Assignment 8: Time Series Analysis

Ana Bishop

Spring 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

```
# 1 load libraries
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- 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(zoo)
```

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

```
library(trend)

# check working directory
getwd()  #looks good
```

```
## [1] "/Users/anabishop/Documents/Data Analytics/ENV872"
```

```
# set my theme
mytheme <- theme_minimal(base_size = 14) +
    theme(axis.text = element_text(color = "black"),
        plot.title = element_text(hjust = 0.5),
        legend.position = "right")

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
one <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv",
    stringsAsFactors = TRUE)
two <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv",
    stringsAsFactors = TRUE)
three <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv",
    stringsAsFactors = TRUE)
four <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv",
    stringsAsFactors = TRUE)
five <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv",
    stringsAsFactors = TRUE)
six <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv",
    stringsAsFactors = TRUE)
seven <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv",
    stringsAsFactors = TRUE)
eight <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv",
    stringsAsFactors = TRUE)
nine <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv",
```

```
    stringsAsFactors = TRUE)
ten <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv",
    stringsAsFactors = TRUE)

epaair <- rbind(one, two, three, four, five,
    six, seven, eight, nine, ten)
```

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
epaair$Date <- mdy(epaair$Date)
class(epaair$Date)  #looks good
```

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

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

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

# 6
GaringerOzone <- left_join(days, ozone, by = "Date")
```

