Assignment 8: Time Series Analysis

Jaleesia Amos

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

```r
#1
#---------Check working directory---------#
getwd()
```

```
## [1] "/Users/jaleesiad.amos/Documents/EDA-Spring2023"
```

```r
#--------Loading packages---------#
pacman::p_load(tidyverse, lubridate, zoo, trend, here)

#--------Set ggplot theme---------#
jatheme <- theme_bw(base_size = 12) +
  theme(plot.title = element_text(face ="bold", hjust = 0.5),
        axis.title = element_text(face = "bold"),
        legend.position = "right")
theme_set(jatheme)
```

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
#----------Load 10 EPA air datasets 2010-2019-----------#
#1
EPAair.2010 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv"), stringsAsFa
#2
EPAair.2011 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv"), stringsAsFa
#3
EPAair.2012 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv"), stringsAsFa
#4
EPAair.2013 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv"), stringsAsFa
#5
EPAair.2014 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv"), stringsAsFa
#6
EPAair.2015 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv"), stringsAsFa
#7
EPAair.2016 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv"), stringsAsFa
#8
EPAair.2017 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv"), stringsAsFa
#9
EPAair.2018 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv"), stringsAsFa
#10
EPAair.2019 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv"), stringsAsFa

#----------Combine 10 EPA air datasets 2010-2019-----------#
GaringerOzone <- rbind(EPAair.2010, EPAair.2011, EPAair.2012, EPAair.2013, EPAair.2014, EPAair.2015, EP
```

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.

```
#3
#--------Change date column to date class--------#
#___Checking structure of data set____#
str(GaringerOzone)
```

```
## 'data.frame':    3589 obs. of  20 variables:
```

```
##  $ Date                              : Factor w/ 3589 levels "01/01/2010","01/02/2010",..: 1 2 3 4
##  $ Source                            : Factor w/ 2 levels "AQS","AirNow": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Site.ID                           : int  371190041 371190041 371190041 371190041 371190041 37119
##  $ POC                               : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Daily.Max.8.hour.Ozone.Concentration: num  0.031 0.033 0.035 0.031 0.027 0.033 0.035 0.032 0.032 0
##  $ UNITS                             : Factor w/ 1 level "ppm": 1 1 1 1 1 1 1 1 1 1 ...
##  $ DAILY_AQI_VALUE                   : int  29 31 32 29 25 31 32 30 30 28 ...
##  $ Site.Name                         : Factor w/ 1 level "Garinger High School": 1 1 1 1 1 1 1 1 1
##  $ DAILY_OBS_COUNT                   : int  17 17 17 17 17 17 17 17 17 17 ...
##  $ PERCENT_COMPLETE                  : num  100 100 100 100 100 100 100 100 100 100 ...
##  $ AQS_PARAMETER_CODE                : int  44201 44201 44201 44201 44201 44201 44201 44201 44201 4
##  $ AQS_PARAMETER_DESC                : Factor w/ 1 level "Ozone": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CBSA_CODE                         : int  16740 16740 16740 16740 16740 16740 16740 16740 16740 1
##  $ CBSA_NAME                         : Factor w/ 1 level "Charlotte-Concord-Gastonia, NC-SC": 1 1 1
##  $ STATE_CODE                        : int  37 37 37 37 37 37 37 37 37 37 ...
##  $ STATE                             : Factor w/ 1 level "North Carolina": 1 1 1 1 1 1 1 1 1 1 ...
##  $ COUNTY_CODE                       : int  119 119 119 119 119 119 119 119 119 119 ...
##  $ COUNTY                            : Factor w/ 1 level "Mecklenburg": 1 1 1 1 1 1 1 1 1 1 ...
##  $ SITE_LATITUDE                     : num  35.2 35.2 35.2 35.2 35.2 ...
##  $ SITE_LONGITUDE                    : num  -80.8 -80.8 -80.8 -80.8 -80.8 ...
```

```
#___Change Date column to date object___#
GaringerOzone$Date <- mdy(GaringerOzone$Date)

#4
#-----Wrangling dataset------#
GaringerOzone_processed <- GaringerOzone %>%
  select(Date, Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)

#5
#------Generate daily dataset-------#
Days <- as.data.frame(seq(as.Date("2010-01-01"), as.Date("2019-12-31"), by = "day"))

#-----Change column name------#
colnames(Days)[1] ="Date"

