Assignment 8: Time Series Analysis

Meg Manning

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

```
# Set up session
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
```

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

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

```
library(here)
```

```
## here() starts at /Users/margaretmanning/Desktop/GitHub/ENVIRON 872/EDE_Fall2023
```

```
here()
```

```
## [1] "/Users/margaretmanning/Desktop/GitHub/ENVIRON 872/EDE_Fall2023"
```

```
# Set theme
mytheme <- theme_classic(base_size = 14) +
  theme(axis.text = element_text(color = "black"),
        legend.position = "top",
        panel.background = element_rect(fill = "beige"))
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 Import 10 datasets
EPAair_2010 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2011 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2012 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2013 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2014 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2015 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2016 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2017 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2018 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2019 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv",
                        stringsAsFactors = TRUE)

# Combine into one dataframe
GaringerOzone <- rbind(EPAair_2010,
                       EPAair_2011,
```

```
                          EPAair_2012,
                          EPAair_2013,
                          EPAair_2014,
                          EPAair_2015,
                          EPAair_2016,
                          EPAair_2017,
                          EPAair_2018,
                          EPAair_2019)
```

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 Fix Date
GaringerOzone$Date <- as.Date(GaringerOzone$Date, format = "%m/%d/%Y")
class(GaringerOzone$Date)
```

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

```
# 4 Wrangle data for Date, Daily.Max.8.hour.Ozone.Concentration, and DAILY_AQI_VALUE
GaringerOzone_processed <-GaringerOzone %>%
  select(Date,
         Daily.Max.8.hour.Ozone.Concentration,
         DAILY_AQI_VALUE)

# 5 Create new dataframe with sequence of dates from 2010-01-01 to 2019-12-31

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

# 6 Use "left_join" to combine the dataframes
GaringerOzone <- left_join(Days, GaringerOzone_processed)
```

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

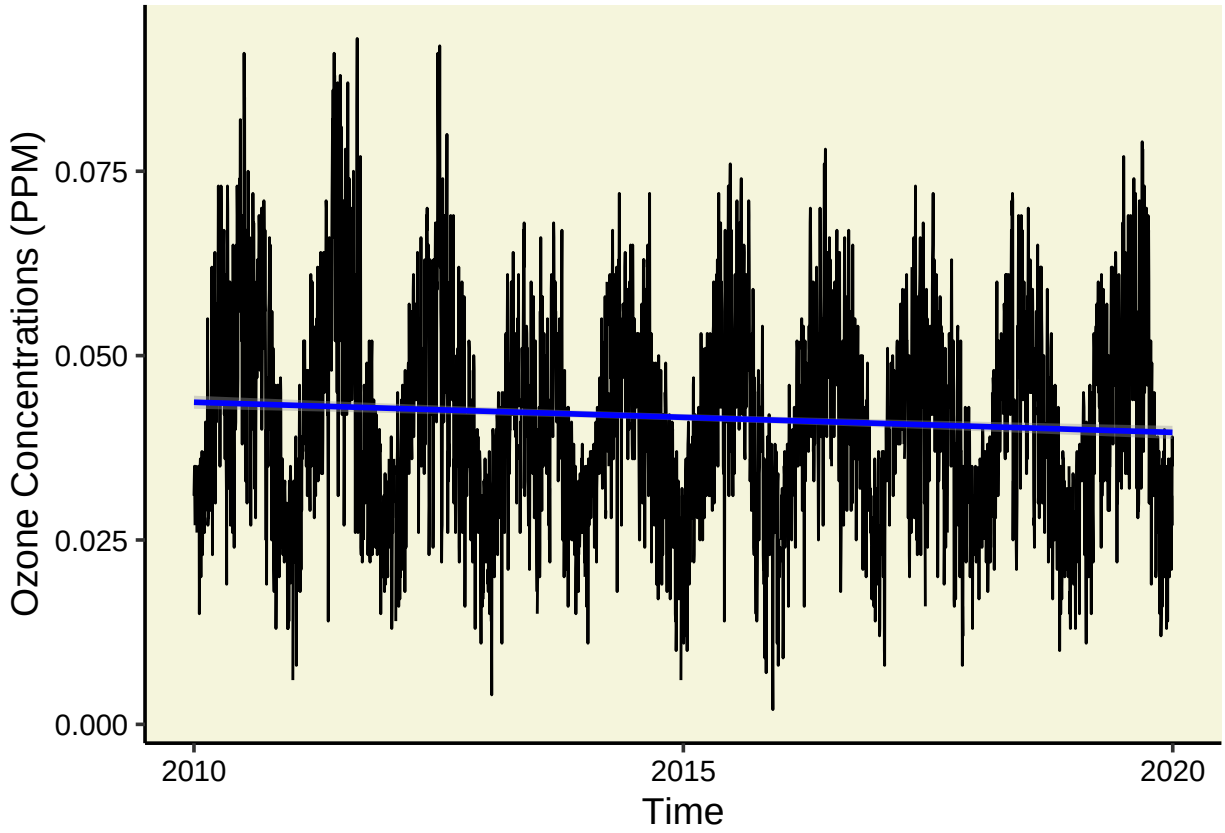
Visualize

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?