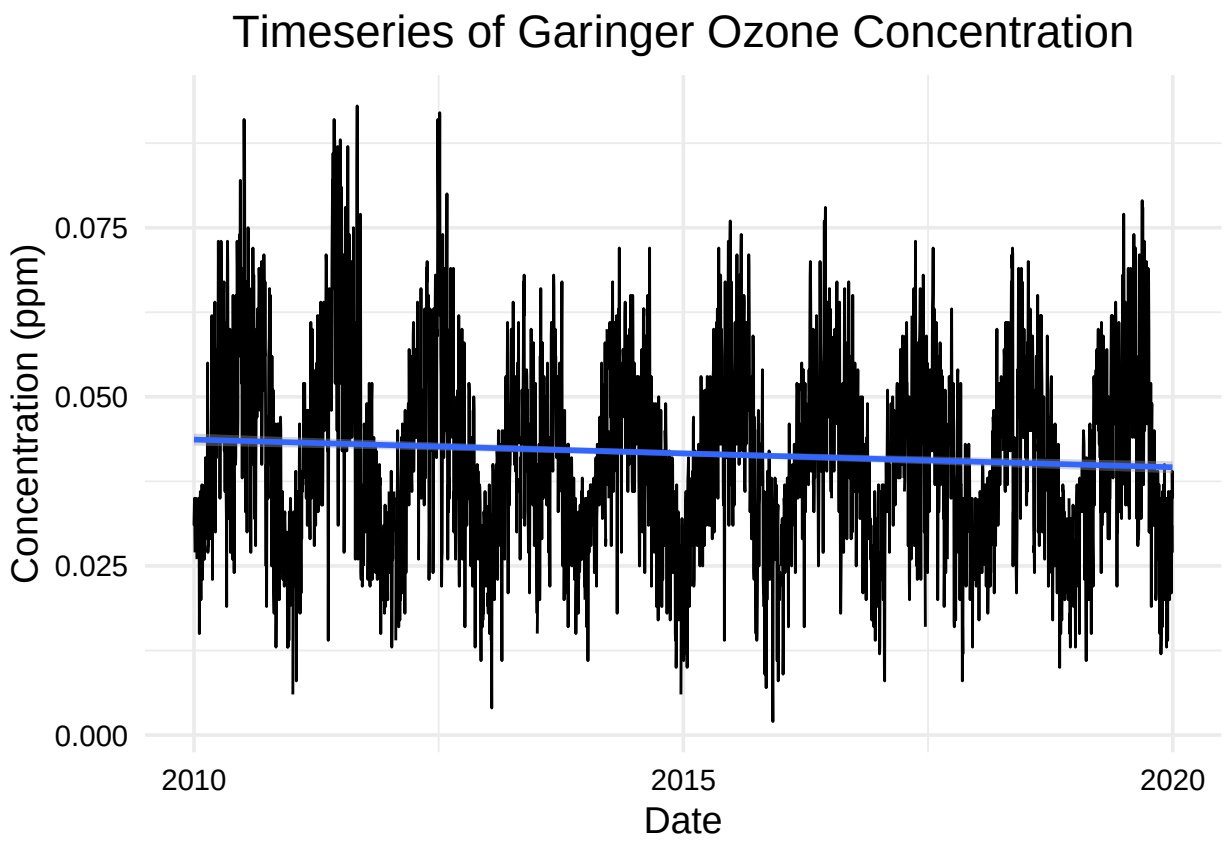
Visualize

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
ggplot(GaringerOzone, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
    geom_line() + geom_smooth(method = "lm") +
    labs(x = "Date", y = "Concentration (ppm)") +
    ggtitle("Timeseries of Garinger Ozone Concentration")
```

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

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



Answer: The trendline appears to be nearly horizontal, although a slight decrease is visible. Therefore, though there does not seem to be a strong trend in the data, since the data oscillates back and forth at the same levels fairly regularly over the years, a slightly decreasing trend is apparent.

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.Co
```

Answer: Linear interpolation assumes that the missing value falls between the previous and next datapoint according to a linear equation drawn through the entire dataset, and estimates a value that falls between those other two values based on that linear relationship. In comparison, piecewise constant interpolation is assumed to be equal to the nearest known datapoint, and spline

interpolation uses a quadratic relationship to represent the data instead of a linear relationship. We used a linear interpolation because the data was only missing in small pieces in between two datapoints, which does not merit the use of piecewise constant interpolation, and the relationship within the data appears to follow a linear relationship, as opposed to a quadratic 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 %>%
    mutate(Year = year(Date)) %>%
    mutate(Month = month(Date)) %>%
    aggregate(Daily.Max.8.hour.Ozone.Concentration ~
        Month + Year, mean)

GaringerOzone.monthly <- GaringerOzone.monthly %>%
    mutate(Date = seq(as.Date("2010-01-01"),
        as.Date("2019-12-01"), "month"))
```

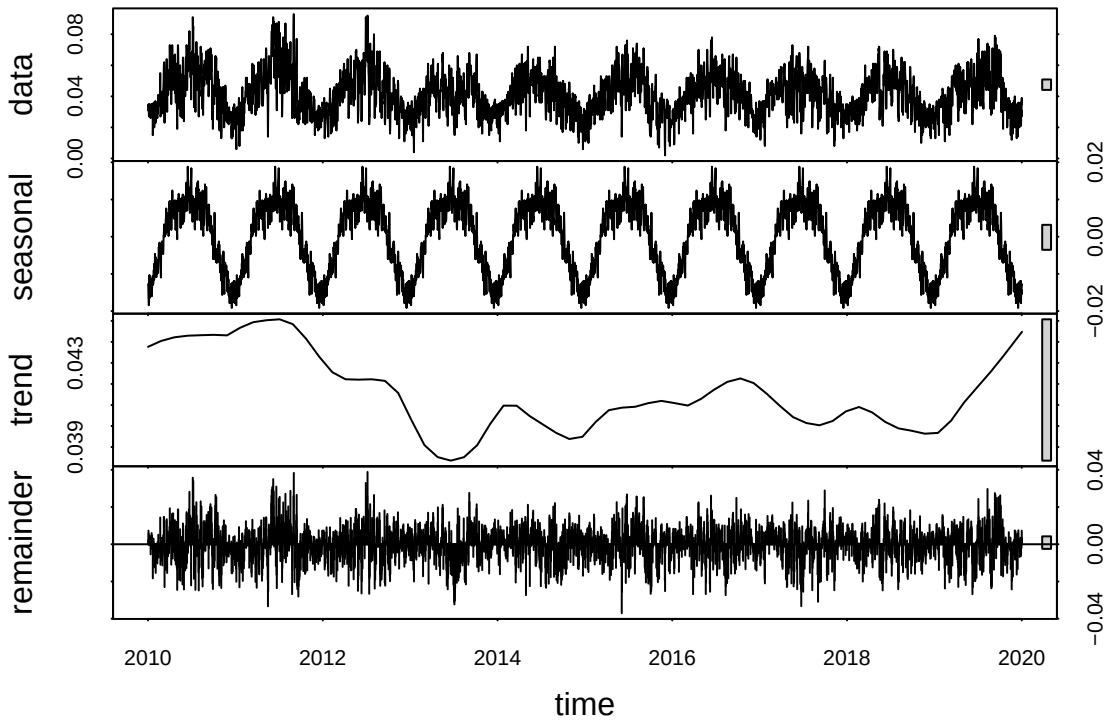
10. Generate two time series objects. Name the first `GaringerOzone.daily.ts` and base it on the dataframe of daily observations. Name the second `GaringerOzone.monthly.ts` and base it on the monthly average ozone values. Be sure that each specifies the correct start and end dates and the frequency of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

