Assignment 8: Time Series Analysis

Nusrat Noor

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 1.9.3     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(zoo)
```

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

```
library(trend)
library(here)
```

```
## here() starts at /home/guest/R/EDE_Fall2023
```

```
#loaded necessary packages
here() #checked directory
```

```
## [1] "/home/guest/R/EDE_Fall2023"
```

```
my_theme <- theme_gray(base_size = 14) +
  theme(legend.position = "right")
theme_set(my_theme)
#created theme and set it
```

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

GaringerOzone <- bind_rows(lapply(GaringerOzone_path, read.csv, stringsAsFactors = TRUE))
#the lapply function applies the read.csv command to the folder so that all the files are read
#bind_rows makes the files into one data frame
dim(GaringerOzone) #checked dimensions
```

```
## [1] 3589   20
```

```
#used the help of Chat GPT and google to figure out how to load in all the files at once
```

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$Date <- as.Date(GaringerOzone$Date, format = "%m/%d/%Y")
# set date column as date class
# 4
GaringerOzone_wr <- select(GaringerOzone, "Date", "Daily.Max.8.hour.Ozone.Concentration",
                           "DAILY_AQI_VALUE")
#wrangled data to only include the asked for columns

# 5
```

```
Days <- as.data.frame(seq(from = as.Date("2010-01-01"), to = as.Date("2019-12-31"),
                          by = "day")) #created new data frame of dates
colnames(Days) <- c("Date") #changed column name
dim(Days) #check dimensions
```

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

```
# 6
GaringerOzone <- left_join(Days, GaringerOzone_wr, by = "Date")
#joined the Days and the wrangled ozone data frames and called it "GaringerOzone"
dim(GaringerOzone) #checked dimensions
```

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

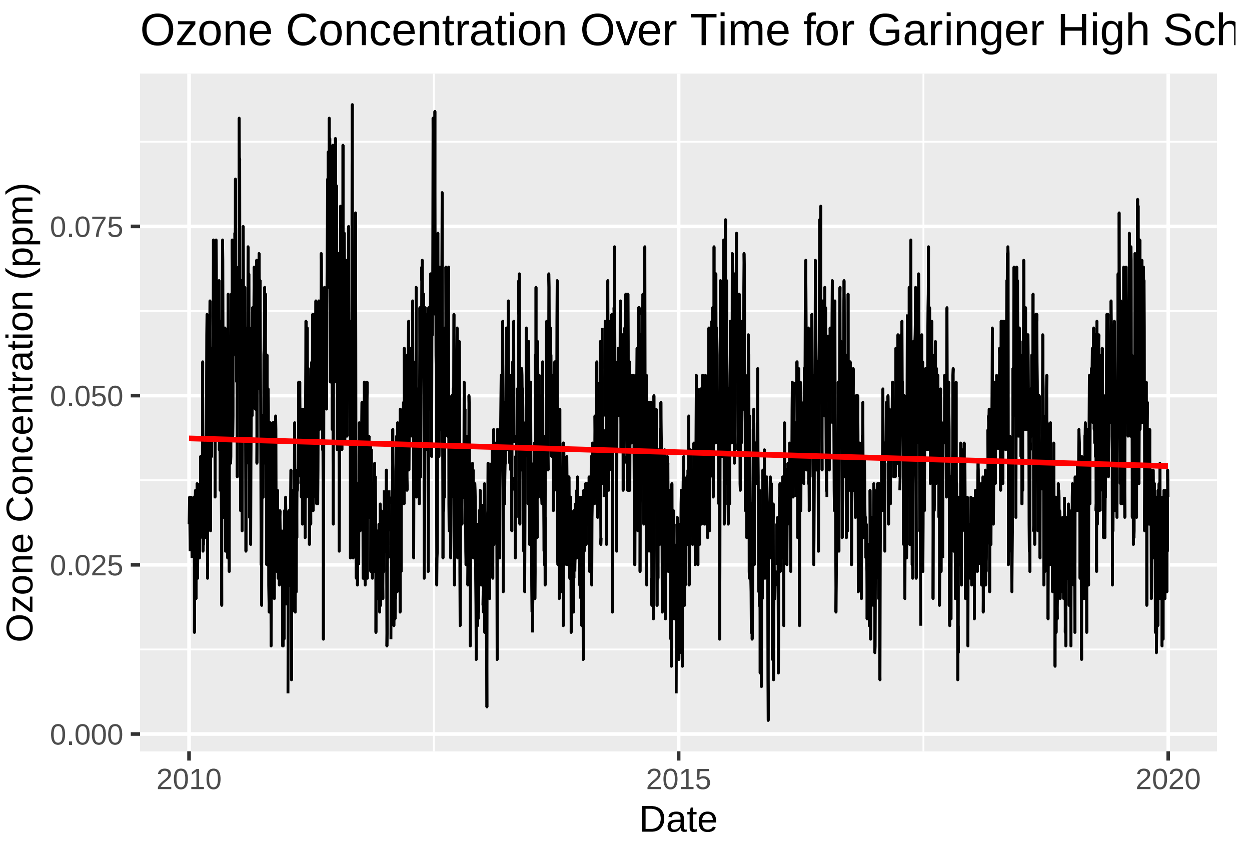
Visualize

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
ggplot(GaringerOzone,
       aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(x = "Date", y = "Ozone Concentration (ppm)",
       title = "Ozone Concentration Over Time for Garinger High School")
```

