Assignment 8: Time Series Analysis

Queenie Wei

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

```
#install.packages("vctrs")
library(vctrs)
#install.packages("vctrs", version = "0.6.3")
#library(vctrs)
#install.packages("tidyverse")
library(tidyverse)
library(lubridate)
#install.packages("trend")
library(trend)
#install.packages("zoo")
library(zoo)
#install.packages("Kendall")
library(Kendall)
#install.packages("tseries")
library(tseries)
library(here)
#install.packages("purrr")
library(purrr)
```

```
library(ggplot2)
mytheme <- theme_classic(base_size = 15) +
  theme(axis.text = element_text(color = "black"),
        legend.position = "top")
theme_set(mytheme)
getwd()
```

```
## [1] "C:/Users/ziyaw/Downloads/EDE_Fall2023"
```

```
getwd()
```

```
## [1] "C:/Users/ziyaw/Downloads/EDE_Fall2023"
```

```
here()
```

```
## [1] "C:/Users/ziyaw/Downloads/EDE_Fall2023"
```

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
# O32010 <- read.csv("./EPAair_O3_GaringerNC2010_raw.csv",
#                            stringsAsFactors = TRUE)
# O32011 <- read.csv("./EPAair_O3_GaringerNC2011_raw.csv",
#                            stringsAsFactors = TRUE)
# O32012 <- read.csv("./EPAair_O3_GaringerNC2012_raw.csv",
#                            stringsAsFactors = TRUE)
# O32013 <- read.csv("./EPAair_O3_GaringerNC2013_raw.csv",
#                            stringsAsFactors = TRUE)
# O32014 <- read.csv("./EPAair_O3_GaringerNC2014_raw.csv",
#                            stringsAsFactors = TRUE)
# O32015 <- read.csv("./EPAair_O3_GaringerNC2015_raw.csv",
#                            stringsAsFactors = TRUE)
# O32016 <- read.csv("./EPAair_O3_GaringerNC2016_raw.csv",
#                            stringsAsFactors = TRUE)
# O32017 <- read.csv("./EPAair_O3_GaringerNC2017_raw.csv",
#                            stringsAsFactors = TRUE)
# O32018 <- read.csv("./EPAair_O3_GaringerNC2018_raw.csv",
#                            stringsAsFactors = TRUE)
# O32019 <- read.csv("./EPAair_O3_GaringerNC2019_raw.csv",
#                            stringsAsFactors = TRUE)
# O3Garinger <- rbind(O32010,O32011,O32012,O32013,O32014,O32015,O32016,O32017, O32018,O32019)
#Create an empty list to store the data frames
data_frames_list <- list()

# # Define a vector of years for the loop
 years <- 2010:2019
#
# # Loop through each year and read the corresponding CSV file
```

```
 for (year in years) {
   file_path <- paste("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC", year, "_raw.csv", sep = "")
   data_frame <- read.csv(file_path, stringsAsFactors = TRUE)
   data_frames_list[[as.character(year)]] <- data_frame
 }

# Combine all data frames into a single data frame
 GaringerOzone <- do.call(rbind, data_frames_list)
####works cited: chatgpt. Prompt: I fed it my huge chunk of code and asked it
###"how do i simplify this with a for loop"
```

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 the date column to date data type
GaringerOzone$Date <- as.Date(GaringerOzone$Date, format = "%m/%d/%Y")

# 4
#select only the three columns
GaringerOzone <- dplyr::select(GaringerOzone, Daily.Max.8.hour.Ozone.Concentration, Date, DAILY_AQI_VALU

# 5
Days <- as.data.frame(seq(as.Date("2010-01-01"), as.Date("2019-12-31"), by = 1))
summary(Days)
```

```
##  seq(as.Date("2010-01-01"), as.Date("2019-12-31"), by = 1)
##  Min.   :2010-01-01
##  1st Qu.:2012-07-01
##  Median :2014-12-31
##  Mean   :2014-12-31
##  3rd Qu.:2017-07-01
##  Max.   :2019-12-31
```

```
colnames(Days)[1] <- "Date"

# 6
GaringerOzone <- dplyr ::left_join(Days,GaringerOzone)
```

