Assignment 8: Time Series Analysis

Chrissie Pantoja

Fall 2024

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

```
# Check your current working directory
getwd()

# Load necessary libraries
library(tidyverse)
library(lubridate)
library(zoo)
library(trend)

# Create a custom ggplot theme
custom_theme <- theme_minimal() +
  theme(
    text = element_text(family = "Arial", size = 12, color = "black"),
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, size = 14),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text = element_text(size = 12),
```

```
    legend.position = "bottom",
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 10),
    panel.grid.major = element_line(color = "grey80"),
    panel.grid.minor = element_line(color = "grey90"),
    panel.border = element_blank(),
    plot.background = element_rect(fill = "white", color = NA)
  )

# Set the custom theme as the default
theme_set(custom_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.

```
# Set the directory path for the folder containing the data files
folder_path <- "Data/Raw/Ozone_TimeSeries"

# List all CSV files in the directory
file_list <- list.files(path = folder_path, pattern = "*.csv", full.names = TRUE)

# Load and combine all files into one dataframe
GaringerOzone <- do.call(rbind, lapply(file_list, read.csv))

# Display the dimensions of the dataset
cat("Dimensions of GaringerOzone: ", dim(GaringerOzone)[1], " rows and ", dim(GaringerOzone)[2], " colu
```

```
## Dimensions of GaringerOzone:  3589  rows and  20  columns
```

Wrangle

3. Set your date column as a date class.

```
# Convert the Date column to Date class
GaringerOzone$Date <- as.Date(GaringerOzone$Date, format = "%m/%d/%Y")

# Verify the change
class(GaringerOzone$Date)
```

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

4. Wrangle your dataset so that it only contains the columns Date, Daily.Max.8.hour.Ozone.Concentration, and DAILY_AQI_VALUE.

```
# Select the specified columns
OzoneData <- GaringerOzone %>%
  select(Date, Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)

# Verify the result
str(OzoneData)
```

```
## 'data.frame':    3589 obs. of  3 variables:
##  $ Date                                : Date, format: "2010-01-01" "2010-01-02" ...
##  $ Daily.Max.8.hour.Ozone.Concentration: num  0.031 0.033 0.035 0.031 0.027 0.033 0.035 0.032 0.032 0
##  $ DAILY_AQI_VALUE                     : int  29 31 32 29 25 31 32 30 30 28 ...
```

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".

```
# Create a sequence of dates from 2010-01-01 to 2019-12-31
Days <- as.data.frame(seq(from = as.Date("2010-01-01"),
                          to = as.Date("2019-12-31"),
                          by = "day"))

# Rename the column to "Date"
colnames(Days) <- "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.

```
# Combine Days and OzoneData with a left join
GaringerOzone <- left_join(Days, OzoneData, by = "Date")

# Display the dimensions of the dataset
cat("Dimensions of GaringerOzone: ", dim(GaringerOzone)[1], " rows and ", dim(GaringerOzone)[2], " colu
```