```
#7 Create line plot of ozone concentrations over time
ozonebytime <- ggplot(GaringerOzone,
                          aes(x = Date,
                              y = Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line() +
  geom_smooth(method = "lm",
              col = "blue") +
  labs(x = "Time",
       y = "Ozone Concentrations (PPM)")
print(ozonebytime)
```

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

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



Answer: Based on this plot, it looks like there may be a slightly negative trend as seen by the blue linear regression with a slight negative slope. However, since it's hard to tell if this is a trend of the data, we will need to run a few statistical tests.

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 to fill NAs for Ozone Concentration
GaringerOzone_clean <- GaringerOzone %>%
  mutate(Daily.Max.8.hour.Ozone.Concentration =
           zoo::na.approx(Daily.Max.8.hour.Ozone.Concentration))
#Check for NAs
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: We used a linear interpolation because this method is similar to a "connect the dots" approach. Any missing data are assumed to fall between the previous and next measurement, with a straight line drawn between the known points determining the values of the interpolated data on any given date. We used this because we have small gaps within the data through time, if there were obvious gaps at the beginning or end, or a large enough gap in the middle, we would've had to split the data into different dataframes. This method allows us to interpolate the few missing values and still see ozone concentrations over time.

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

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

```
#Create new Date column with each month-year combo set as first of month
GaringerOzone.monthly <- GaringerOzone.monthly %>%
  mutate(Date = make_date(Year, Month, 1))
```

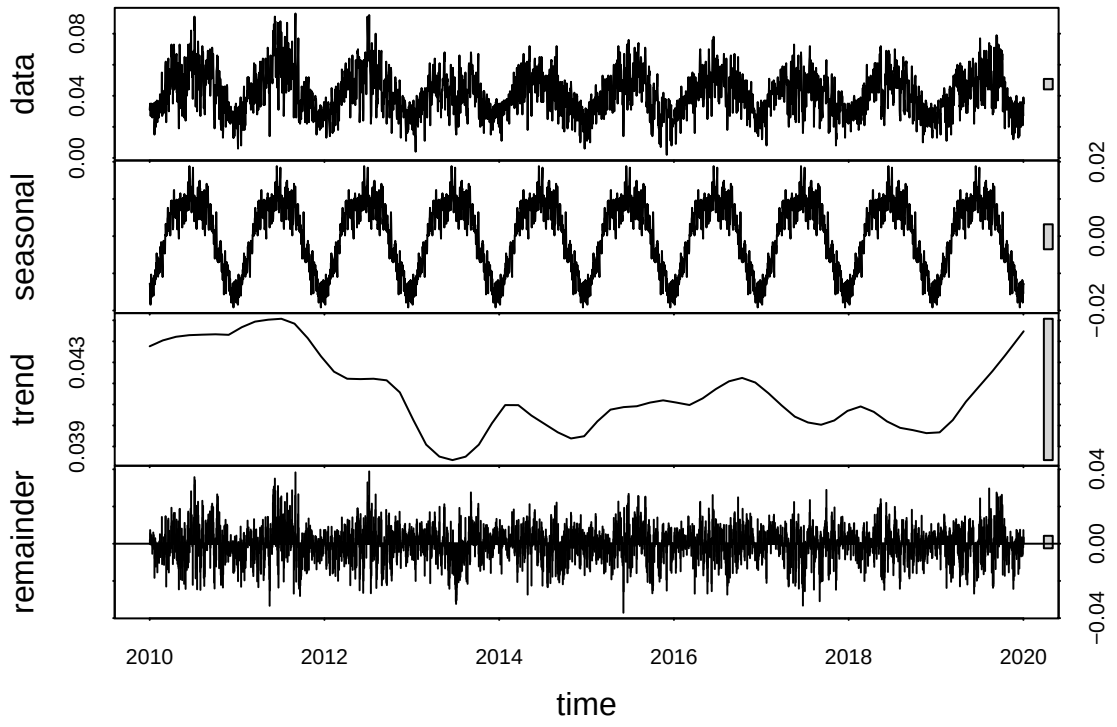
10. Generate two time series objects. Name the first `GaringerOzone.daily.ts` and base it on the dataframe of daily observations. Name the second `GaringerOzone.monthly.ts` and base it on the monthly average ozone values. Be sure that each specifies the correct start and end dates and the frequency of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

