Assignment 8: Time Series Analysis

Dwiti Bagadia

Spring 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

```
#1
getwd()
```

```
## [1] "/Users/d/Desktop/UNC/Spring:23/DUKE 872 L1 - Environmental Data Analysis/EDA-Spring23"
```

```
library(tidyverse)
library(lubridate)
library(zoo)
library(ggthemes)
library(trend)
library(formatR)

#setting and creating theme
library(ggthemes)
tweetheme <- theme_classic(base_size = 14) +
  theme(axis.text = element_text(color = "darkgrey"),
        axis.ticks = element_line(color = "darkgrey"),
        plot.background = element_rect(color = "white"),
        legend.position = "top")
theme_set(tweetheme)

#formatting
knitr::opts_chunk$set(tidy.opts=list(width.cutoff=35), tidy=TRUE)
```

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.

```
# 2 loading data
GarOzone10 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv",
    stringsAsFactors = TRUE)
GarOzone11 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv",
    stringsAsFactors = TRUE)
GarOzone12 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv",
    stringsAsFactors = TRUE)
GarOzone13 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv",
    stringsAsFactors = TRUE)
GarOzone14 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv",
    stringsAsFactors = TRUE)
GarOzone15 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv",
    stringsAsFactors = TRUE)
GarOzone16 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv",
    stringsAsFactors = TRUE)
GarOzone17 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv",
    stringsAsFactors = TRUE)
GarOzone18 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv",
    stringsAsFactors = TRUE)
GarOzone19 = read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv",
    stringsAsFactors = TRUE)
GarOzone = GarOzone10 %>%
    full_join(GarOzone11) %>%
    full_join(GarOzone12) %>%
    full_join(GarOzone13) %>%
    full_join(GarOzone14) %>%
    full_join(GarOzone15) %>%
    full_join(GarOzone16) %>%
    full_join(GarOzone17) %>%
    full_join(GarOzone18) %>%
    full_join(GarOzone19)
```

```
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
## Joining with ‘by = join_by(Date, Source, Site.ID, POC,
## Daily.Max.8.hour.Ozone.Concentration, UNITS, DAILY_AQI_VALUE, Site.Name,
## DAILY_OBS_COUNT, PERCENT_COMPLETE, AQS_PARAMETER_CODE, AQS_PARAMETER_DESC,
## CBSA_CODE, CBSA_NAME, STATE_CODE, STATE, COUNTY_CODE, COUNTY, SITE_LATITUDE,
## SITE_LONGITUDE)‘
```

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”.

