# Assignment 8: Time Series Analysis

Tani Valdez Rivas

Fall 2023

## OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on generalized linear models.

## Directions

1. Rename this file `<FirstLast>_A08_TimeSeries.Rmd` (replacing `<FirstLast>` with your first and last name).
2. Change "Student Name" on line 3 (above) with your name.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure to **answer the questions** in this assignment document.
5. When you have completed the assignment, **Knit** the text and code into a single PDF file.

## Set up

1. Set up your session:

- Check your working directory
- Load the tidyverse, lubridate, zoo, and trend packages
- Set your ggplot theme

```r
# Installing packages
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(lubridate)
install.packages("trend")
```

```
## Installing package into '/home/guest/R/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```r
library(trend)
install.packages("zoo")
```

```
## Installing package into '/home/guest/R/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```r
library(zoo)
```

```
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
install.packages("Kendall")
```

```
## Installing package into '/home/guest/R/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```r
library(Kendall)
install.packages("tseries")
```

```
## Installing package into '/home/guest/R/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```r
library(tseries)
```

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

```r
library(dplyr)


# Checking working directory
library(here)
```

```
## here() starts at /home/guest/R/EDE_Fall2023
```

```r
here()
```

```
## [1] "/home/guest/R/EDE_Fall2023"
```

```r
# Building ggplot themes
mytheme <- theme_gray(base_size = 12) +
  theme(axis.title = element_text(color = "black"),
          legend.position = "right",
        legend.title = element_text(color = "black"),
        plot.title = element_text(hjust = 0.5))

theme_set(mytheme)
```

2. Import the ten datasets from the Ozone_TimeSeries folder in the Raw data folder. These contain ozone concentrations at Garinger High School in North Carolina from 2010-2019 (the EPA air database only allows downloads for one year at a time). Import these either individually or in bulk and then combine them into a single dataframe named `GaringerOzone` of 3589 observation and 20 variables.

```r
#1

# Importing raw data
Grainger2010 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2010_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2011 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2011_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2012 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2012_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2013 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2013_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2014 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2014_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2015 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2015_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2016 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2016_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2017 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2017_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2018 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2018_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2019 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2019_raw.csv'),
                        stringsAsFactors = TRUE)

# Creating a single dataframe using the bind_rows function
GaringerOzone <- bind_rows(Grainger2010, Grainger2011, Grainger2012, Grainger2013,
                           Grainger2014, Grainger2015, Grainger2016, Grainger2017,
                           Grainger2018, Grainger2019)
```

```r
dim(GaringerOzone) # Checking dimensions
```

```
## [1] 3589   20
```

## Wrangle

3. Set your date column as a date class.
4. Wrangle your dataset so that it only contains the columns Date, Daily.Max.8.hour.Ozone.Concentration, and DAILY_AQI_VALUE.
5. Notice there are a few days in each year that are missing ozone concentrations. We want to generate a daily dataset, so we will need to fill in any missing days with NA. Create a new data frame that contains a sequence of dates from 2010-01-01 to 2019-12-31 (hint: `as.data.frame(seq())`). Call this new data frame Days. Rename the column name in Days to "Date".
6. Use a `left_join` to combine the data frames. Specify the correct order of data frames within this function so that the final dimensions are 3652 rows and 3 columns. Call your combined data frame GaringerOzone.

```r
# 3

# Using the as.Date function to set date column as date class
GaringerOzone$Date <- mdy(GaringerOzone$Date)

# 4

# Wrangling to select Date, Daily.Max.8.hour.Ozone.Concentration, and DAILY_AQI_VALUE
GaringerOzone <- GaringerOzone %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration,DAILY_AQI_VALUE)

# 5

# Creating daily dataset using as.data.frame(seq())
start_date <- as.Date("2010-01-01")
end_date <- as.Date("2019-12-31")
DailyGaringerOzone <-
  as.data.frame(seq(start_date, end_date, by = "days"))
colnames(DailyGaringerOzone) <- "Date" # changing column name

# 6

# Using the left_join function to combine two datasets
GaringerOzone <- left_join(DailyGaringerOzone,GaringerOzone, by = "Date")
dim(GaringerOzone) # Checking dimensions
```

```
## [1] 3652    3
```