```
#10 Create 2 time series objects
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,
```

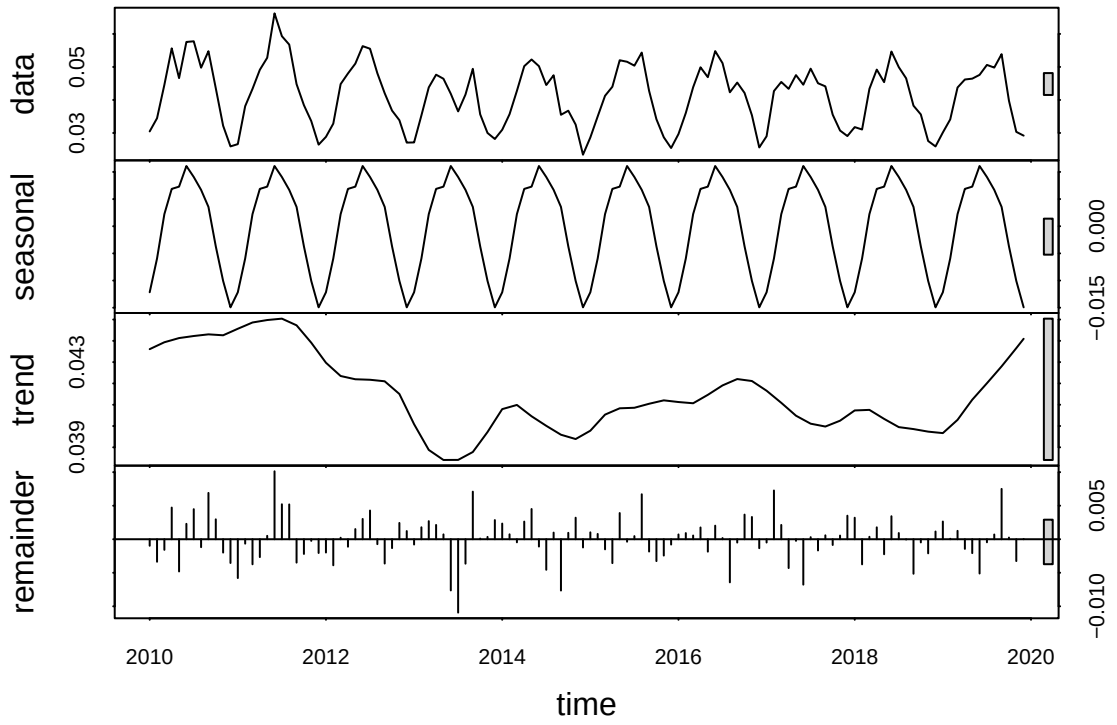
```
                    start=c(f_year,f_month),
                    frequency=365)
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$MonthlyOzone,
                    start=c(f_year,f_month),
                    frequency=12)
```

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

