Assignment 8: Time Series Analysis

Vicky Fong

Fall 2024

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     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## 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(here)
```

```
## here() starts at /home/guest/EDE_Fall2024
```

```
library(zoo)
```

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

```
library(dplyr)
library(trend)
library(Kendall)
getwd()
```

```
## [1] "/home/guest/EDE_Fall2024"
```

```
my_theme <-  theme(
    plot.title = element_text(face = "bold", size = 12),
    panel.background = element_rect(fill = "white", colour = NA),
    panel.border = element_rect(fill = NA, colour="grey50"),
    panel.grid.major = element_line(colour = "black", size = 0.01),
    panel.grid.minor =  element_line(colour = "black", size = 0.01),
    axis.text = element_text(size = 10),
    axis.ticks = element_blank(),
  )
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

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
file_folder <- here('Data','Raw','Ozone_TimeSeries')
files <- dir(file_folder, pattern = "*.csv")
setwd(file_folder)
GaringerOzone <- files %>%
  map(read_csv) %>%
  reduce(rbind)
```

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.

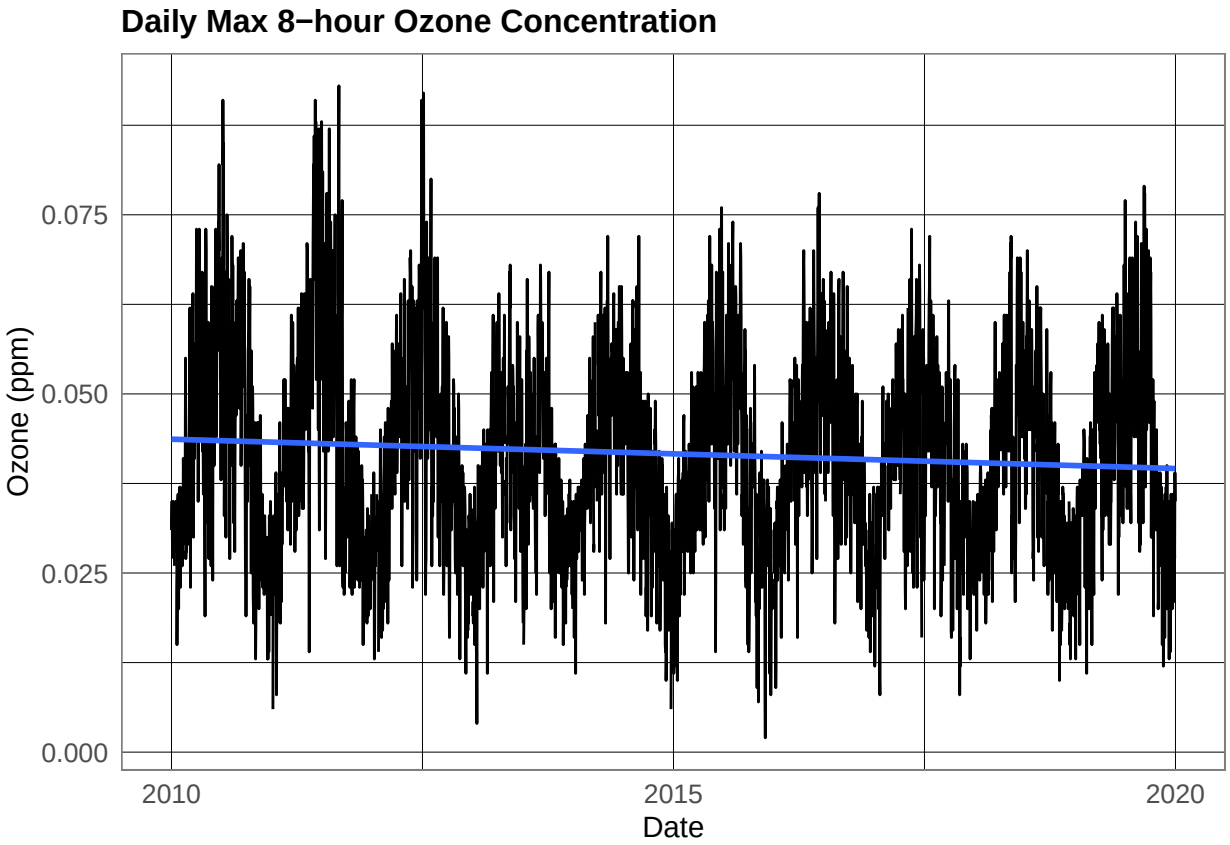
```
# 3
GaringerOzone$Date <- mdy(GaringerOzone$Date)

