Assignment 8: Time Series Analysis

Jordan Mullens

Spring 2023

OVERVIEW

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

Directions

1. Rename this file `<JordanMullens>_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
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.0
## v tibble  3.1.8     v stringr 1.5.0
## v tidyr   1.2.1     v forcats 0.5.2
## v readr   2.1.3
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(trend)
library(zoo)
```

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

```
library(Kendall)
library(tseries)
```

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

```
library(here)
```

```
## here() starts at /home/guest/EDA-Spring2023
```

```
here()
```

```
## [1] "/home/guest/EDA-Spring2023"
```

```
mytheme <- theme_classic(base_size = 14) +
  theme(axis.text = element_text(color = "darkblue"),
        legend.position = "top")
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.

```
#2
O3.NC2010 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv"), stringsAsFact

O3.NC2011 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv"), stringsAsFact

O3.NC2012 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv"), stringsAsFact

O3.NC2013 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv"), stringsAsFact

O3.NC2014 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv"), stringsAsFact

O3.NC2015 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv"), stringsAsFact

O3.NC2016 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv"), stringsAsFact

O3.NC2017 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv"), stringsAsFact

O3.NC2018 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv"), stringsAsFact

O3.NC2019 <- read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv"), stringsAsFact

GaringerOzone <- rbind(O3.NC2010, O3.NC2011, O3.NC2012, O3.NC2013, O3.NC2014, O3.NC2015, O3.NC2016, O3.
```

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
GaringerOzone <- GaringerOzone %>%
  mutate(Date = mdy(Date))

class(GaringerOzone$Date)
```

```
## [1] "Date"
```

```
#4
GaringerOzone.wrangled <- select(GaringerOzone, Date, Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_V

#5
Days <- as.data.frame(seq(as.Date("2010-01-01"), as.Date("2019-12-31"), by = "day"))
colnames(Days)[1] = "Date"
```

```r
#6
GaringerOzone <-left_join(Days, GaringerOzone.wrangled)
```

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

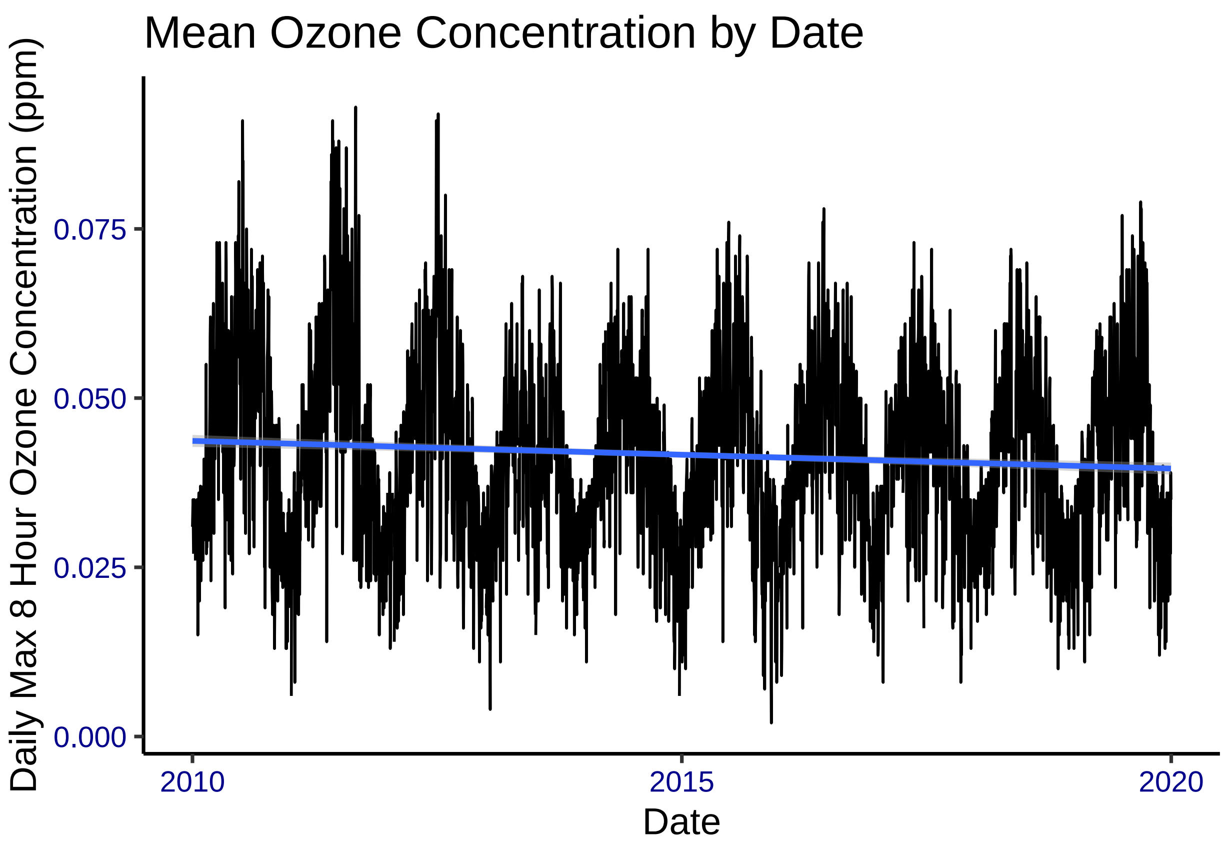
Visualize

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
ggplot(GaringerOzone, aes(x=Date, y=Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line() +
  geom_smooth(method="lm") +
  labs(
    title ='Mean Ozone Concentration by Date',
    x = 'Date',
    y = 'Daily Max 8 Hour Ozone Concentration (ppm)')
```