```
#11 Decompose the ts objects and plot using plot()
GaringerOzone.daily.decomp <- stl(GaringerOzone.daily.ts,s.window = "periodic")
plot(GaringerOzone.daily.decomp)
```




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



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 using the Mann-Kendall
GaringerOzone.monthly.trend <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
GaringerOzone.monthly.trend
```

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

```
summary(GaringerOzone.monthly.trend)
```

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

```
#Run SMK Test
GaringerOzone.monthly.trend2 <- trend::smk.test(GaringerOzone.monthly.ts)
# Inspect results
GaringerOzone.monthly.trend2
```

```
##
##  Seasonal Mann-Kendall trend test (Hirsch-Slack test)
##
## data:  GaringerOzone.monthly.ts
## z = -1.963, p-value = 0.04965
## alternative hypothesis: true S is not equal to 0
## sample estimates:
##    S varS
##  -77 1499
```

```
summary(GaringerOzone.monthly.trend2)
```

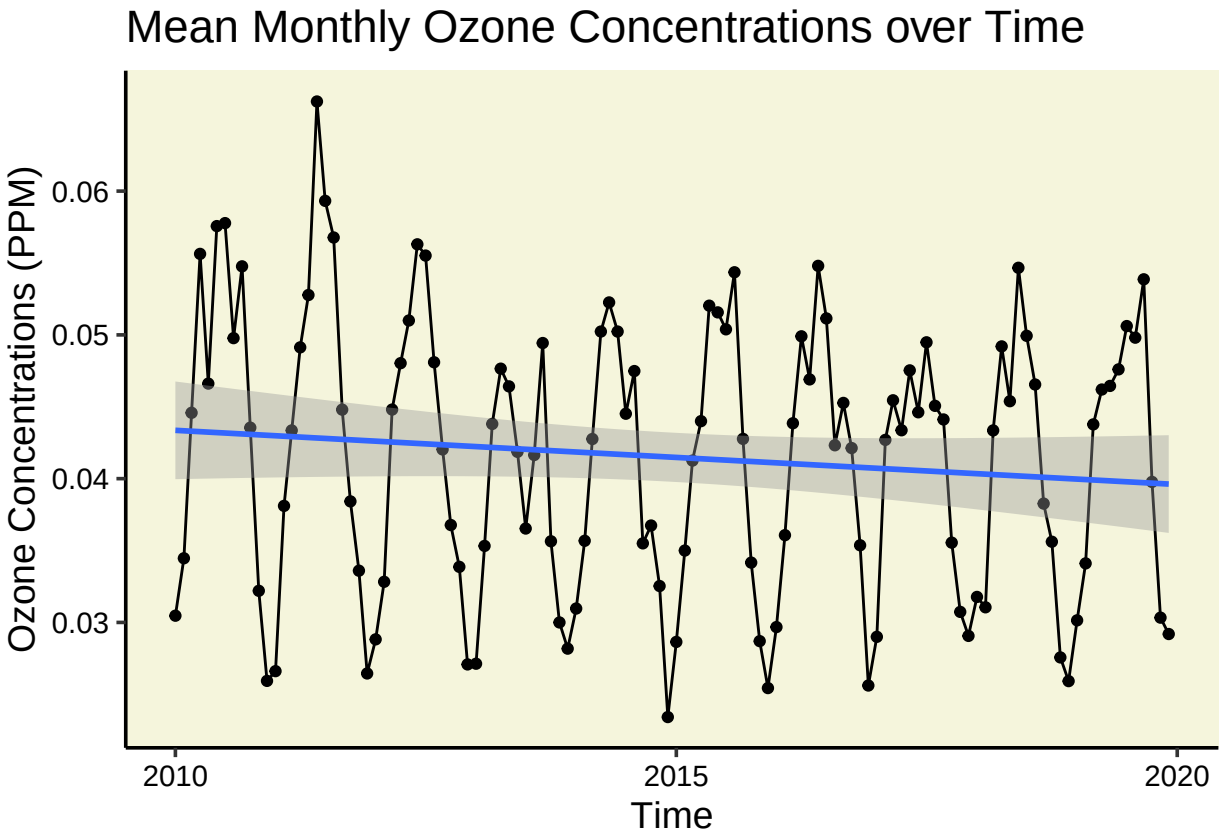
```
##
##  Seasonal Mann-Kendall trend test (Hirsch-Slack test)
##
## data: GaringerOzone.monthly.ts
## alternative hypothesis: two.sided
##
## Statistics for individual seasons
##
## H0
##                       S varS    tau      z Pr(>|z|)
## Season 1:   S = 0   15  125  0.333  1.252  0.21050
## Season 2:   S = 0   -1  125 -0.022  0.000  1.00000
## Season 3:   S = 0   -4  124 -0.090 -0.269  0.78762
## Season 4:   S = 0  -17  125 -0.378 -1.431  0.15241
## Season 5:   S = 0  -15  125 -0.333 -1.252  0.21050
## Season 6:   S = 0  -17  125 -0.378 -1.431  0.15241
## Season 7:   S = 0  -11  125 -0.244 -0.894  0.37109
## Season 8:   S = 0   -7  125 -0.156 -0.537  0.59151
## Season 9:   S = 0   -5  125 -0.111 -0.358  0.72051
## Season 10:   S = 0 -13  125 -0.289 -1.073  0.28313
## Season 11:   S = 0 -13  125 -0.289 -1.073  0.28313
## Season 12:   S = 0  11  125  0.244  0.894  0.37109
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Answer: For this analysis we are interested in the concentration of ozone in ppm on average for each month. Since we want to assess whether our data, in this case the ozone concentration, increases or decreases over time and whether the trend in either direction is statistically significant, we used the Mann-Kendall test to figure out if a monotonic trend exists.