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

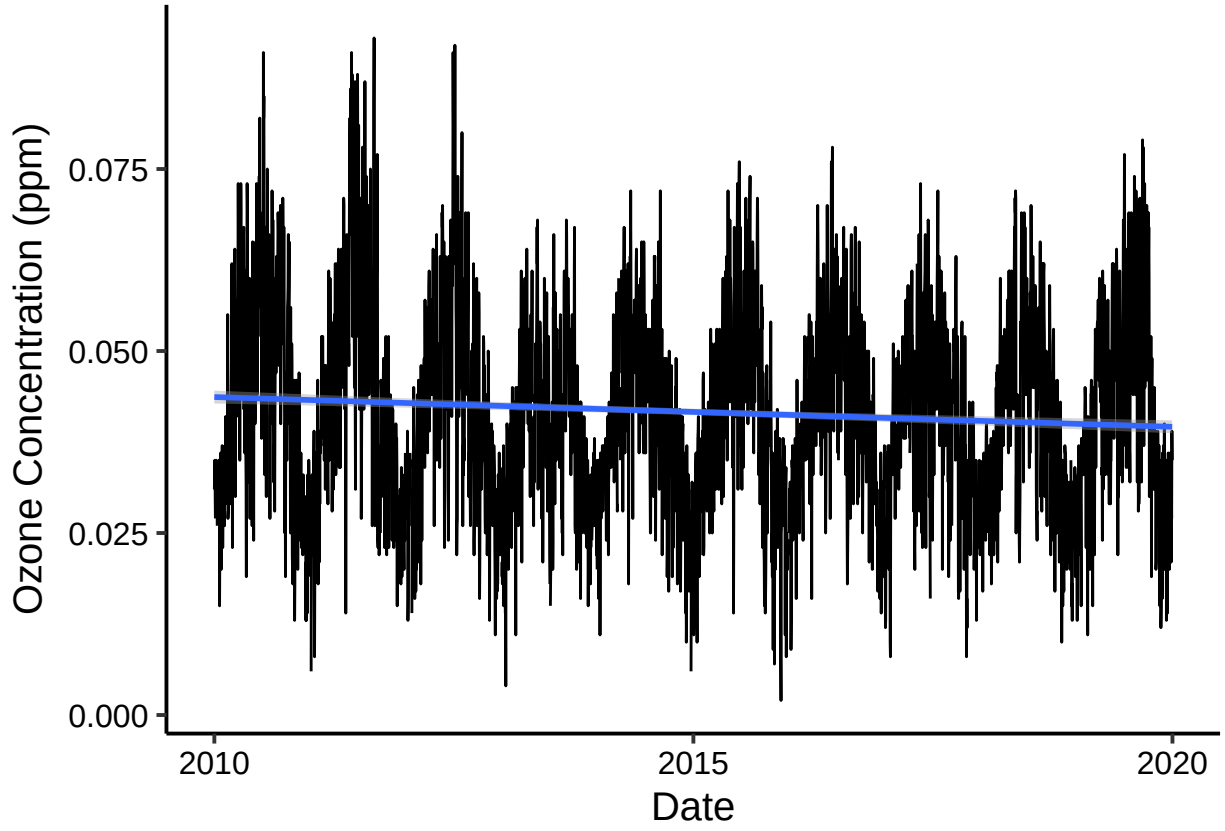
Visualize

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(GaringerOzone) +
  geom_line(aes(y = Daily.Max.8.hour.Ozone.Concentration, x = Date)) +
  geom_smooth(aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration), method = lm) +
  ylab(expression("Ozone Concentration (ppm)"))
```

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

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




```
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
```

Answer: The lm function has a negative slope, indicating that there is a negative correlation between the two variables (Ozone is decreasing 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



f_month <- lubridate::month(dplyr::first(GaringerOzone$Date))
f_year <- lubridate::year(dplyr::first(GaringerOzone$Date))
GaringerOzone$Daily.Max.8.hour.Ozone.Concentration <- zoo::na.approx(GaringerOzone$Daily.Max.8.hour.Ozo
#any(is.na(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration))
```

Answer: Why didn't we use a piecewise constant or spline interpolation? Spline interpolation does not make sense to be used in this case because there is no reason for quadratic function to be involved in the interpolation (the trends did not warrant this). Splines also tend to obsecure or change the trend of the original data, which is not ideal for this set of data. Piecewise constant does not capture the overall trend of the data. It simply fills in the gap with the nearest data, which might obscure the trend.