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

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



Answer: Yes. The plot suggests ozone concentration is decreasing over time. We are unsure whether this decrease is noteworthy, so a test is required to determine whether or not it's significant.

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
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
```

```
summary(GaringerOzone$Daily.Max.8.hour.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
```

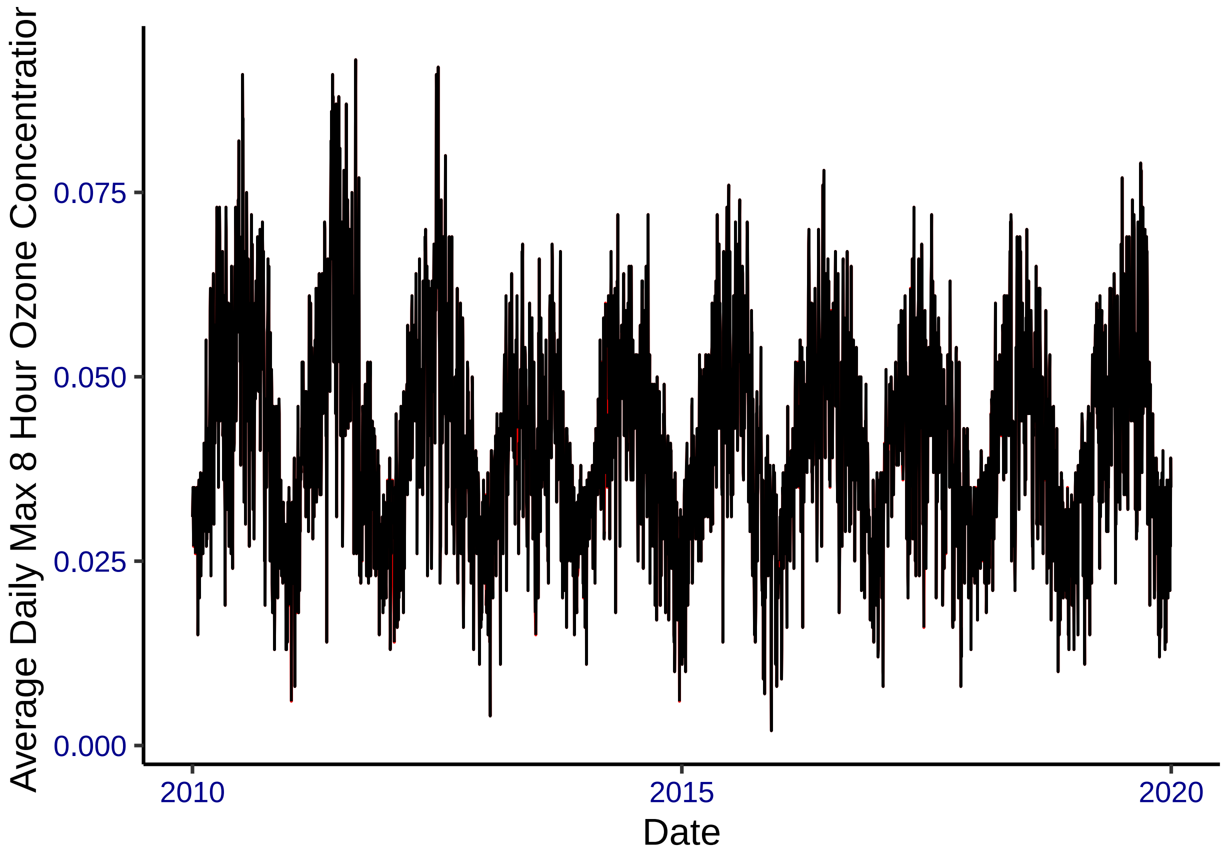
```
# Adding new column with no missing obs
# In real applications you will simply replace NAs
GaringerOzone_clean <-
  GaringerOzone %>%
  mutate(Daily.Ozone.clean = zoo::na.approx(Daily.Max.8.hour.Ozone.Concentration))