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

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



```
#created line plot of ozone concentration by date and added linear model
#trend line and cleaned up labels
```

Answer:There is a slight decrease in Ozone concentration over time.

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$Daily.Max.8.hour.Ozone.Concentration <-
  na.approx(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration)
#used linear interpolation to fill in the missing daily ozone concentrations
```

Answer: We didn't use the piecewise because for missing values, it takes the closest point and copies it while the linear and spline takes the average of the above and below measurements. This makes the trend more gradual and smooth. And we didn't use the spline because it brings in more variability by using the quadratic formula and doesn't give us the straight line that we want.

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 %>% #created new monthly data frame
```

```
  mutate(Year = year(Date), Month = month(Date)) %>% #created month and year columns
  group_by(Year, Month) %>% #grouped by year and then month
  summarise(mean_ozone = mean(Daily.Max.8.hour.Ozone.Concentration)) %>%
  #summarised the mean ozone concentration for each month
  mutate(Date2 = my(paste0(Month, "-", Year))) #created new date column of grouped together dates
```

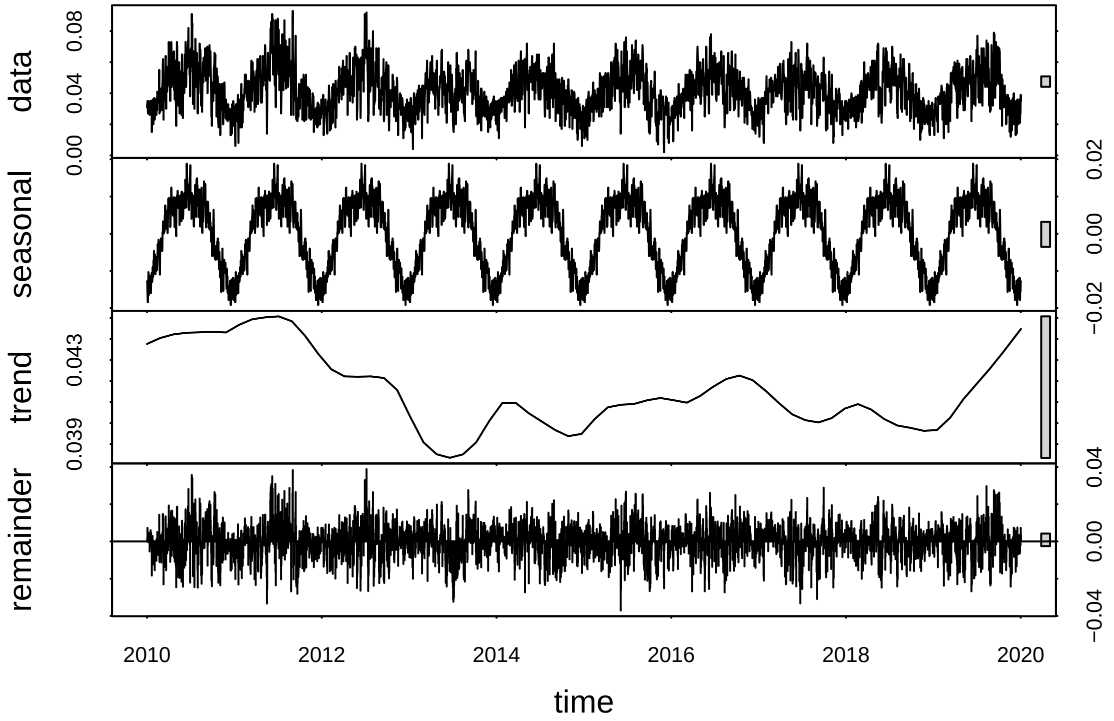
```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

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.

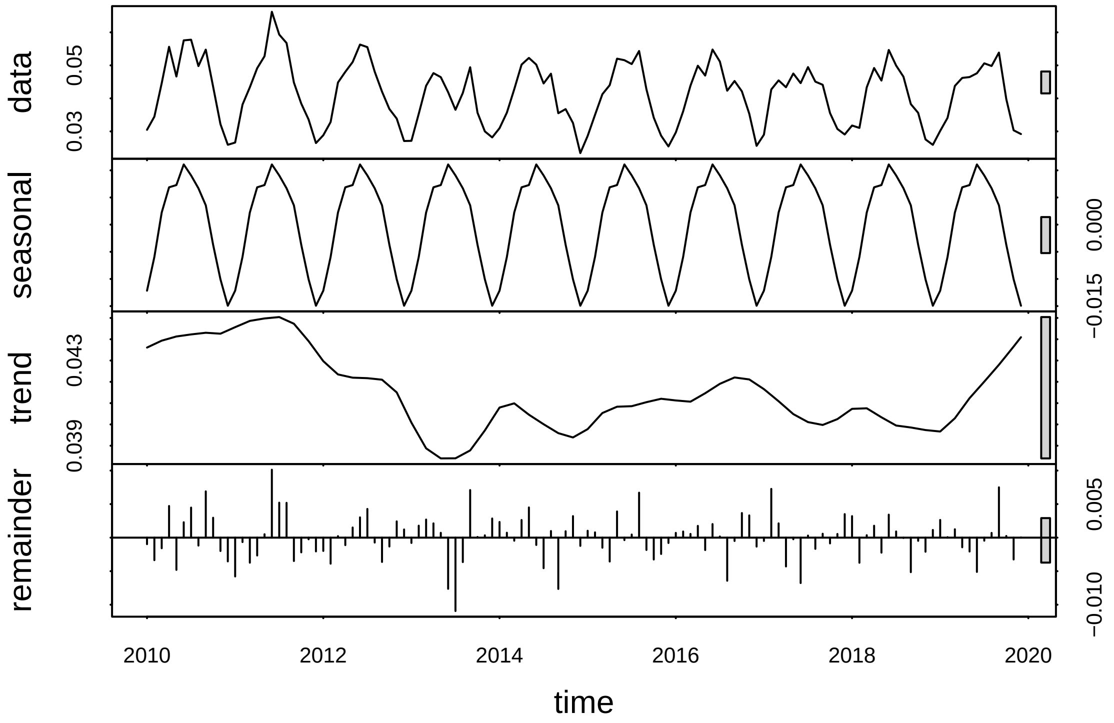
```
#10
GaringerOzone.daily.ts <- ts(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration,
                             start = c(2010,1), frequency = 365)
#created time series object for daily ozone concentration observations
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$mean_ozone,
                               start = c(2010,1), frequency = 12)
#created time series object for monthly ozone concentration observations
```

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

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



```
#decomposed the daily time series and plotted it
GaringerOzone.monthly.decom <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
plot(GaringerOzone.monthly.decom)
```



```
#decomposed the monthly time series and plotted it
```

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
GOMonthly.trend <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
#ran the seasonal Mann-Kendall trend analysis
GOMonthly.trend
```

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

```
summary(GOMonthly.trend)
```

