Assignment 7: Time Series Analysis

Julia Weinberg

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
getwd()
```

```
## [1] "/Users/juliaweinberg/Desktop/github repos/Environmental_Data_Analytics_2022/Assignments"
```

```
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   1.1.3     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(Kendall)
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(xts)
```

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
library(readr)

my.theme <- theme_gray(base_size = 14) + #set base theme and size
  theme(axis.text = element_text(color = "darkblue"),  #color text dark blue
        legend.position = "bottom") #align legend on bottom
theme_set(my.theme) #set 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
EPAair_2010 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2011 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2012 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2013 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2014 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2015 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2016 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2017 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv",
                        stringsAsFactors = TRUE)
EPAair_2018 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv",
                        stringsAsFactors = TRUE)
```

```
EPAair_2019 <- read.csv("../Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv",
                        stringsAsFactors = TRUE)


GaringerOzone <- list.files(path="../Data/Raw/Ozone_TimeSeries", full.names = TRUE) %>%
  lapply(read_csv) %>%
  bind_rows
```

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

# 4

GaringerOzone <- GaringerOzone%>%
  select(Date, `Daily Max 8-hour Ozone Concentration`, DAILY_AQI_VALUE)
#select data

# 5

Days <- as.data.frame(seq(as.Date("2010-01-01"), as.Date("2019-12-31"), by = "1 day"))
names(Days)[1] <- "Date" #create data frame and rename column


# 6
GaringerOzone <- left_join(Days, GaringerOzone) #join data frames
```

```
## Joining, by = "Date"
```

Visualize

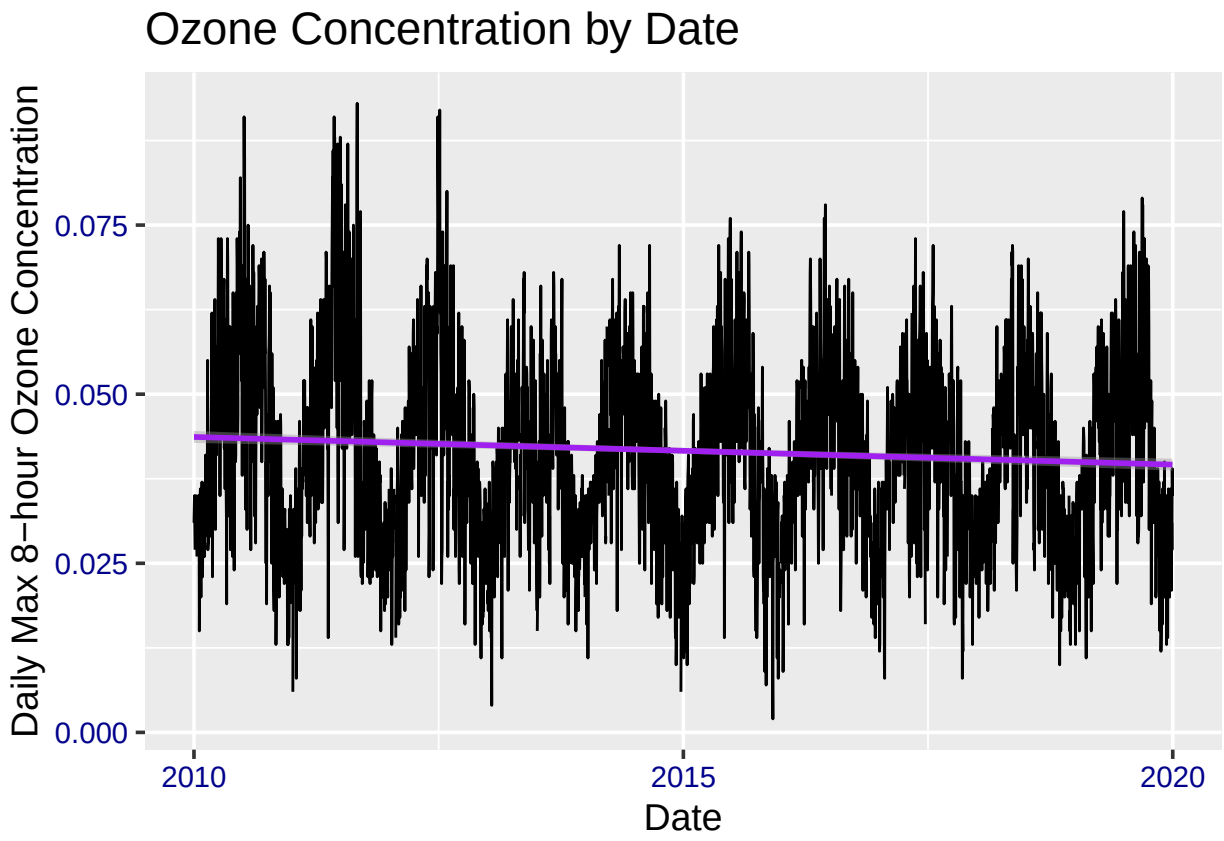
7. Create a line plot depicting ozone concentrations over time. In this case, we will plot actual concentrations in ppm, not AQI values. Format your axes accordingly. Add a smoothed line showing any linear trend of your data. Does your plot suggest a trend in ozone concentration over time?