summary(GaringerOzone_clean$Daily.Ozone.clean)
```

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

```
#Note the NA is gone

ggplot(GaringerOzone_clean) +
  geom_line(aes(x = Date, y = Daily.Ozone.clean), color = "red") +
  geom_line(aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration), color = "black") +
  ylab("Average Daily Max 8 Hour Ozone Concentration")
```



Answer: Linear interpolation assumes the missing data falls between adjacent data points. This interpolation enabled us to approximate a continuous function from this discrete set of data points. Spline interpolation is useful when readings are taken at irregular intervals and when data sets are particularly noisy with outliers. Our data set has readings at regular intervals. Piecewise constant interpolation is useful for datasets that exhibit sudden changes or discontinuities; utilize when you need to approximate a continuous function that represents the dataset. Because we filled in the discontinuities, a piecewise interpolation may not be appropriate.

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) #group by month first, not by year in the time series on number 9. If the data is out of order, the time series will look like a hump.

```r
#9
#Add month and year columns
GaringerOzone.monthly <- GaringerOzone_clean %>%
  mutate(Month = month(Date)) %>%
  mutate(Year = year(Date))

#create new date column with each month-year combo being set as the first day of the month
GaringerOzone.monthly <- mutate(GaringerOzone.monthly, Date = my(paste0(Month,"-",Year)))

#create aggregated data
```

```
GaringerOzone.monthly <- GaringerOzone.monthly %>%
  group_by(Year, Month, Date) %>%
  summarise(Mean.Monthly.Ozone = mean(Daily.Ozone.clean))
```

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

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.

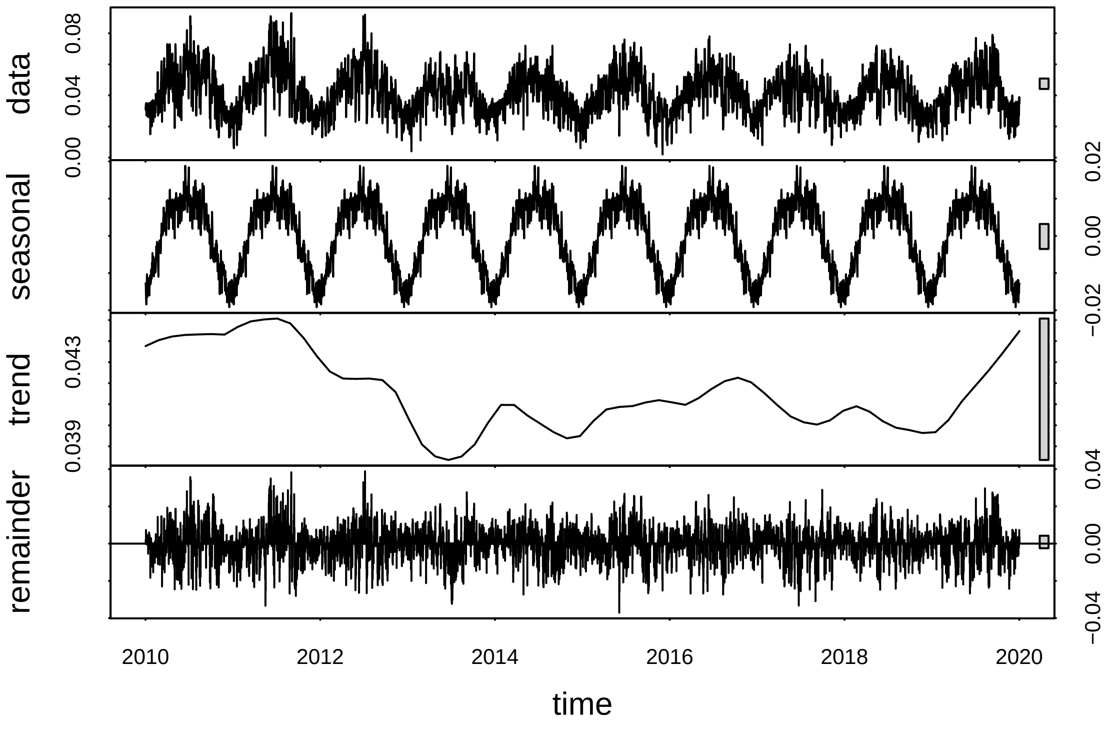
```
#10
#Daily ts
f_month <- month(first(GaringerOzone_clean$Date))
f_year <- year(first(GaringerOzone_clean$Date))
GaringerOzone.daily.ts <- ts(GaringerOzone_clean$Daily.Ozone.clean, frequency = 365, start=c(f_year, f_

