Assignment 8: Time Series Analysis

Allison Barbaro

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
getwd()
```

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

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.0
## v tibble  3.1.8     v dplyr   1.1.0
## 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()
```

```r
library(lubridate)
```

```
## Loading required package: timechange
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
library(zoo)
```

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

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

```
## here() starts at /home/guest/R/EDA-Spring2023
```

```r
library(dplyr)
library(ggplot2)

mytheme <- theme_classic(base_size = 14) +
  theme(axis.text = element_text(color = "black"),
        legend.position = "top", legend.title = element_text(
      color='purple'))
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.

```r
#2
ozone_2010 <-
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv", stringsAsFactors = TRUE)
ozone_2011 <-
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv", stringsAsFactors = TRUE)
ozone_2012 <-
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv", stringsAsFactors = TRUE)
ozone_2013 <-
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv", stringsAsFactors = TRUE)
ozone_2014 <-
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv", stringsAsFactors = TRUE)
ozone_2015 <-
```

```
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv", stringsAsFactors = TRUE)
ozone_2016 <-
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv", stringsAsFactors = TRUE)
ozone_2017 <-read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv", stringsAsFactors = T
ozone_2018 <-
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv", stringsAsFactors = TRUE)
ozone_2019 <-
  read.csv("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv", stringsAsFactors = TRUE)

Garinger_Ozone<-
  bind_rows(ozone_2010,ozone_2011,ozone_2012,ozone_2013,
            ozone_2014,ozone_2015,ozone_2016,ozone_2017,ozone_2018,
            ozone_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
Garinger_Ozone$Date <-mdy(Garinger_Ozone$Date)

#4
Garinger_Ozone_wrangled <- Garinger_Ozone %>%
  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"), by = 1))
colnames(Days)[1] = "Date"

#6
GaringerOzone <- left_join(Days, Garinger_Ozone_wrangled)
```

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

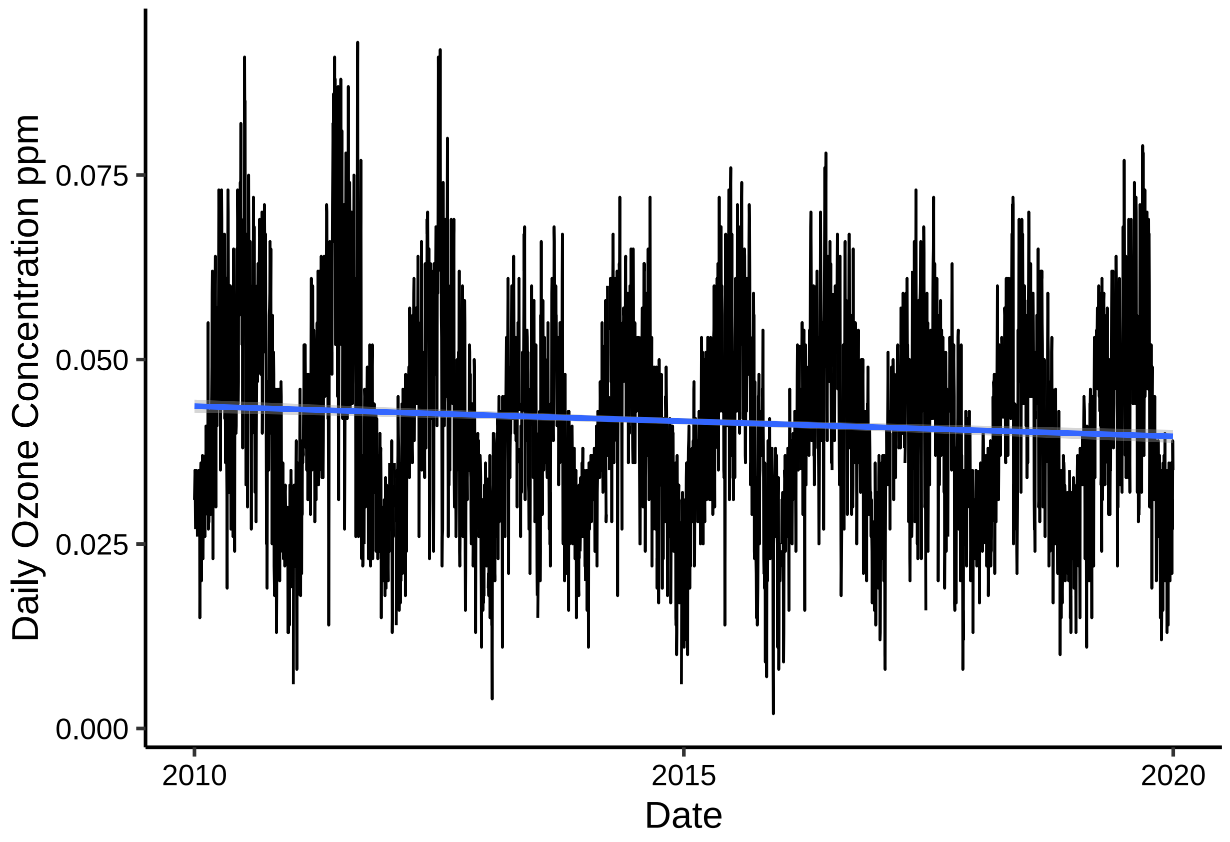
Visualize

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?

