Assignment 8: Time Series Analysis

Claire Pajka

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

```
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 errors
```

```
library(lubridate)
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 C:/Users/cepaj/OneDrive/Documents/EDE_Fall2023
```

```
getwd()
```

```
## [1] "C:/Users/cepaj/OneDrive/Documents/EDE_Fall2023"
```

```
mytheme <- theme_classic(base_size=12)+
  theme(
    plot.title= element_text(size= 12, color = "darkgreen",
                             face="bold", hjust = 0.5),
    axis.text = element_text(size = 12, color = "black"),
    legend.position = 'right',
    legend.background = element_blank(),
    legend.box.background = element_rect(colour = "black"),
    plot.background = element_rect(color = "black"),
    axis.line = element_line(size = 0.65, color = "black"))
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
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.

```
#1
#use rbind to combine into one dataframe or maybe merge. not sure which one
#here()
ozone2010 <- read.csv(
```

```
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2010_raw.csv"),
  stringsAsFactors = TRUE)
ozone2011 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2011_raw.csv"),
  stringsAsFactors = TRUE)
ozone2012 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2012_raw.csv"),
  stringsAsFactors = TRUE)
ozone2013 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2013_raw.csv"),
  stringsAsFactors = TRUE)
ozone2014 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2014_raw.csv"),
  stringsAsFactors = TRUE)
ozone2015 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2015_raw.csv"),
  stringsAsFactors = TRUE)
ozone2016 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2016_raw.csv"),
  stringsAsFactors = TRUE)
ozone2017 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2017_raw.csv"),
  stringsAsFactors = TRUE)
ozone2018 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2018_raw.csv"),
  stringsAsFactors = TRUE)
ozone2019 <- read.csv(
  here("Data", "Raw", "Ozone_TimeSeries", "EPAair_O3_GaringerNC2019_raw.csv"),
  stringsAsFactors = TRUE)
GaringerOzone2010_2019 <- rbind(ozone2010, ozone2011,
                                ozone2012, ozone2013, ozone2014,
                                ozone2015, ozone2016, ozone2017,
                                ozone2018, ozone2019)
```

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
GaringerOzone2010_2019$Date <- mdy(GaringerOzone2010_2019$Date)
class(GaringerOzone2010_2019$Date) #checking that this is now a date class
```

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

```
view(GaringerOzone2010_2019)
# 4
subsetGaringerOzone <- GaringerOzone2010_2019 %>%
  select(Date, Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
view(subsetGaringerOzone)

# 5
subsetGaringerOzone$Date[subsetGaringerOzone$Date == ""] <- NA
Garinger_sequence <- seq(as.Date("2010-01-01"), as.Date("2019-12-31"), by= "days")
Days <- data.frame(Date= Garinger_sequence)
view(Days)

# 6
GaringerOzone <- left_join(Days, subsetGaringerOzone, by= "Date")
view(GaringerOzone)
dim(GaringerOzone)
```

