# Assignment 7: Time Series Analysis

Andrew Brantley

## 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 Monday, March 14 at 7:00 pm.

### Set up

1. Set up your session:

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

```
#1

# checking working directory
getwd()
```

```
## [1] "/Users/AndrewBrantley/Library/CloudStorage/Box-Box/Environmental Data Analytics/GithubRepos/Envi
```

```
# loading packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

```
## -- 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
```

```
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

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

```
library(trend)
library(plyr)
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following object is masked from 'package:purrr':
##
##     compact
```

```
library(Kendall)

# building personal theme
Andrew.Theme <- theme_gray(base_size = 14) +
  theme(axis.text = element_text(colour = "black", face = "italic"),
        legend.position = "right",
        panel.grid.major.x = element_line(colour = "black", linetype = 3, size = 0.5),
        panel.grid.major.y =  element_line(colour = "black", linetype = 3, size = 0.5))

theme_set(Andrew.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

# importing ten Ozone datasets
OzoneFiles = list.files(path = "../Data/Raw/Ozone_TimeSeries/", pattern="*.csv", full.names=TRUE)
OzoneFiles
```

```
##  [1] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2010_raw.csv"
##  [2] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2011_raw.csv"
```

```
##  [3] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2012_raw.csv"
##  [4] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2013_raw.csv"
##  [5] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2014_raw.csv"
##  [6] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2015_raw.csv"
##  [7] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2016_raw.csv"
##  [8] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2017_raw.csv"
##  [9] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2018_raw.csv"
## [10] "../Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2019_raw.csv"
```

```
# combining datasets into one dataframe
GaringerOzone <- OzoneFiles %>%
  ldply(read.csv)
dim(GaringerOzone)
```

```
## [1] 3589   20
```

## 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

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

# 4

# filtering dataset for Date, Daily.Max.8.hour.Ozone.Concentration, and DAILY_AQI_VALUE columns
GaringerOzone.Subset <- GaringerOzone %>%
                          select(Date, Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)

# 5

# creating daily dataset
Days <- as.data.frame(seq(as.Date("2010-01-01"), as.Date("2019-12-31"), by = "days"))

# renaming column name to "Date"
colnames(Days) <- c("Date")

# 6

# joining dataframes
GaringerOzone <- left_join(Days, GaringerOzone.Subset, by = c("Date"))
dim(GaringerOzone)
```

```
## [1] 3652    3
```

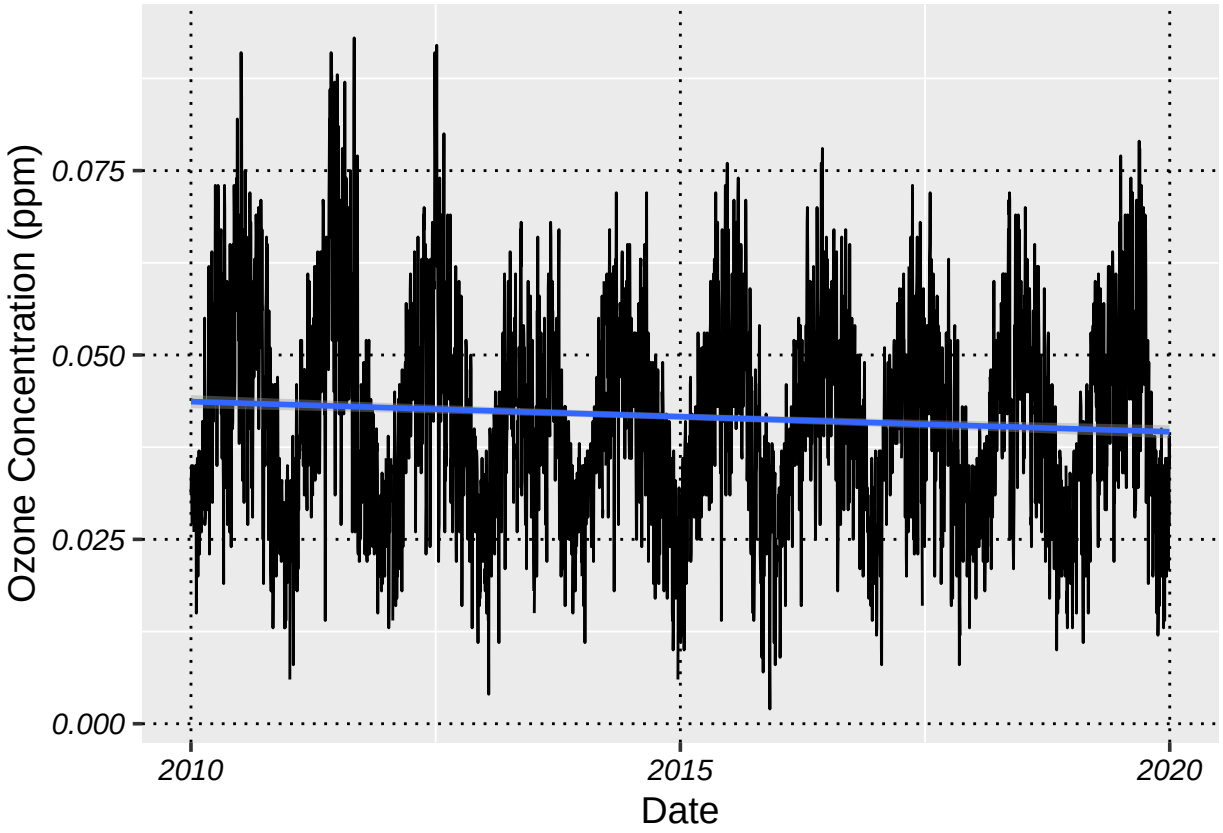
## Visualize

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

# graphing ozone data over time

OzoneConc.Plot <- ggplot(GaringerOzone, aes(Date, Daily.Max.8.hour.Ozone.Concentration)) +
                      geom_line() +
                      geom_smooth(method = "lm") +
                      labs(x = "Date", y = "Ozone Concentration (ppm)")
OzoneConc.Plot
```

