Assignment 8: Time Series Analysis

Sakina Shahid

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

```
getwd()
```

```
## [1] "/home/guest/EDE_Fall2023"
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

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

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

```
library(trend)
library(dplyr)
library(here)
```

```
## here() starts at /home/guest/EDE_Fall2023
```

```
theme <- theme_linedraw(base_size = 13) +
  theme(axis.text = element_text
    (color = "black"),plot.title = element_text(color="black",size=12),
    axis.title=element_text(size=10),
plot.background=element_rect(fill="lightyellow"),
panel.border=element_rect(linewidth = 1.5,linetype = "solid"),
axis.ticks= element_line(color="darkgrey",size=2),
panel.grid.major = element_line(size=0.1))
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
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.

```
#1

ozone_2010.df <- read.csv(here('Data','Raw',                'Ozone_TimeSeries',
'EPAair_O3_GaringerNC2010_raw.csv'),
    stringsAsFactors = TRUE)

ozone_2011.df <- read.csv(here('Data','Raw',                'Ozone_TimeSeries',
     'EPAair_O3_GaringerNC2011_raw.csv'),
        stringsAsFactors = TRUE)

ozone_2012.df <- read.csv(here('Data','Raw',                'Ozone_TimeSeries',
 'EPAair_O3_GaringerNC2012_raw.csv'),
```

```
    stringsAsFactors = TRUE)

ozone_2013.df <- read.csv(here('Data','Raw',                    'Ozone_TimeSeries',
 'EPAair_O3_GaringerNC2013_raw.csv'),
    stringsAsFactors = TRUE)

ozone_2014.df <- read.csv(here('Data','Raw',                    'Ozone_TimeSeries',
 'EPAair_O3_GaringerNC2014_raw.csv'),
  stringsAsFactors = TRUE)

ozone_2015.df <- read.csv(here('Data','Raw',                    'Ozone_TimeSeries',
 'EPAair_O3_GaringerNC2015_raw.csv'),
  stringsAsFactors = TRUE)

ozone_2016.df <- read.csv(here('Data','Raw',                    'Ozone_TimeSeries',
'EPAair_O3_GaringerNC2016_raw.csv'),
  stringsAsFactors = TRUE)

ozone_2017.df <- read.csv(here('Data','Raw',                    'Ozone_TimeSeries',
 'EPAair_O3_GaringerNC2017_raw.csv'),
  stringsAsFactors = TRUE)

ozone_2018.df <- read.csv(here('Data','Raw',                    'Ozone_TimeSeries',
   'EPAair_O3_GaringerNC2018_raw.csv'),
      stringsAsFactors = TRUE)

ozone_2019.df <- read.csv(here('Data','Raw',                    'Ozone_TimeSeries',
   'EPAair_O3_GaringerNC2019_raw.csv'),
         stringsAsFactors = TRUE)

GaringerOzone <- rbind(ozone_2010.df,
                       ozone_2011.df,ozone_2012.df,ozone_2013.df,ozone_2014.df,
                       ozone_2015.df,ozone_2016.df,
                       ozone_2017.df,ozone_2018.df,
                       ozone_2019.df)
```

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.

```r
# 3

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

```
## [1] "Date"
```

```r
# 4

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

# 5

Days <- data.frame(seq(x=GaringerOzone.columns$Date, as.Date("2010-01-01"),
as.Date("2019-12-31"), by = "day"))
colnames(Days)
```

```
## [1] "seq.x...GaringerOzone.columns.Date..as.Date..2010.01.01....as.Date..2019.12.31...."
```

```r
names(Days)[names(Days) == "seq.x...GaringerOzone.columns.Date..as.Date..2010.01.01....as.Date..2019.12