```
#7
Ozone_time_plot <- ggplot(GaringerOzone, aes(x = Date, y = `Daily Max 8-hour Ozone Concentration`)) + #
  my.theme+ #include theme
  geom_line()+ #create line plot
```

```
  geom_smooth(method = "lm", color = "purple") + #add best fit line
  labs(title = "Ozone Concentration by Date", x = "Date", y = "Daily Max 8-hour Ozone Concentration")
print(Ozone_time_plot)
```

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

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



Answer: The plot shows a slight overall reduction in ozone concentration over time, however it has seasonal highs and lows.

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) #get summary
```

```
##       Date            Daily Max 8-hour Ozone Concentration DAILY_AQI_VALUE
##  Min.   :2010-01-01   Min.   :0.00200                      Min.   :  2.00
##  1st Qu.:2012-07-01   1st Qu.:0.03200                      1st Qu.: 30.00
##  Median :2014-12-31   Median :0.04100                      Median : 38.00
##  Mean   :2014-12-31   Mean   :0.04163                      Mean   : 41.57
##  3rd Qu.:2017-07-01   3rd Qu.:0.05100                      3rd Qu.: 47.00
##  Max.   :2019-12-31   Max.   :0.09300                      Max.   :169.00
##                       NA's   :63                           NA's   :63
```

```
GaringerOzone <- GaringerOzone %>%
  mutate(`Daily Max 8-hour Ozone Concentration` = zoo ::
  na.approx(`Daily Max 8-hour Ozone Concentration`)) #fill in NA's

summary(GaringerOzone) #get summary
```

```
##       Date             Daily Max 8-hour Ozone Concentration DAILY_AQI_VALUE
##  Min.   :2010-01-01   Min.   :0.00200                      Min.   :  2.00
##  1st Qu.:2012-07-01   1st Qu.:0.03200                      1st Qu.: 30.00
##  Median :2014-12-31   Median :0.04100                      Median : 38.00
##  Mean   :2014-12-31   Mean   :0.04151                      Mean   : 41.57
##  3rd Qu.:2017-07-01   3rd Qu.:0.05100                      3rd Qu.: 47.00
##  Max.   :2019-12-31   Max.   :0.09300                      Max.   :169.00
##                                                            NA's   :63
```

Answer: Since there were many occurances of NA's and seasonal fluctuations were at play, a linear interpolation works best to fill in the NAs.

