Assignment 8: Time Series Analysis

Xia Meng Howey

Fall 2025

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

```
#load libraries
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 error
```

```
library(lubridate)
library(here)
```

```
## here() starts at /home/guest/R/EDE_Fall2025
```

```
library(zoo)
```

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

```
library(trend)

#check working directory
getwd()
```

```
## [1] "/home/guest/R/EDE_Fall2025"
```

```
#define my theme, customize panel background

mytheme <- theme_minimal(base_size = 14) +
  theme(panel.background = element_rect(fill = "#ead1dc"),
        axis.title.x = element_text(color = "#0964BD", face = "bold"),
        axis.title.y = element_text(color = "#0964BD", face = "bold"),
        axis.text = element_text(color = "#0964BD"))


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.

```
#1
#upload 10 raw data files
csv.files <- list.files("./Data/Raw/Ozone_TimeSeries", pattern = "*.csv", full.names = TRUE)

#read each CSV file into a list
data.list <- lapply(csv.files, read.csv)

#combine the list of data frames into a single data frame
O3.df <- do.call(rbind, data.list)
```

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. set date column as date class
O3.df$Date <- mdy(O3.df$Date)

# 4 Take slice of df, grabbing date, daily max 8 hr Ozone conc.,
# and daily AQI value columns
O3.df2 <- select(O3.df, Date, Daily.Max.8.hour.Ozone.Concentration,
                 DAILY_AQI_VALUE)

# 5 make new df with all days, rename column to Date
Days <- as.data.frame(seq(as.Date("2010-01-01"), as.Date("2019-12-31"),
                          by = "day"))
names(Days) <- "Date"

# 6 use left_join to combine the data frames
GaringerOzone <- left_join(Days, O3.df2)
```

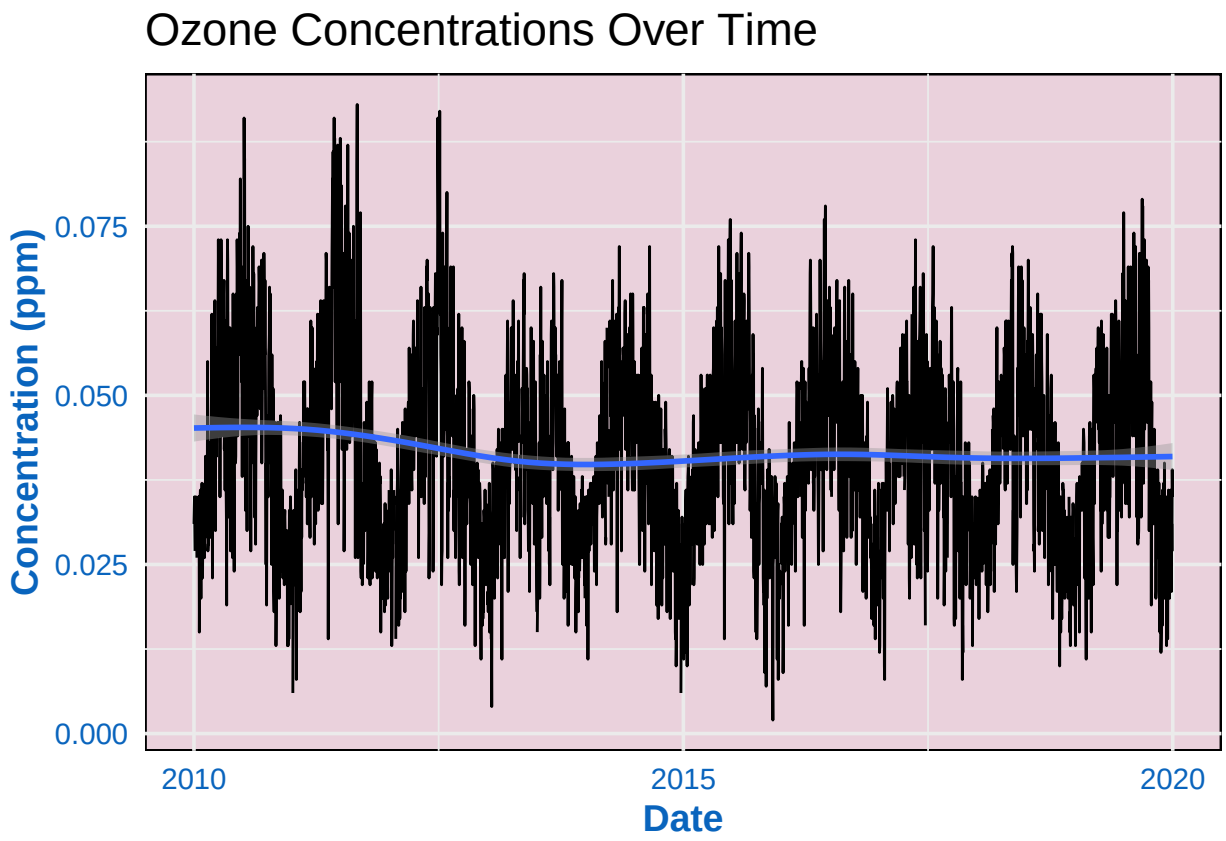
```
## Joining with `by = join_by(Date)`
```

