Assignment 8: Time Series Analysis

Nagarajan Vaidya Subramanian

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
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 1.0.0
## -- 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
```

```
#install.packages("zoo")
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 4.2.3
```

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

```
#install.packages("trend")
library(trend)
```

```
## Warning: package 'trend' was built under R version 4.2.3
```

```
library(here)
```

```
## here() starts at X:/ENV 872 Environmental Data Analytics/Git_codes/EDA-Spring2023
```

```
getwd()
```

```
## [1] "X:/ENV 872 Environmental Data Analytics/Git_codes/EDA-Spring2023"
```

```
my_theme <- theme_minimal() +
  theme(
    legend.position = "bottom",
    legend.title = element_text(colour = "#440000"),
    plot.title = element_text(colour = "blue",
                              hjust = 0.5),
    plot.subtitle = element_text(hjust = 0.5),
    axis.title = element_text(colour = "#4169e1")
  )
theme_set(my_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.

```
#2
GaringerOzone <- list.files(path = 'Data/Raw/Ozone_TimeSeries',
                            pattern = "*.csv",full.names =TRUE) %>%
  lapply(read.csv) %>%
```

```
  bind_rows


#GaringerOzone <- list.files(path = here('Data/Raw/Ozone_TimeSeries/'),
#                             pattern = ".csv") %>%
#  import_list(rbind = TRUE)

#ozone2010 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv'), strin
#ozone2011 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv'), strin
#ozone2012 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv'), strin
#ozone2013 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv'), strin
#ozone2014 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv'), strin
#ozone2015 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv'), strin
#ozone2016 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv'), strin
#ozone2017 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv'), strin
#ozone2018 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv'), strin
#ozone2019 <- read.csv(file = here('Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv'), strin
#GaringerOzone <- rbind(ozone2010, ozone2011, ozone2012, ozone2013, ozone2014, ozone2015, ozone2016, oz
```

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
glimpse(GaringerOzone)
```

```
## Rows: 3,589
## Columns: 20
## $ Date                                  <chr> "01/01/2010", "01/02/2010", "01/0~
## $ Source                                <chr> "AQS", "AQS", "AQS", "AQS", "AQS"~
## $ Site.ID                               <int> 371190041, 371190041, 371190041, ~
## $ POC                                   <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ Daily.Max.8.hour.Ozone.Concentration <dbl> 0.031, 0.033, 0.035, 0.031, 0.027~
## $ UNITS                                 <chr> "ppm", "ppm", "ppm", "ppm", "ppm"~
## $ DAILY_AQI_VALUE                       <int> 29, 31, 32, 29, 25, 31, 32, 30, 3~
## $ Site.Name                             <chr> "Garinger High School", "Garinger~
## $ DAILY_OBS_COUNT                       <int> 17, 17, 17, 17, 17, 17, 17, 17, 1~
## $ PERCENT_COMPLETE                      <dbl> 100, 100, 100, 100, 100, 100, 100~
## $ AQS_PARAMETER_CODE                    <int> 44201, 44201, 44201, 44201, 44201~
## $ AQS_PARAMETER_DESC                    <chr> "Ozone", "Ozone", "Ozone", "Ozone~
## $ CBSA_CODE                             <int> 16740, 16740, 16740, 16740, 16740~
```

```
## $ CBSA_NAME                              <chr> "Charlotte-Concord-Gastonia, NC-S~
## $ STATE_CODE                             <int> 37, 37, 37, 37, 37, 37, 37, 37, 3~
## $ STATE                                  <chr> "North Carolina", "North Carolina~
## $ COUNTY_CODE                            <int> 119, 119, 119, 119, 119, 119, 119~
## $ COUNTY                                 <chr> "Mecklenburg", "Mecklenburg", "Me~
## $ SITE_LATITUDE                          <dbl> 35.2401, 35.2401, 35.2401, 35.240~
## $ SITE_LONGITUDE                         <dbl> -80.78568, -80.78568, -80.78568, ~
```

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

#4
GaringerOzone_sub <- GaringerOzone %>%
  select(Date, Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE) %>%
  rename(ozone = Daily.Max.8.hour.Ozone.Concentration)


#5
startDate <- ymd("2010-01-01")
endDate <- ymd("2019-12-31")
Days <- as.data.frame(seq(from = startDate, to = endDate, by = 1))
colnames(Days) <- "Date"

#6
GaringerOzone_cts <- left_join(Days, GaringerOzone_sub) %>%
  select(-c(DAILY_AQI_VALUE))
```