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

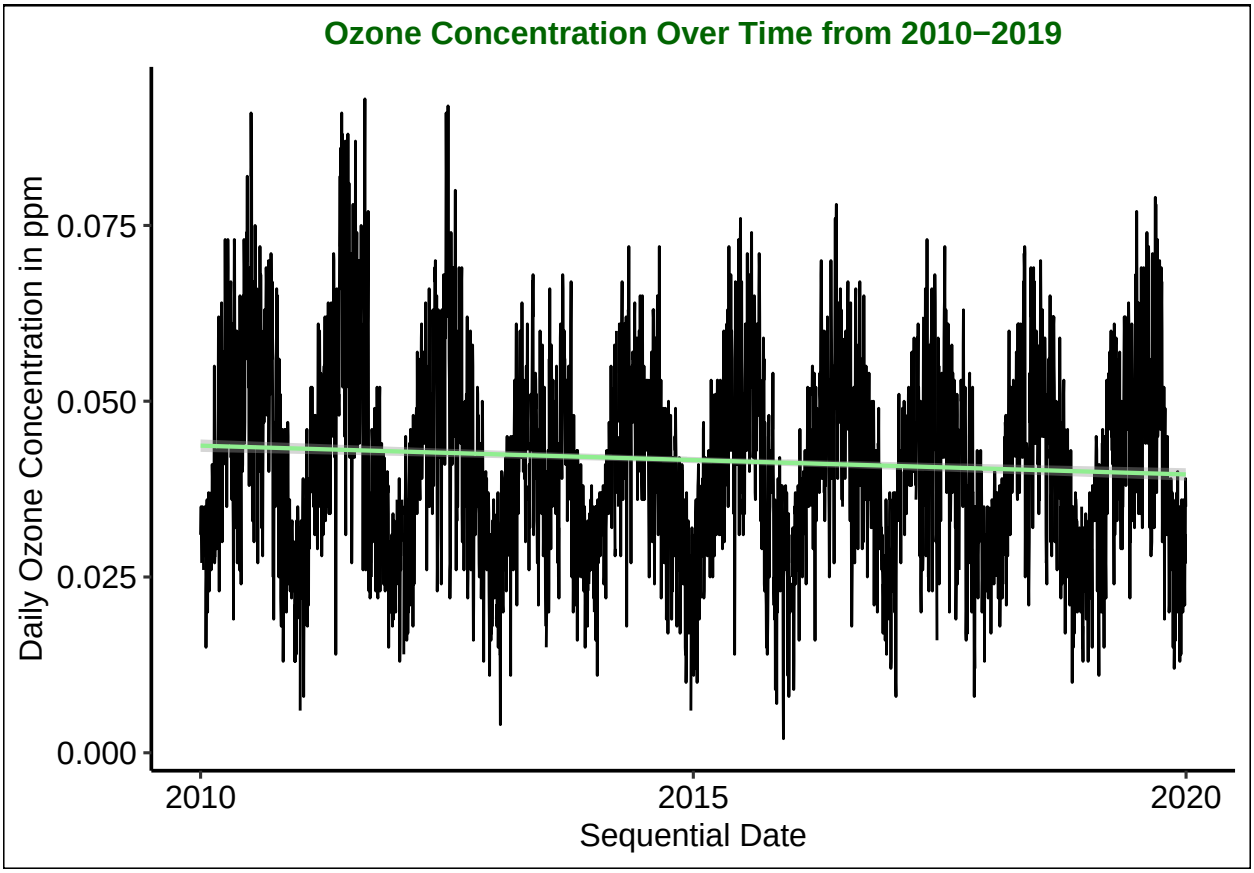
Visualize

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?

```
#7
Ozone_time_plot <-
  ggplot(GaringerOzone,
         aes(
           y=Daily.Max.8.hour.Ozone.Concentration,
           x= Date))+
  geom_line()+
  mytheme+
  geom_smooth(method = lm, linewidth = .75, color = "lightgreen")+
  labs(title= "Ozone Concentration Over Time from 2010-2019",
       fontface = "bold")+
  xlab("Sequential Date")+
  ylab("Daily Ozone Concentration in ppm")
print(Ozone_time_plot)
```

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

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



Answer: The trend line suggests a slight decrease in daily ozone concentrations from 2010 to 2019. We can see that there are natural fluctuations of season for each year, with the lows of each period being the December and January daily ozone concentrations for each year. Looking at the seasonal patterns alone, it is hard to discern whether there is an overall trend in daily ozone concentration, but if we add a smoothed trend line (in light green on this graph), we can see the daily ozone concentration decreases slightly from 2010 to 2019.

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
GaringerOzone_clean <-
  GaringerOzone %>%
  mutate(Daily.Max.8.hour.Ozone.Concentration = zoo::na.approx(Daily.Max.8.hour.Ozone.Concentration))
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
```

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

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

Answer: Linear interpolation is appropriate for time series data because by connecting points on either side of the data, it assumes that our daily data relationship is linear. Using spline interpolation would use the quadratic formula to connect the data, leading to a curved line between interpolated points. Using a piecewise constant assumes that the data is equal to the value directly before or after it, and looking at our data, we can see that the values follow a linear relationship.

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
GaringerOzone.monthly <-
  GaringerOzone_clean %>%
  mutate(Year = year(Date)) %>%
  mutate(Month = month(Date)) %>%
  group_by(Year, Month) %>%
  summarise(MonthlyAverage = mean(Daily.Max.8.hour.Ozone.Concentration))
```

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

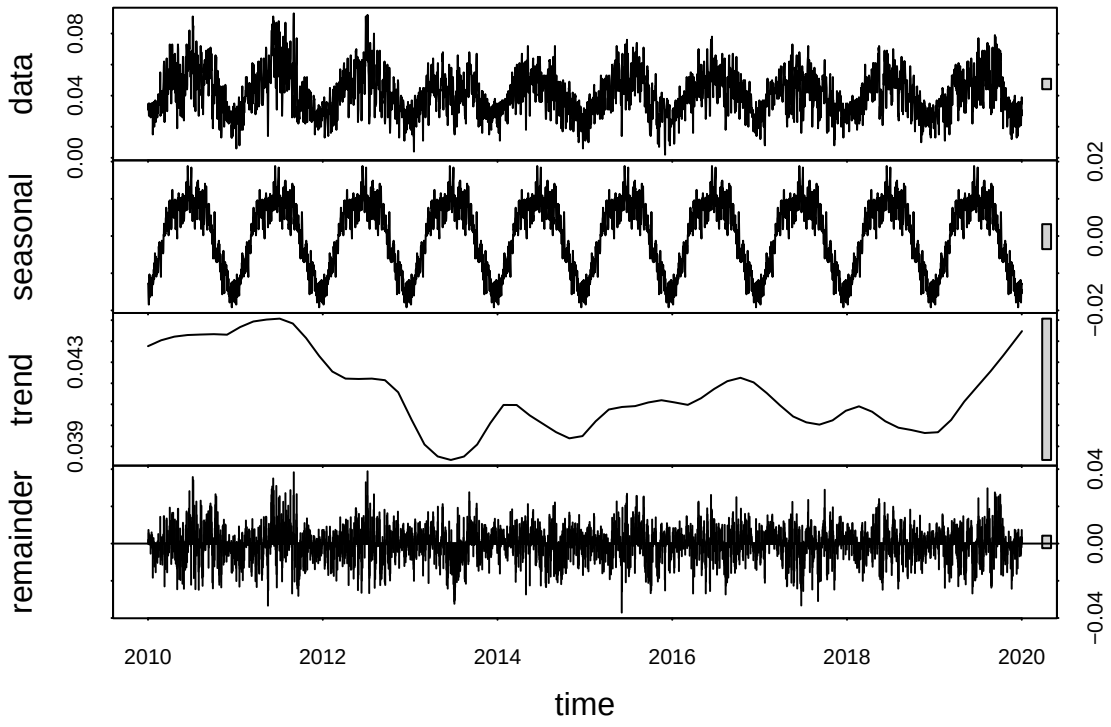
```
GaringerOzone.monthly <- GaringerOzone.monthly %>%
  mutate(Date= make_date(Year, Month, 1))
  view(GaringerOzone.monthly)
```