#Monthly ts
f_month_M <- first(GaringerOzone.monthly$Month)
f_year_M <- first(GaringerOzone.monthly$Year)
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$Mean.Monthly.Ozone, frequency = 12, start=c(f_year
```

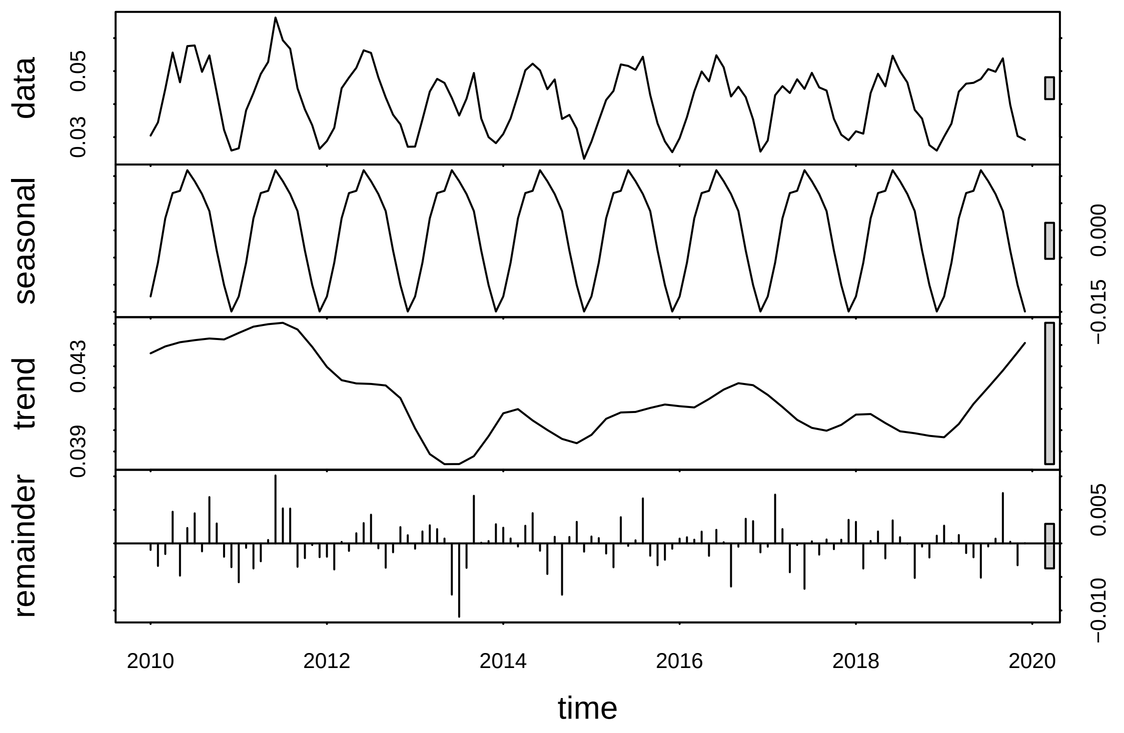
11. Decompose the daily and the monthly time series objects and plot the components using the `plot()` function.