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(Month = month(Date), Year = year(Date)) %>% #mutate columns
  mutate(Date = my(paste0(Month, "-", Year))) %>%
  dplyr::group_by(Date, Month, Year) %>% #group data
  dplyr::summarise(
    mean.ozone = mean(`Daily Max 8-hour Ozone Concentration`)) %>%
  select(mean.ozone, Date)
```

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

```
## Adding missing grouping variables: `Month`
```

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.

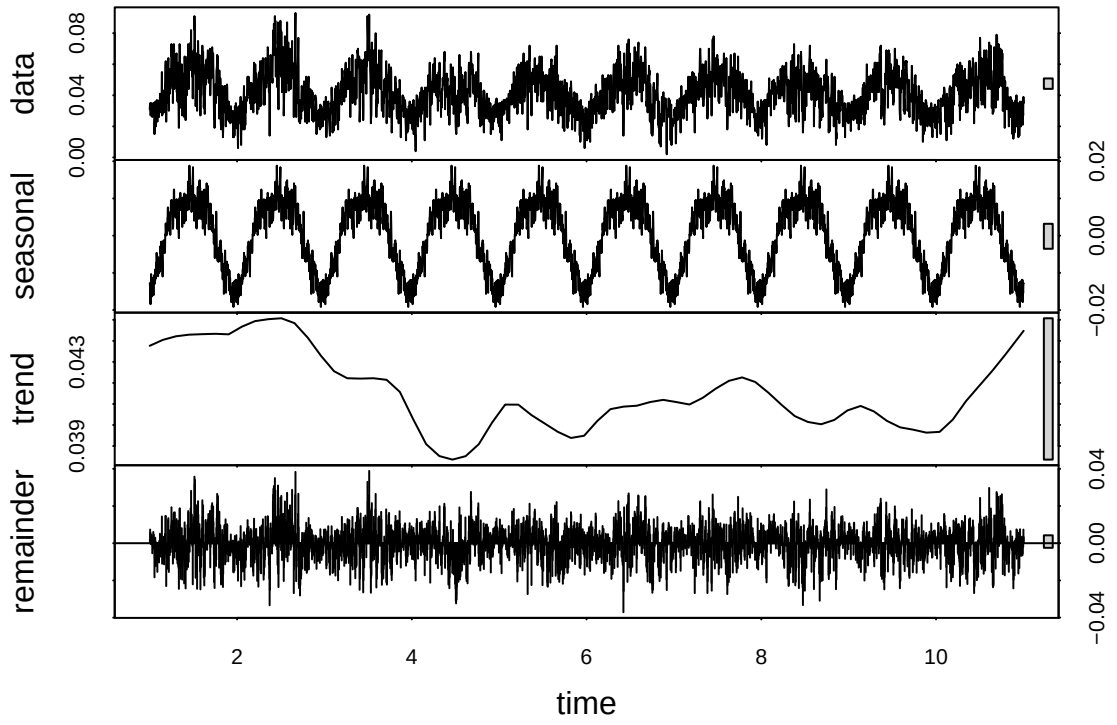
```
#10
f_day <- day(first(GaringerOzone$Date)) #set first day
f_month <- month(first(GaringerOzone$Date)) #set first month
f_year <- year(first(GaringerOzone$Date)) #set first year
GaringerOzone.daily.ts <-
ts(GaringerOzone$`Daily Max 8-hour Ozone Concentration`,
start = c(f_day, f_month, f_year), frequency = 365) #create daily time series