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
library(dplyr)
library(lubridate)
# Create the monthly ozone data frame

GaringerOzone.monthly <- GaringerOzone %>%
  mutate(Year = year(Date), ## adding the year column
         Month = month(Date)) %>%  ###adding the month column
  group_by(Year, Month) %>% #grouping the data by year and month
  summarize(Ozone_mean = mean(Daily.Max.8.hour.Ozone.Concentration))
```

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

```
GaringerOzone.monthly$Date <- as.Date(paste(GaringerOzone.monthly$Year, GaringerOzone.monthly$Month, "0
###Works cited: Bard. Prompt: the prompt for this question. Bard cited
###"Sources: github.com/ENV872/EDA-Fall2022".... Below I will try to recreate
#that code without Bard's help since they basically used past course material.
GaringerOzone.monthly2 <- GaringerOzone
GaringerOzone.monthly2$Year <- year(GaringerOzone.monthly2$Date)
GaringerOzone.monthly2$Month <- month(GaringerOzone.monthly2$Date)
GaringerOzone.monthly2 <- GaringerOzone.monthly2 %>%
  group_by(Year, Month) %>%
  summarize(Ozone_mean = mean(Daily.Max.8.hour.Ozone.Concentration))
```

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

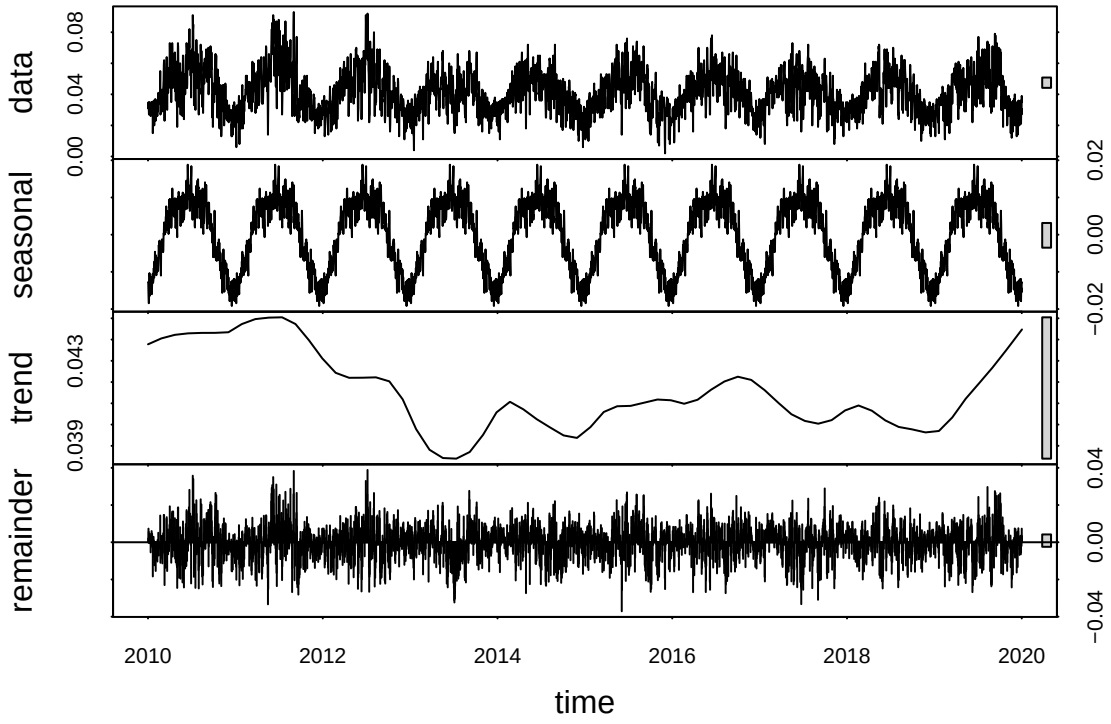
```
GaringerOzone.monthly2$Date <- lubridate::ym(paste(GaringerOzone.monthly$Year, GaringerOzone.monthly$Mo
```

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.

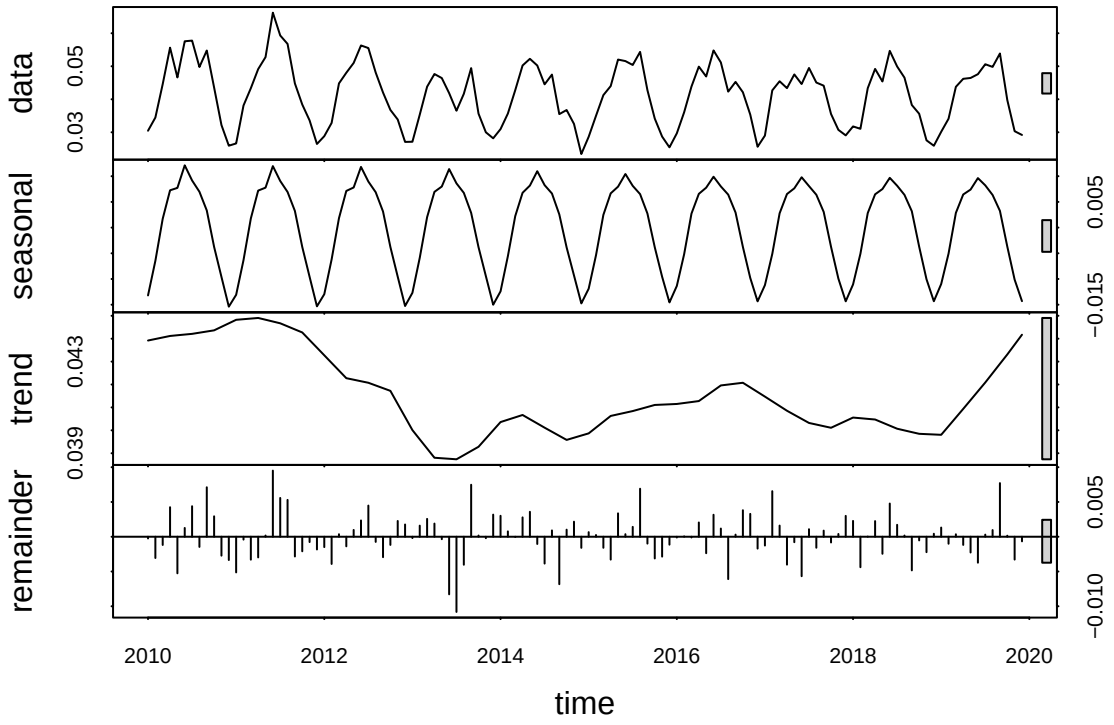
```
#10
f_month <- lubridate::month(dplyr::first(GaringerOzone$Date))
f_year <- lubridate::year(dplyr::first(GaringerOzone$Date))
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$Ozone_mean,
                   start=c(f_year,f_month, 01),
                   frequency=12)
GaringerOzone.daily.ts <- ts(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration,
                   start=c(f_year,f_month, 01),
                   frequency=365)
```

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