# 6

GaringerOzone <- merge(Days,
                       GaringerOzone.columns,
                       by = "Date", all.x = TRUE)
```

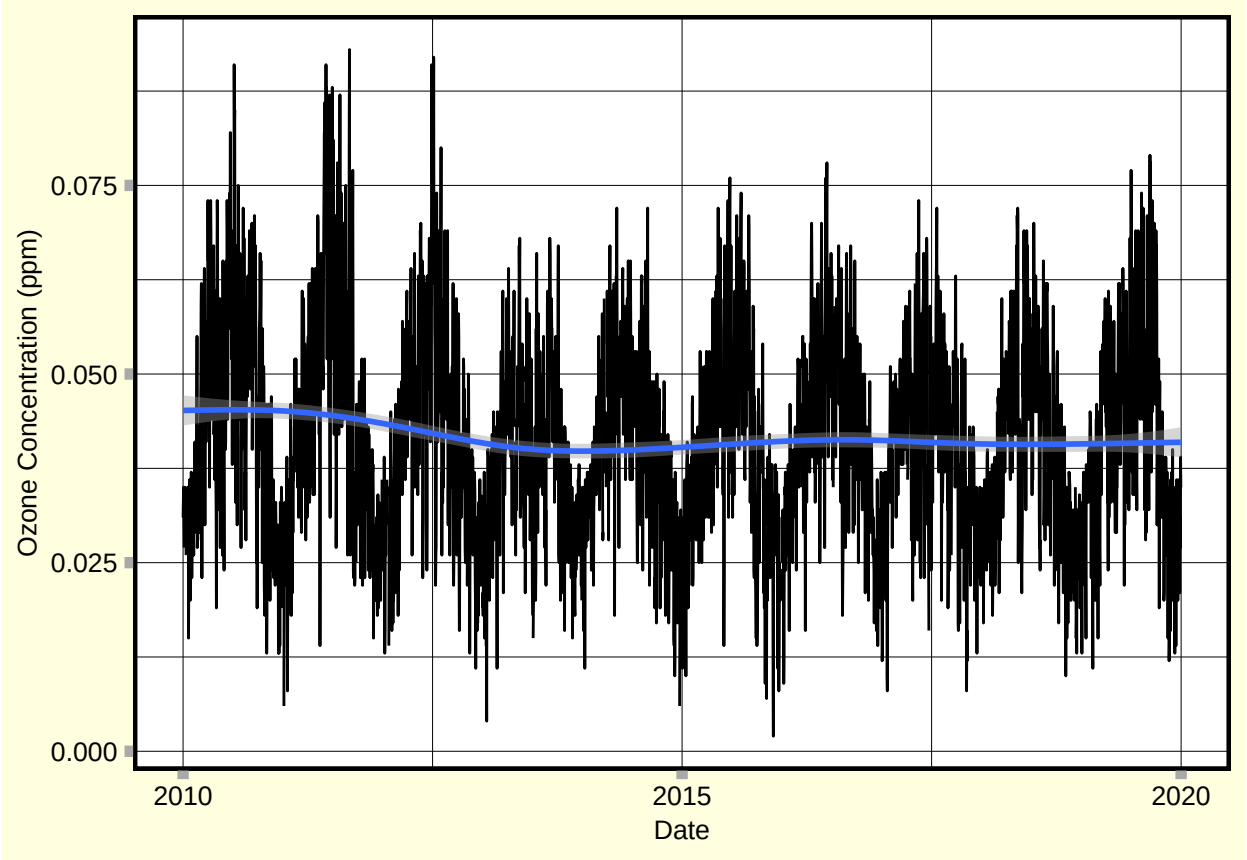
Visualize

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?

```r
#7

plot <- ggplot(GaringerOzone,
    aes(x=Date,y=Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line() +
  geom_smooth() + ylab("Ozone Concentration (ppm)")

print(plot)
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

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



Answer: Based on the plot above it is difficult to notice any trend in the ozone levels. There seems to be a dip in ozone concentrations in 2014 but this levels out over time.