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
f_month <- month(first(GaringerOzone_clean$Date))
f_year <- year(first(GaringerOzone_clean$Date))
GaringerOzone.daily.ts <- ts(GaringerOzone_clean$Daily.Max.8.hour.Ozone.Concentration,
                             frequency=365, start=c(f_year, f_month))
#GaringerOzone.daily.ts


f_month <- month(first(GaringerOzone.monthly$Date))
f_year <- year(first(GaringerOzone.monthly$Date))
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$MonthlyAverage, frequency=12,  start=c(f_year, f_m
#GaringerOzone.monthly.ts
```

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

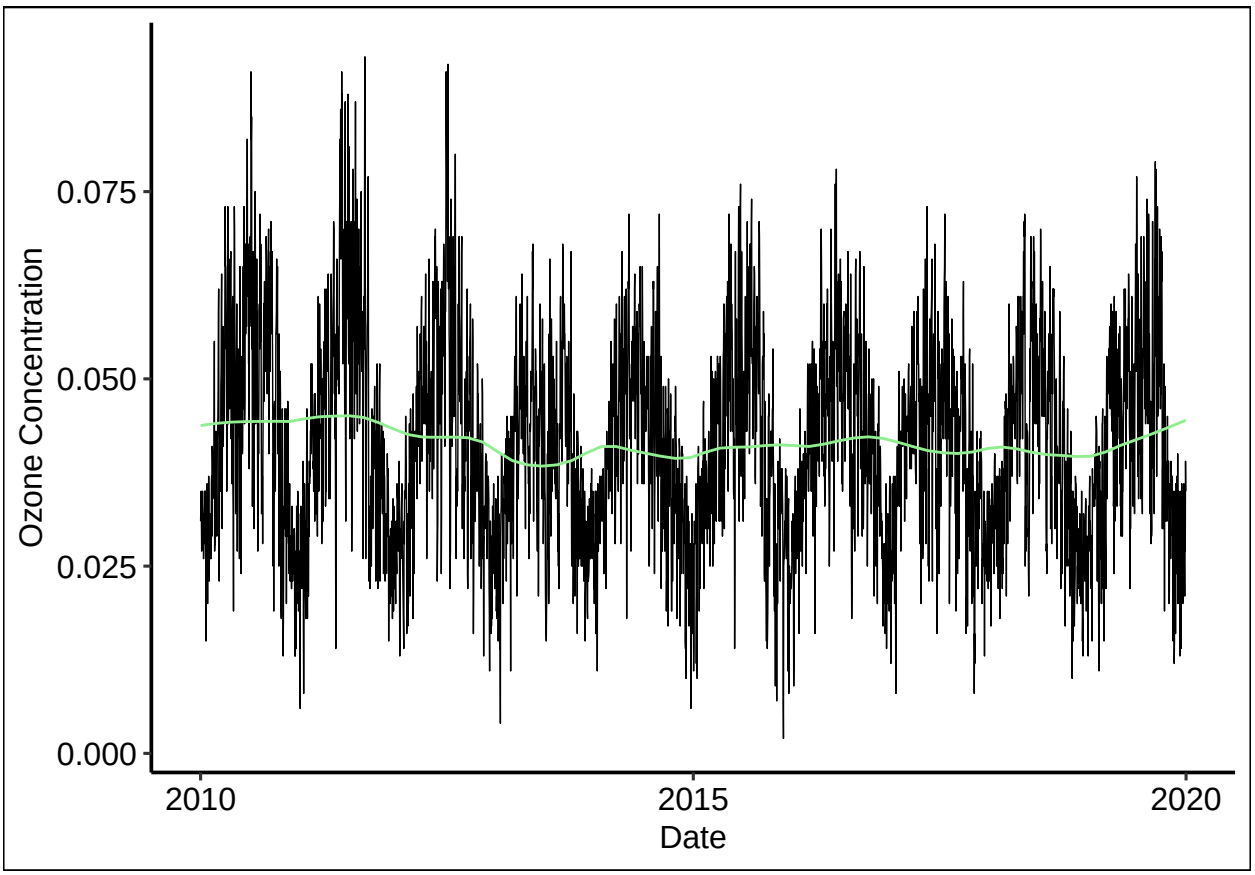
```
#11