#6
#------Combine datasets while retaining all rows-------#
GaringerOzone <- left_join(Days, GaringerOzone_processed)
```

```
## Joining with `by = join_by(Date)`
```

```
#----Changeing Ozone colname to simplify code------#
colnames(GaringerOzone)[2] = "Ozone_Concentration"
```

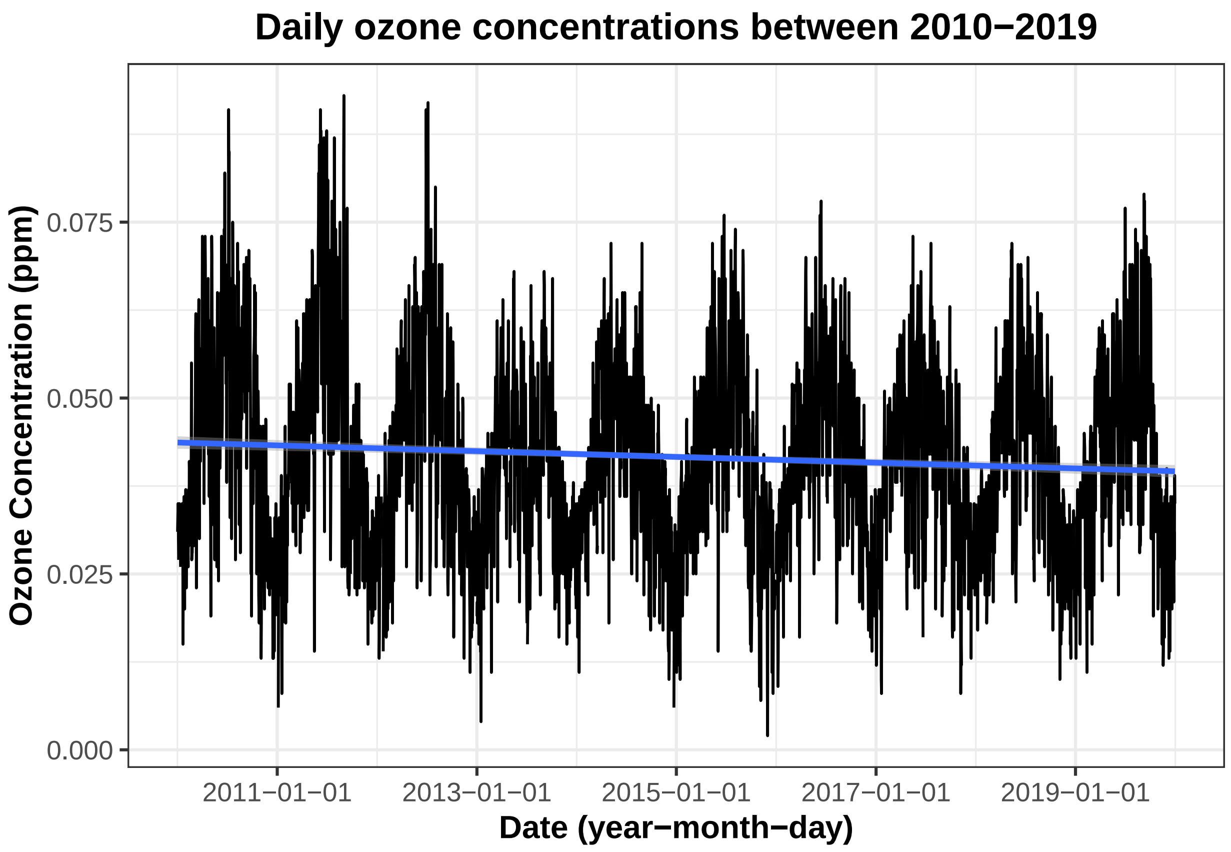
Visualize

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?