```r
#7
ggplot(GaringerOzone, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line() +
  geom_smooth(method = "lm") +
  labs(x = "Date", y = "Daily Ozone Concentration ppm")
```

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

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



Answer: This plot suggests 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?

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

Answer: We used a linear interpolation because our data has a linear trend based upon our initial plot. Spline interpolation uses a quadratic function and piecewise uses a nearest neighbor approach, neither of which fit our data as well as the linear method.

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 <- Daily.Ozone.Garinger %>%
  mutate(Month = month(Date)) %>%
  mutate(Year = year(Date)) %>%
  mutate(Day = "01") %>%
  mutate(DateMY = mdy(paste(Month, Day, Year, sep = "-"))) %>%
  group_by(Month) %>%
  mutate(monthly.mean.ozone =
  mean(Daily.Max.8.hour.Ozone.Concentration.Daily.Ozone.Garinger))
```

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.

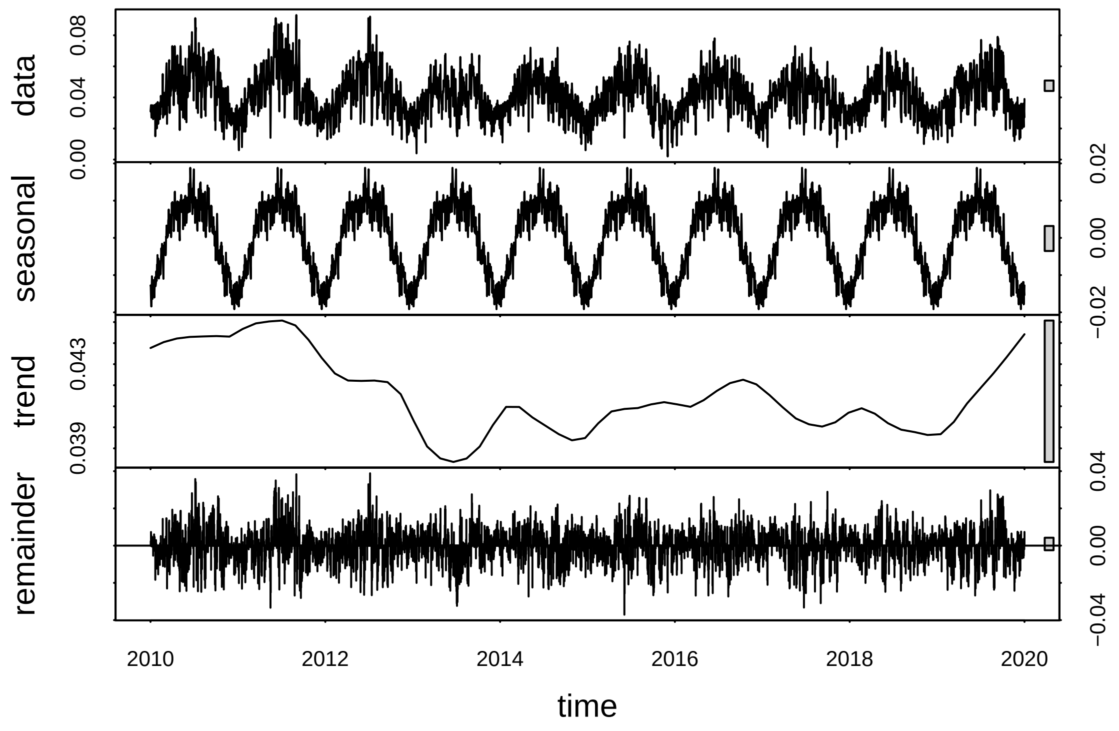
```
#10
GaringerOzone.daily.ts <-
ts(Daily.Ozone.Garinger$Daily.Max.8.hour.Ozone.Concentration.Daily.Ozone.Garinger,
   start = c(2010,1), end = c(2020, 1), frequency = 365)

GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$monthly.mean.ozone,
    start = c(2010,1), end = c(2019, 12), frequency = 12)
```