```
#11
daily_decomp <- stl(GaringerOzone.daily.ts,s.window = 365)
plot(daily_decomp)
```



```
monthly_decomp <- stl(GaringerOzone.monthly.ts,s.window = 12)
plot(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


# Run SMK test
monthly_trend <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
```

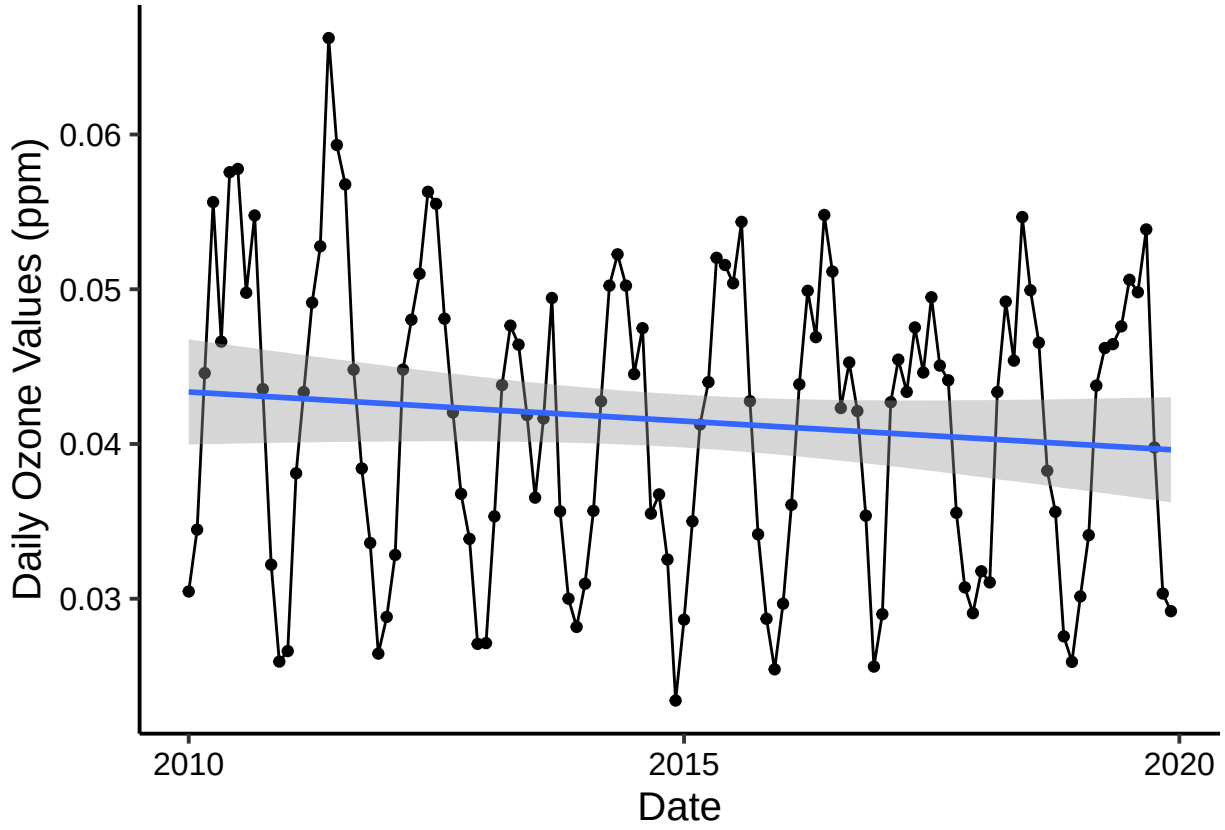
Answer: The seaonal Mann-Kendall is the most appropriate because it is the only trend analysis that accounts for seasonality, which we have in this graph.

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_plot <-
ggplot(GaringerOzone.monthly, aes(x = Date, y = Ozone_mean)) +
  geom_point() +
  geom_line() +
```

```
  ylab("Daily Ozone Values (ppm)") +
  geom_smooth(method = lm ) +
  mytheme
print(monthly_plot)
```

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




```
summary(monthly_trend)
```

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

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 score is negative (-77), which means that there is a decreasing trend – the O3 levels are decreasing over time. However, the small tau (-0.143)/close to zero indicates taht there is a very weak association between the two variables.On the other hand, the p value of 0.0467 indicates that there is a close to significant trend. Overall, the data suggests that Ozone has decreased at this station over the 2010s.

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
montly.dec <- decompose(GaringerOzone.monthly.ts)
seasonal <- montly.dec$seasonal
no_seasonal.ts <- GaringerOzone.monthly.ts - seasonal


#16

no_seasonal.trend <- Kendall::MannKendall(no_seasonal.ts)
summary(no_seasonal.trend)
```

```
## Score =  -1149 , Var(Score) = 194365.7
## denominator =  7139.5
## tau = -0.161, 2-sided pvalue =0.0092157
```

```
# > summary(monthly_trend)
# Score =  -77 , Var(Score) = 1499
# denominator =  539.4972
# tau = -0.143, 2-sided pvalue =0.046724
# > no_seasonal.trend <- Kendall::MannKendall(no_seasonal.ts)
# > summary(no_seasonal.trend)
# Score =  -1149 , Var(Score) = 194365.7
# denominator =  7139.5
# tau = -0.161, 2-sided pvalue =0.0092157
```

Answer: The Score is much higher for the latter(-1149 as opposed to -77), indicating that the trend is a lot stronger without the seasonal component. However, the Var(Score) is a lot larger(194365.7 as opposed to 1499) , which means that there is a higher uncertainty if seasonal trends are not accounted for. The p value is smaller for the latter, indicating a stronger significance for the trend analysis. Overall, there is a stronger correlation/significance shown in the latter case but with higher uncertainty.