dailyozone_decomp <- stl(GaringerOzone.daily.ts, s.window = "periodic")
plot(dailyozone_decomp)
```



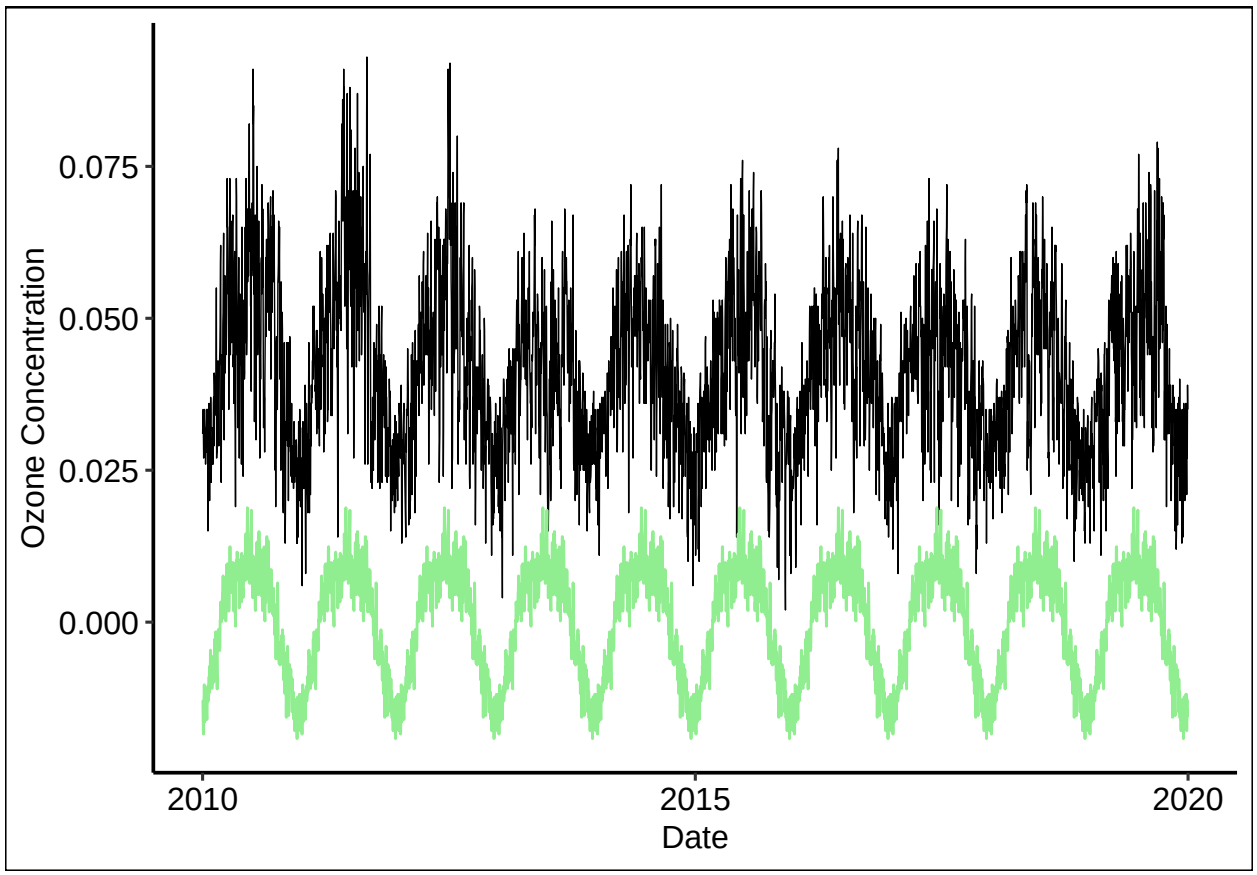

```
Dailyozone_Components <- as.data.frame(dailyozone_decomp$time.series [,1:3])
Dailyozone_Components <- mutate(Dailyozone_Components,
                                Observed = GaringerOzone_clean$Daily.Max.8.hour.Ozone.Concentration,
                                Date = GaringerOzone_clean$Date)

Dailyozone_Components_trendplot <-
  ggplot(Dailyozone_Components) +
  geom_line(aes(y= Observed, x = Date), size = 0.25)+
  geom_line(aes(y = trend, x = Date), color = "lightgreen")+
  ylab("Ozone Concentration")
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