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?

```
#7 make a line plot
O3.plot <- ggplot(GaringerOzone,
              aes(
                x = Date,
                y = Daily.Max.8.hour.Ozone.Concentration
              ))+
  geom_line()+
  geom_smooth()+
  ylab("Concentration (ppm)")+
  labs(title = "Ozone Concentrations Over Time")

print(O3.plot)
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'

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



Answer: Overall there is no trend over time, but there is a seasonal trend.

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
# use linear interpolation to fill in missing daily data for ozone concentration
Gar.Oz.complete <-
  GaringerOzone %>%
  mutate( Daily.Max.8.hour.Ozone.Concentration = zoo::na.approx(Daily.Max.8.hour.Ozone.Concentration))

summary(Gar.Oz.complete$Daily.Max.8.hour.Ozone.Concentration)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00200 0.03200 0.04100 0.04151 0.05100 0.09300
```

Answer: Linear interpolation is good for interpolation here because we see the estimated linear value from the point before to the point after the missing after. Piecewise constant would not be good, because we see the data is dynamically shifting with the seasons, and not staying constant so the piecewise would grab the same value before or after. Spline uses a quadratic rather than a straight line.

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
#wrangle data to make new df with monthly means of ozone
 GaringerOzone.monthly <-  Gar.Oz.complete %>%
  mutate(Year = year(Date),
         Month = month(Date)) %>%
  group_by(Year, Month) %>%
  summarise(Mean = mean(Daily.Max.8.hour.Ozone.Concentration))
```

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

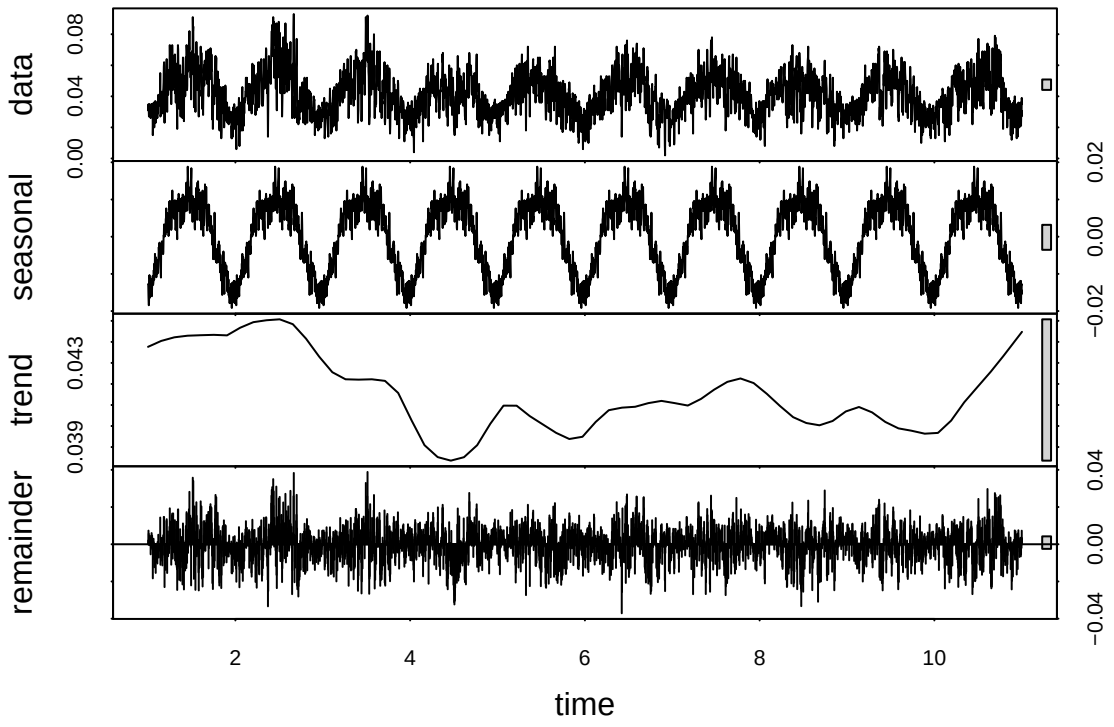
```
GaringerOzone.monthly <- GaringerOzone.monthly %>%
  mutate( Date = my(paste0(Month,"-",Year)))
```

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.

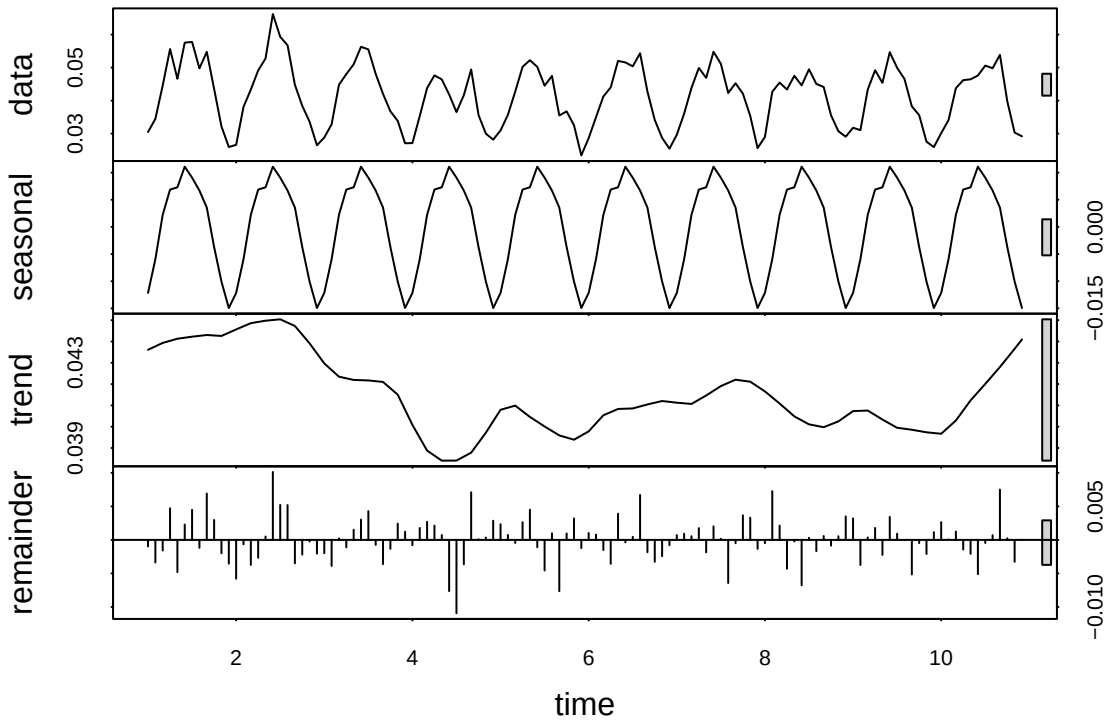
```
#10
#create two time series objects
f_day <- day(first(Gar.Oz.complete$Date))
f_month <- month(first(GaringerOzone.monthly$Date))

GaringerOzone.daily.ts <- ts(Gar.Oz.complete$Daily.Max.8.hour.Ozone.Concentration,
                             start = c(f_day,f_month),
                             frequency = 365)

GaringerOzone.monthly.ts <-
ts(GaringerOzone.monthly$Mean,
   start = c(f_day,f_month),
   frequency = 12)
```

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

```
#11 decompose both ts and plot
GOdaily.decomp <- stl(GaringerOzone.daily.ts,s.window = "periodic")
plot(GOdaily.decomp)
```




```
GOmonthlyts.decomp <- stl(GaringerOzone.monthly.ts,s.window = "periodic")
plot(GOmonthlyts.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
#Run SMK test
monthlyOtrend <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)