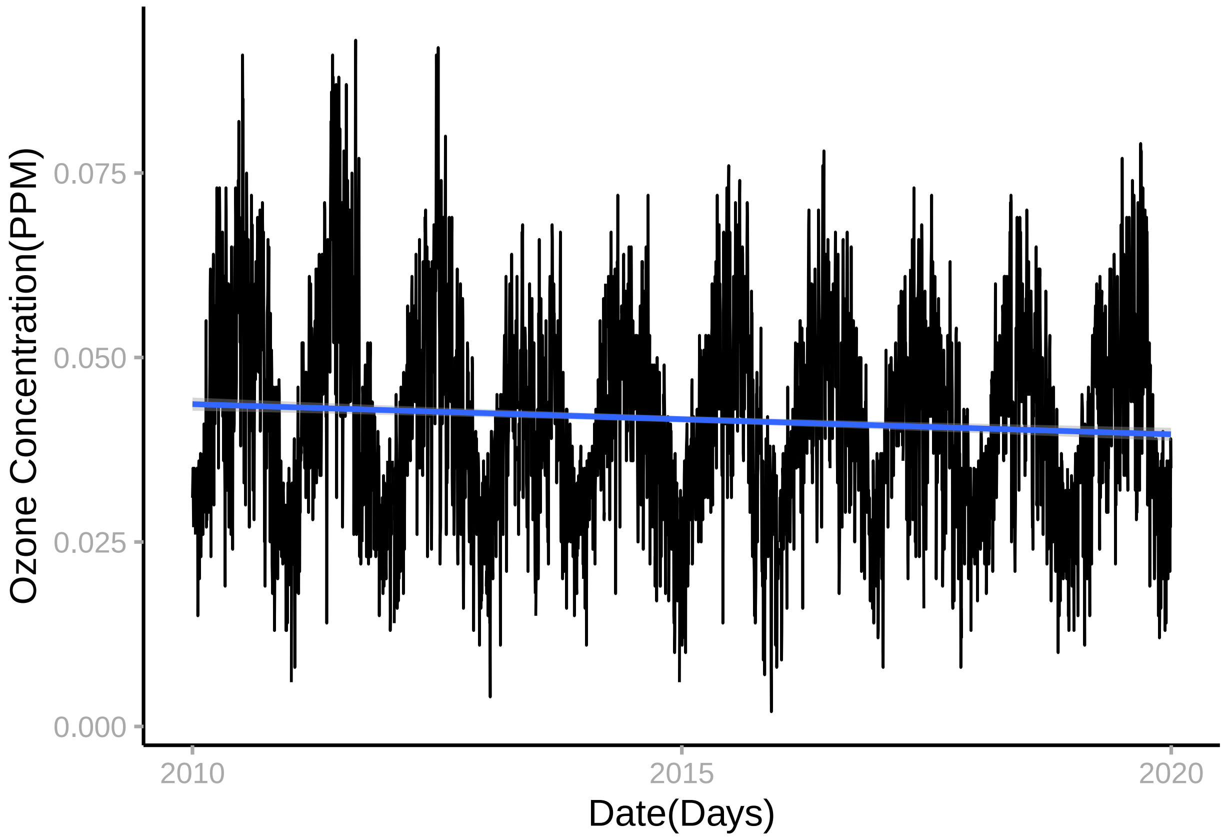
6. Use a `left_join` to combine the data frames. Specify the correct order of data frames within this function so that the final dimensions are 3652 rows and 3 columns. Call your combined data frame GaringerOzone.

```r
# 3 setting date column as date
# class
GarOzone$Date = mdy(GarOzone$Date)

# 4 wrangling datasets
GarOzone = select(GarOzone, Date, Daily.Max.8.hour.Ozone.Concentration,
    DAILY_AQI_VALUE)

# 5 creating new data frame to
# generate a daily dataset
Days = as.data.frame(seq(mdy("1/1/2010"),
    mdy("12/31/2019"), "days"))
colnames(Days) = c("Date")

# 6 combine data frames using
# leftjoin
GarOzone = left_join(Days, GarOzone)
```

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 creating line plots
ggplot(GarOzone, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
    geom_line() + geom_smooth(method = "lm") +
    xlab("Date(Days)") + ylab("Ozone Concentration(PPM)")
```



Answer: The plot shows a downward trend in ozone concentration 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?