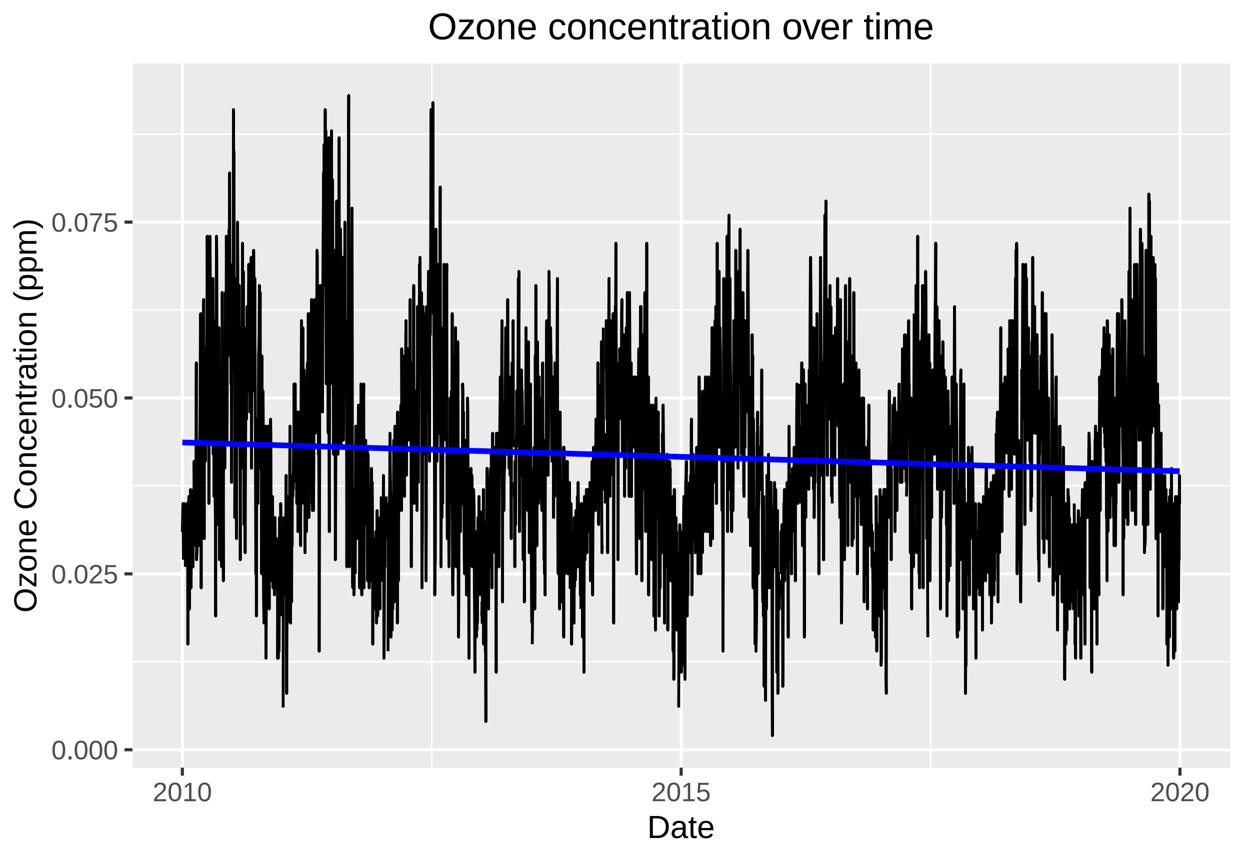
## Visualize

7. Create a line plot depicting ozone concentrations over time. In this case, we will plot actual concentrations in ppm, not AQI values. Format your axes accordingly. Add a smoothed line showing any linear trend of your data. Does your plot suggest a trend in ozone concentration over time?

```r
#7

# Using the ggplot to plot Ozone concentrations over time
Ozone_Time <-
  ggplot(GaringerOzone, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line() +
  labs(x = "Date", y = "Ozone Concentration (ppm)",
       title = "Ozone concentration over time") +
  mytheme +
  geom_smooth(method = "lm", se = FALSE, color = "blue")
Ozone_Time
```

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

```
## Warning: Removed 63 rows containing non-finite values (`stat_smooth()`).
```



Answer: Ozone concentrations levels are decreasing over time.