first_month <- month(first(GaringerOzone.Monthly$Date)) #set first month
first_year <- year(first(GaringerOzone.Monthly$Date)) #set first year
GaringerOzone.monthly.ts <- ts(GaringerOzone.Monthly$mean.ozone,
start = c(first_month,first_year),frequency = 12) #create monthly time series
```

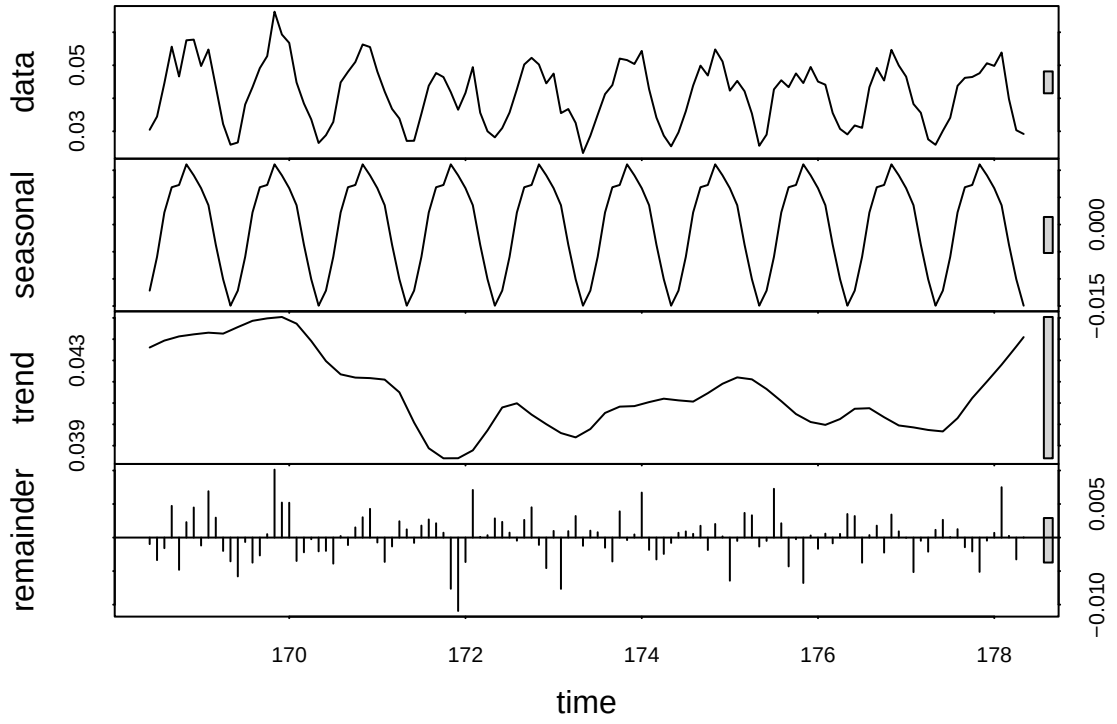
11. Decompose the daily and the monthly time series objects and plot the components using the `plot()` function.

```
#11

ozone_daily_decomp <- stl(GaringerOzone.daily.ts,s.window = "periodic")
plot(ozone_daily_decomp)
```




```
ozone_monthly_decomp <- stl(GaringerOzone.monthly.ts,s.window = "periodic")
plot(ozone_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_ozone_trend <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)