```
# 8 linear interpolations to fill
# missing daily data
GarOzone$Daily.Max.8.hour.Ozone.Concentration = zoo::na.approx(GarOzone$Daily.Max.8.hour.Ozone.Concentra
```

Answer: Using piecewise constant would generate non-smooth results and it will not adequately represent underlying data; with spline interpolation we may need a large quantity of data to effectively simulate the curve as it is compurationally demanding.

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)

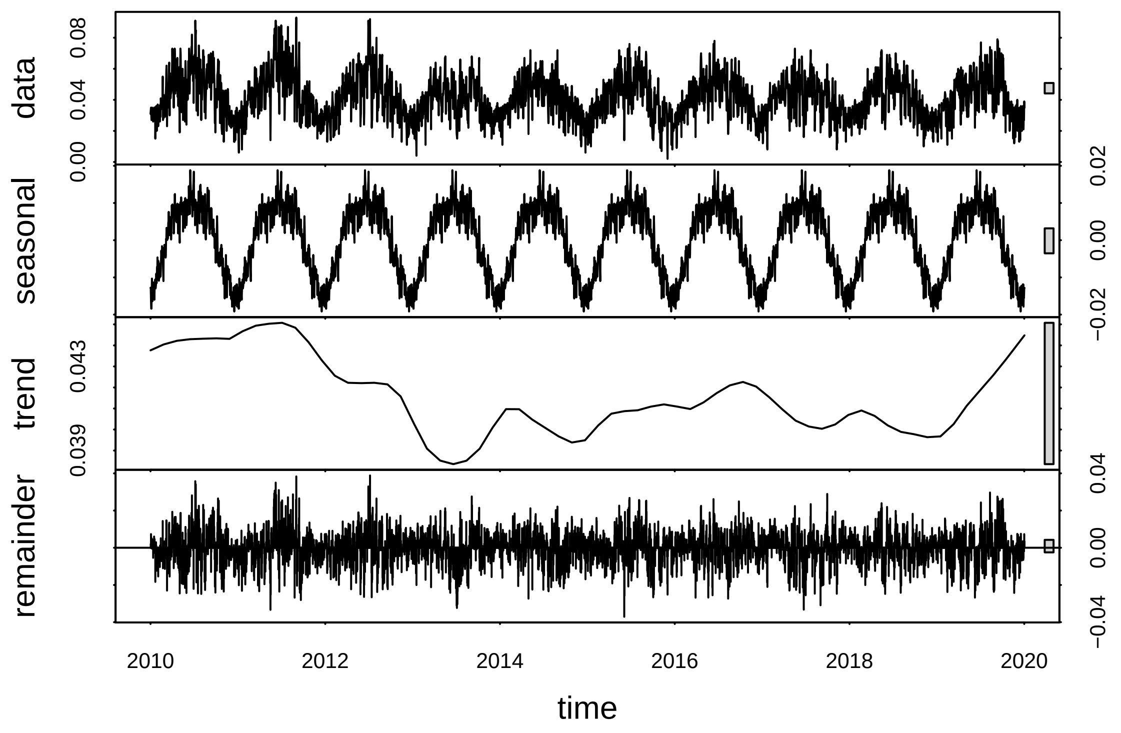
```
# 9 new data frame creation
GaringerOzone.monthly = GarOzone %>%
    select(Date, Daily.Max.8.hour.Ozone.Concentration) %>%
    mutate(Year = year(Date), Month = month(Date)) %>%
    group_by(Year, Month) %>%
    summarise(MeanOzone = mean(Daily.Max.8.hour.Ozone.Concentration)) %>%
    mutate(Date = ymd(paste0(Year, "-",
        Month, "-", "01")))
```

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.

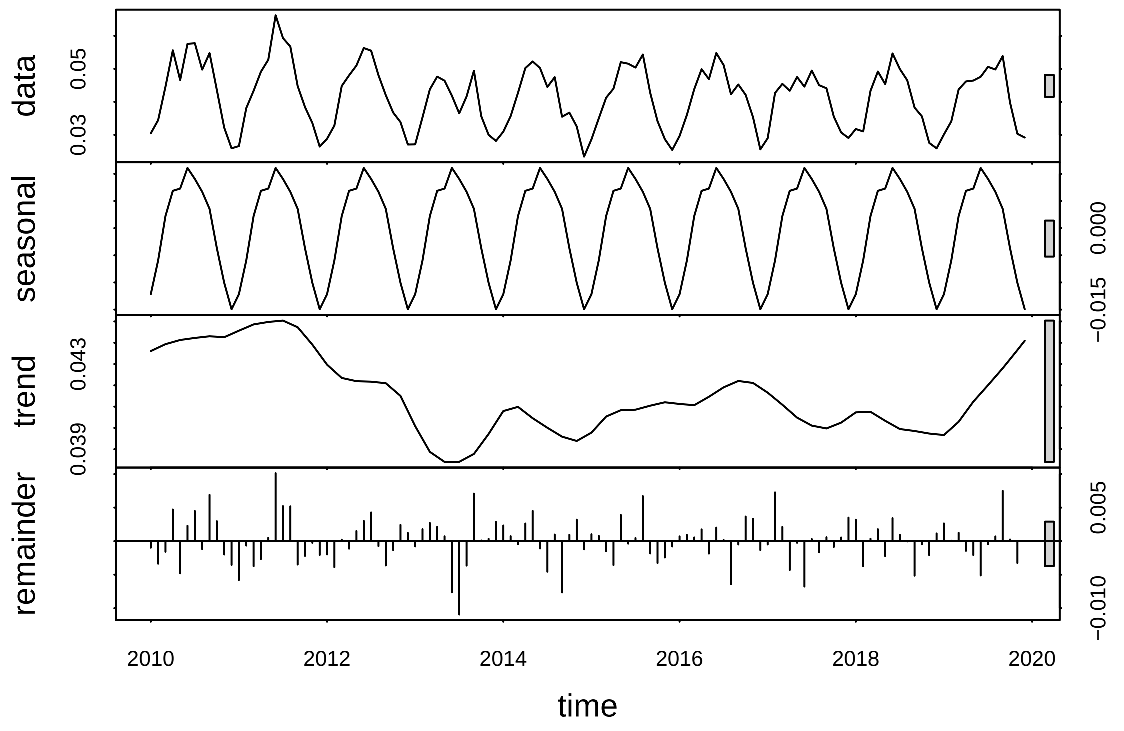
```
# 10 generating 2 new time series
# objects
GaringerOzone.daily.ts = ts(GarOzone$Daily.Max.8.hour.Ozone.Concentration,
    start = c(2010, 1), frequency = 365)
GaringerOzone.monthly.ts = ts(GaringerOzone.monthly$MeanOzone,
    start = c(2010, 1), frequency = 12)
```

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