```
#7
#--------Line plots of O3 vs time-------#
ggplot(GaringerOzone, aes(y = Ozone_Concentration, x = Date)) +
  geom_line() +
  geom_smooth(method = "lm") +
  labs(x = "Date (year-month-day)", y = "Ozone Concentration (ppm)") +
  scale_x_date(breaks = "2 year") +
  ggtitle("Daily ozone concentrations between 2010-2019")
```

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

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



Answer: There appears to be a slight downward trend over the 10 year period, but this is difficult to see.

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 Interpolation for missing data------------#
#____checking for missing data____#
summary(GaringerOzone$Ozone_Concentration)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## 0.00200 0.03200 0.04100 0.04163 0.05100 0.09300      63
```

```
#____Use linear interpolation_____#
GaringerOzone_clean <-
  GaringerOzone %>%
  mutate(Ozone_Concentration = zoo::na.approx(GaringerOzone$Ozone_Concentration))
#___Checking NAs removed____#
summary(GaringerOzone_clean$Ozone_Concentration)
```

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

Answer: Both piecewise and spline interpolation are more likely to introduce false values into the dataset. A linear interpolation is most likely to produce values close to the true value as it is within range of its previous and subsequent measurements. Piecewise would fill in NAs with the value of the nearest neighbor; a quick look at the dataset indicates none of the values are repeated, also a quadratic function not be appropriate here.

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
#------Create new data frame------#
#______add year and month before grouping______#
GaringerOzone.monthly <- GaringerOzone_clean %>%
  mutate(Year = year(Date), Month = month(Date)) %>%
  group_by(Year, Month) %>%
  summarise(meanozone = mean(Ozone_Concentration))
```

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

```
#______add new date column______#
GaringerOzone.monthly$YearMonth <- as.yearmon(paste(GaringerOzone.monthly$Year, GaringerOzone.monthly$Mo
```

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.

```r
#10
#-----------Creating daily ts------------#
#____checking start and end dates____#
summary(GaringerOzone_clean$Date)
```

```
##          Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
## "2010-01-01" "2012-07-01" "2014-12-31" "2014-12-31" "2017-07-01" "2019-12-31"
```

```r
#____Daily ts____#
GaringerOzone.daily.ts <- ts(GaringerOzone_clean$Ozone_Concentration, start = c(2010,1),
                             end = c(2019, 12), frequency = 365)

#----------Creating monthly ts-----------#
#____checking start and end dates____#
range(GaringerOzone.monthly$YearMonth)
```

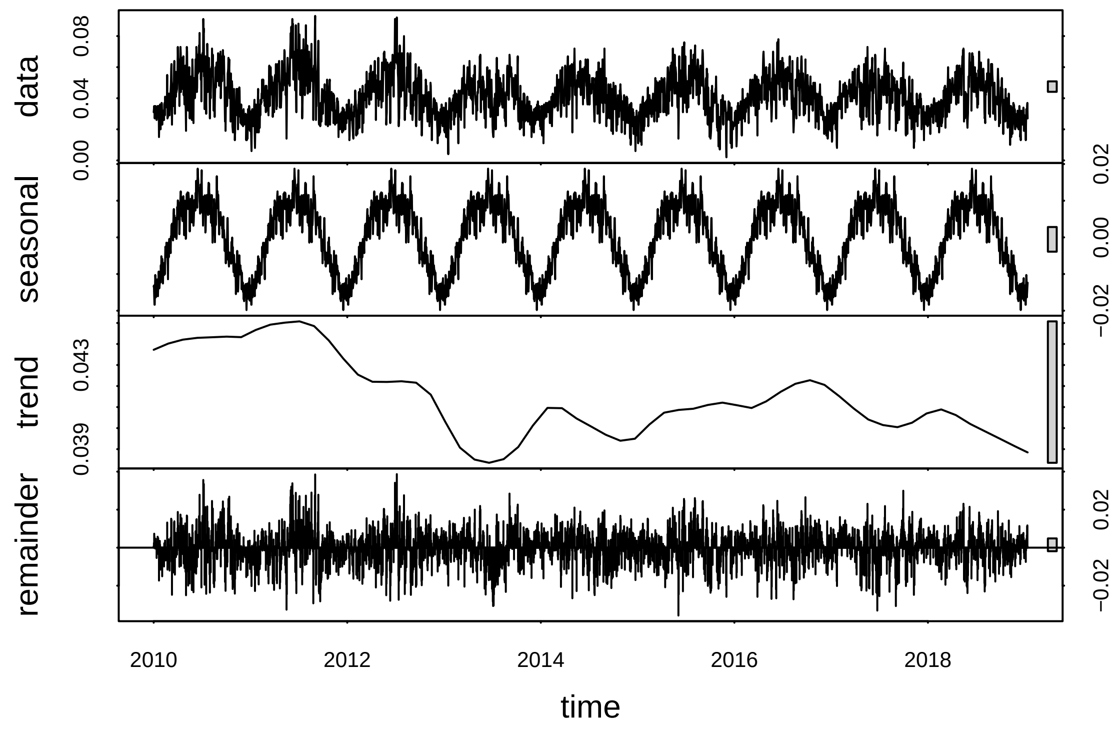
```
## [1] "Jan 2010" "Dec 2019"
```

