Assignment 8: Time Series Analysis

Hanna Karnei

Fall 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 Load libraries and set theme

getwd()
```

```
## [1] "/Users/hannakarnei"
```

```
setwd('/Users/hannakarnei/Desktop/EDA/EDE_Fall2023/')
getwd()
```

```
## [1] "/Users/hannakarnei/Desktop/EDA/EDE_Fall2023"
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.0      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
library(zoo)
```

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

```r
library(trend)
library(here)
```

```
## here() starts at /Users/hannakarnei/Desktop/EDA/EDE_Fall2023
```

```r
mytheme <- theme_classic(base_size = 10) +
  theme(axis.text = element_text(color = "grey"),
        legend.position = "top")

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.

```r
#2 Import data

here()
```

```
## [1] "/Users/hannakarnei/Desktop/EDA/EDE_Fall2023"
```

```r
x2019<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv"),
                              stringsAsFactors = TRUE)

x2018<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv"),
                              stringsAsFactors = TRUE)

x2017<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv"),
                              stringsAsFactors = TRUE)

x2016<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv"),
                              stringsAsFactors = TRUE)

x2015<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv"),
                              stringsAsFactors = TRUE)

x2014<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv"),
                              stringsAsFactors = TRUE)

x2013<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv"),
                              stringsAsFactors = TRUE)

x2012<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv"),
                              stringsAsFactors = TRUE)

x2011<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv"),
                              stringsAsFactors = TRUE)

x2010<-read.csv(here("Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv"),
                              stringsAsFactors = TRUE)

GaringerOzone <-
  bind_rows(x2010, x2011, x2012, x2013, x2014, x2015, x2016, x2017, x2018, x2019)
```

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 Filter dataset

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

```
head(GaringerOzone)
```

```
##         Date Daily.Max.8.hour.Ozone.Concentration DAILY_AQI_VALUE
## 1 2010-01-01                                0.031              29
## 2 2010-01-02                                0.033              31
## 3 2010-01-03                                0.035              32
## 4 2010-01-04                                0.031              29
## 5 2010-01-05                                0.027              25
## 6 2010-01-07                                0.033              31
```

```
# 5 Create a new df and rename a column

Days <- as.data.frame(seq(as.Date("2010-01-01"),
                          as.Date("2019-12-31"),
                          "day"))

colnames(Days) <- "Date"

#6 Merge dfs

GaringerOzone <- Days %>%
  left_join(GaringerOzone, by = "Date")


head(GaringerOzone)
```

