giulio Grandi/IE/MBD/Term I/Mod
ules/R/Assignment/Group Assignment/weather.xlsx')
str(weather)
weather$date <- as.Date(weather$date)

View(weather)

# Importing daily data (72 .csv) ----
temp <- list.files()
myfiles <- lapply(temp, read.csv)

string_length <- str_length(temp)

ID <- vector("numeric", length(temp))

for (i in seq_along(temp)) {

  ID[i] <- substr(temp[i], 13, string_length[i] - 4)

}

ID <- gsub("_", "-", ID)

for (i in seq_along(myfiles)) {
  myfiles[[i]]$Date <- rep(ID[i], nrow(myfiles[[i]]))
}

daily <- data.table::rbindlist(myfiles, use.names = TRUE)
str(daily)

daily$station <- NULL

daily$date <- as.Date(paste0('20',daily$Date, '-', daily$day))

daily$Date <- NULL
daily$day <- NULL

daily$parameter <- as.factor(daily$parameter)

str(daily)
View(daily)

# Parameters measured ----

measured_variables <- unique(daily$parameter)
```

```
length(measured_variables)

formula <- c('SO_2', 'CO', 'NO', 'NO_2','PM2.5','PM10', 'NOx', 'O_3', 'TOL','BEN', 'EBE', 'MXY', 'P
XY','OXY', 'TCH', 'CH4', 'NMHC')
parameter <- c('1','6','7','8','9','10','12','14','20','30','35','37','38','39','42','43','44')

parameters_1 <- data.frame(formula, parameter)
str(parameters_1)

temp_2 <- merge(daily, parameters_1 , by = 'parameter', all.x = T)
temp_2$parameter <- NULL

str(temp_2)
View(temp_2)

# Computing a data frame returning the daily average values of each parameter ----

daily_mean <- temp_2 %>%
  group_by (date, formula) %>%
  summarize (averaged_value = mean(value))

str(daily_mean)

daily_mean_parameter <- as.data.frame(tidyr::spread(daily_mean,formula,averaged_value))

str(daily_mean_parameter)
View(daily_mean_parameter)

# Merging weather with means & adding the seasons ----
str(weather)

daily_data <- merge(daily_mean_parameter, weather, by = 'date', sort = T)

str(daily_data)
View(daily_data)

# Adding the seasons to daily_data
seasons = function(x){

  if(x %in% 3:5) return("Spring")
  if(x %in% 6:8) return("Summer")
  if(x %in% 9:11) return("Fall")
  if(x %in% c(12,1,2)) return("Winter")

}

daily_data$season <-as.factor(sapply(month(daily_data$date), seasons))
str(daily_data)

# Final daily dataset ----
requested_parameters <- c('NO_2', 'SO_2', 'O_3', 'PM2.5', 'temp_avg','precipitation','wind_avg_spee
d')

daily_final <- daily_data[ ,c('date',requested_parameters,'season')]
sum(is.na(daily_final))

total_NAs<-apply(daily_final,2, function(var) { return(sum(is.na(var)))})
total_NAs

for (i in seq_along(daily_final)){
  Sys.sleep(0.1)
```

```
    print(colnames(daily_data[i]))
    for (j in 1:nrow(daily_final)) {
      if (is.na(daily_final[j,i])){
        daily_final[j,i] <- median(daily_final[,i], na.rm = T)
      }
    }
  }

  sum(is.na(daily_final))

  str(daily_final)
  View(daily_final)

  # Plotting ----

  # Correlation Matrix
  param <- subset(daily_final, select = c(2:8))
  str(param)

  cor_matrix <- cor(param)
  cor_plot <- corrplot(cor_matrix,method="color", type = "upper", title = "Parameters Correlation Mat
  rix", addCoef.col = "black", mar=c(0,0,1,0))

  # NO_2 Plot
  ggplot(daily_final, aes(date,NO_2,colour = NO_2))+
    geom_point(size = 0.75)+
    scale_color_gradient(low = "navyblue", high = "red") +
    theme_dark()+
    facet_wrap(vars(season))+
    labs(title = 'NO_2 accross the  seasons')+
    geom_smooth(colour = 'white', method = loess, formula = y ~ x)

  #SO_2 Plot
  ggplot(daily_final, aes(date,SO_2,colour = SO_2))+
    geom_point(size = 0.75)+
    scale_color_gradient(low = "navyblue", high = "red") +
    theme_dark()+
    facet_wrap(vars(season))+
    labs(title = 'SO_2 accross the  seasons')+
    geom_smooth(colour = 'white', method = loess, formula = y ~ x)

  #O_3 Plot
  ggplot(daily_final, aes(date,O_3,colour = O_3))+
    geom_point(size = 0.75)+
    scale_color_gradient(low = "navyblue", high = "red") +
    theme_dark()+
    facet_wrap(vars(season))+
    labs(title = 'O_3 accross the  seasons')+
    geom_smooth(colour = 'white', method = loess, formula = y ~ x)

  # PM2.5 Plot
  ggplot(daily_final, aes(date,PM2.5,colour = PM2.5))+
    geom_point(size = 0.75)+
    scale_color_gradient(low = "navyblue", high = "red") +
    theme_dark()+
    facet_wrap(vars(season))+
    labs(title = 'PM2.5 accross the  seasons')+
    geom_smooth(colour = 'white', method = loess, formula = y ~ x)

  #temp_avg Plot
  ggplot(daily_final, aes(date,temp_avg,colour = temp_avg))+
    geom_point(size = 0.75)+
    scale_color_gradient(low = "navyblue", high = "red") +
    theme_dark()+
    facet_wrap(vars(season))+
    labs(title = 'temp_avg accross the  seasons')+
    geom_smooth(colour = 'white', method = loess, formula = y ~ x)

  #precipitation Plot
  ggplot(daily_final, aes(date,precipitation,colour = precipitation))+
    geom_point(size = 0.75)+
    scale_color_gradient(low = "navyblue", high = "red") +
    theme_dark()+
    facet_wrap(vars(season))+
    labs(title = 'precipitation accross the  seasons')+
    geom_smooth(colour = 'white', method = loess, formula = y ~ x)

  #wind_avg_speed Plot
  ggplot(daily_final, aes(date,wind_avg_speed,colour = wind_avg_speed))+
    geom_point(size = 0.75)+
    scale_color_gradient(low = "navyblue", high = "red") +
    theme_dark()+
    facet_wrap(vars(season))+
    labs(title = 'wind_avg_speed accross the  seasons')+
    geom_smooth(colour = 'white', method = loess, formula = y ~ x)



  # NO_2 Multiple Regression ----
  daily_final <- as.data.table(daily_final)

  NO_2_model <- lm(NO_2~., data = daily_final[ ,-'date', with=F])
  summary(NO_2_model)
```