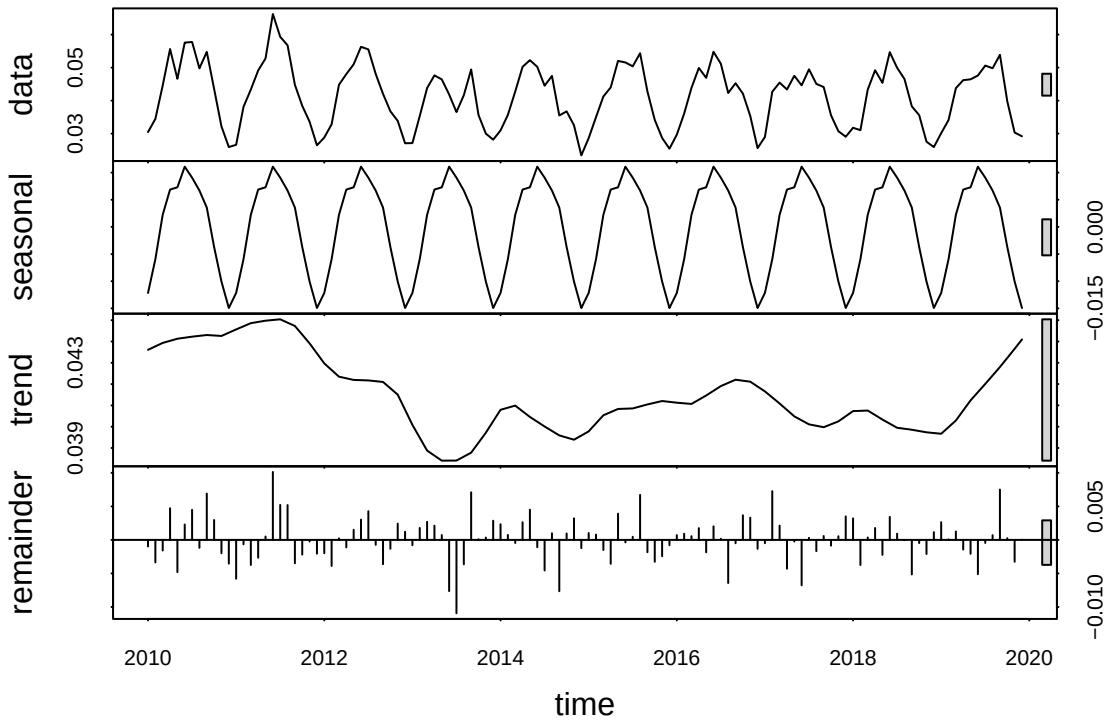
```
plot(Dailyozone_Components_trendplot)
```



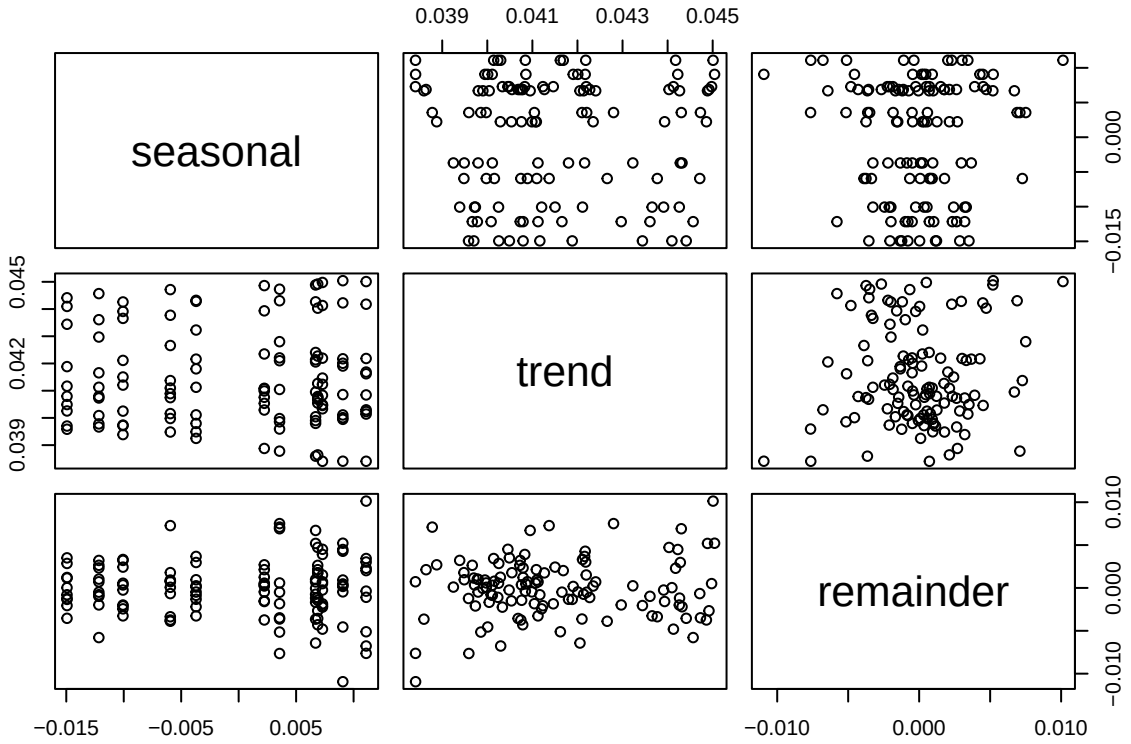

```
Dailyozone_Components_seasonalplot <-
  ggplot(Dailyozone_Components) +
  geom_line(aes(y= Observed, x = Date), size = 0.25)+
  geom_line(aes(y = seasonal, x = Date), color = "lightgreen")+
  ylab("Ozone Concentration")
plot(Dailyozone_Components_seasonalplot)
```