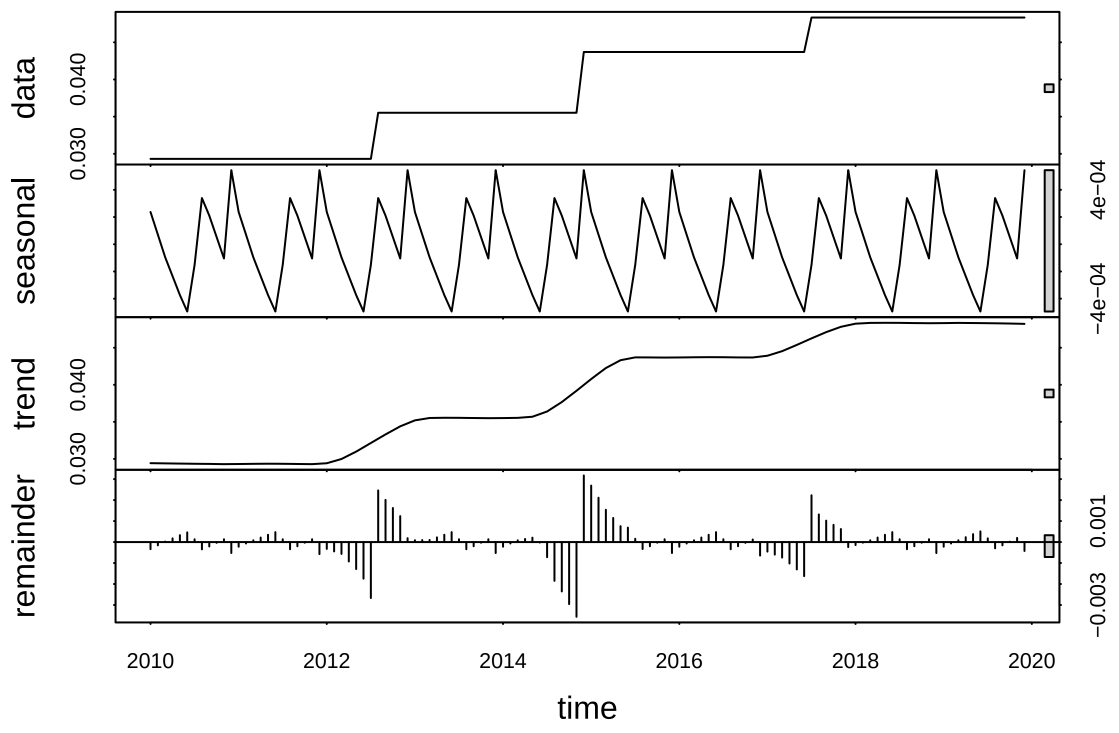
11. Decompose the daily and the monthly time series objects and plot the components using the `plot()` function.

```
#11

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



```
GaringerOzone.monthly.decomposed <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
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
Ozone.monthly.smk <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
summary(Ozone.monthly.smk)
```

```
## Score =  444 , Var(Score) = 1388
## denominator =  489.653
## tau = 0.907, 2-sided pvalue =< 2.22e-16
```

Answer: The seasonal Mann Kendall is appropriate for our data because the data shows seasonality. All other monotonic trend analysis methods require no seasonality.

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
Mean.monthly.plot <-
  ggplot(GaringerOzone.monthly, aes(x = Date, y = monthly.mean.ozone)) +
  geom_point() +
  geom_line() +
  labs(x = "Date", y = "Ozone Concentration (ppm)")
```

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: Ozone concentrations have not changed over the 2010s at this station. Our plot above illustrates this lack of change, depicting that monthly averages over the year are relatively similar, with the same minimums and maximums occurring each season. This plot supports our results from the Seasonal Mann Kendall test, which provided a statistically insignificant p-value (smaller than 0.05), leading us to accept the null hypothesis (no change over time).

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[,2:3])
#Decomposing without the seasonal component

GaringerOzone.monthly.120.var <-GaringerOzone.monthly %>%
  distinct(DateMY, .keep_all = TRUE)
#Creating a dataframe for the monthly average data that has only one value per month.
#This allows us to add the categories of observed and date to our
#decomposed dataframe (see code lines just below).

GaringerOzone.monthly.components <-
  mutate(GaringerOzone.monthly.components,
         Observed = GaringerOzone.monthly.120.var$monthly.mean.ozone,
         Date = GaringerOzone.monthly.120.var$DateMY)



#16

GaringerOzone.components.ts <- ts(GaringerOzone.monthly.components$Observed,
    start = c(2010,1), end = c(2019, 12), frequency = 12)

Ozone.monthly.mk <- Kendall::MannKendall(GaringerOzone.components.ts)
summary(Ozone.monthly.mk)
```

```
## Score =  -80 , Var(Score) = 192866.7
## denominator =  6864.692
## tau = -0.0117, 2-sided pvalue =0.85724
```

Answer: The results of this Mann Kendall test on our non-seasonal data illustrate that there is change in our data over time. Our P-value is larger than 0.05, which means we reject the null hypothesis. This is different from our Seasonal Mann-Kendall results.