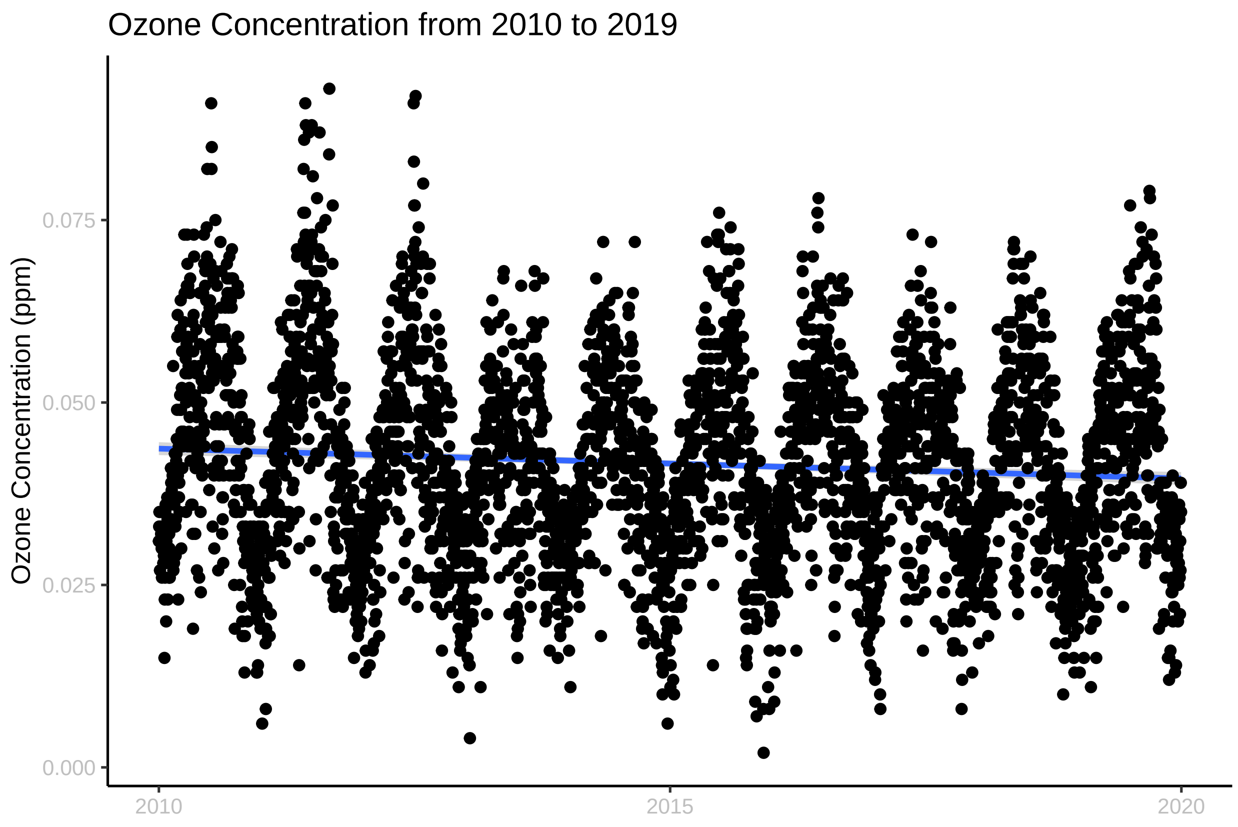
```
##         Date Daily.Max.8.hour.Ozone.Concentration DAILY_AQI_VALUE
## 1 2010-01-01                                0.031              29
## 2 2010-01-02                                0.033              31
## 3 2010-01-03                                0.035              32
## 4 2010-01-04                                0.031              29
## 5 2010-01-05                                0.027              25
## 6 2010-01-06                                   NA              NA
```

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 Plot ozone concentration

plot.ozone.concentration<-
  ggplot(GaringerOzone, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
  geom_smooth(method = "lm") +
  ggtitle('Ozone Concentration from 2010 to 2019') +
  ylab('Ozone Concentration (ppm)') +
  xlab (' ')+
  geom_point()

print(plot.ozone.concentration)
```

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

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

## Warning: Removed 63 rows containing missing values (geom_point).
```



Answer: There is a clear seasonal trend in the distribution of data. Values fluctuate within each year.

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?

```r
#8 Fill in N/As using linear interpolation

GaringerOzone$Daily.Max.8.hour.Ozone.Concentration<-
  zoo::na.approx(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration)

head(GaringerOzone)
```

```
##         Date Daily.Max.8.hour.Ozone.Concentration DAILY_AQI_VALUE
## 1 2010-01-01                                0.031              29
## 2 2010-01-02                                0.033              31
## 3 2010-01-03                                0.035              32
## 4 2010-01-04                                0.031              29
## 5 2010-01-05                                0.027              25
## 6 2010-01-06                                0.030              NA
```

Answer: Spline winterpolation would not work in this case because it uses a quadratic function to interpolate, whereas the current dataset uses a linear function. The piecewise constant would not work because it replaces "NAs" with a value nearest to the date, assuming that there is no upward or downward movementin the dataset. Looking at the ozone concentration dataset, we can clearly see a change in daily values, so the better approach is to use the average of the two nearest values.