```
#11
#a good rule of thumb the time series has a monthly trend. the trends for the monthly and daily should

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



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



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
# Run SMK test
GaringerOzone.monthly_data_trend1 <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)

# Inspect results
GaringerOzone.monthly_data_trend1
```

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

```
summary(GaringerOzone.monthly_data_trend1)
```

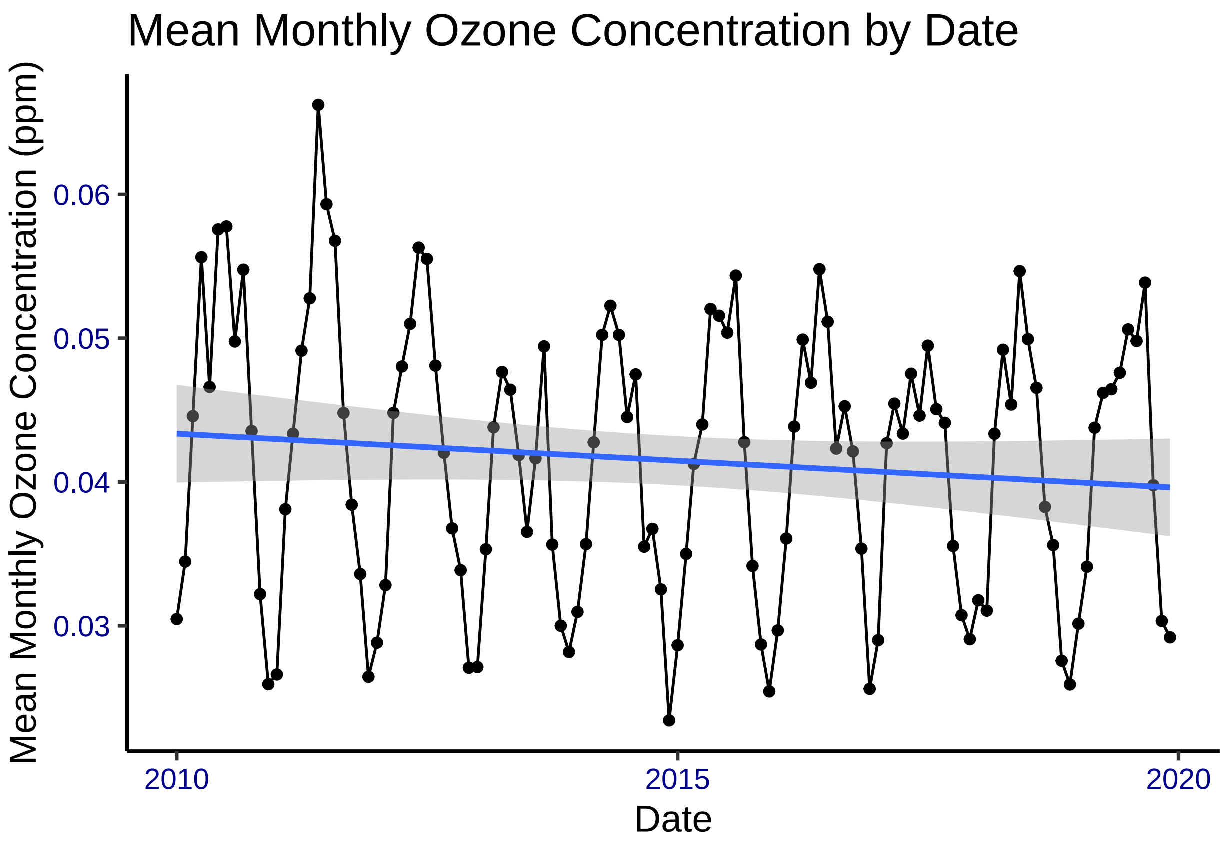
```
## Score =  -77 , Var(Score) = 1499
## denominator =  539.4972
## tau = -0.143, 2-sided pvalue =0.046724
```

Answer: The seasonal Mann-Kendall is most appropriate because there is seasonality in the data and the data is non-parametric. A seasonal Mann-Kendall is appropriate when there is a time series data set that exhibits a seasonal pattern. You would utilize a Mann Kendall when data is missing. We interpolated missing data and no longer have missing data so we can run a seasonal Mann Kendall.

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.

```r
#13
Garinger.mean.monthly.ozone.plot <-
ggplot(GaringerOzone.monthly, aes(x = Date, y = Mean.Monthly.Ozone)) +
  geom_point() +
  geom_line() +
  geom_smooth( method = lm ) +
  labs(
    title ='Mean Monthly Ozone Concentration by Date',
    x = 'Date',
    y = 'Mean Monthly Ozone Concentration (ppm)')
print(Garinger.mean.monthly.ozone.plot)
```

```
## `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: If we hypothesized that mean monthly ozone concentrations change over timne (2010 to 2019), then we would reject the null hypothesis (there is no significant change in between mean monthly ozone concentrations from 2010 to 2019) Initial p value is slightly significant (0.049672), which suggests that mean monthly ozone concentrations are changing over time.