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

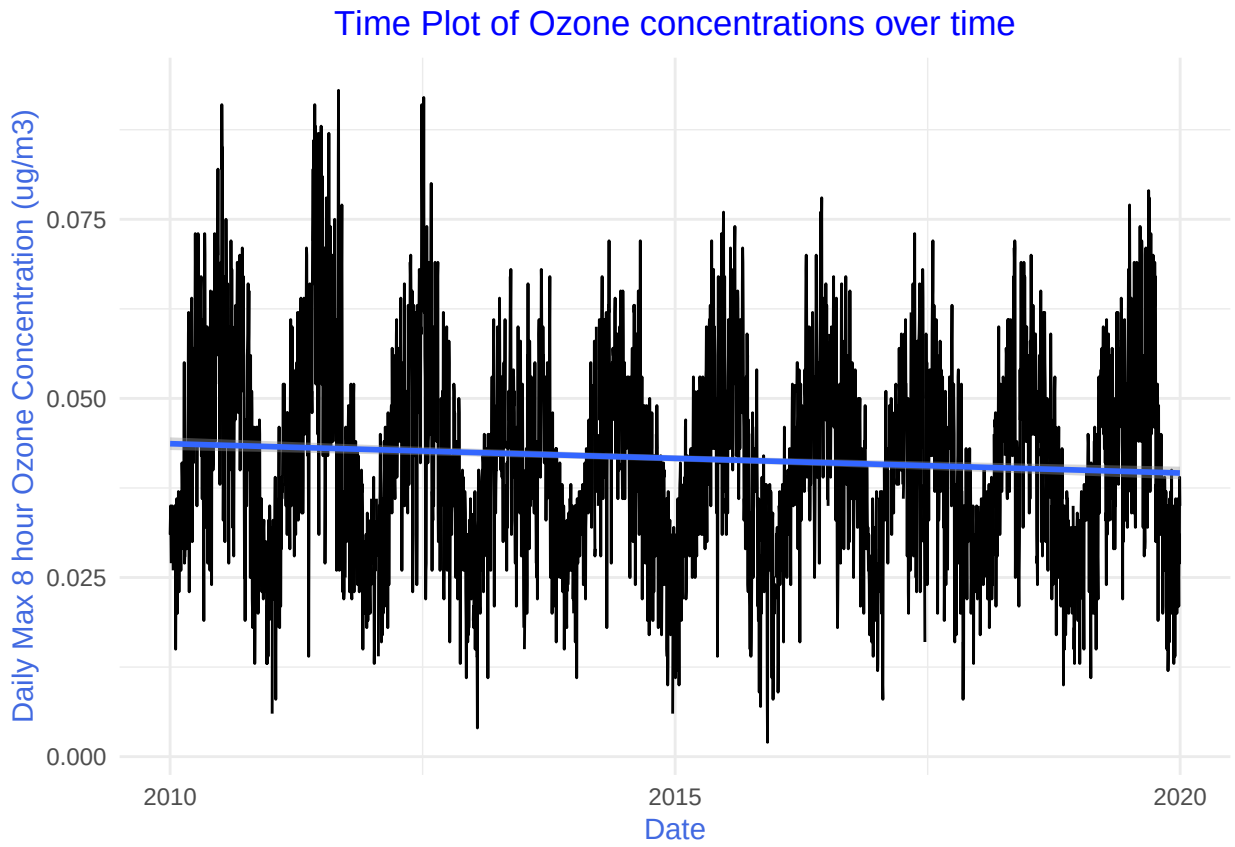
Visualize

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
ggplot(data = GaringerOzone_cts,
       mapping = aes(x = Date,
                     y = ozone)) +
  geom_line() +
  geom_smooth(method = "lm") +
  labs(title = "Time Plot of Ozone concentrations over time",
       x = "Date",
       y = "Daily Max 8 hour Ozone Concentration (ug/m3)") +
  scale_x_date()
```

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

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



Answer: There appears to be a trend of Ozone concentrations gradually decreasing over time, as seen by the small negative slope of the smoothed trend 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
summary(GaringerOzone_cts)
```

```
##       Date                ozone        
##  Min.   :2010-01-01   Min.   :0.00200  
##  1st Qu.:2012-07-01   1st Qu.:0.03200  
##  Median :2014-12-31   Median :0.04100  
##  Mean   :2014-12-31   Mean   :0.04163  
##  3rd Qu.:2017-07-01   3rd Qu.:0.05100  
##  Max.   :2019-12-31   Max.   :0.09300  
##                       NA's   :63
```

```
GaringerOzone_cts <- GaringerOzone_cts %>%