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

Garinger_Ozone_clean <-
  GaringerOzone %>%
mutate( Daily.Max.8.hour.Ozone.Concentration.clean =
          zoo::na.approx
(Daily.Max.8.hour.Ozone.Concentration))

summary(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
## 0.00200 0.03200 0.04100 0.04163 0.05100 0.09300      63
```

```
summary(Garinger_Ozone_clean$Daily.Max.8.hour.Ozone.Concentration.clean)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00200 0.03200 0.04100 0.04151 0.05100 0.09300
```

Answer: We used a linear interpolation because this fits the trend of our data better. In the case of a piecewise function, it assumes that the missing points are the same as the data points above or below it instead of estimating an average value between the points. A spline interpolation is more appropriate for data that shows an exponential trend since the interpolation relies on the quadratic function.

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
# mean ozone concentrations for each month

GaringerOzone.date <- Garinger_Ozone_clean %>%
  mutate(Year = year(Date), Month = month(Date), Day = day(Date))  %>% group_by(Year,Month) %>%
  summarise(sum_ozone = sum(Daily.Max.8.hour.Ozone.Concentration.clean))
```

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

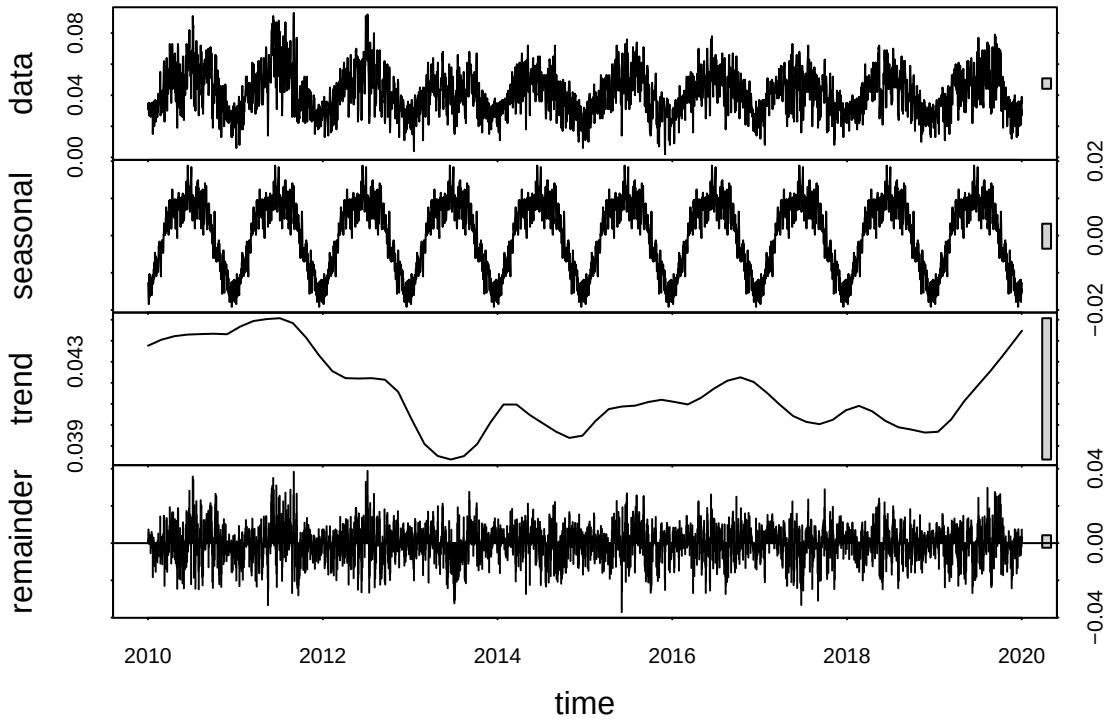
```
GaringerOzone.monthly <- GaringerOzone.date %>%
  mutate( Date = my(paste0(Month,"-",Year)))
```

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.