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

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




Answer: From this plot it seems that there is a very slight downward trend over this decade. There is also a very noticeable seasonal trend that oscillates the ozone concentrations on a yearly basis.

## 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

# filling missing data with linear interpolation
GaringerOzone.Clean <-
  GaringerOzone %>%
  mutate(OzoneConc_Clean = zoo::na.approx(Daily.Max.8.hour.Ozone.Concentration))
```

Answer: We don't use the piecewise constant interpolation because it is based on a nearest neighbor approach and each point in the dataset has two equally near neighbors (day before the NA and day after the NA). We don't use the spline because relationships between data points is likely to be much more linear than polynomial. Linear interpolation is also best used when you are missing values sporadically and not in big chunks.

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

# creating monthly average dataset
GaringerOzone.monthly <- GaringerOzone.Clean %>%
                            mutate(Month = month(Date),
                                   Year = year(Date)) %>%
                            mutate(Month_Year = my(paste0(Month, "-", Year))) %>%
                            dplyr::group_by(Month_Year, Month, Year) %>%
                            dplyr::summarise(MeanOzone = mean(OzoneConc_Clean))
```

```
## `summarise()` has grouped output by 'Month_Year', 'Month'. You can override using the `.groups` argu
```

```
head(GaringerOzone.monthly)
```

```
## # A tibble: 6 x 4
## # Groups:   Month_Year, Month [6]
##   Month_Year Month  Year MeanOzone
##   <date>     <dbl> <dbl>     <dbl>
## 1 2010-01-01     1  2010    0.0305
## 2 2010-02-01     2  2010    0.0345
## 3 2010-03-01     3  2010    0.0446
## 4 2010-04-01     4  2010    0.0556
## 5 2010-05-01     5  2010    0.0466
## 6 2010-06-01     6  2010    0.0576
```

