Assignment 7: Time Series Analysis

Ardath Dixon

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on time series analysis.

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, submit the completed exercise (PDF file) to the dropbox in Sakai. Add your last name into the file name (e.g., "Fay_A07_TimeSeries.Rmd") prior to submission.

The completed exercise is due on Tuesday, March 16 at 11:59 pm.

Set up

1. Set up your session:

- Check your working directory
- Load the tidyverse, lubridate, zoo, and trend packages
- Set your ggplot theme

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
getwd() ## checking working directory. setwd() if need to change
```

```
## [1] "C:/Users/ardat/OneDrive/Documents/DataAnalytics/Environmental_Data_Analytics_2021"
```

```
library(tidyverse)
library(lubridate)
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 4.0.4
```

```
library(trend)
```

```
## Warning: package 'trend' was built under R version 4.0.4
```

```
## set ggplot theme
mytheme <- theme_light(base_size = 10)+
  theme(axis.text = element_text(color = "black"))
theme_set(mytheme)

library(plyr) ## package to enable ldply function below
```

```
## bulk import of ozone files & combine into 1 df
GaringerOzoneFiles = list.files(path = "./Data/Raw/Ozone_TimeSeries/",
                           pattern = "*.csv", full.names = TRUE)
GaringerOzone <- GaringerOzoneFiles %>%
  ldply(read.csv)
```

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.

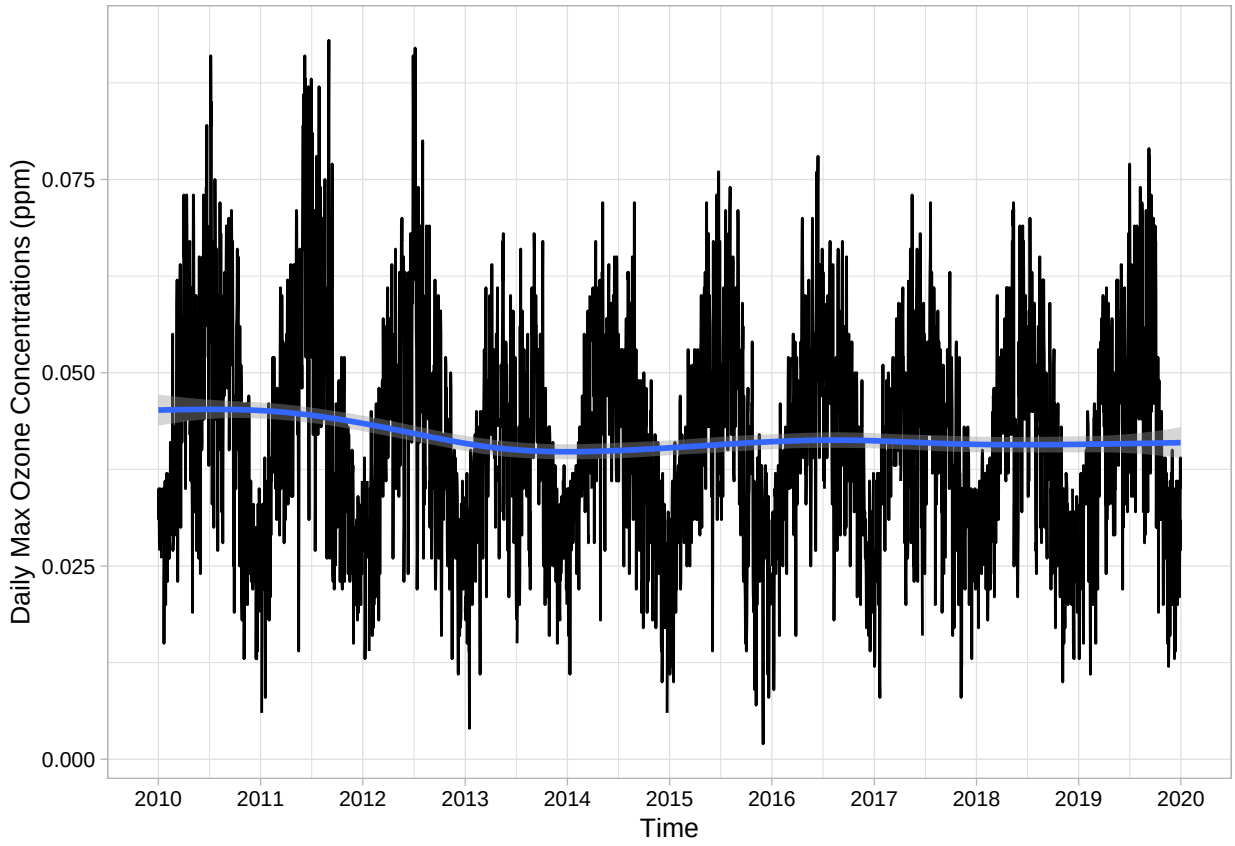
```
# 3 set date column as date class
GaringerOzone$Date <- as.Date(GaringerOzone$Date, format = "%m/%d/%Y")

