Assignment 8: Time Series Analysis

Fiona Kelley

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

```r
#Installing and loading the necessary packages.
library(tidyverse)
library(lubridate)
#install.packages("trend")
library(trend)
#install.packages("zoo")
library(zoo)
library(dplyr)
library(here)
#install.packages("Kendall")
library(Kendall)
#install.packages("tseries")
library(tseries)

#Checking the working directory.
getwd()
```

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

```
#Creating a ggplot theme.

MyTheme <- theme_classic() + theme(
        #Sets text on the axes as black.
        axis.text = element_text(color = 'black'),
        #Positions the legend to the right of plots.
        legend.position = ("right"),
        #Outlines and fills the legend pink.
        legend.background = element_rect(color = 'pink', fill = 'pink'),
        #Outlines and fills the background light blue.
        plot.background = element_rect(color = 'lightblue', fill = 'lightblue'),
        #Sets grid lines for the plot as grey.
        panel.grid.minor = element_line(color = "grey"),
        panel.grid.major = element_line(color = "grey")
)

#Setting MyTheme as the default theme.
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 Importing the data in bulk and combining the data into a single data frame.

#List.files creates a list of files in the designated folder.
O3_data_path <- list.files(here("Data/Raw/Ozone_TimeSeries"),
                           pattern = "\\.csv$", full.names = TRUE)

#The lapply function applies the read.cvs function to the entire list created above.
#The bind_rows combines the lists into a single data frame.
O3_data <- bind_rows(lapply(O3_data_path, read.csv, stringsAsFactors = TRUE))

#Chat GPT was used to help generate the code for importing the data in bulk.
```

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.

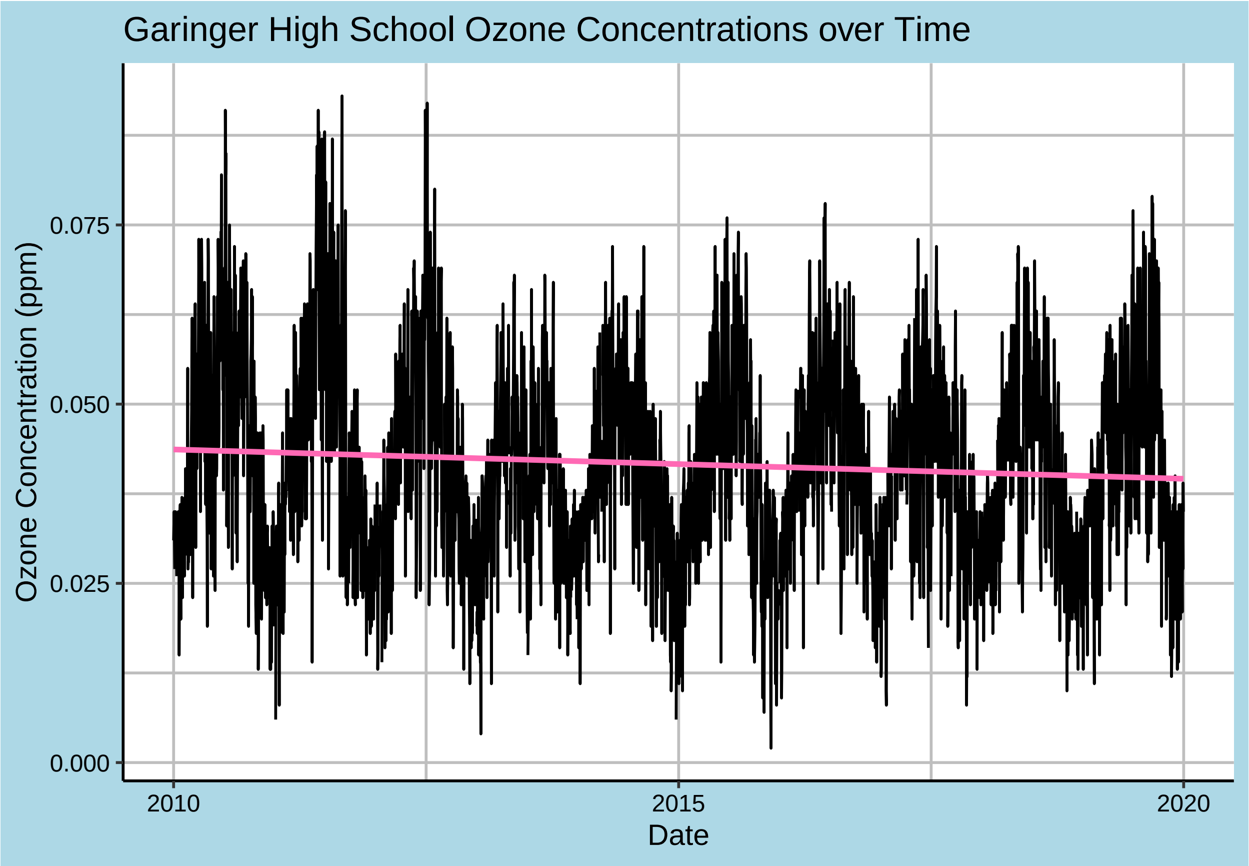
```
#3 Setting the year column to a date object.
O3_data$Date <- as.Date(O3_data$Date, format = "%m/%d/%Y")

#4 Wrangling the data set so it only includes the desired columns.
O3_wrangled_data <- O3_data %>%
  select(Date, Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)

#5 Creating a daily data set.
Days <- as.data.frame(seq(from = as.Date("2010-01-01"),
                          to = as.Date("2019-12-31"),
                          by = "day"))
colnames(Days) = "Date"

#6 Combining the data frames.
GaringerOzone <- left_join(Days, O3_wrangled_data, 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 Plotting ozone concentrations over time.

O3_plot <- ggplot(GaringerOzone, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line(linewidth = 0.5) +
  geom_smooth(method = "lm", se = FALSE, color = "hotpink") +
  labs(x = "Date",
       y = "Ozone Concentration (ppm)",
       title = "Garinger High School Ozone Concentrations over Time")

O3_plot
```