## Time Series Analysis

Study question: Have ozone concentrations changed over the 2010s at this station?

8. Use a linear interpolation to fill in missing daily data for ozone concentration. Why didn't we use a piecewise constant or spline interpolation?

```r
#8

# Using the head and summary function for analysis
head(GaringerOzone)
```

```
##         Date Daily.Max.8.hour.Ozone.Concentration DAILY_AQI_VALUE
## 1 2010-01-01                                0.031              29
## 2 2010-01-02                                0.033              31
## 3 2010-01-03                                0.035              32
## 4 2010-01-04                                0.031              29
## 5 2010-01-05                                0.027              25
## 6 2010-01-06                                   NA              NA
```

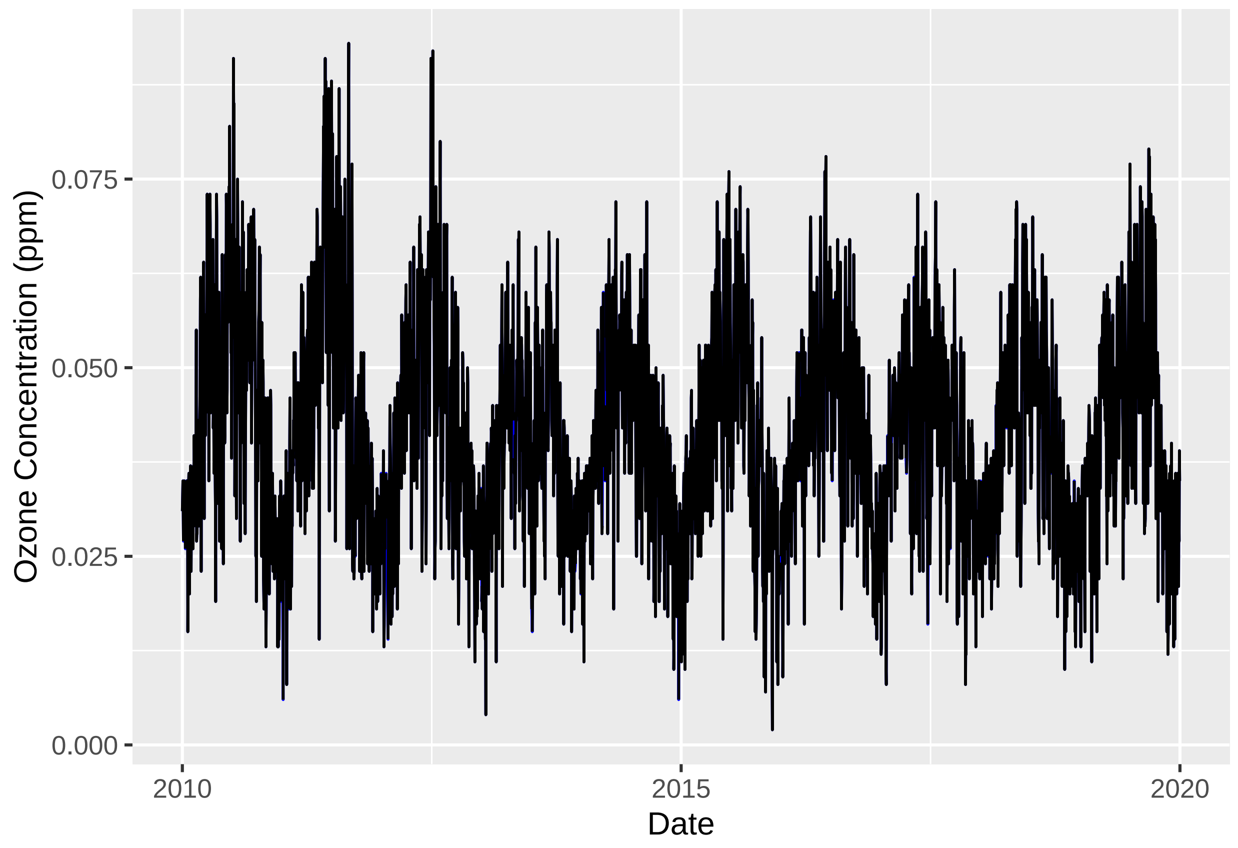
```r
summary(GaringerOzone)
```

```
##       Date            Daily.Max.8.hour.Ozone.Concentration DAILY_AQI_VALUE
##  Min.   :2010-01-01   Min.   :0.00200                      Min.   :  2.00
##  1st Qu.:2012-07-01   1st Qu.:0.03200                      1st Qu.: 30.00
##  Median :2014-12-31   Median :0.04100                      Median : 38.00
##  Mean   :2014-12-31   Mean   :0.04163                      Mean   : 41.57
##  3rd Qu.:2017-07-01   3rd Qu.:0.05100                      3rd Qu.: 47.00
##  Max.   :2019-12-31   Max.   :0.09300                      Max.   :169.00
##                       NA's   :63                           NA's   :63
```

```r
# Using the zoo package
GaringerOzone_clean <- GaringerOzone %>%
  mutate(Daily.Max.8.hour.Ozone.Concentration.clean =
           zoo::na.approx(Daily.Max.8.hour.Ozone.Concentration))

summary(GaringerOzone_clean$Daily.Max.8.hour.Ozone.Concentration.clean)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00200 0.03200 0.04100 0.04151 0.05100 0.09300
```

```r
# Creating ggplot
GaringerOzone_Interpolation <- ggplot(GaringerOzone_clean ) +
  geom_line(aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration.clean),
            color = "blue") +
  geom_line(aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration),
            color = "black") +
  ylab("Ozone Concentration (ppm)")
GaringerOzone_Interpolation
```