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 Create plot of monthly ozone by time
ozonebytime.monthly <- ggplot(GaringerOzone.monthly,
                              aes(x = Date,
                                  y = MonthlyOzone)) +
  geom_point() +
  geom_line() +
  geom_smooth(method = "lm") +
  labs(x = "Time",
       y = "Ozone Concentrations (PPM)",
       title = "Mean Monthly Ozone Concentrations over Time")
print(ozonebytime.monthly)
```

```
## `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: This graph is depicting the mean monthly ozone between 2010 and 2019, using the Mann-Kendall test to see if there is a trend in the change in ozone concentrations over time by month. With a linear regression plotted on top of the graph, we can visually see what the trend of ozone concentration through time might look like. (Based on the Mann-Kendall test, we can conclude that the trend is significant because our p-value is less than 0.05, and that the slope is negative, or ozone is decreasing over time, because of the tau statistic from the smk test)

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 Subtract seasonal component from the ts
GaringerOzone_Components <- as.data.frame(GaringerOzone.monthly.decomp$time.series[,1:3])

GaringerOzone_noseasonal <- mutate(GaringerOzone_Components,
        Observed = GaringerOzone.monthly$MonthlyOzone,
        Date = GaringerOzone.monthly$Date,
        Observed_NoSeasonality = Observed - seasonal)

GaringerOzone.monthly.ts2 <- ts(GaringerOzone_noseasonal$Observed_NoSeasonality,
```

```
                  start= 2010,
                  frequency=12)

#16 Run Mann-Kendall on the new dataframe without the season component

GaringerOzone.monthly.trend3 <- Kendall::MannKendall(GaringerOzone.monthly.ts2)
GaringerOzone.monthly.trend3
```

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

```
summary(GaringerOzone.monthly.trend3)
```

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

Answer: When removing the seasonal component of the ozone data and running the Mann-Kendall test, we can see that the slope has decreased to -0.165 based on the tau statistic. Our p-value is also less than 0.05 and is much smaller than when we included the seasonal component of the data. This means that the trend is in fact significant and it is a negative trend based on the negative slope.