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.

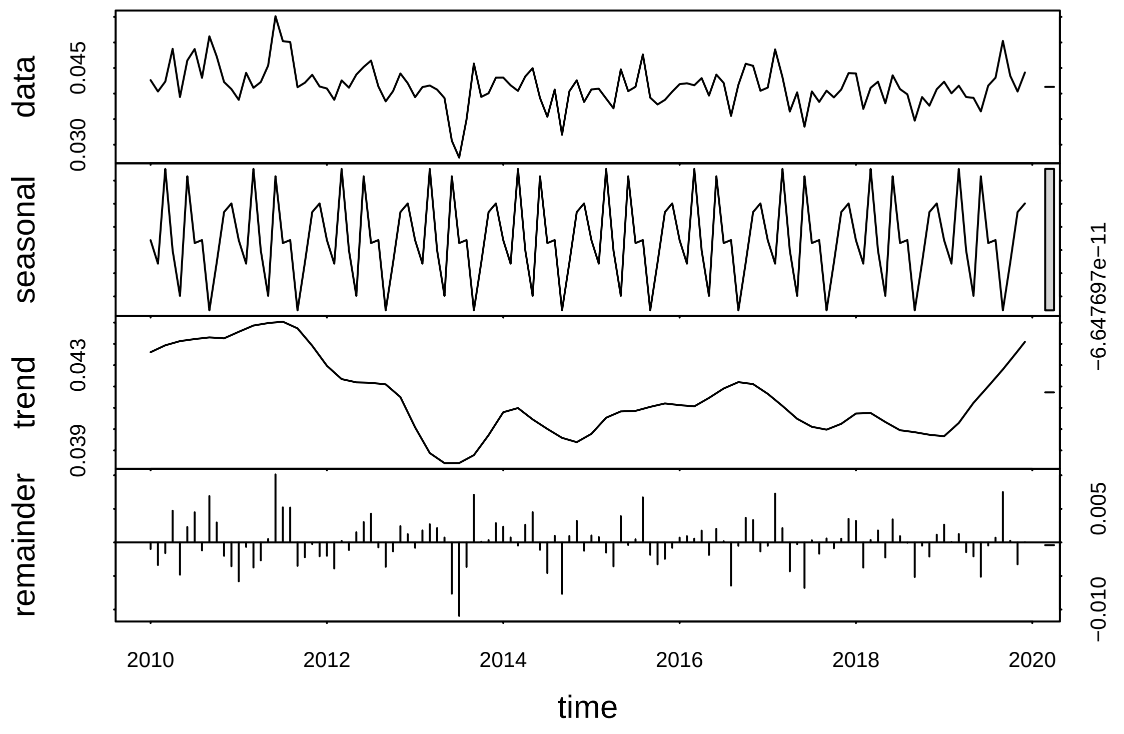
```
#15
# We can extract the components and turn them into data frames
GaringerOzone.monthly_Components <- as.data.frame(GaringerOzone.monthly_Decomposed$time.series[,1:3])

GaringerOzone.monthly_Components <- mutate(GaringerOzone.monthly_Components,
        Observed = GaringerOzone.monthly$Mean.Monthly.Ozone,
        Date = GaringerOzone.monthly$Date)

GaringerOzone.monthly.nonseasonal = GaringerOzone.monthly_Components[,4] - GaringerOzone.monthly_Compone

GaringerOzone.monthly.nonseasonal.ts <- ts(GaringerOzone.monthly.nonseasonal, start= c(2010,1), frequen

GaringerOzone.monthly.nonseasonal.Decomposed <- stl( GaringerOzone.monthly.nonseasonal.ts, s.window = "
plot(GaringerOzone.monthly.nonseasonal.Decomposed)
```



```
#16
# Run SMK test
GaringerOzone.monthly.nonseasonal_data_trend1 <- Kendall::MannKendall(GaringerOzone.monthly.nonseasonal

GaringerOzone.monthly.nonseasonal_data_trend1
```

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

```
summary(GaringerOzone.monthly.nonseasonal_data_trend1)
```

```
## Score =  -1179 , Var(Score) = 194365.7
## denominator =  7139.5
## tau = -0.165, 2-sided pvalue =0.0075402
```

> Answer: The final p value is 0.00754 which is more significant than our initial p value when we included the seasonal information. When you do remove the seasonal, you may have a significant trend. With the seasonal you may not have a significant trend.