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 Aggregate data by month

GaringerOzone.monthly <- GaringerOzone %>%
  mutate(
    Year = year(Date),
    Month = month(Date)
  ) %>%
  group_by(Year, Month) %>%
  summarise(
    MeanOzone = mean(Daily.Max.8.hour.Ozone.Concentration, na.rm = TRUE)
  )
```

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

```
GaringerOzone.monthly$NewDate <-
  as.Date(paste(GaringerOzone.monthly$Year, GaringerOzone.monthly$Month, "01", sep = "-"))

head(GaringerOzone.monthly)
```

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

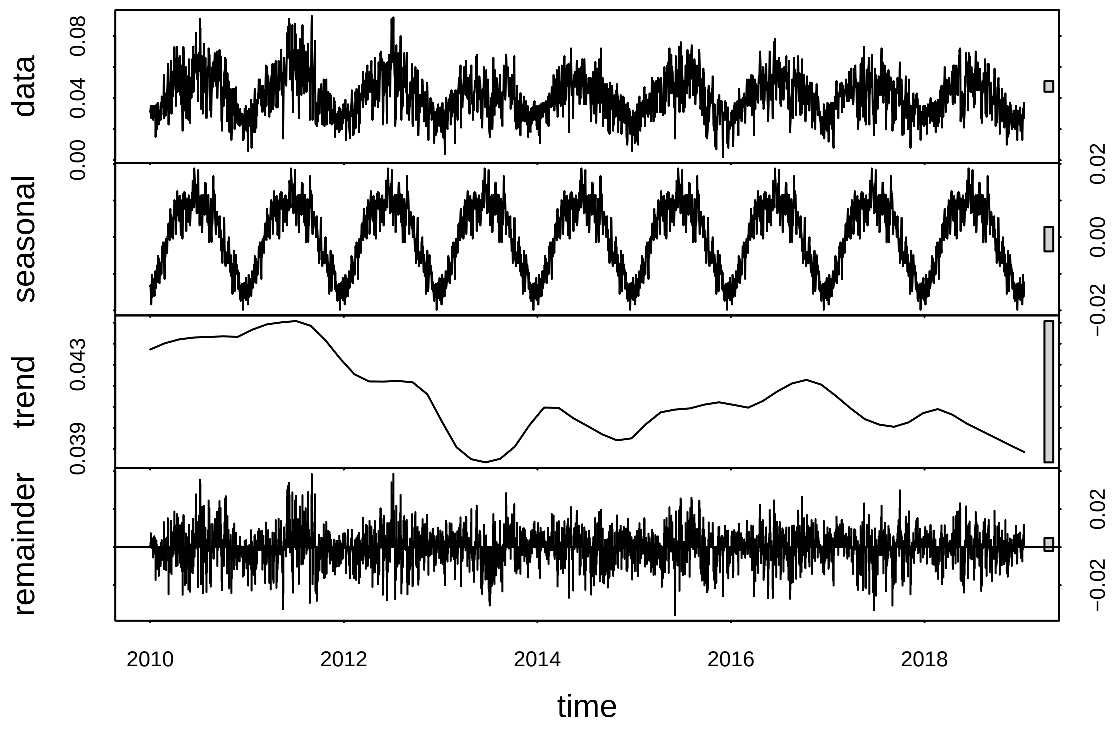
10. Generate two time series objects. Name the first `GaringerOzone.daily.ts` and base it on the dataframe of daily observations. Name the second `GaringerOzone.monthly.ts` and base it on the

monthly average ozone values. Be sure that each specifies the correct start and end dates and the frequency of the time series.

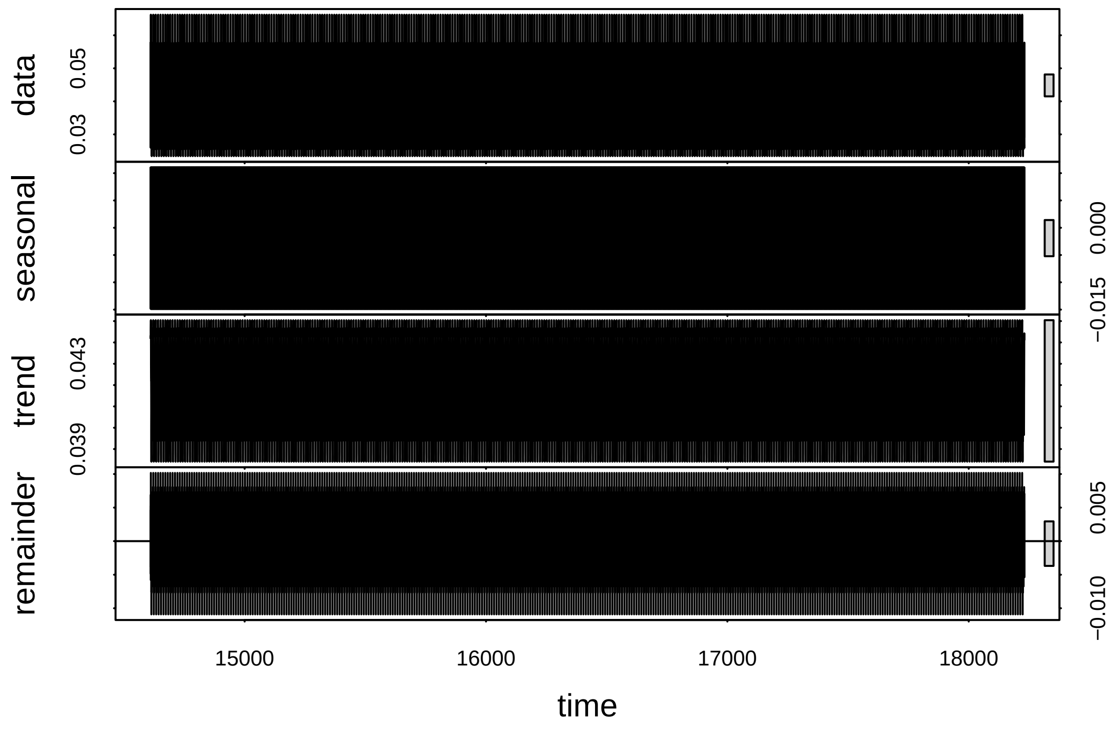
```
#10 Generate ts dataframes