Answer: Linear interpolation uses data that comes before and after the unknown value and assumes that the unkown value is between those measurements. As ozone concentrations are unlikley to be equal, it does not make sense to use a piecewise constant. Spline interpolation

9. Create a new data frame called `GaringerOzone.monthly` that contains aggregated data: mean ozone concentrations for each month. In your pipe, you will need to first add columns for year and month to form the groupings. In a separate line of code, create a new Date column with each month-year combination being set as the first day of the month (this is for graphing purposes only)

```r
#9

# Piping to get mean ozone for each month
GaringerOzone.monthly <- GaringerOzone_clean %>%
  mutate(Year = lubridate::year(Date), Month = lubridate::month(Date)) %>%
  # used Chatgpt for assistance with this code
  group_by(Year,Month) %>%
  summarise(MeanOzone = mean(Daily.Max.8.hour.Ozone.Concentration.clean))
```

```
## ‘summarise()‘ has grouped output by ’Year’. You can override using the
## ‘.groups‘ argument.
```

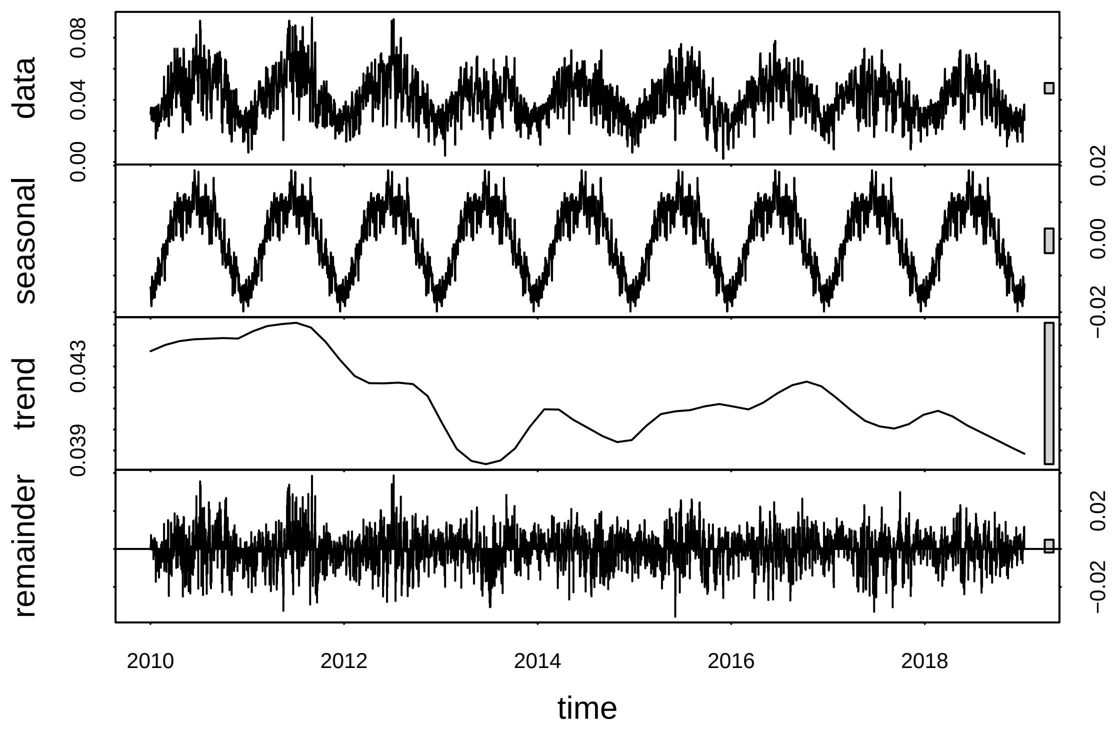
```r
# Combining and creating new date column
GaringerOzone.monthly$Date <-