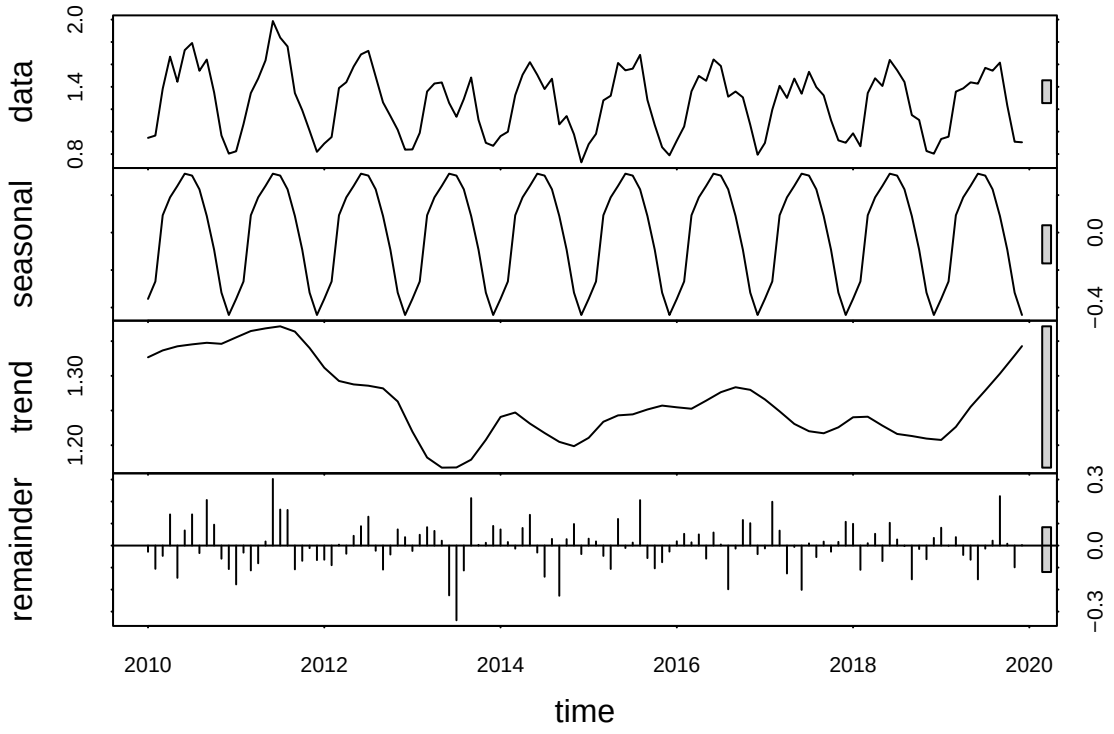
```
#10

f_month<- month(first(GaringerOzone.monthly$Date))
f_year<- year(first(GaringerOzone.monthly$Date))

GaringerOzone.daily.ts <-  ts(Garinger_Ozone_clean$Daily.Max.8.hour.Ozone.Concentration.clean,
                              frequency=365,start=c(f_year,f_month))

GaringerOzone.monthly.ts <-  ts(GaringerOzone.monthly$sum_ozone,
                              frequency=12,start=c(f_year,f_month))
```

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

```r
#11

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




```r
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_trend <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
monthly_trend
```

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