  mutate(ozone = zoo::na.approx(ozone))
summary(GaringerOzone_cts)
```

```
##       Date                ozone        
##  Min.   :2010-01-01   Min.   :0.00200  
##  1st Qu.:2012-07-01   1st Qu.:0.03200  
##  Median :2014-12-31   Median :0.04100  
##  Mean   :2014-12-31   Mean   :0.04151  
##  3rd Qu.:2017-07-01   3rd Qu.:0.05100  
##  Max.   :2019-12-31   Max.   :0.09300
```

Answer: We avoid a piece-wise constant interpolation because that would set the missing values to be equal to one of their nearest neighbours. Whereas we know that is not true of the data.

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_cts %>%
  mutate(Year = year(Date),
         Month = month(Date)) %>%
  group_by(Year, Month) %>%
  mutate(meanOzone = mean(ozone)) %>%
  select(Date, Month, Year, meanOzone) %>%
  mutate(Date = floor_date(Date, unit = "month")) %>%
  distinct()
```

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.

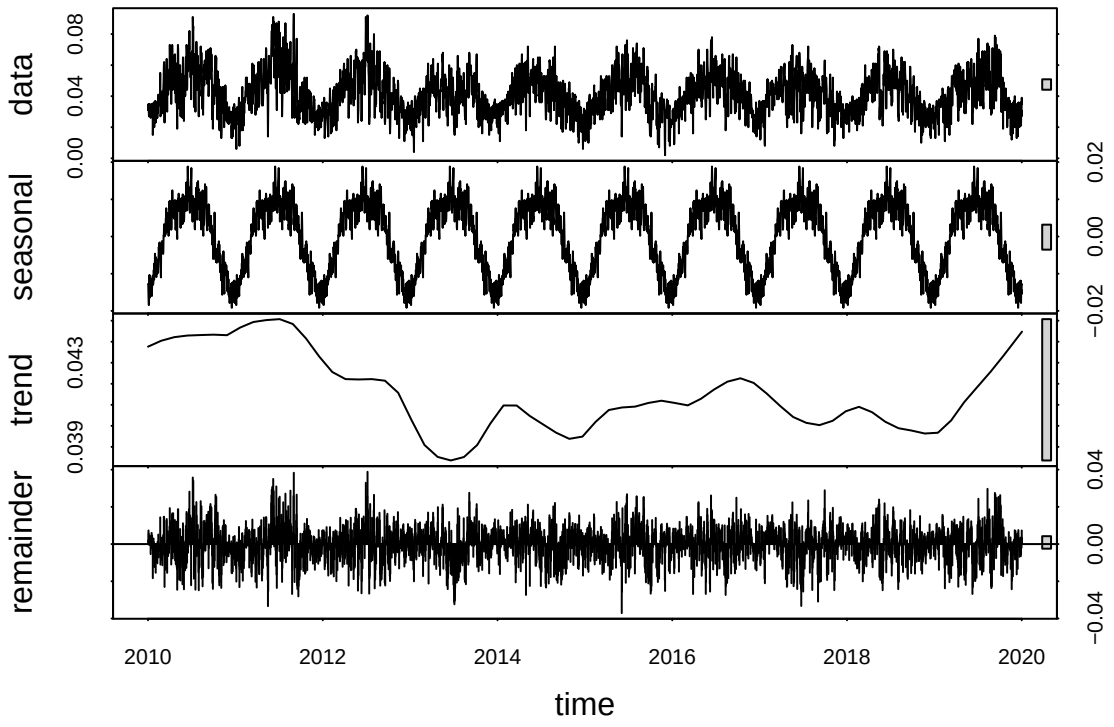
```
#10
GaringerOzone.daily.ts <- ts(data = GaringerOzone_cts$ozone,
                             start = c(year(first(GaringerOzone_cts$Date)),
                                       month(first(GaringerOzone_cts$Date))),
                             frequency = 365)