  as.Date(paste(GaringerOzone.monthly$Year, sprintf("%02d",
  GaringerOzone.monthly$Month), 1, sep = "-"))
```

10. Generate two time series objects. Name the first `GaringerOzone.daily.ts` and base it on the dataframe of daily observations. Name the second `GaringerOzone.monthly.ts` and base it on the monthly average ozone values. Be sure that each specifies the correct start and end dates and the frequency of the time series.

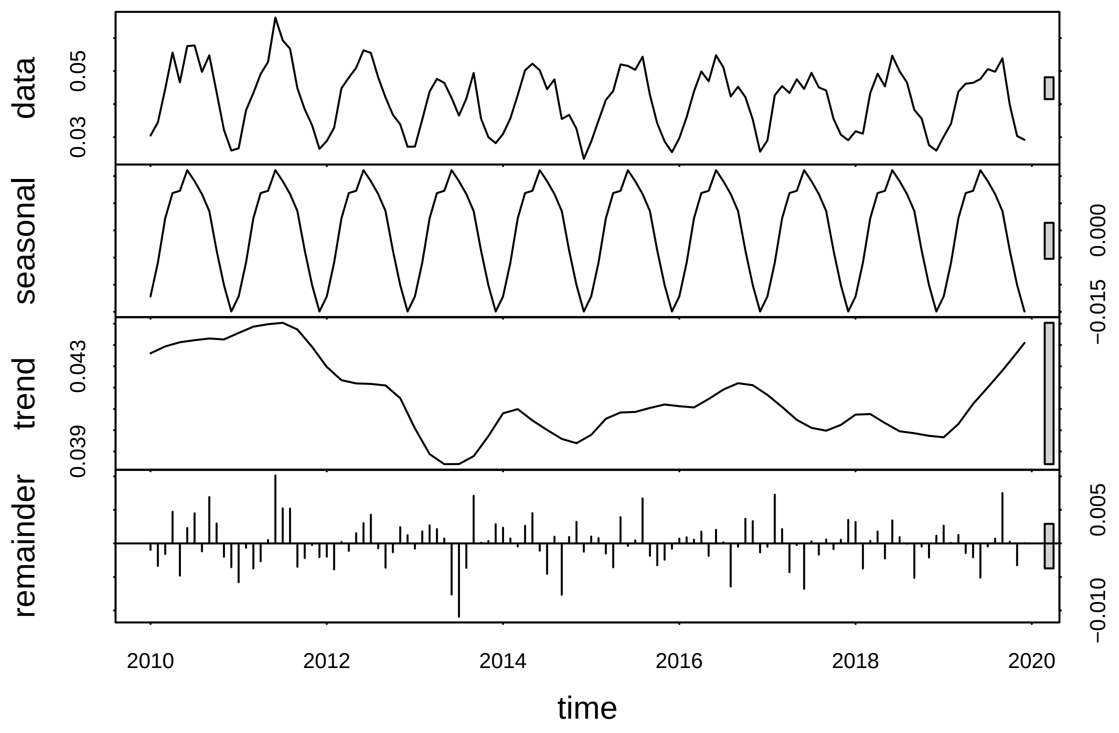
```
#10

# Creating start and end dates
f_month <- month(first(GaringerOzone.monthly$Date))
f_year <- year(first(GaringerOzone.monthly$Date))
l_month <- month(last(GaringerOzone.monthly$Date))
l_year <- year(last(GaringerOzone.monthly$Date))

# Using the ts function to produce timeseries
GaringerOzone.daily.ts <- ts(GaringerOzone_clean$Daily.Max.8.hour.Ozone.Concentration.clean,
                   start=c(f_year,f_month),
                   end = c(l_year,l_month),
                   frequency=365)


# Monthly timeseries
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$MeanOzone,
                   start=c(f_year,f_month),
                   end = c(l_year,l_month),
                   frequency=12)
```