monthly_ozone_trend
```

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

```
summary(monthly_ozone_trend)
```

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

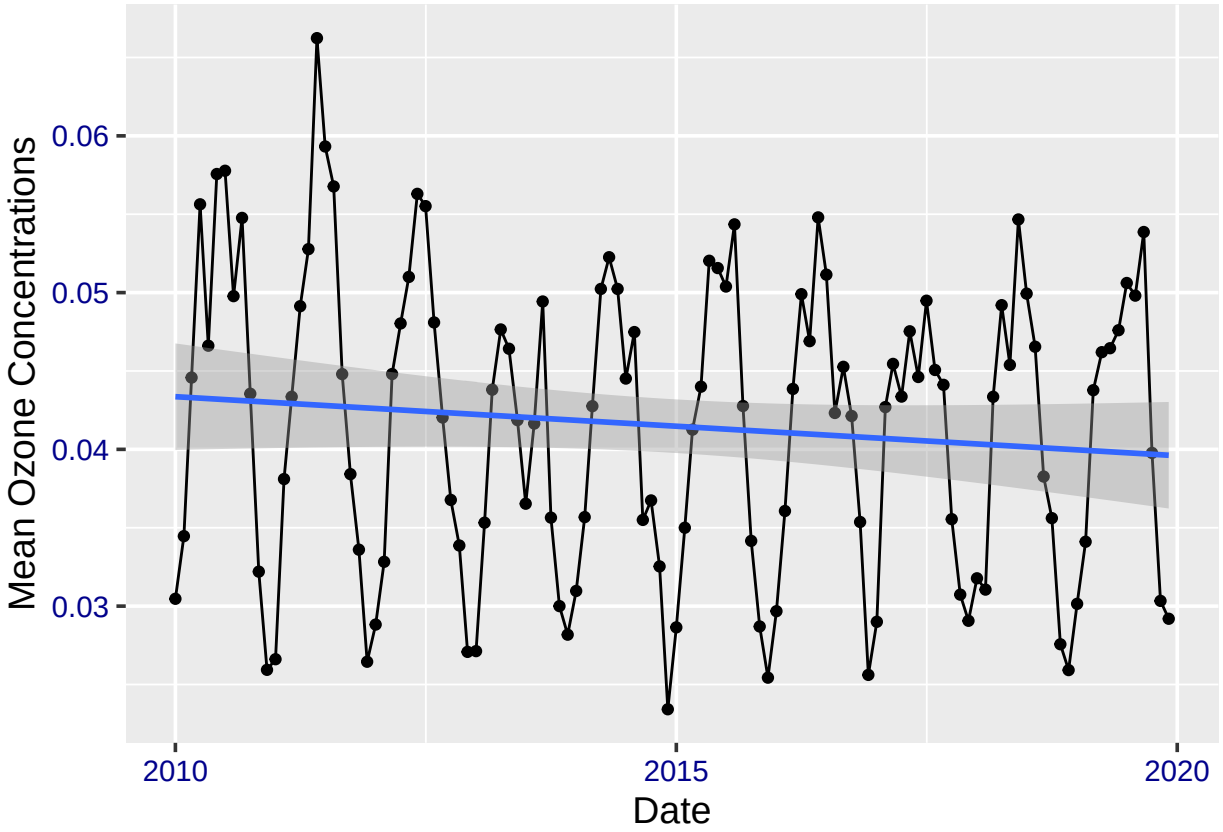
Answer: the Seasonal Mann Kendall is best since the results are grouped my month. The Seasonal Mann Kendall accounts for seasonality which is something to consider when analyzing by month.

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 <- ggplot(GaringerOzone.Monthly,
  aes(x = Date, y = mean.ozone)) + #plot ozone by time
  my.theme+ #include theme
  geom_line()+ #create line plot
  geom_point()+
  ylab("Mean Ozone Concentrations") +
  geom_smooth(method = lm)
print(monthly_ozone_plot)
```

```
## `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 graph shows seasonal fluctuations in ozone levels, however throughout the 10 year observed period, an overall trend of a reduction in ozone levels can be observed. Additionally, the Mann Kendall test returned a p-value of less than 0.05, which indicates a significant overall change in ozone levels.

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

GaringerOzone.Monthly.components <- mutate(GaringerOzone.Monthly.components,
  Observed = GaringerOzone.Monthly$mean.ozone,
  Date = GaringerOzone.Monthly$Date)

f_month2 <- month(first(GaringerOzone.Monthly.components$Date))
f_year2 <- year(first(GaringerOzone.Monthly.components$Date))
GaringerOzone.monthly.ts2 <- ts(GaringerOzone.Monthly.components$Observed, start = c(f_month2, f_year2)


#16
```

```
monthly_ozone_trend2 <- Kendall::MannKendall(GaringerOzone.monthly.ts2)

monthly_ozone_trend2
```

```
## tau = -0.0594, 2-sided pvalue =0.33732
```

```
summary(monthly_ozone_trend2)
```

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

```
summary(monthly_ozone_trend)
```

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

Answer: The seasonal Mann Kendall test has a lower p-value than the non-seasonal Mann Kendall test, and that of the non-seasonal test is not below 0.05. Therefore, no statistically significant interpretations can be made through the non-seasonal test.