```r
#____Monthly ts____#
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$meanozone,
                               start = c(2010,1), end = c(2019, 12), frequency = 12)
```

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

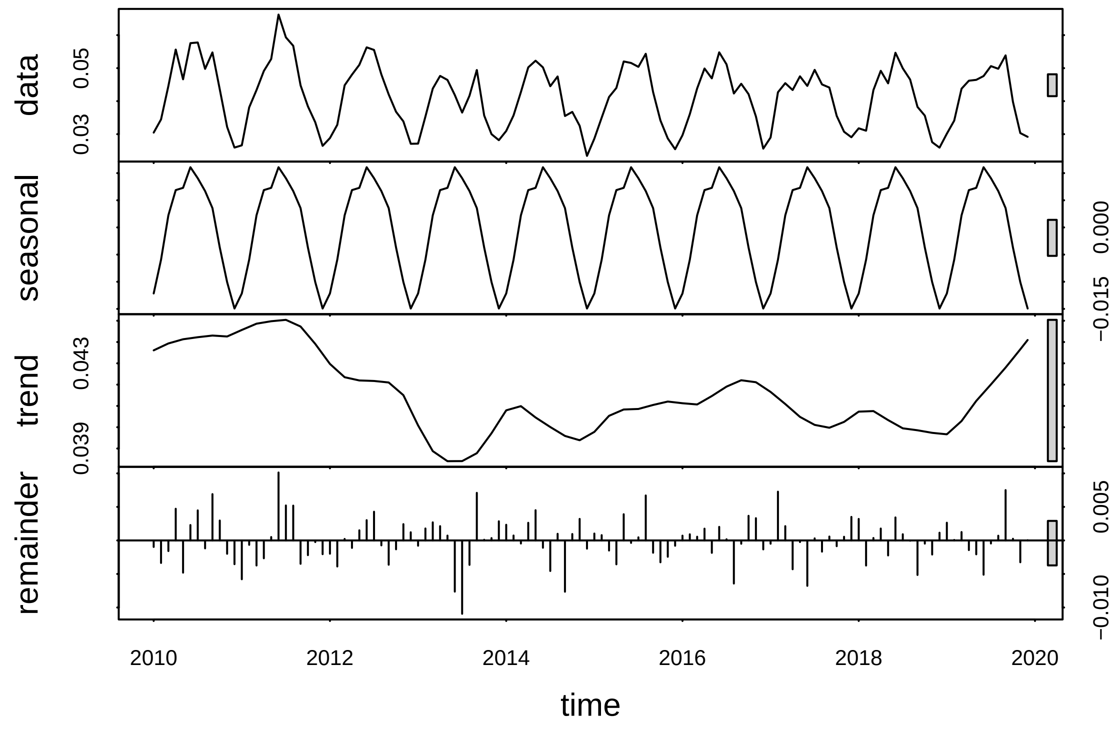
```r
#11
#---------Daily Decomposition----------#
GaringerOzone.daily.ts.D <-stl(GaringerOzone.daily.ts, s.window = "periodic")

#--------Visualize daily decomp-----------#
plot(GaringerOzone.daily.ts.D)
```



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

#--------Visualize monthly decomp-----------#
plot(GaringerOzone.monthly.ts.D)
```



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 on the monthly ts--------#
MonthlyTrend<- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)