```
summary(monthly_trend)
```

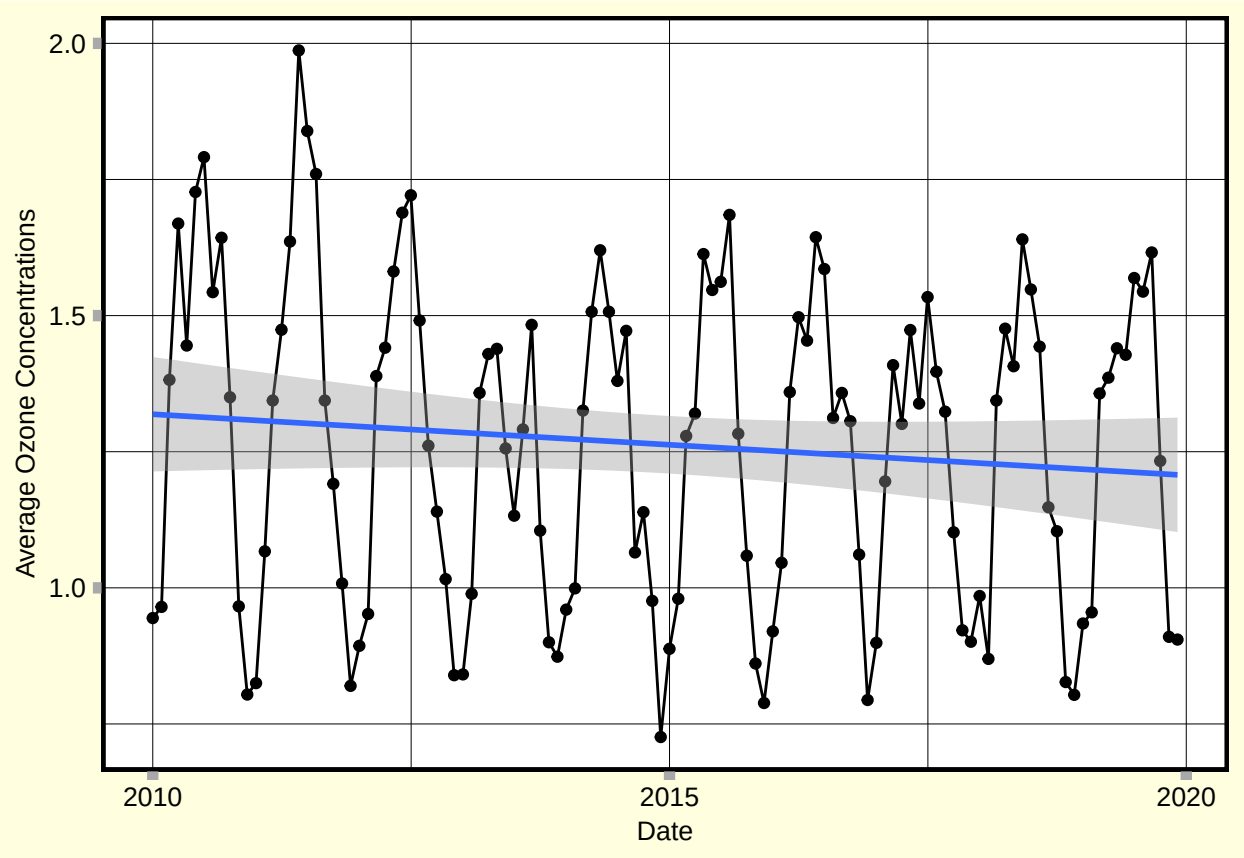
```
## Score =  -77 , Var(Score) = 1499
## denominator =  539.4972
## tau = -0.143, 2-sided pvalue =0.046724
```

Answer: A seasonal Man-Kendall is most appropriate because we are trying to understand the change in ozone levels over a long period of time. Using a seasonal function allows us to figure out whether there are daily or climate related fluctuations in ozone. The data also assumes a non parametric distribution and our data does not necessarily follow normal distribution.

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
ozone_monthly_plot <-
ggplot(GaringerOzone.monthly, aes(x = Date, y = sum_ozone)) +
  geom_point() +
  geom_line() +
  ylab("Average Ozone Concentrations") +
  geom_smooth( method = lm )
print(ozone_monthly_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: Based on the visual output, there is a slight decrease in Ozone levels from 2010 and 2020. On average to answer the question, it does ozone concentrations have changed in a statistically significant way. This is supported by the Man Kendall p value of 0.047 which is smaller than 0.05. This indicates that are able to reject our null hypothesis.

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.

```r
#15

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

monthly_decomposed_noseason <- GaringerOzone.monthly.ts-monthly_components$seasonal

#16

monthly_noseason <- Kendall::MannKendall(monthly_decomposed_noseason)
monthly_noseason
```

```
## tau = -0.165, 2-sided pvalue =0.0077466
```

Answer: When we remove the seasonal variation from the ozone trend, there is a more statistically significant result of p=0.0077. This indicates that when we include seasonality, there is a trend in ozone levels that is being hidden. This provides further evidence to support the decrease of ozone levels between 2010 and 2020.