#Inspect results
monthlyOtrend
```

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

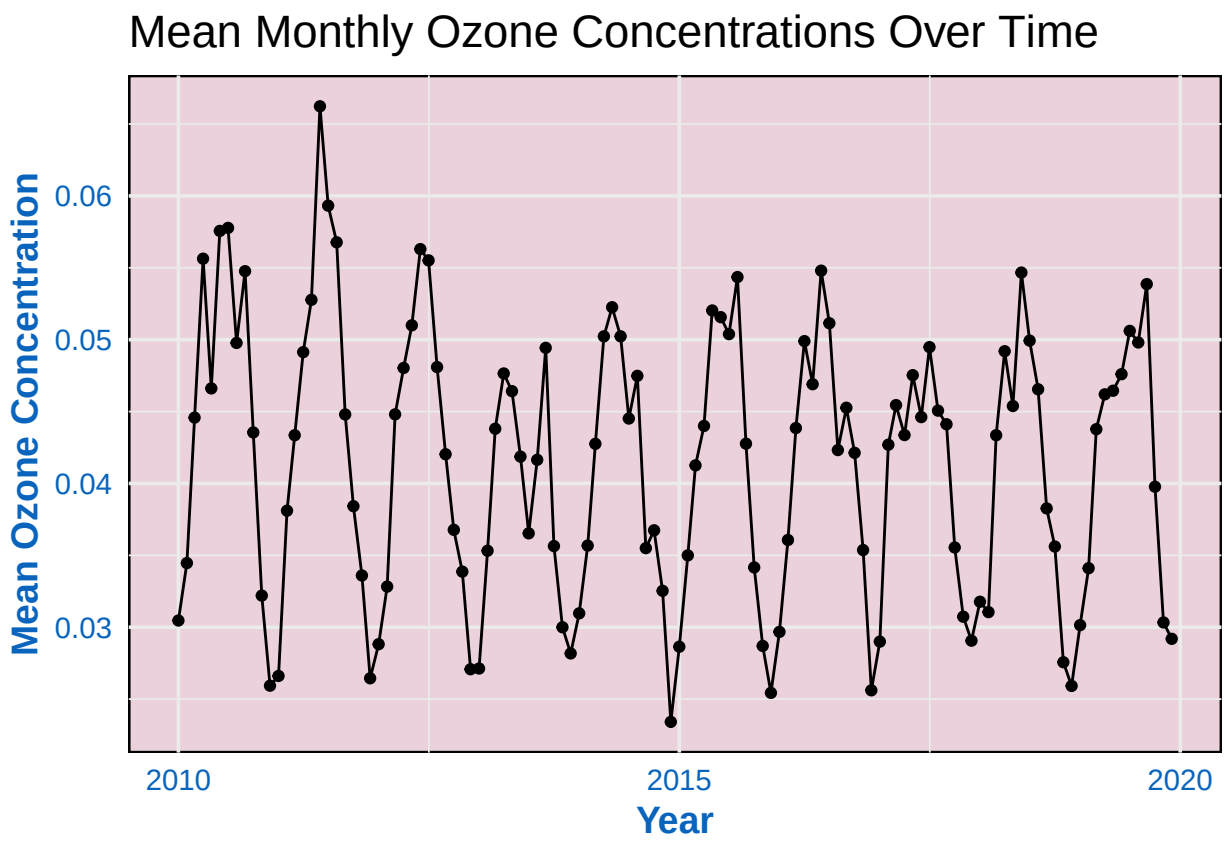
```
summary(monthlyOtrend)
```

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

Answer: Because this is the only test that works with seasonal data

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
Mean.Oz.plot <- ggplot(GaringerOzone.monthly,
                       aes(
                         x = Date,
                         y = Mean
                       ))+
  geom_point()+
  geom_line()+
  ylab("Mean Ozone Concentration")+
  xlab("Year")+
  labs(title = "Mean Monthly Ozone Concentrations Over Time")

print(Mean.Oz.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: There is a statistically significant decreasing trend in the data. (p value was under 0.05, although it was close at 0.046, which means we reject the null hypothesis that the data is stationary)

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
#substract seasonal component from monthly ts
GaringerOzone.monthly.stl <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
GaringerOzone.monthly.deseasonalized <- GaringerOzone.monthly.ts - GaringerOzone.monthly.stl$time.series

#16
#run non-seasonal Mann Kendall
monthlyOtrend.nonseasonal <- Kendall::MannKendall(GaringerOzone.monthly.deseasonalized)


#Inspect results
monthlyOtrend.nonseasonal
```

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

```
summary(monthlyOtrend.nonseasonal)
```

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

Answer: The p value is much lower at 0.0075 than the seasonal Mann Kendall test so there is stronger stastistical significance of a downward trend when the seasonality is removed.