```
# 10
GaringerOzone.daily.ts <- ts(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration,
    start = c(2010, 1), frequency = 365)
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$Daily.Max.8.hour.Ozone.Concentration,
    start = c(2010, 1), frequency = 12)
```

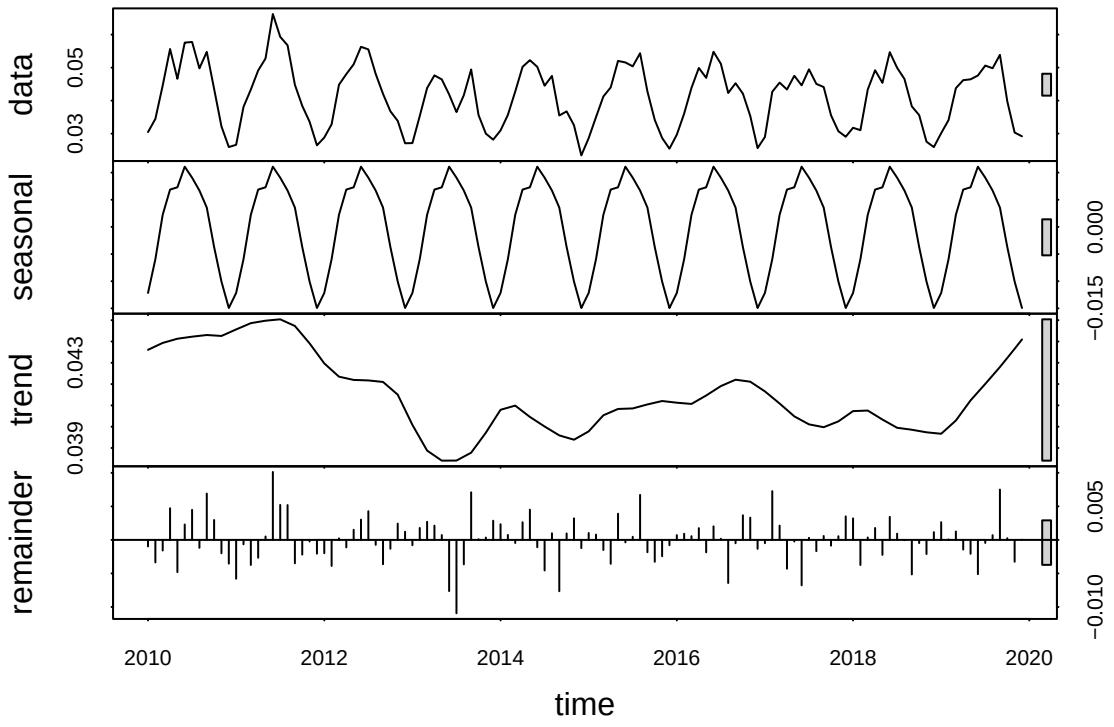
11. Decompose the daily and the monthly time series objects and plot the components using the `plot()` function.

```
# 11
GaringerOzone.daily_decomposed <- stl(GaringerOzone.daily.ts,
    s.window = "periodic")
GaringerOzone.monthly_decomposed <- stl(GaringerOzone.monthly.ts,
    s.window = "periodic")

plot(GaringerOzone.daily_decomposed)
```




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

# show results
monthly_analysis
```

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

```
summary(monthly_analysis)
```