# 4
GaringerOzone <- GaringerOzone %>%
  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 = "days"))
colnames(Days) <- "Date"

# 6
GaringerOzone <- left_join(Days,GaringerOzone, 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
GaringerOzone %>%
  ggplot(aes(x=Date, y=`Daily Max 8-hour Ozone Concentration`)) +
  geom_line() +
  geom_smooth(method=lm, se=FALSE) +
  labs(x="Date", y="Ozone (ppm)",
       title="Daily Max 8-hour Ozone Concentration") +
  my_theme
```

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

## Warning: Removed 63 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```



Answer: The plot shows postive then negative seasonal trends in ozone within each year, and there is an overall decrease in ozone concentration from 2010 to 2020.

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 <-  GaringerOzone %>%
  mutate( Ozone = zoo::na.approx(`Daily Max 8-hour Ozone Concentration`),
          ) %>%
  select(Date,Ozone) #using cleaner column names
```

Answer: Since there appears to be positive and negative linear trends in ozone wihin each year, linear interpolation is the most apprropriate method to "connect the dots" and find the average value between the previous and next ozone measurement. Piecewise constant is not approapriate since using the "nearest neighbor" approach would under- or over-estimate the ozone values. Spline interpolation is not appropriate either as there is no quadratic trend in ozone values.

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), month = month(Date)) %>%
  group_by(year, month) %>%
  summarise(mean_ozone = mean(Ozone))
```

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

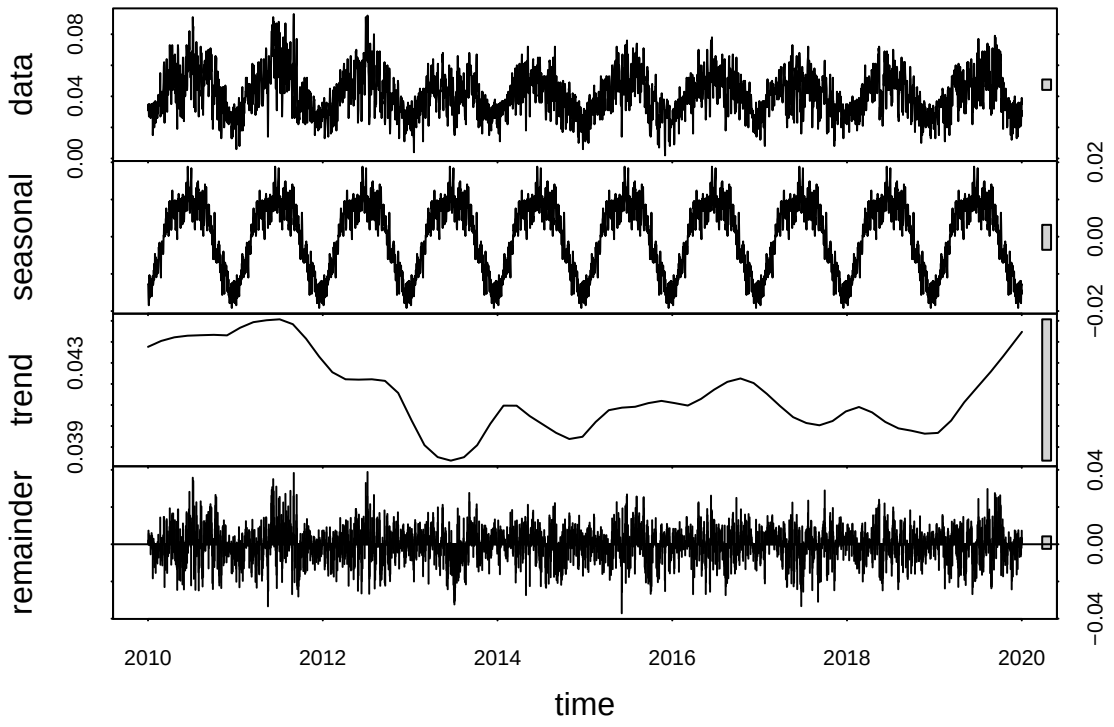
```
GaringerOzone.monthly$date = ymd(
  paste(GaringerOzone.monthly$year, GaringerOzone.monthly$month, "1"))
```

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.

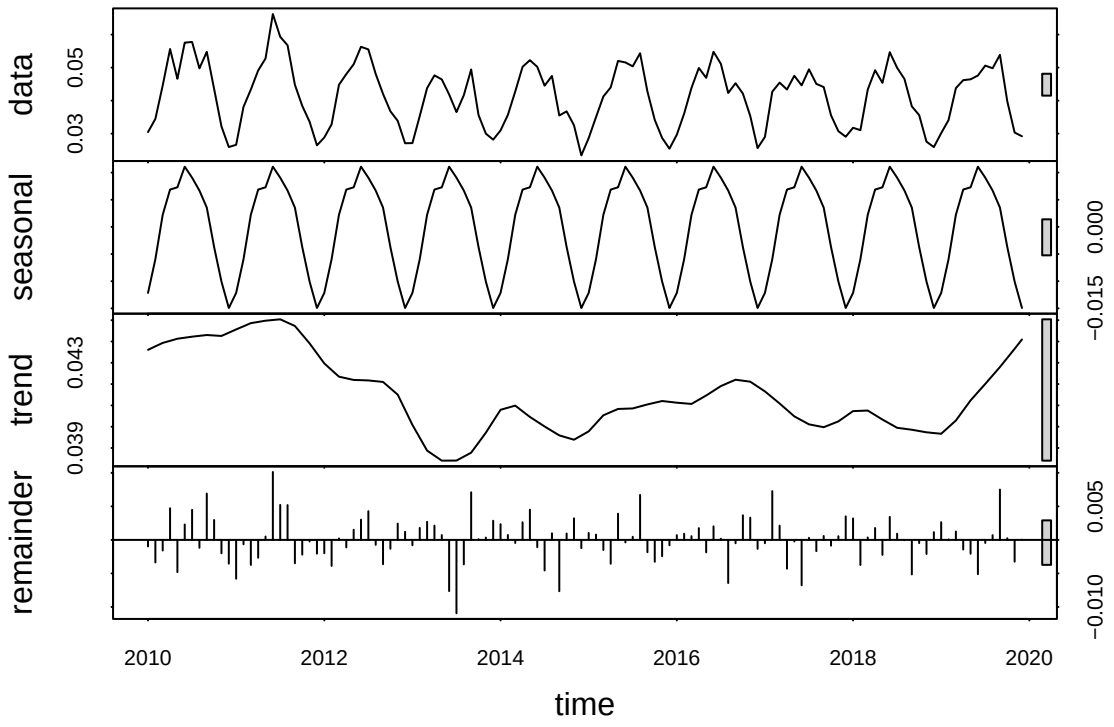
```
#10
f_month <- first(GaringerOzone.monthly$month)
f_year <- first(GaringerOzone.monthly$year)