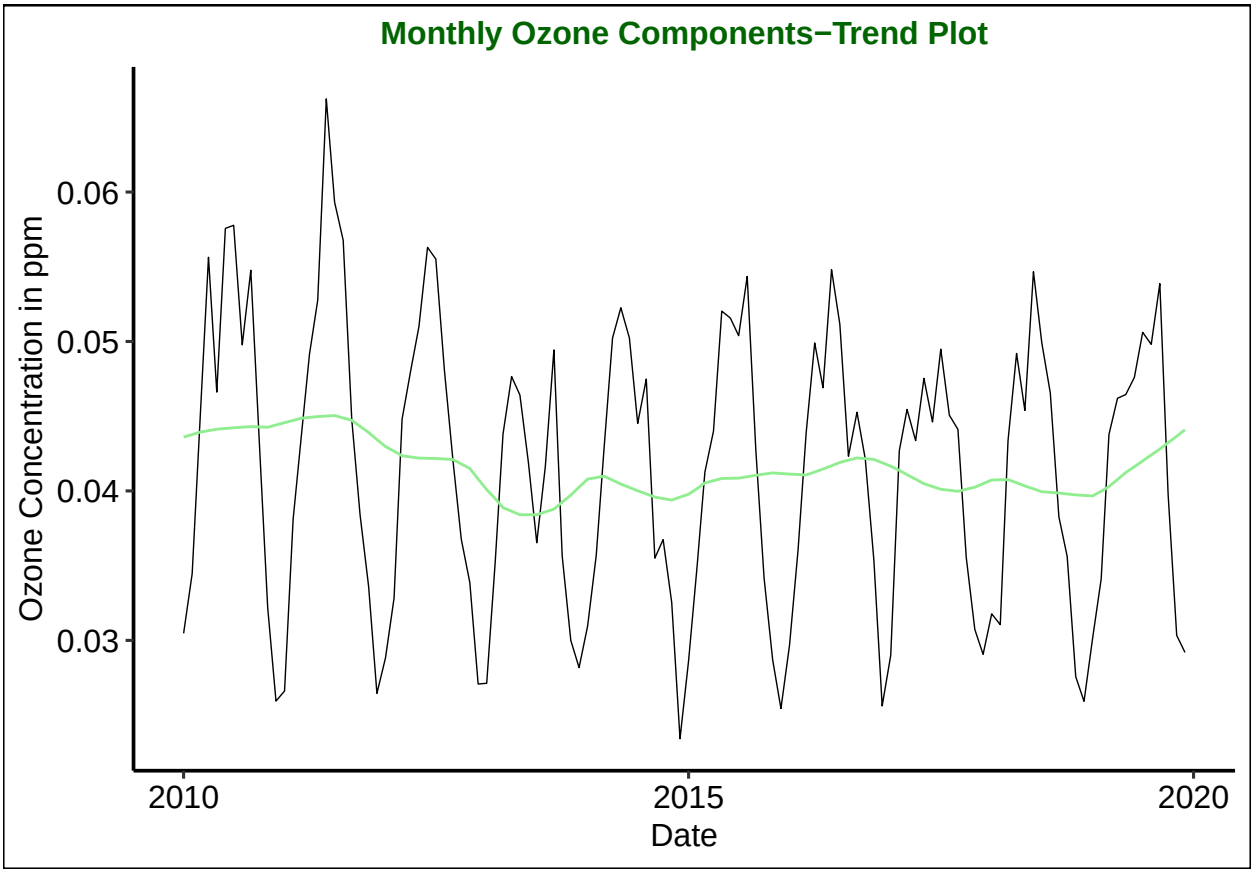

```
#monthly is below
monthlyozone_decomp <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
plot(monthlyozone_decomp)
```




```
Monthlyozone_Components <- as.data.frame(monthlyozone_decomp$time.series [,1:3])
plot(Monthlyozone_Components)
```