```
# 11 decomposing and plotting
GarOzone.daily.ts.dec = stl(GaringerOzone.daily.ts,
    s.window = "periodic")
GarOzone.monthly.ts.dec = stl(GaringerOzone.monthly.ts,
    s.window = "periodic")
plot(GarOzone.daily.ts.dec)
```



```
plot(GarOzone.monthly.ts.dec)
```



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

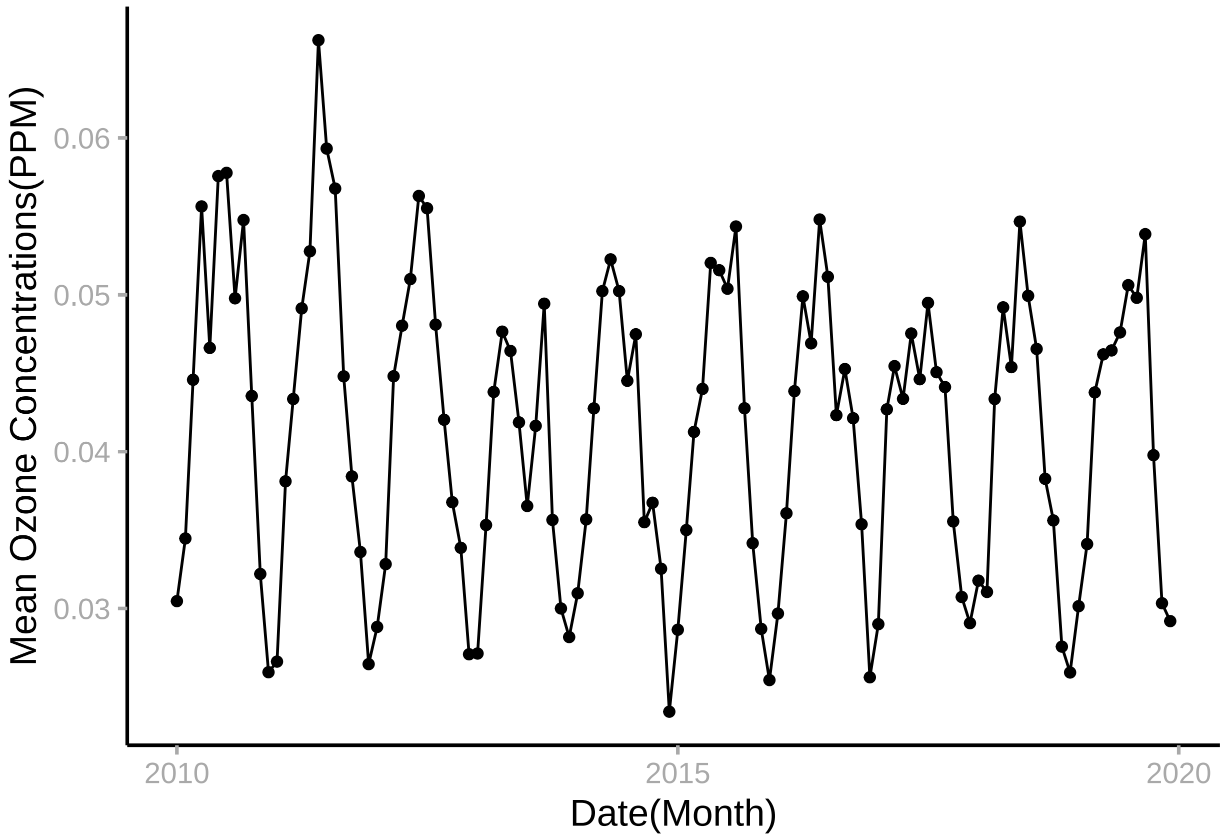
```
# 12 monotonic trend analysis
GarOzone.montly.trend = Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
summary(GarOzone.montly.trend)
```