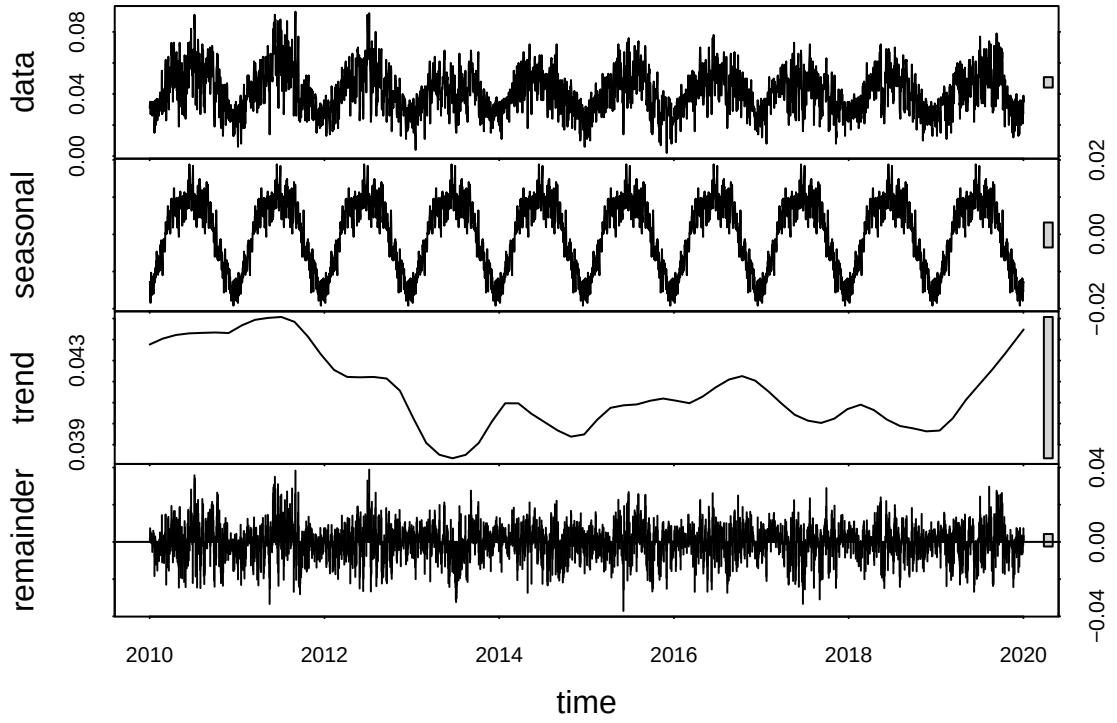
10. Generate two time series objects. Name the first `GaringerOzone.daily.ts` and base it on the dataframe of daily observations. Name the second `GaringerOzone.monthly.ts` and base it on the monthly average ozone values. Be sure that each specifies the correct start and end dates and the frequency of the time series.

```
#10

# creating daily time series
GaringerOzone.daily.ts <- ts(GaringerOzone.Clean$OzoneConc_Clean,
                      start = c(2010, 01, 01), frequency = 365)

# creating monthly time series
```

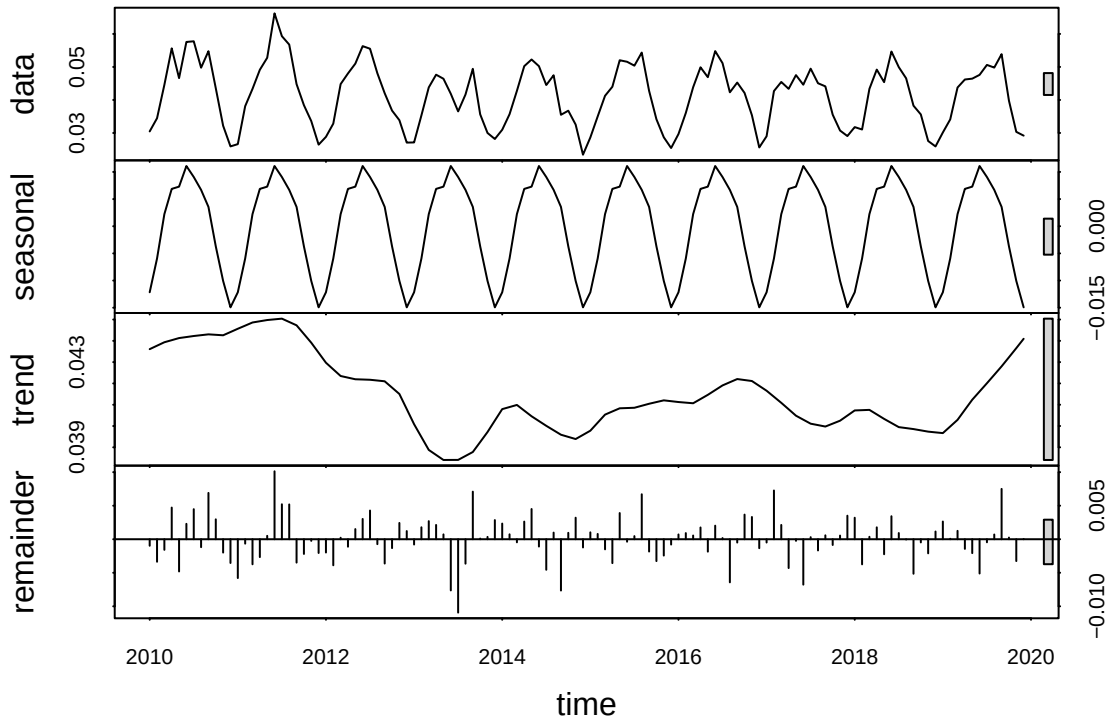
```
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$MeanOzone,
                      start = c(2010, 01, 01), frequency = 12)
```