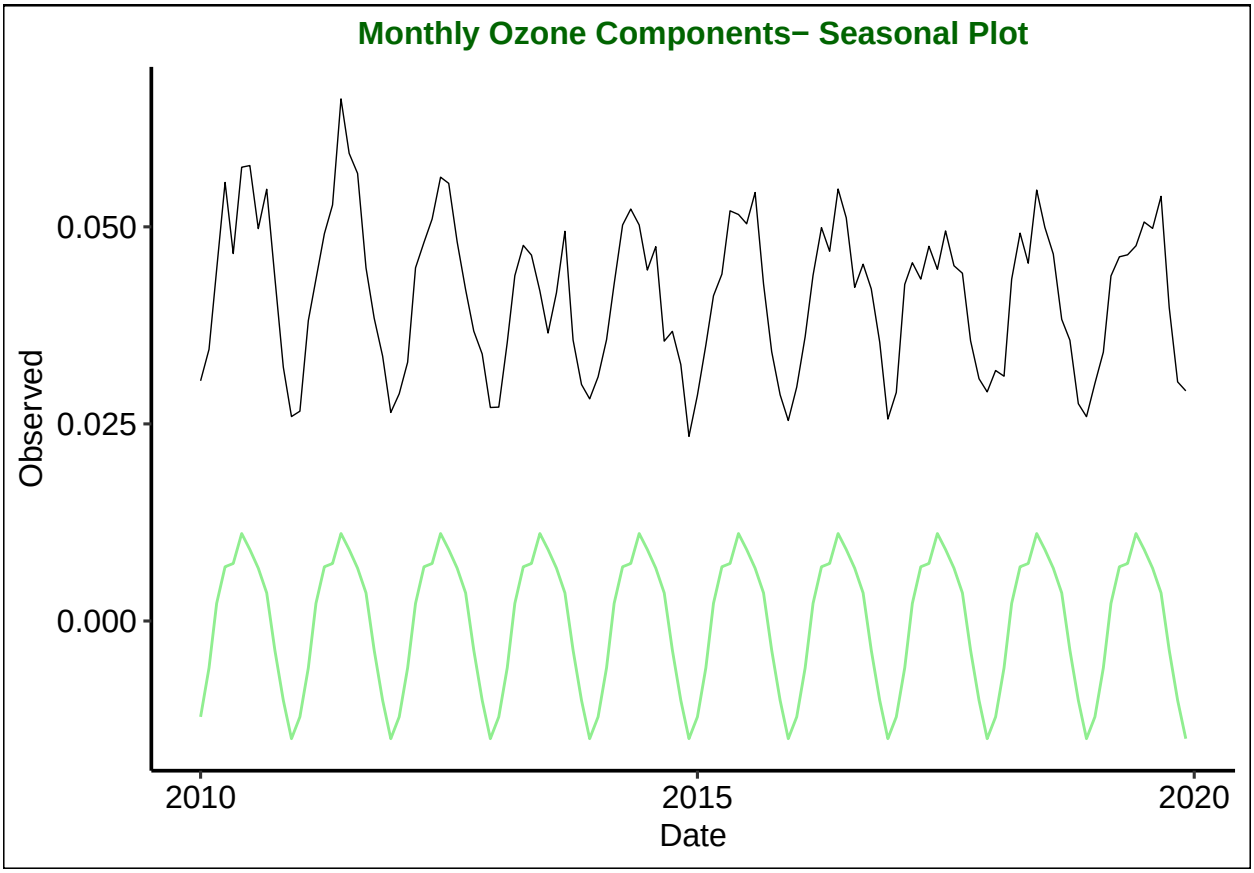

```
Monthlyozone_Components <- mutate(Monthlyozone_Components,
                                  Observed = GaringerOzone.monthly$MonthlyAverage,
                                  Date = GaringerOzone.monthly$Date)

Monthlyozone_Components_trendplot <-
  ggplot(Monthlyozone_Components) +
  geom_line(aes(y= Observed, x = Date), size = 0.25)+
  geom_line(aes(y = trend, x = Date), color = "lightgreen")+
  labs(title="Monthly Ozone Components-Trend Plot")+
  ylab("Ozone Concentration in ppm")
plot(Monthlyozone_Components_trendplot)
```




```
Monthlyozone_Components_seasonalplot <-
  ggplot(Monthlyozone_Components) +
  geom_line(aes(y= Observed, x = Date), size = 0.25)+
  geom_line(aes(y = seasonal, x = Date), color = "lightgreen")+
  labs(title= "Monthly Ozone Components- Seasonal Plot")
  ylab("Ozone Concentration in ppm")
```

```
## $y
## [1] "Ozone Concentration in ppm"
##
## attr(,"class")
## [1] "labels"
```

```
plot(Monthlyozone_Components_seasonalplot)
```