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



Answer: The line plot displays frequent fluctuations in ozone concentrations, ranging from below 0.025 ppm to about 0.090 ppm. The trend line depicts a slight decrease in ozone concentrations from 2010 through 2019, indicating there was little overall change in ozone levels during this decade.

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 Using a linear interpolation to fill in missing data.
GaringerOzone_clean <- GaringerOzone %>%
  mutate( Daily.Max.8.hour.Ozone.Concentration =
            zoo::na.approx(Daily.Max.8.hour.Ozone.Concentration))
```

Answer: A linear interpolation uses neighboring data points to provide missing daily data with a value that linearly connects the points it falls between. This method is typically utilized when the data set does not display any erratic patterns. Piecewise constants fill in missing data with the closest data point, which could significantly alter data trends and produce less gradual changes. A spline interpolation is similar to a linear interpolation but uses quadratic functions to fill in missing data. This method may introduce more variability in the data set and make it difficult to determine trends.

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)

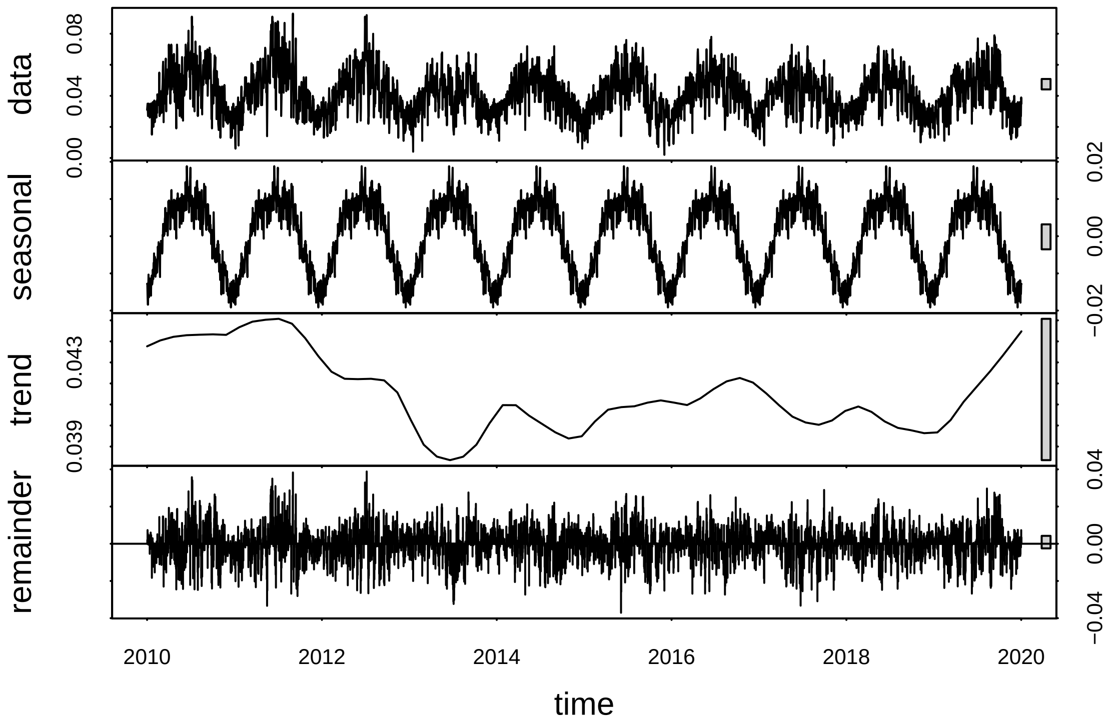
```r
#9 Creating a data frame for monthly average ozone concentrations.

GaringerOzone.monthly <- GaringerOzone_clean %>%
  #Creating a Month and Year column.
  mutate(Month = month(Date), Year = year(Date)) %>%
  #Grouping by the Date column.
  group_by(Date) %>%
  #Formatting the Date column to be set to the first date of the month.
  mutate(Date = my(paste0(Month,"-",Year))) %>%
  #Calculating the mean ozone concentration by Date.
  summarise(Mean_O3_Concentration = mean(Daily.Max.8.hour.Ozone.Concentration))
```

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.

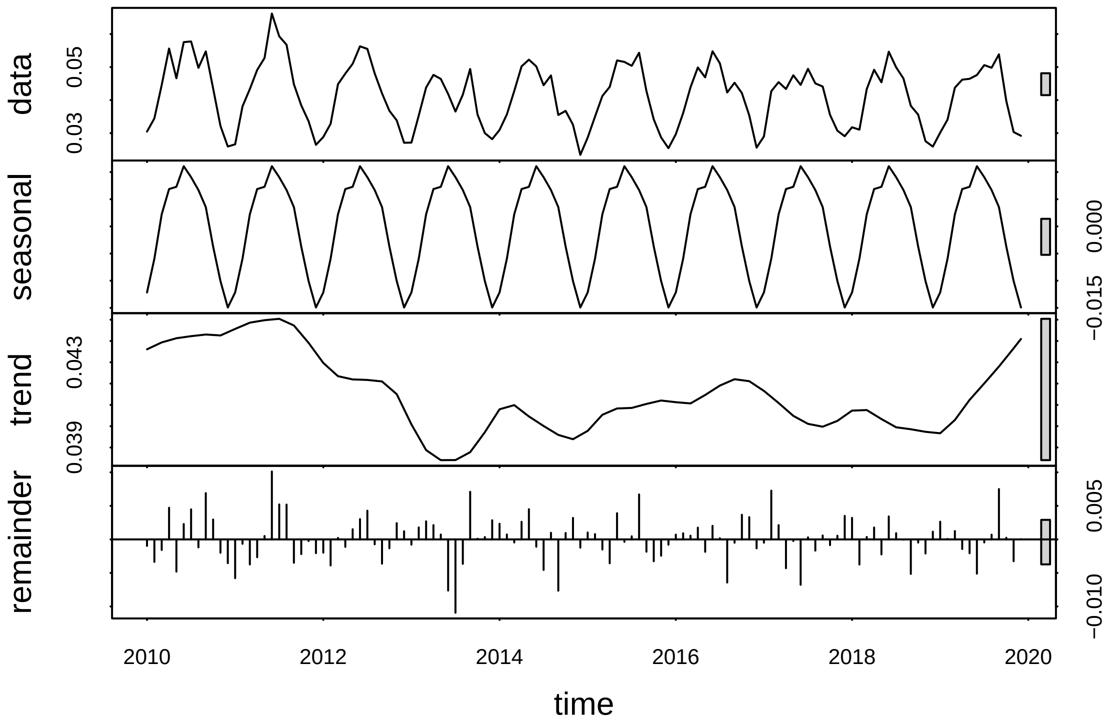
```r
#10 Creating time series objects for daily and monthly ozone values.