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

```
#11

# decomposing daily time series
GaringerOzoneDaily.decomp <- stl(GaringerOzone.daily.ts, s.window = "periodic")
plot(GaringerOzoneDaily.decomp)
```




```
# decomposing monthly time series
GaringerOzoneMonthly.decomp <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
plot(GaringerOzoneMonthly.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?

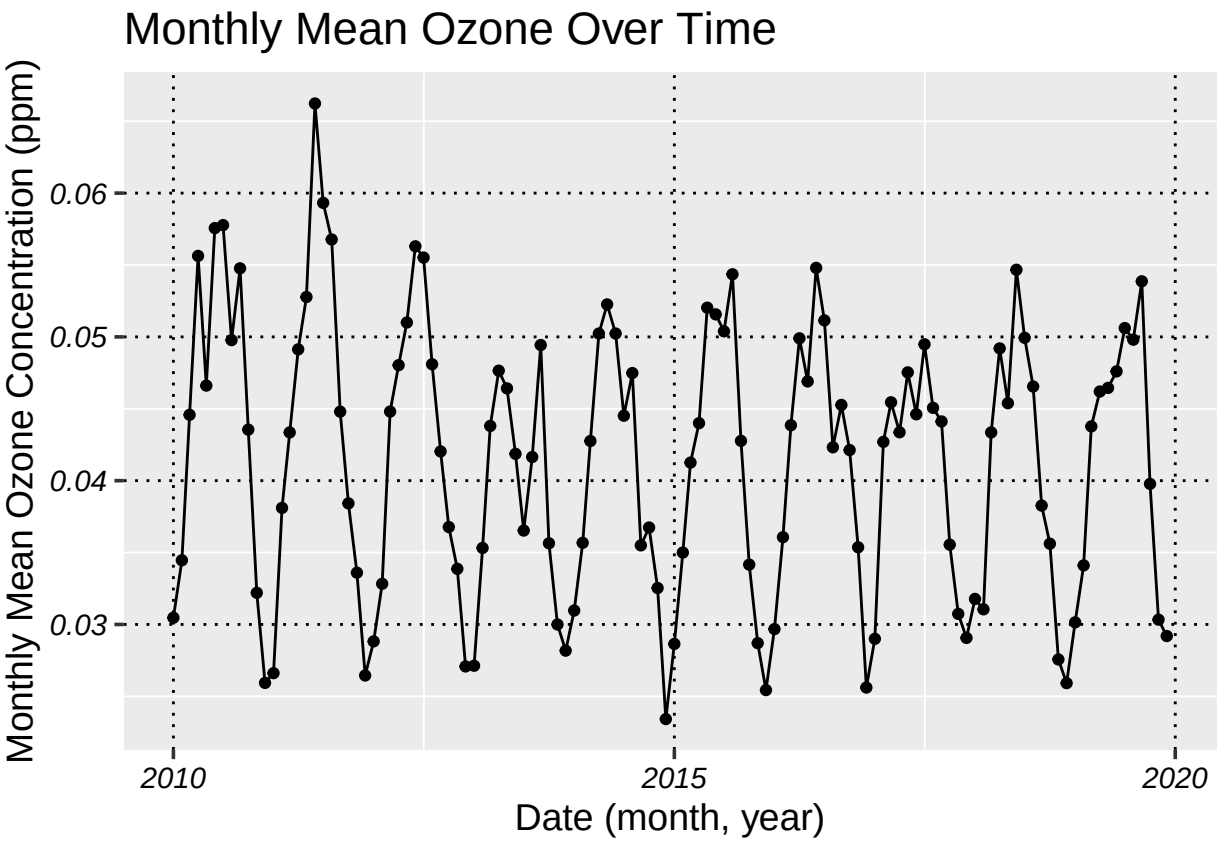
```
#12