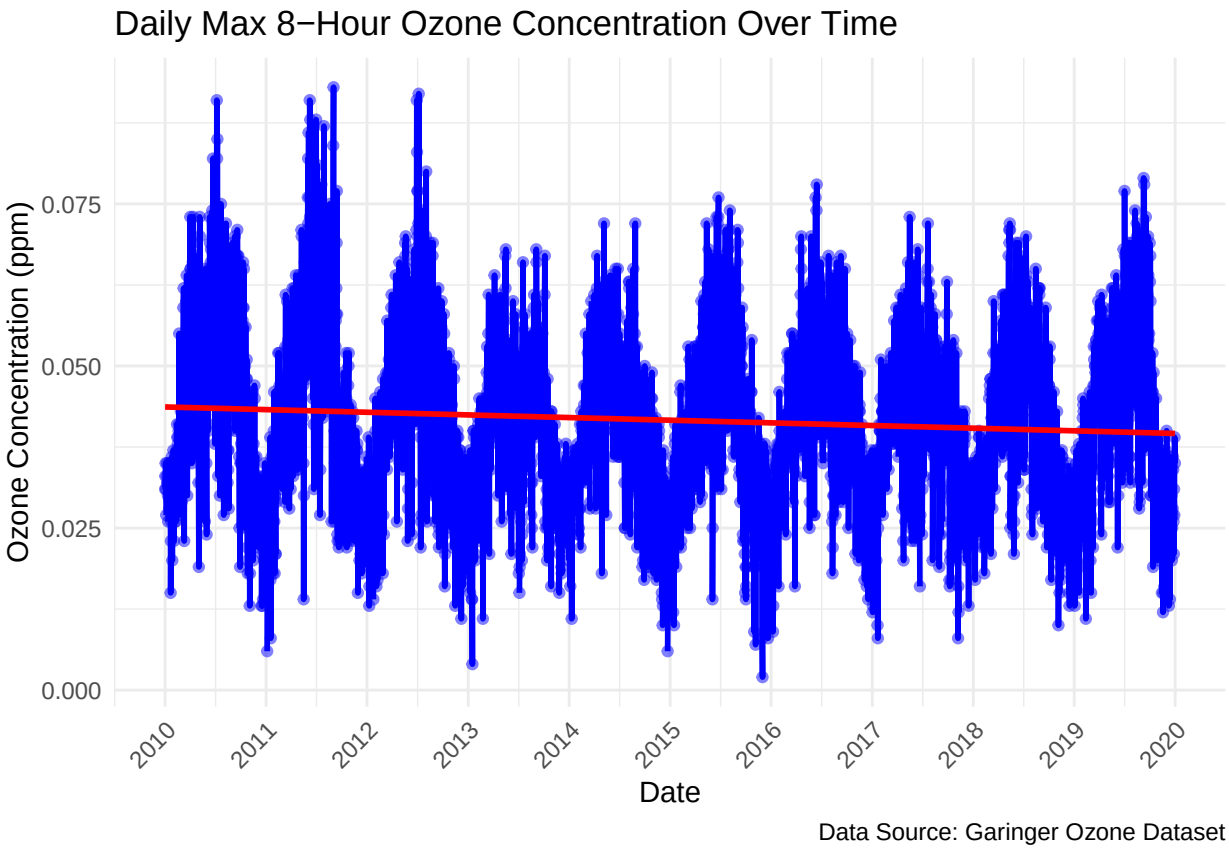
```
## Dimensions of GaringerOzone:  3652  rows and  3  columns
```

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?

```
# Create a line plot of ozone concentrations over time
ggplot(data = GaringerOzone, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line(color = "blue", size = 1) +  # Line for actual concentrations
  geom_point(color = "blue", alpha = 0.5) +  # Points for actual concentrations
  geom_smooth(method = "lm", color = "red", se = FALSE) +  # Smoothed line showing trend
  labs(title = "Daily Max 8-Hour Ozone Concentration Over Time",
       x = "Date",
       y = "Ozone Concentration (ppm)",
       caption = "Data Source: Garinger Ozone Dataset") +
  theme_minimal() +  # Minimal theme for better aesthetics
  scale_x_date(date_breaks = "1 year", date_labels = "%Y") +  # Format x-axis
  theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x-axis labels for readability
```

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



Answer: The plot indicates a slight downward trend in daily maximum 8-hour ozone concentrations over the given period. However, the trend is subtle and there's significant day-to-day variability. While the overall direction suggests a possible decrease in ozone levels, further analysis is needed to confirm the strength and causes of this trend, considering factors like air pollution regulations, meteorological conditions, and natural fluctuations in ozone concentrations.

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?

```
#Check if there is any missing value
sum(is.na(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration))

# Apply linear interpolation to fill missing values
GaringerOzone.clean <- GaringerOzone %>%
  mutate(Daily.Max.8.hour.Ozone.Concentration.clean = zoo::na.approx(Daily.Max.8.hour.Ozone.Concentrati
  mutate(DAILY_AQI_VALUE.clean = zoo::na.approx(DAILY_AQI_VALUE))

# Verify that the missing values are filled
summary(GaringerOzone.clean)
```

Answer: Linear interpolation was chosen to fill missing daily ozone concentration data because it smoothly bridges gaps with minimal complexity, making it well-suited for time series data that typically changes gradually from day to day. This approach maintains a straightforward trend without introducing artificial fluctuations, as would happen with more complex methods like spline interpolation, which may create unnecessary peaks and valleys. Additionally, piecewise constant interpolation was avoided because it holds values constant, which can misrepresent gradual daily changes. Thus, linear interpolation provides a balanced, accurate solution for filling short gaps in the data.