# 4 wrangle dataset to only include 3 columns
GaringerOzone <- select(GaringerOzone, Date,
                        Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)

# 5 create df with all dates in the window
startday = as.Date('2010-01-01', format = '%Y-%m-%d')
endday = as.Date('2019-12-31', format = '%Y-%m-%d')

Days <- as.data.frame(seq(from = startday, to = endday, by = 'day'))
Days <- rename(Days, replace = c('seq(from = startday, to = endday, by = "day")'
                                 = "Date"))

# 6 combine dataframes
GaringerOzone <- left_join(Days, GaringerOzone, by = c("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 create line plot of ozone concentrations over time
OzoneTimePlot <- ggplot(GaringerOzone,
                        aes(x= Date,
                            y = Daily.Max.8.hour.Ozone.Concentration))+
  geom_line()+
  geom_smooth()+
  labs(x = "Time", y = "Daily Max Ozone Concentrations (ppm)")+
  scale_x_date(date_breaks = "years", date_labels = "%Y")
OzoneTimePlot
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
## Warning: Removed 63 rows containing non-finite values (stat_smooth).
```



Answer: This plot suggests a slight decrease in ozone concentration over time. There is a slight negative slope to the smoothed line.

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 linear interpolation
GaringerOzone_clean <-
  GaringerOzone %>%
  mutate(Daily.Max.8.hour.Ozone.Concentration =
           na.approx(Daily.Max.8.hour.Ozone.Concentration)) %>%
  mutate(DAILY_AQI_VALUE = na.approx(DAILY_AQI_VALUE))
```

Answer: Linear interpolation fits here because atmospheric conditions change gradually over time. Logically, when one day's measurement changes to the measurement taken two days later, it must pass through the halfway point. Linear interpolation reasons that the halfway point between the two measurements is what the atmospheric measurement will be for the middle NA day. The piecewise constant method takes the nearest neighbor's value. This assumes a more stepwise change process, where there would be no change and then greater change all at once. For atmospheric changes, a gradual linear shift is a better fit. Spline assumes a quadratic relationship,

but a linear relationship fits better logically for this comparison as well (as ozone concentration increases, the rate of increase doesn't respectively also increase).