```
## Score =  -77 , Var(Score) = 1499
## denominator =  539.4972
## tau = -0.143, 2-sided pvalue =0.046724
```

Answer: Seasonal Mann-Kendall is the most appropriate in this case as it works with seasonal data. The last plot made visualises the data change seasonally.

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 plot to depict mean monthly
# ozone concentrations
ggplot(GaringerOzone.monthly, aes(x = Date,
    y = MeanOzone)) + geom_point() +
    geom_line() + xlab("Date(Month)") +
    ylab("Mean Ozone Concentrations(PPM)")
```



14. To accompany your graph, summarize your results in context of the research question. Include output from the statistical test in parentheses at the end of your sentence. Feel free to use multiple sentences in your interpretation.

    Answer: Monotonic trend analysis result > Score = -1770 , Var(Score) = 69305.33; Denominator = 7725.182; Tau = -0.229, 2-sided pvalue = 1.7751e-11 We can summarize this from the graph. We can how see that the mean of ozone concentration often peaks during summer months and drops to the lowest during the winter months. The monotonic trend analysis shows us that the p-value is far less than alpha of 0.05; this means that the ozone concentration has changed during 2010 and it has been a downward trend ever since.

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.

```
# 15
GarOzone.monthly.comp = as.data.frame(GarOzone.monthly.ts.dec$time.series[,
    2:3])
GarOzone.monthly.comp = GarOzone.monthly.comp %>%
    mutate(Date = GaringerOzone.monthly$Date)



# 16
```

```r
GarOzone.monthly.comp.ts = ts(GarOzone.monthly.comp,
    start = c(2010, 1), frequency = 12)
GarOzone.monthly.comp.ts.trend = Kendall::MannKendall(GarOzone.monthly.comp.ts)
summary(GarOzone.monthly.comp.ts.trend)
```

```
## Score =  19640 , Var(Score) = 5205500
## denominator =  64620
## tau = 0.304, 2-sided pvalue =< 2.22e-16
```

> Answer: The non-seasonal Ozone monthly series shows us the p value is less than equal to 2.22e-16 which is far less than the p value of the Seaonal Mann Kendall series. This shows us that the accuracy becomes higher after removing the seasonal component from the data.