GaringerOzone.monthly.ts <- ts(data = GaringerOzone.monthly$meanOzone,
                               start = c(year(first(GaringerOzone.monthly$Date)),
                                         month(first(GaringerOzone.monthly$Date))),
                               end = c(year(last(GaringerOzone.monthly$Date)),
                                         month(last(GaringerOzone.monthly$Date))),
                               frequency = 12)
```

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

```
#11
GaringerOzone.daily.stl <- stl(GaringerOzone.daily.ts,
                               s.window = "periodic")
GaringerOzone.monthly.stl <- stl(GaringerOzone.monthly.ts,
                                 s.window = "periodic")
print("Plot of the daily time series")
```

```
## [1] "Plot of the daily time series"
```

```
plot(GaringerOzone.daily.stl)
```



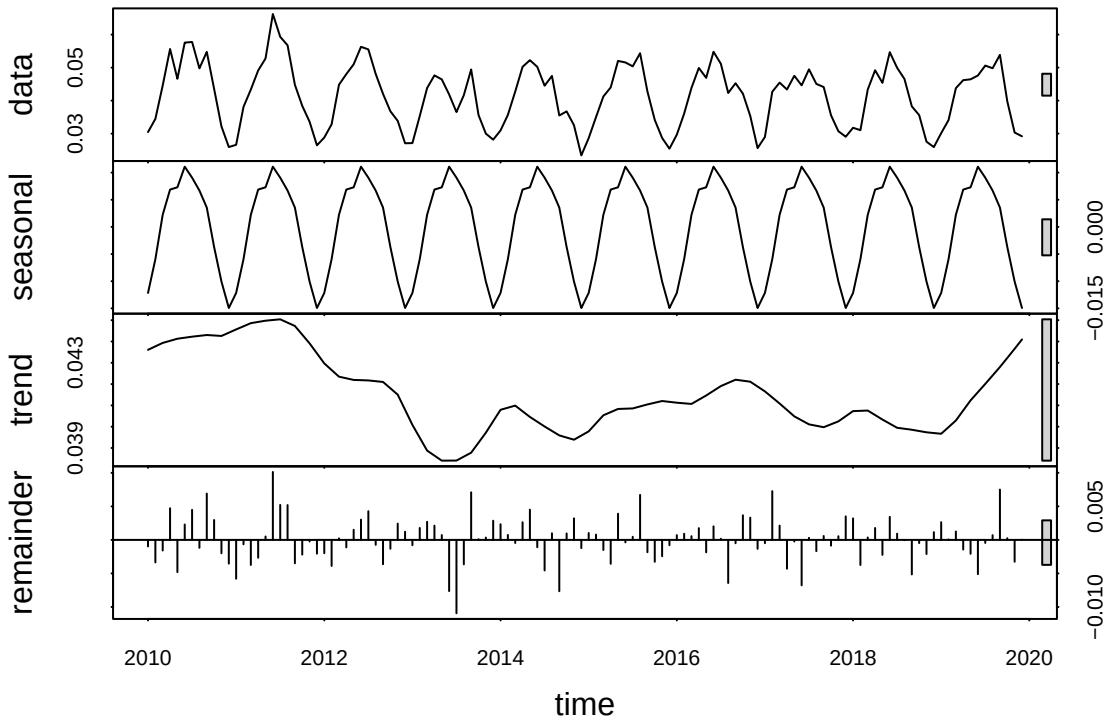

```
print("***")
```

```
## [1] "***"
```

```
print("Plot of the monthly time series")
```

```
## [1] "Plot of the monthly time series"
```

```
plot(GaringerOzone.monthly.stl)
```



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
#install.packages("Kendall")
smk.monthlydata <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
summary(smk.monthlydata)
```