GaringerOzone.daily.ts <- ts(GaringerOzone$Ozone,
                   start=c(f_year,f_month),
                   frequency=365)

GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$mean_ozone,
                   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
daily_decomp <- stl(GaringerOzone.daily.ts,s.window = "periodic")
plot(daily_decomp)
```




```
monthly_decomp <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
plot(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
monthly_trend <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
summary(monthly_trend)
```

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

```
monthly_trend2 <- trend::smk.test(GaringerOzone.monthly.ts)
summary(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|)
```

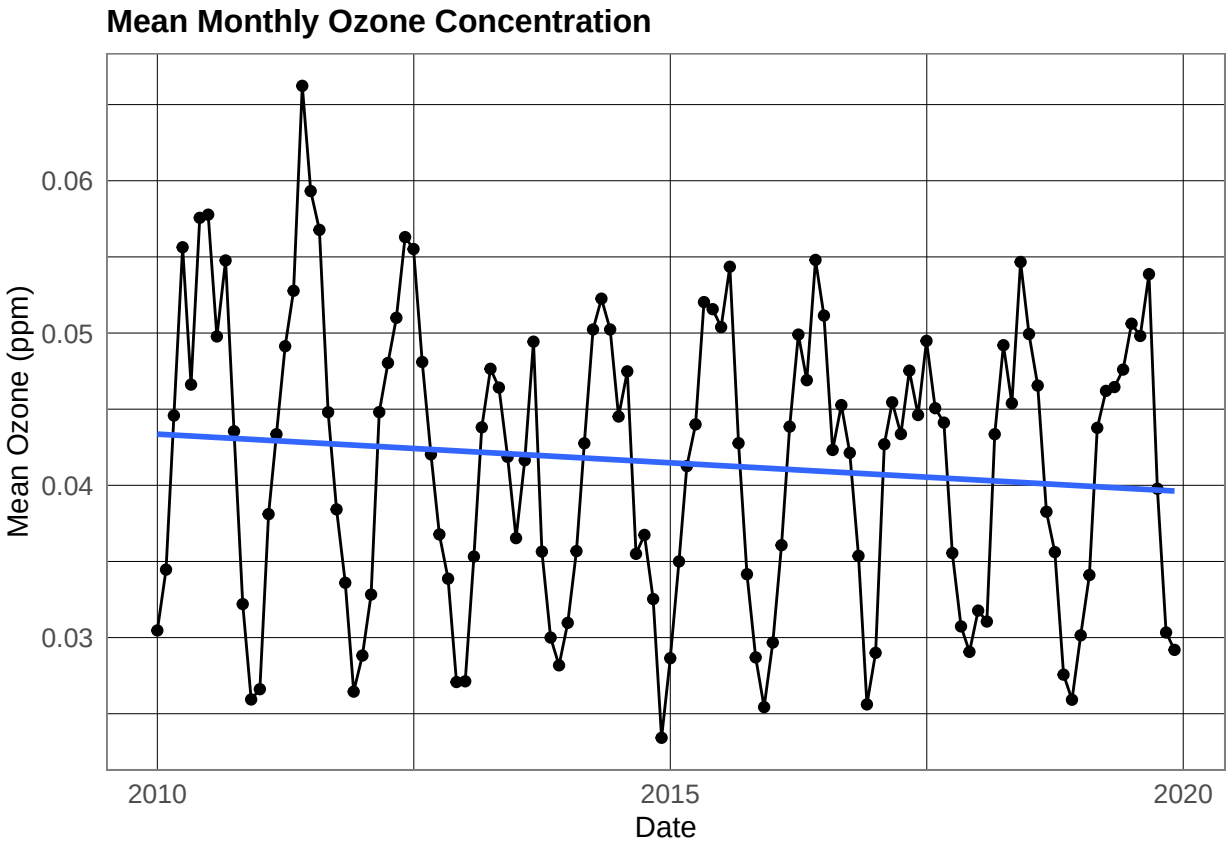
```
## Season 1:   S = 0   15  125  0.333  1.252  0.21050
## Season 2:   S = 0   -1  125 -0.022  0.000  1.00000
## Season 3:   S = 0   -4  124 -0.090 -0.269  0.78762
## Season 4:   S = 0  -17  125 -0.378 -1.431  0.15241
## Season 5:   S = 0  -15  125 -0.333 -1.252  0.21050
## Season 6:   S = 0  -17  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: The p-value frrmo the SeasonalMannKendall test was less than 0.05, indicating that there is a seasonal trend in mean monthly ozone concentrations over time.

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
GaringerOzone.monthly %>%
  ggplot(aes(x=date, y=mean_ozone)) +
  geom_point() +
  geom_line() +
  geom_smooth(method=lm, se=FALSE) +
  labs(x="Date", y="Mean Ozone (ppm)",
       title="Mean Monthly Ozone Concentration") +
  my_theme
```

```
## `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: From 2010 to 2020, mean annual ozone levels have decreased, decreasing faster from 2010 to 2014 then decreasing slower from 2014 to 2019. There is a seasonal trend as the seasonal mann-kendall test had a p-value of less than 0.05, indicating that there is a significant seasonal trend in the data. When looking at the individual months, however, showed no particular month that had significant p-values, suggesting none of the months had a stronger tendency of decrease.

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
monthly_decomp_components <- as.data.frame(monthly_decomp$time.series[,1:3])

monthly_decomp_components <- mutate(monthly_decomp_components,
        Observed = GaringerOzone.monthly$mean_ozone,
        Date = GaringerOzone.monthly$mean_ozone)

#16
GaringerOzone.monthly.ts2 <- ts(monthly_decomp_components$Observed,
```

```
                   start=c(f_year,f_month),
                   frequency=12)
monthly_trend3 <- Kendall::MannKendall(GaringerOzone.monthly.ts2)
summary(monthly_trend3)
```

```
## Score =  -424 , Var(Score) = 194364.7
## denominator =  7139
## tau = -0.0594, 2-sided pvalue =0.33732
```

Answer: The p-value is greater than 0.05, suggesting that there is no trend when running the Mann Kendall test on the non-seasonal Ozone monthly series. Since the p-value is less than 0.05 for the seasonal mothly series, it suggests that seasonality is an important factor when analyzing trends of ozone concentration in the 2010s.