GaringerOzone.daily.ts <- ts(GaringerOzone_clean$Daily.Max.8.hour.Ozone.Concentration,
                             start = c(2010, 1, 01),
                             frequency = 365)

GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$Mean_O3_Concentration,
                               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
# Generate decomposition.
Daily_decomposed <- stl(GaringerOzone.daily.ts,
                        s.window = "periodic")
Monthly_decomposed <- stl(GaringerOzone.monthly.ts,
                          s.window = "periodic")

# Visualize the decomposed series.
plot(Daily_decomposed)
```



```r
plot(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 Running a Seasonal Mann-Kendall trend analysis
Ozone_monthly_trend <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
#Inspect the results.
Ozone_monthly_trend
```

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

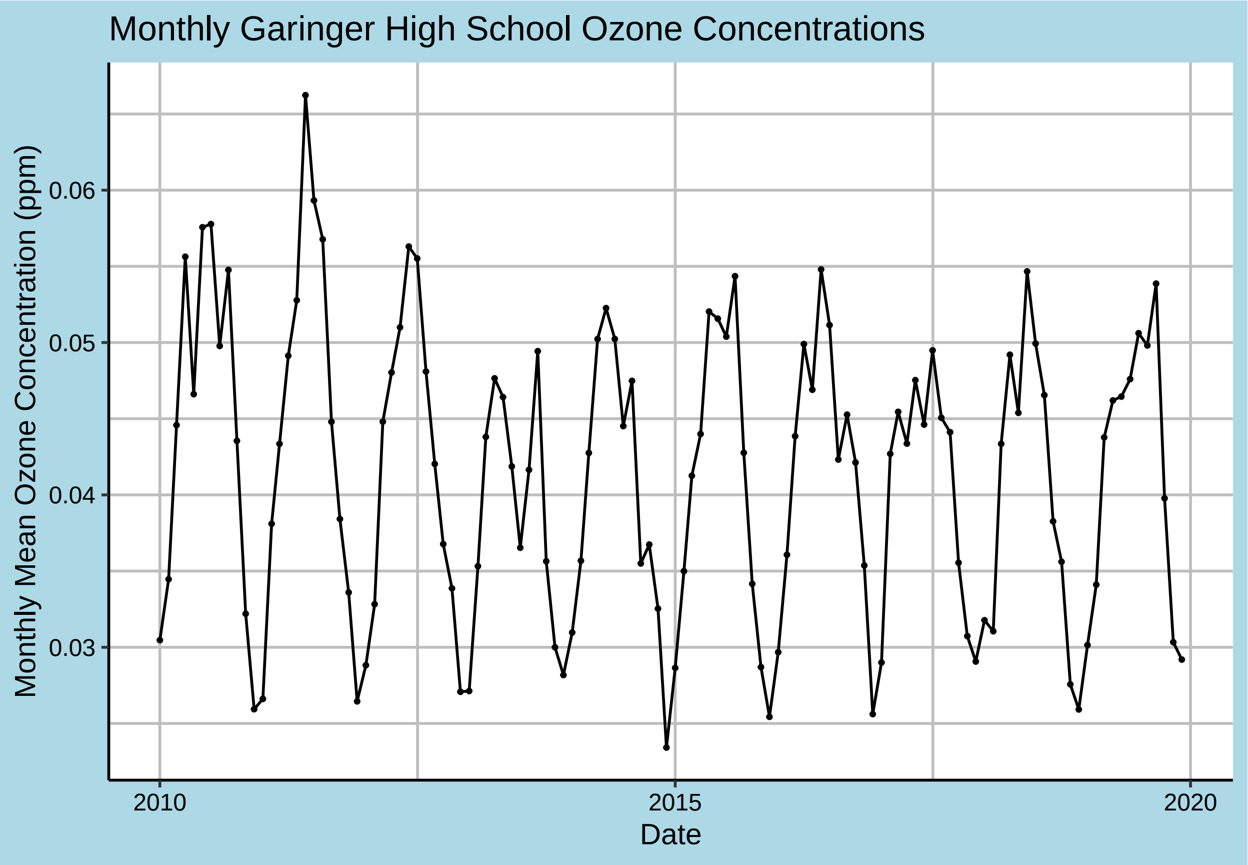
```
summary(Ozone_monthly_trend)
```

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

Answer: A seasonal Mann-Kendall is used in this case because we are examining the potential monthly pattern occurring in mean ozone concentrations. This trend analysis will determine recurring relationships present between mean ozone levels and certain periods of the year. Such results can be used to determine how changing conditions, such as temperature, may influence ozone concentrations.

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 Generating a plot for monthly mean ozone concentrations.

Ozone_monthly_plot <- ggplot(GaringerOzone.monthly,
                             aes(x = Date,
                                 y = Mean_O3_Concentration)) +
  geom_point(size = 0.5) +
  geom_line() +
  labs(x = "Date",
       y = "Monthly Mean Ozone Concentration (ppm)",
       title = "Monthly Garinger High School Ozone Concentrations")