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_clean %>%
  mutate(Month = month(Date)) %>%
  mutate(Year = year(Date)) %>%
  group_by(Year, Month) %>%
  dplyr::summarize(mean_ozone = mean(Daily.Max.8.hour.Ozone.Concentration))
```

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

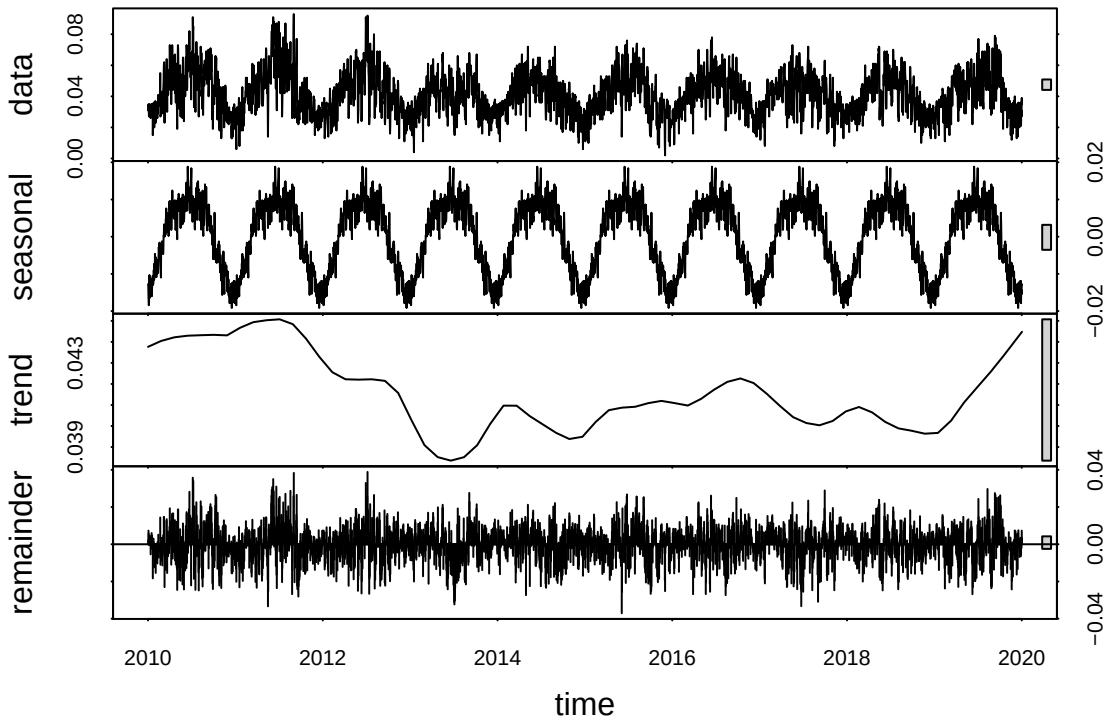
```
## create GraphDate column with outputs indicating the 1st of each mo. (Y-m-01)
create.dates <- (function(m,y) {paste0(y,("-"),m,("-01"))})
GaringerOzone.monthly$GraphDate <- create.dates(GaringerOzone.monthly$Month,
                                                GaringerOzone.monthly$Year)
GaringerOzone.monthly$GraphDate <- as.Date(GaringerOzone.monthly$GraphDate,
                                           format = "%Y-%m-%d")
```

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.

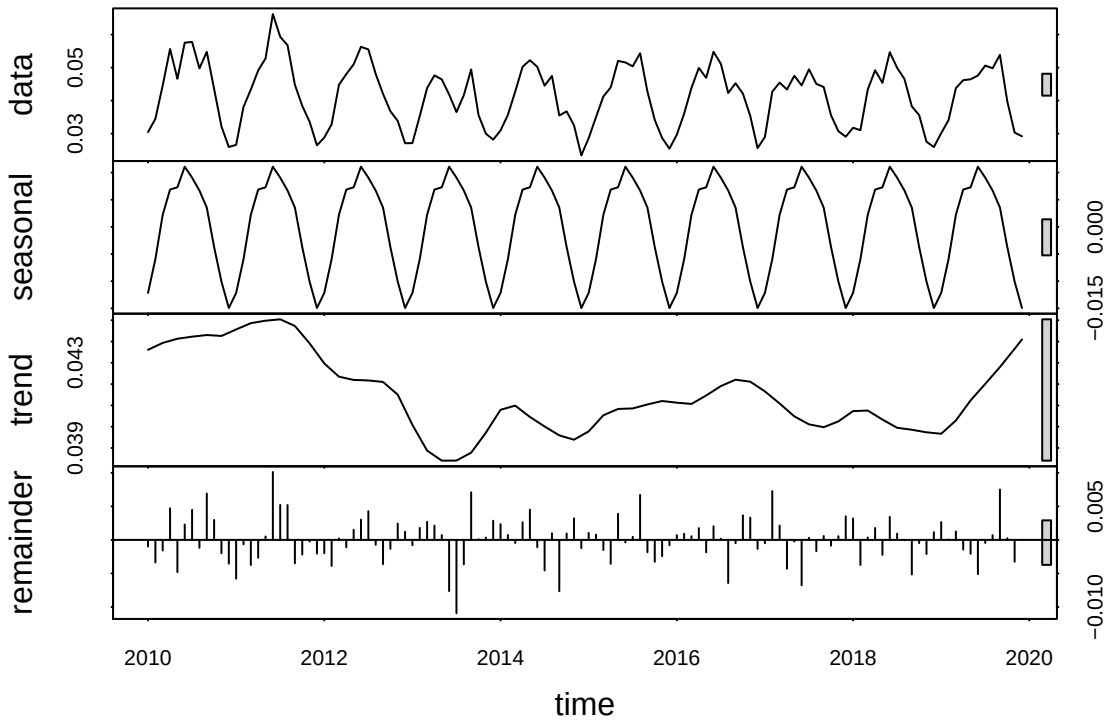
```
#10 Generate two time series objects
GaringerOzone.daily.ts <-
  ts(GaringerOzone_clean$Daily.Max.8.hour.Ozone.Concentration,
                              start = c(2010, 1), frequency = 365)

GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$mean_ozone,
                               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 Decompose and plot the daily & monthly time series objects

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?