11. Decompose the daily and the monthly time series objects and plot the components using the `plot()` function.

```
#11

# Using the stl function to decompose timeseries

#Daily
DailyGaringerOzone_Decompose <- stl(GaringerOzone.daily.ts, s.window = "periodic")
plot(DailyGaringerOzone_Decompose)
```



```
# Monthly
MonthlyGaringerOzone_Decompose <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
plot(MonthlyGaringerOzone_Decompose)
```



12. Run a monotonic trend analysis for the monthly Ozone series. In this case the seasonal Mann-Kendall is most appropriate; why is this?

```
#12

# Using the Kendall::SeasonalMannKendall function to run a monotonic trend
GaringerOzone.monthly.ts_trend <-
  Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
GaringerOzone.monthly.ts_trend
```

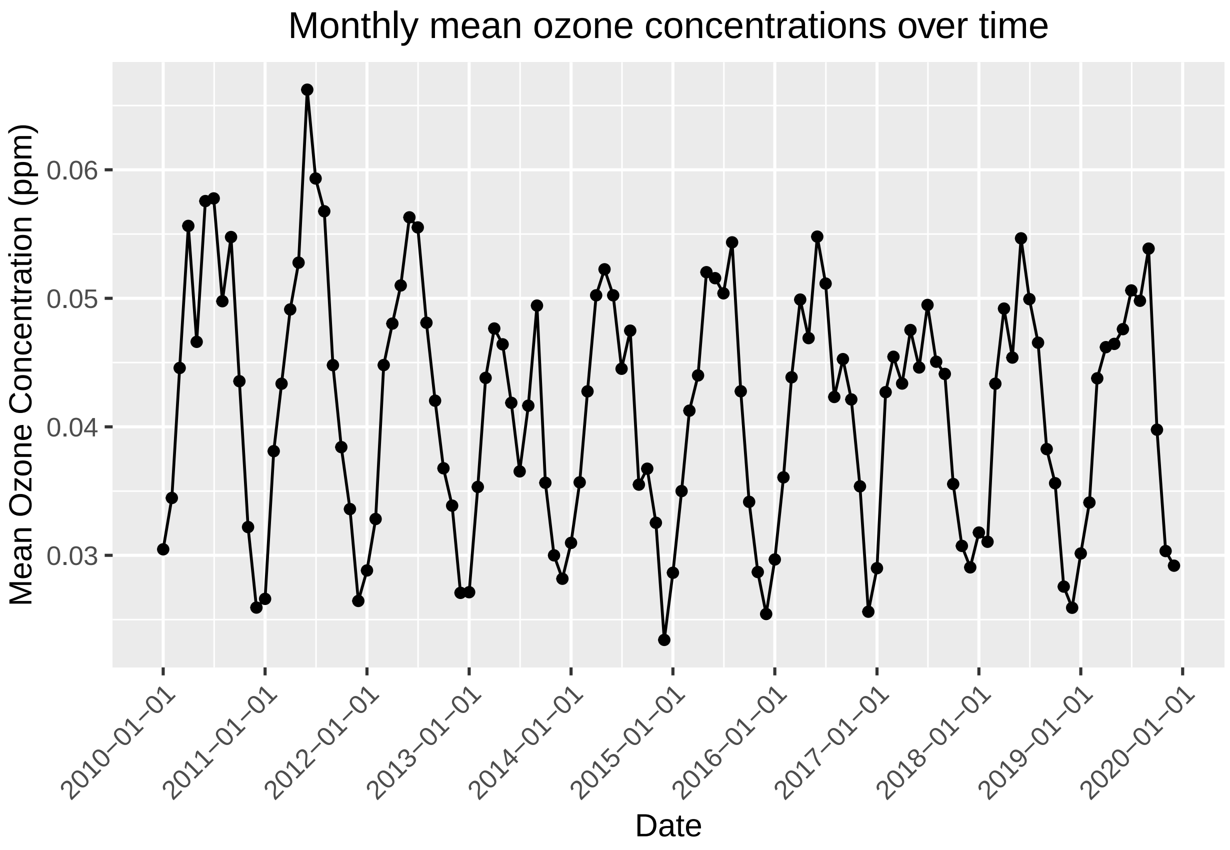
```
## tau = -0.143, 2-sided pvalue =0.046724
```