GaringerOzone.daily.ts <- ts(
  GaringerOzone$Daily.Max.8.hour.Ozone.Concentration,
  start = c(year(min(GaringerOzone$Date)), month(min(GaringerOzone$Date))),
  end = c(year(max(GaringerOzone$Date)), month(max(GaringerOzone$Date))),
  frequency = 365
)

GaringerOzone.monthly.ts <- ts(
  GaringerOzone.monthly$MeanOzone,
  start = c(min(GaringerOzone.monthly$NewDate)),
  end = c(max(GaringerOzone.monthly$NewDate)),
  frequency = 12
)
```

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

```
#11 Decompose ts dfs and plot them

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



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

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

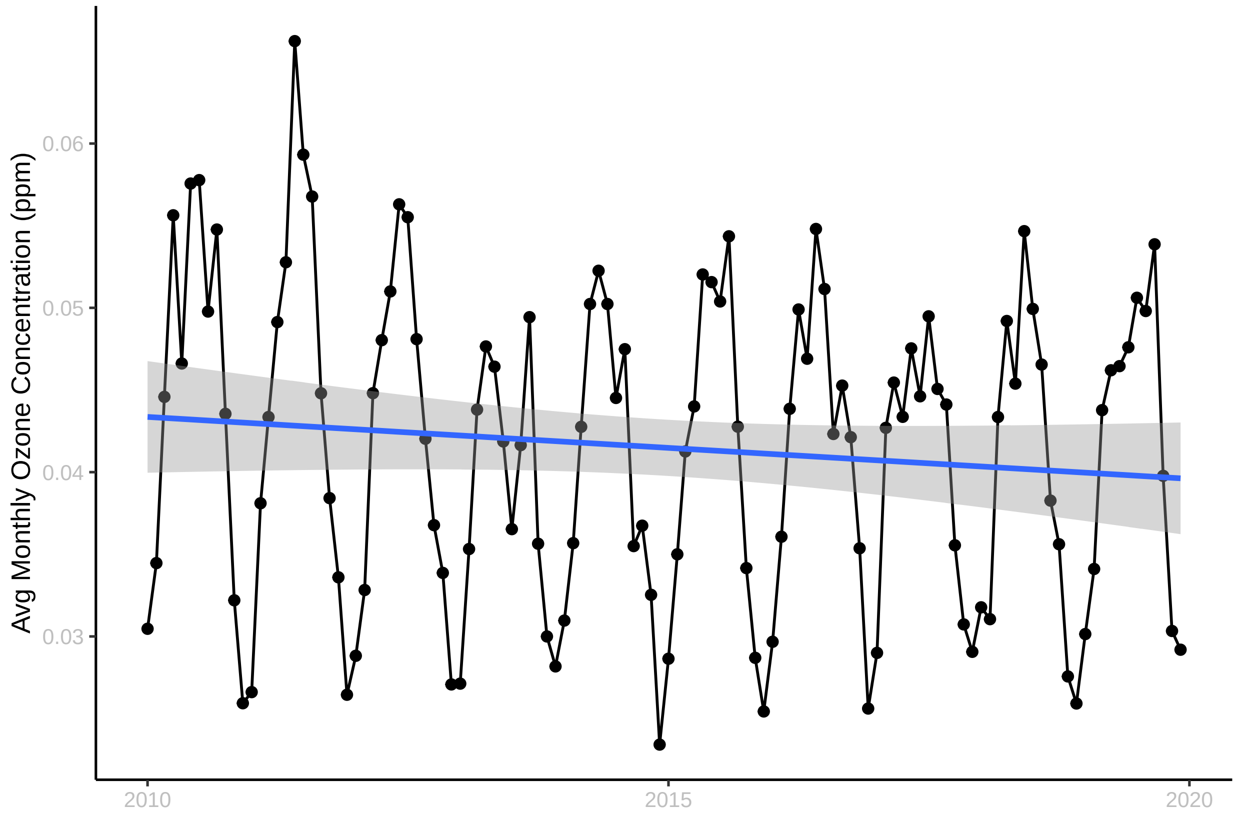
```
## Score =  -13757 , Var(Score) = 62666232320
## denominator =  74556452
## tau = -0.000185, 2-sided pvalue =0.95617
```

Answer: The seasonal Mann-Kendall is the most appropriate test because it is designed to analyze seasonal, non-parametric data. There is a clear seasonal trend in our data. All other tests we learned about could only be used for non-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 Create a plot depicting mean monthly ozone concentrations

monthly_ozone_plot <-
ggplot(GaringerOzone.monthly, aes(x = NewDate, y = MeanOzone)) +
  geom_point() +
  geom_line() +
  xlab ('') +
  ylab("Avg Monthly Ozone Concentration (ppm)") +
  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 negative score (-13757) and tau (-0.000185) indicate a decreasing trend in the time series data. However, the high p-value (0.95617) does not allow us to reject the null hypothesis. Therefore, we conclude that ozone concentrations have not changed over the 2010s at this station.

15. Subtract the seasonal component from the `GaringerOzone.monthly.ts`. Hint: Look at how we extracted the series components for the EnoDischarge in 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 Extract seasonal component from the time series

monthly_decomposed_components <- as.data.frame(monthly_decomposed$time.series[,1:3])
monthly_decomposed_deaseasoned <-
  GaringerOzone.monthly.ts-monthly_decomposed_components$seasonal

#16 Run the Mann Kendall test on the non-seasonal data

monthly_ozone_trend_noseason <- Kendall::MannKendall(monthly_decomposed_deaseasoned)
summary(monthly_ozone_trend_noseason)
```

```
## Score =  -242548 , Var(Score) = 9.11591e+12
## denominator =  940063168
## tau = -0.000258, 2-sided pvalue =0.93597
```

Answer: Based on the results of the Mann Kendall test, we are not able to reject the null hypothesis (p=0.93597). Hence the answer is no, the ozone concentrations have not changed over the 2010s at the Garinger station. The p-value of 0.95617 in the seasonal Mann Kendall test also led us to the same conclusion.