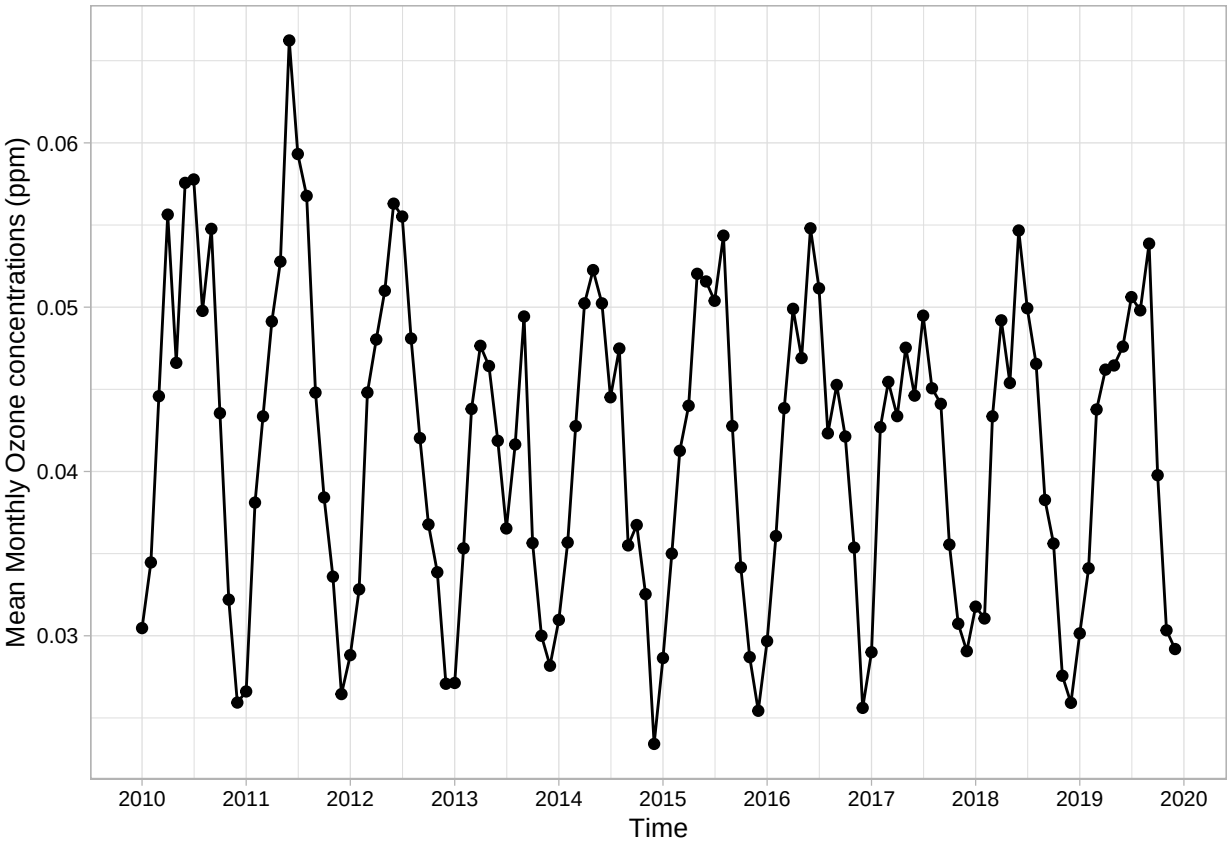
```
#12 monotonic trend analysis with seasonal Mann-Kendall

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

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

Answer: The seasonal Mann-Kendall is most appropriate in this case because the data shows seasonal trends. Plotting the time series objects from #11 show lines that go up and down each year, and therefore seasonality. The other monotonic trend analysis tests (linear regression, Mann-Kendall, Spearman Rho) require the data to have 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 plot of mean monthly ozone concentrations over time
GaringerOzone.monthly.plot <- ggplot(GaringerOzone.monthly,
                                     aes(x = GraphDate, y = mean_ozone))+
  geom_point()+
  geom_line()+
  labs(x = "Time", y = "Mean Monthly Ozone concentrations (ppm)")+
  scale_x_date(date_breaks = "year", date_labels = "%Y")
print(GaringerOzone.monthly.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: Data suggests that ozone concentrations have changed at this station over the 2010s decade ($p = 0.047$). This incorporated seasonal changes with the Seasonal Mann-Kendall. Ozone concentrations show greater seasonal ranges (higher max & lower min) over the earlier years, and become generally more subdued over time. The mean max ozone concentration levels also show a gradual, general decrease 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.

```
#15 subtract seasonal component from monthly data
GaringerOzone.monthly.components <-
  as.data.frame(GaringerOzone.monthly.decomposed$time.series[,1:3])

GaringerOzone.monthly.components <-
  mutate(GaringerOzone.monthly.components,
         Observed = GaringerOzone.monthly$mean_ozone,
         Date = GaringerOzone.monthly$GraphDate,
         NoSeasonal = Observed - seasonal)

GaringerOzone.monthly_noseasonaltrend_ts <-
  ts(GaringerOzone.monthly.components$NoSeasonal,
```

```
     start = c(2010,1), frequency = 12)

#16 run the mann kendall on this non-seasonal data
GaringerOzone.trend2 <-
  Kendall::MannKendall(GaringerOzone.monthly_noseasonaltrend_ts)
summary(GaringerOzone.trend2)
```

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

Answer: The non-seasonal monthly Ozone data gives a more significant Mann Kendall test result than the seaonal monthy data's Seasonal Mann Kendall result. Mann Kendall gives a p-value = 0.00754, and Seasonal Mann Kendall gives a p-value = 0.0467.