Ozone_monthly_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: During the 2010s, ozone concentrations consistently fluctuated, producing a relatively small overall change in mean concentrations from 2010 to the end of 2019. The tau value (-0.143) of the seasonal Mann-Kendall analysis supports this observation, revealing a slight negative trend. The p-value (0.046724) is less than the significance value (0.05), meaning the result is statistically significant.

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 Extracting series components.

Monthly_components <- as.data.frame(Monthly_decomposed$time.series[,1:3])
Monthly_components <- mutate(Monthly_components,
                             Observed = GaringerOzone.monthly$Mean_O3_Concentration,
                             Date = GaringerOzone.monthly$Date,
                             Nonseason = Observed - seasonal)

Monthly_components.ts <- ts(Monthly_components$Nonseason,
                               start = c(2010, 1),
                               end = c(2019,12),
                               frequency=12)


#16 Running a Non-Seasonal Mann-Kendall trend analysis
Ozone_monthly_trend2 <- Kendall::MannKendall(Monthly_components.ts)
#Inspect the results.
Ozone_monthly_trend2
```

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

```
summary(Ozone_monthly_trend2)
```

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

Answer: The tau value of the non-seasonal trend is more negative (-0.165) than that of the seasonal trend (-0.143). A tau of zero indicates there is no trend, therefore specifying that there is a stronger trend found when the seasonal component is extracted from the monthly data set. The two-sided p-value for the non-seasonal trend (0.0075402) is less than the p-value for the seasonal trend (0.046724), indicating greater statistical significance when the seasonal component is extracted.