Answer: This is the most appropiate as Mann-Kendall test can show patterns, does not make assumptions, and can detect the smalled of changes in the data.

13. Create a plot depicting mean monthly ozone concentrations over time, with both a geom_point and a geom_line layer. Edit your axis labels accordingly.

```
# 13

# Creating a plot that looks at the monthly mean concentration of Ozone over time
MeanOzone_Time <- ggplot(GaringerOzone.monthly,aes(x = Date, y = MeanOzone))+
  geom_line() +
  geom_point() +
```

```r
  labs(x = "Date", y = "Mean Ozone Concentration (ppm)",
       title = "Monthly mean ozone concentrations over time") +
  scale_x_date(date_labels = "%Y-%m-%d", date_breaks = "1 year") +
  mytheme +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
MeanOzone_Time
```



Answer: The graoh has shown that ozone levels have started to decrease With a p-value of 0.046724 from the seasonal Mann-Kendall, the relationship between ozone concentrations and time demonstrates a correlation. This finding is significant as it indicates air quality may be effected and it is critical to conduct contiunous studies and monitoring to learn the adverse effects.

15. Subtract the seasonal component from the `GaringerOzone.monthly.ts`. Hint: Look at how we extracted the series components for the EnoDischarge on the lesson Rmd file.
16. Run the Mann Kendall test on the non-seasonal Ozone monthly series. Compare the results with the ones obtained with the Seasonal Mann Kendall on the complete series.

```r
#15

#Using the decomposed time series object to subtract the seasonal component
GaringerOzone.monthly_Components <-
  as.data.frame(MonthlyGaringerOzone_Decompose$time.series[,1:3])
```

```r
 nonseasonal_GaringerOzone.monthly.ts <-
   GaringerOzone.monthly.ts - GaringerOzone.monthly_Components$seasonal


#16

 # Using the Kendall::MannKendall() function to rund a trend
Monthly_Nonseasonal_GraingerOzone <-
  Kendall::MannKendall(nonseasonal_GaringerOzone.monthly.ts)
Monthly_Nonseasonal_GraingerOzone
```

```
## tau = -0.165, 2-sided pvalue =0.0075402
```

> Answer: The non-seasonal Mann-Kendall test showed a tau of -0.165 and a 2-sided pvalue of 0.0075402. The seasonal Mann-Kendall test showed a tau of -0.143 and a2-sided pvalue of 0.046724. Both results indicate that mean ozone concentration levels are decreasing over time. Both show it is statistically significant.