#----Reveiw results------#
summary(MonthlyTrend)
```

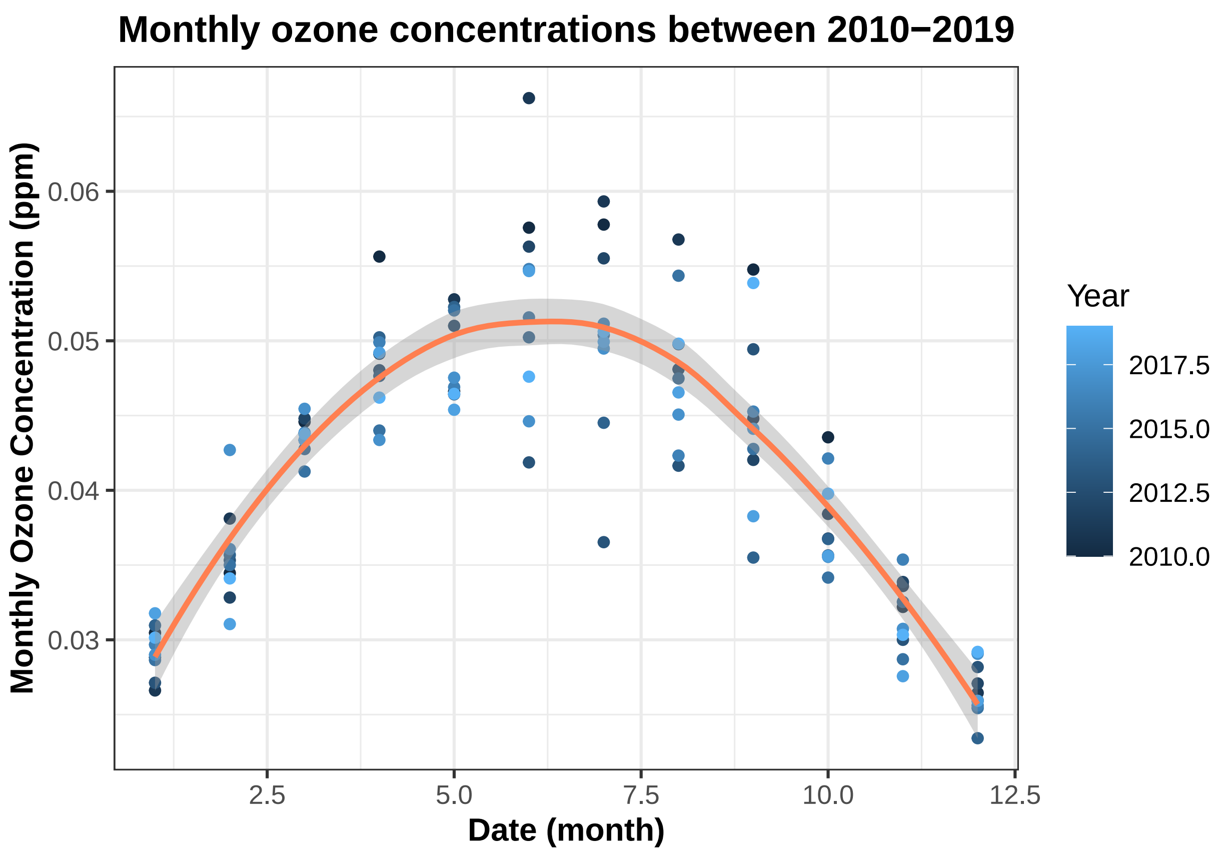
```
## Score =  -77 , Var(Score) = 1499
## denominator =  539.4972
## tau = -0.143, 2-sided pvalue =0.046724
```

Answer: A seasonal Mann-Kendall test is most appropriate here because there appears to be regular changes that occur through out the course of the year. This can be seen in the decomposed plot in the seasonal section.

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
#---------Monthly mean ozone concentrations--------#
ggplot(GaringerOzone.monthly, aes(y = meanozone, x = Month)) +
  geom_point(aes(color = Year)) +
  geom_smooth(method = "loess", color = "coral") +
  labs(x = "Date (month)", y = "Monthly Ozone Concentration (ppm)") +
  ggtitle("Monthly ozone concentrations between 2010-2019") +
  theme(legend.position = "right")
```

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



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: Research question:Have ozone concentrations changed over the 2010s at this station? From the seasonal Mann-Kendall test our p-value is so close to 0.05 an argument could be made that the data is stationary. What is clear the seasonality of the ozone concentrations.

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
#--------Extract seasonal components from monthly ts----------#
GaringerOzone.monthly.ts_Components <- (GaringerOzone.monthly.ts.D$time.series[,2:3])

#16
#-------Monotonic trend on the monthly ts--------#
MonthlyTrend2<- Kendall::MannKendall(GaringerOzone.monthly.ts_Components)
summary(MonthlyTrend2)
```

```
## Score =  -16300 , Var(Score) = 1545533
## denominator =  28680
## tau = -0.568, 2-sided pvalue =< 2.22e-16
```

Answer: Once we remove the seasonal component, it is clear that there is a trend as the p-value is much lower than 0.05 meaning that we reject the null hypothesis that the data is stationary.