# running trend analysis on monthly data
Monthly.SMK<- SeasonalMannKendall(GaringerOzone.monthly.ts)
summary(Monthly.SMK)
```

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

Answer: The Seasonal Mann Kendall test is most appropriate here because there is clear sesonality in our data that must be taken into consideration. This test also allows for non-parametric data to be used.

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

# monthly ozone plot over time
MonthlyOzone.Plot <- ggplot(GaringerOzone.monthly) +
                        geom_point(aes(Month_Year, MeanOzone)) +
                        geom_line(aes(Month_Year, MeanOzone)) +
                        labs(x = "Date (month, year)", y = "Monthly Mean Ozone Concentration (ppm)",
                             title = "Monthly Mean Ozone Over Time")
MonthlyOzone.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.

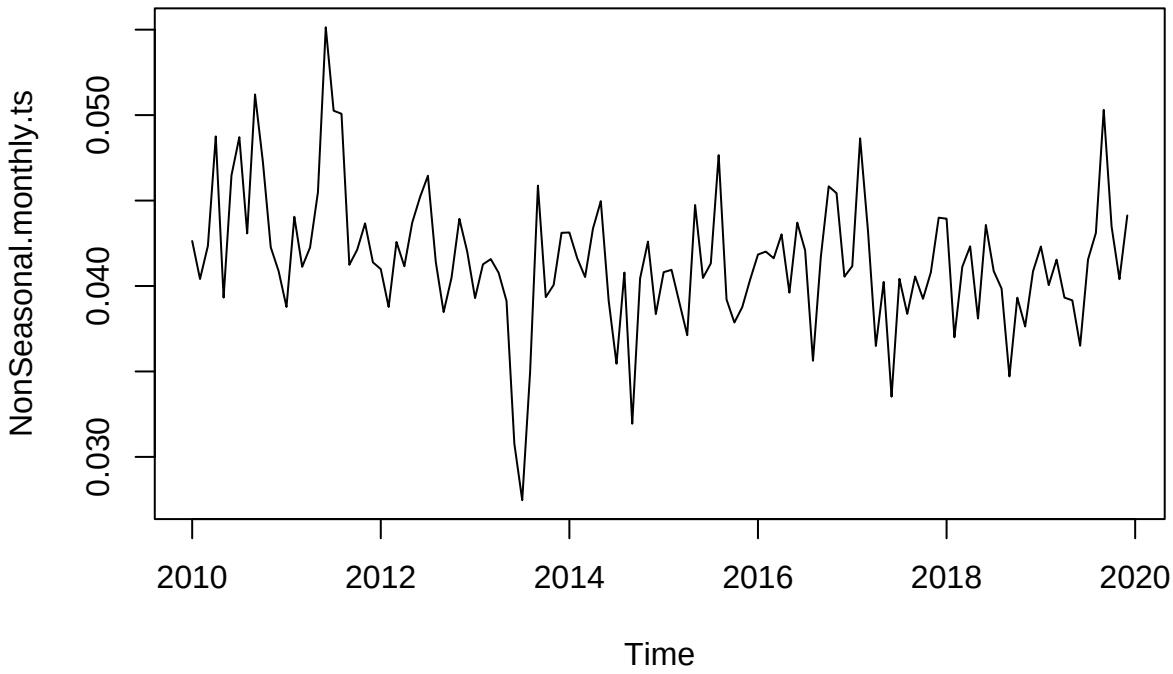
    Answer: Our research question is asking if ozone concentrations have changed over time at the Garinger station. In our preliminary graph we were able to spot a slight downward trend in the trendline we applied to the data. While this alone couldn't be used to determine that there in fact has been a decrease over time, the results of our Seasonal Mann Kendall test can be. The result of this shows that there is a significant difference over time in the mean ozone values, specifically a significant decrease over time as shown by the tau value provided in the results (tau = -0.143, 2-sided pvalue =0.046724).

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

# adding components into a dataframe, removing seasonal component
MonthlyOzone.Components <- as.data.frame(GaringerOzoneMonthly.decomp$time.series[,2:3])

# adding trend and remainder for analysis
NonSeasonalOzone.Monthly <- mutate(MonthlyOzone.Components,
                                   Seasonality_Removed = MonthlyOzone.Components$trend +
                                     MonthlyOzone.Components$remainder)

#16
```

```r
# creating non-seasonal time series
NonSeasonal.monthly.ts <- ts(NonSeasonalOzone.Monthly$Seasonality_Removed,
                     start = c(2010, 01, 01), frequency = 12)
plot(NonSeasonal.monthly.ts)
```




```r
# running Mann Kendall on time series with seasonality removed
MannKendall(NonSeasonal.monthly.ts)
```

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

Answer: After removing seasonality we still get a significant result with the Mann Kendall test that there is a change in the trend of ozone concentrations over time and this trend is downward due to the negative tau value (tau = -0.165, 2-sided pvalue =0.0075402). This negaitve trend exists regardless of the presence of seasonality in the data.