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

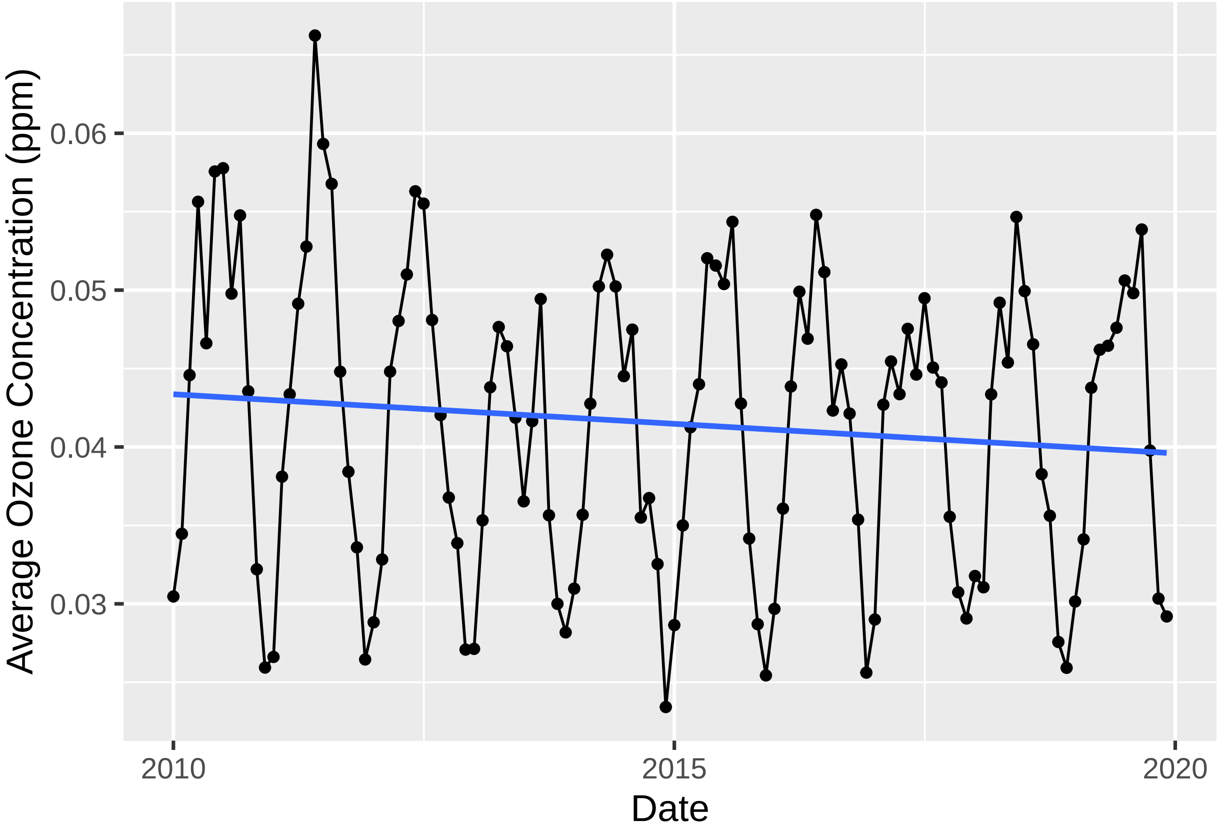
```
#inspected the results
```

Answer: It is the most appropriate because the data is seasonal and non-parametric.

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
ggplot(GaringerOzone.monthly, aes(x = Date2, y = mean_ozone)) +
  geom_point() +
  geom_line() +
  ylab("Average Ozone Concentration (ppm)") +
  xlab("Date") +
  geom_smooth(method = "lm", se = FALSE)
```

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



```
#created graph of mean monthly ozone concentration over time with
#both a point and line layer and changed labels and added linear model trend line
```

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 ozone concentrations have decreased very slightly in the 2010s (tau = -0.143, p-value = 0.0467).

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
GaringerOzone.monthly.components <- as.data.frame(GaringerOzone.monthly.decom$time.series[,1:3])
#created new data frame of the extracted components of the time.series column
```

```
GaringerOzone.monthly.components <- mutate(GaringerOzone.monthly.components,
                                           Observed = GaringerOzone.monthly$mean_ozone,
                                           Date = GaringerOzone.monthly$Date2,
                                           noseason = Observed - seasonal)
#created 3 more columns of the Observed, Date, and noseason
GO.monthly.comp.ts <- ts(GaringerOzone.monthly.components$noseason,
                         start = c(2010,1), end = c(2019,12), frequency = 12)
#made time series object of the extracted components

#16
GOMonthly.trend.2 <- Kendall::MannKendall(GO.monthly.comp.ts)
#ran non-seasonal Mann-Kendall analysis
GOMonthly.trend.2
```

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

```
summary(GOMonthly.trend.2)
```

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

```
#inspected the results
```

Answer: When doing the non-seasonal, the tau is larger and the p-value is smaller than the seasonal series so that makes the nonseasonal series more significant.