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

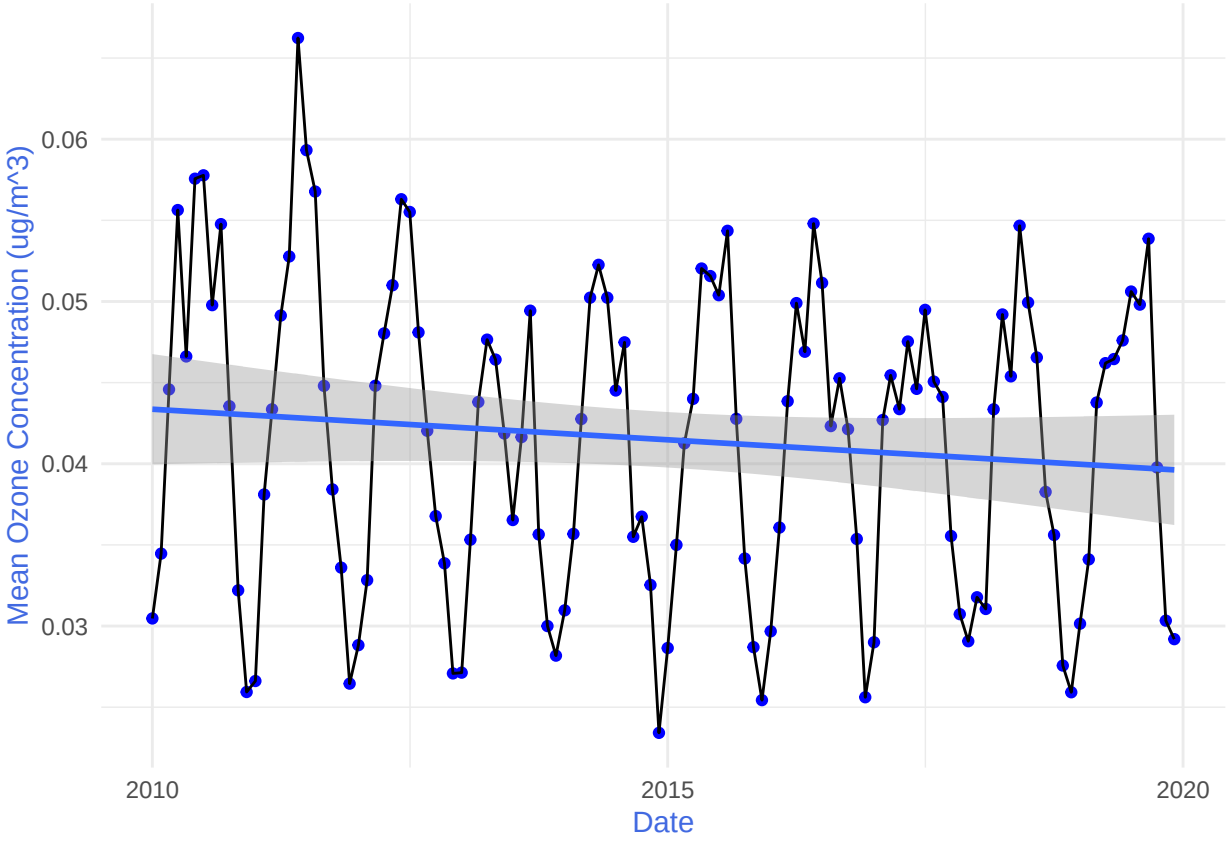
Answer: The regular Mann-Kendall does not account for seasonality of the data. Since our data has a yearly seasonality, we will need the seasonal Mann-Kendall.

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
ggplot(data = GaringerOzone.monthly,
       mapping = aes(x = Date,
                     y = meanOzone)) +
  geom_point(color = "blue") +
  geom_line() +
```

```
  geom_smooth(method = "lm") +
  labs(x = "Date",
       y = "Mean Ozone Concentration (ug/m^3)")
```

```
## `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: The null hypothesis of the seasonal Mann-Kendall test is that observations are independent, randomly ordered, and without serial correlation. The p-value of the 2-sided test on our monthly data is 0.047 which is less than 0.05. Therefore we can reject the null hypothesis at 90% confidence level, but accept it with a 95% confidence level.

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
GaringerOzone.monthly.nonseasonal <- GaringerOzone.monthly.stl$time.series[,1:3]
```

```r
#16
smk.nonseasonal  <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.nonseasonal)
summary(smk.nonseasonal)
```

```
## Score =  -226 , Var(Score) = 36200
## denominator =  4942.631
## tau = -0.0457, 2-sided pvalue =0.2349
```

Answer: The seasonal Mann-Kendall test for the non-seasonal data has a p-value of 0.23 which is greater than 0.5. We can therefore accept the null hypothesis, and conclude that there is no significant dependency, order, or serial correlation in our data.