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
monthly_ozone_SMK <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
monthly_ozone_SMK
```

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

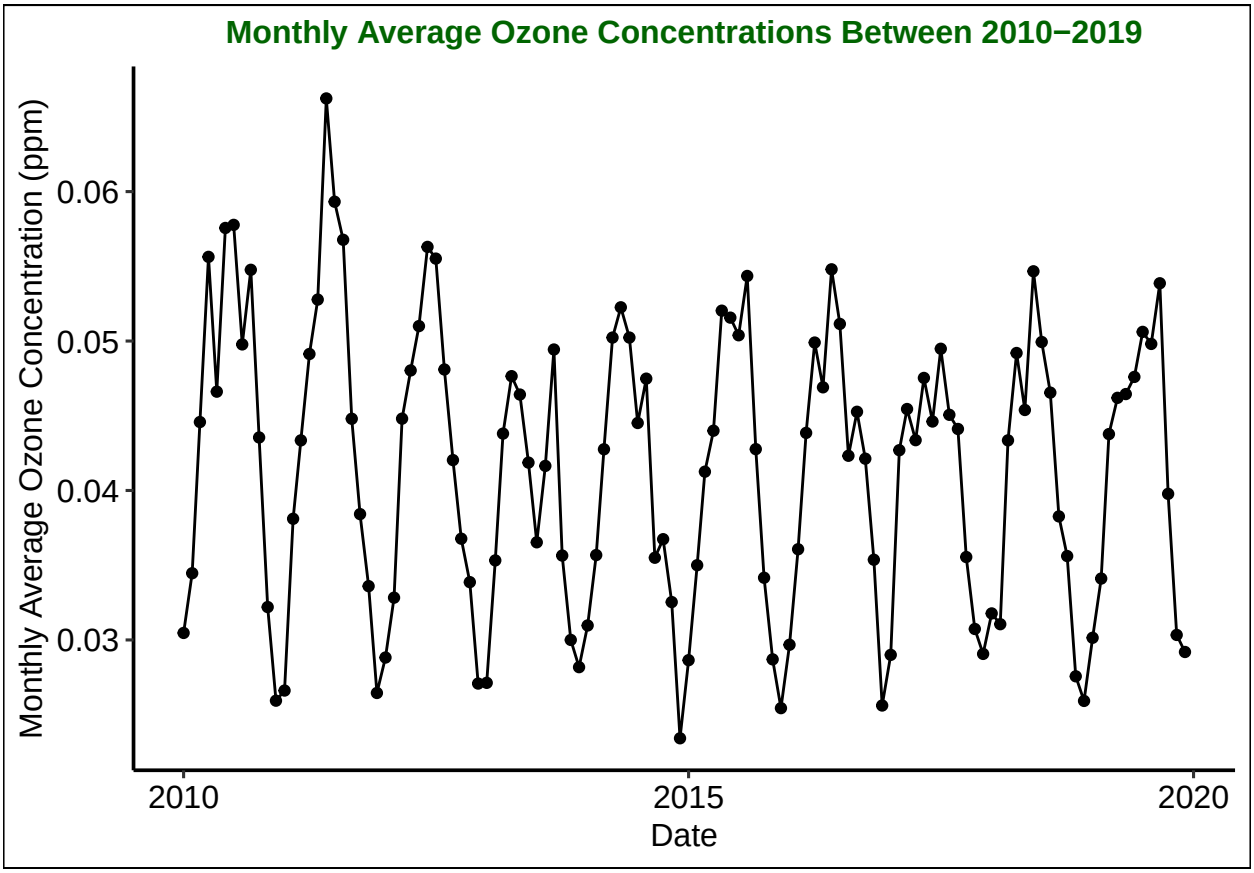
```
summary(monthly_ozone_SMK)
```

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

Answer: The seaonsal Mann-Kendall test is most appropriate for the monthly Ozone series because it accounts for the seasonal variation pattern that occurs. From plotting our data, we can see that the Ozone concentrations rise in the spring and summer, decrease in the fall, and reach their lowest in the winter months. This pattern occurs every year in the data, so the seasonal Mann-Kendall test is best to account for the trends between seasons (like Fall data every year, instead of Fall vs. Winter 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.

```r
# 13
monthly_ozone_plot <-
  ggplot(GaringerOzone.monthly,
         aes(
           y=MonthlyAverage,
           x= Date))+
  geom_point()+
  geom_line()+
  #geom_smooth(method = lm, linewidth = .75, color = "black")+
  labs(title= "Monthly Average Ozone Concentrations Between 2010-2019",
       fontface = "bold")+
  xlab("Date")+
  ylab("Monthly Average Ozone Concentration (ppm)")+
    mytheme
print(monthly_ozone_plot)
```



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: The data suggests that average ozone concentrations are more elevated in warmer summer months than in cooler winter months. This seasonal pattern seems to repeat every year, with December or January of every year from 2010 to 2019 representing the lowest monthly average (in ppm) for each respective year.

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
seasonal_monthly_component <- monthlyozone_decomp$time.series [,1]
ozone_no_seasonal <- monthlyozone_decomp$time.series [,2:3]
#print(ozone_no_seasonal)


#16

MK_ozone_no_seasonal <- Kendall::MannKendall(ozone_no_seasonal)
MK_ozone_no_seasonal
```

```
## tau = -0.568, 2-sided pvalue =< 2.22e-16
```

```
summary(MK_ozone_no_seasonal)
```

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

Answer: The non-seasonal Mann Kendall: Score = -16300 , Var(Score) = 1545533, denominator = 28680 tau = -0.568, 2-sided pvalue =< 2.22e-16 Seasonal Mann Kendall: Score = -77 , Var(Score) = 1499, denominator = 539.4972 tau = -0.143, 2-sided pvalue =0.046724 Both scores are negative, which indicates a negative trend for both series. Both p values are below 0.05, which means that we reject the null that there is no trend present (we can conclude that there is a monotonic trend present) and both taus are negative, which mean that the monotonic trend is decreasing. The p value of the seasonal Mann Kendall test is larger than the pvalue of the non seasonal Mann-Kendall test.