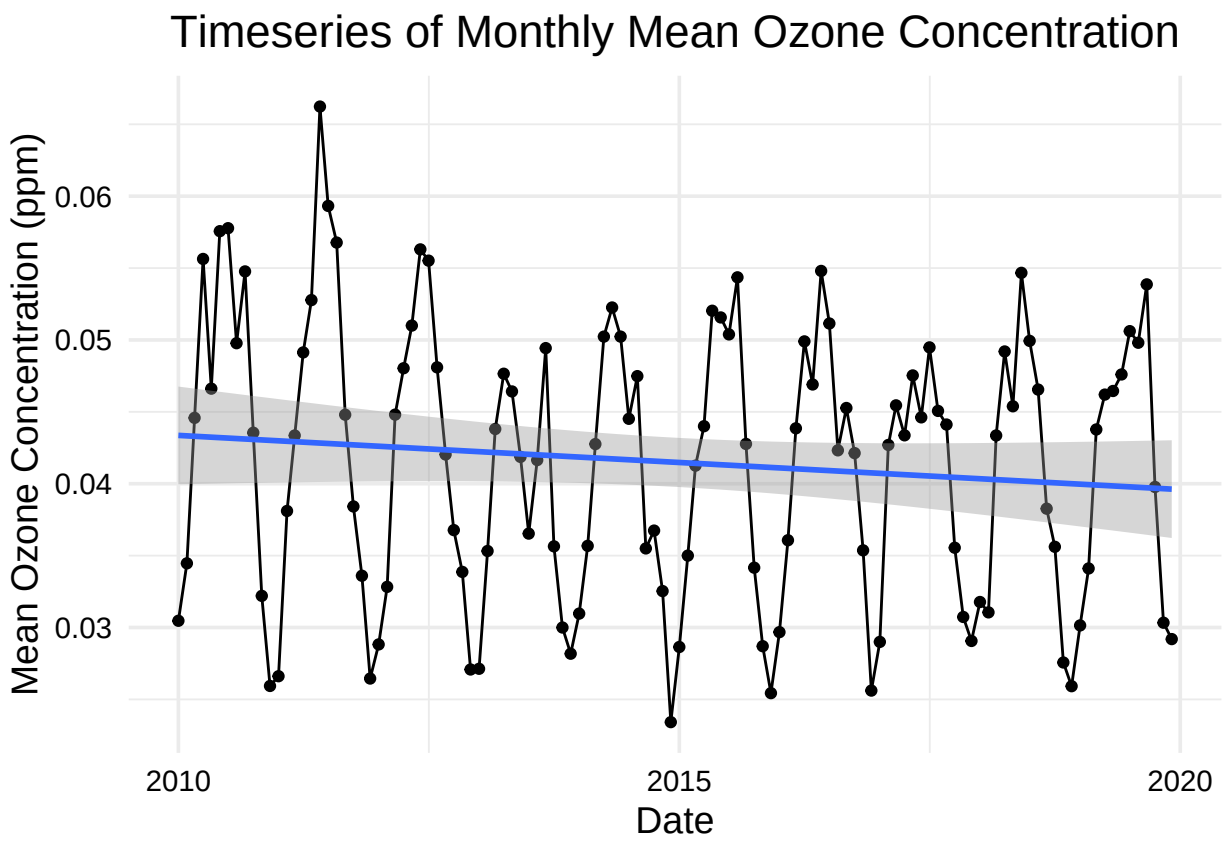
```
## Score =  -77 , Var(Score) = 1499
## denominator =  539.4972
## tau = -0.143, 2-sided pvalue =0.046724
```

Answer: The seasonal Mann-Kendall test is most appropriate because there was clear seasonality in the data (the ozone concentration consistently rose and fell over even time increments throughout the duration of the study period), and the seasonal Mann-Kendall is the only test that is capable of handling seasonality in its analysis.

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
ggplot(GaringerOzone.monthly, aes(x = Date,
    y = Daily.Max.8.hour.Ozone.Concentration)) +
    geom_line() + geom_point() + geom_smooth(method = "lm") +
    labs(x = "Date", y = "Mean Ozone Concentration (ppm)") +
    ggtitle("Timeseries of Monthly Mean Ozone Concentration")
```

```
## `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: Based on the visualization of the data and the statisical results, I can conclude that the ozone concentration did change over the 2010s at this station. The visualization of the data shown above reveals a decreasing linear trendline, and the results of the seasonal Mann-Kendall test performed on the data confirmed that the earlier measured data was significantly less than the later measured data (score = -77, p = 0.047, tau = -0.143). Therefore, the results show that the ozone concentration decreased over the 2010s at the Garinger station.

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.

```r
# 15
GaringerOzone_monthly_Components <- as.data.frame(GaringerOzone.monthly_decomposed$time.series[,
    1:3])

GaringerOzone_monthly_Components <- mutate(GaringerOzone_monthly_Components,
    Observed = GaringerOzone.monthly$Daily.Max.8.hour.Ozone.Concentration,
    Date = GaringerOzone.monthly$Date)

GaringerOzone_monthly_noseasonality <- GaringerOzone_monthly_Components %>%
    mutate(NoSeasonality = (Observed - seasonal))

# 16 set time series object
GaringerOzone.monthly.noseasonality.ts <- ts(GaringerOzone_monthly_noseasonality$NoSeasonality,
    start = c(2010, 1), frequency = 12)

# conduct analysis
monthly_noseasonality_analysis <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.noseasonality.ts)

# show results
monthly_noseasonality_analysis
```

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

```r
summary(monthly_noseasonality_analysis)
```

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

Answer: The results of this test were exactly the same as the previous test. From these results, it appears that manually removing seasonality from the data achieved the same result as the seasonal Mann-Kendall test on the initial dataset.