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)

```
# Step 1: Add columns for year and month, then calculate monthly mean ozone concentrations
GaringerOzone.monthly <- GaringerOzone.clean %>%
  mutate(
    year = year(Date),          # Extract year from Date column
    month = month(Date)          # Extract month from Date column
  ) %>%
  group_by(year, month) %>%
  summarize(
    mean_ozone = mean(Daily.Max.8.hour.Ozone.Concentration.clean, na.rm = TRUE)
  ) %>%
  ungroup()

# Step 2: Create a new Date column for graphing, setting each month-year combination to the first day o
GaringerOzone.monthly <- GaringerOzone.monthly %>%
  mutate(
    Date = as.Date(paste(year, month, "01", sep = "-"))
  )
```

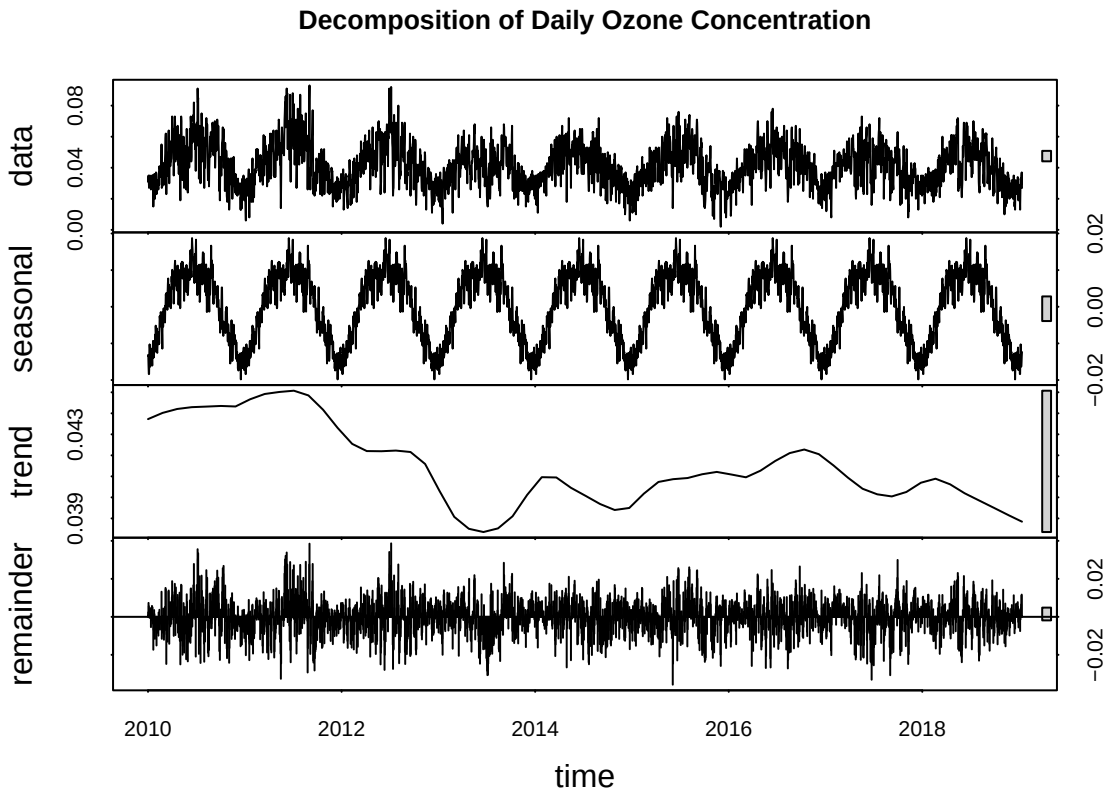
10. Generate two time series objects. Name the first `GaringerOzone.daily.ts` and base it on the dataframe of daily observations. Name the second `GaringerOzone.monthly.ts` and base it on the monthly average ozone values. Be sure that each specifies the correct start and end dates and the frequency of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

```
library(dplyr)

# Extract the starting and ending years and months for monthly time series
s_year <- year(first(GaringerOzone$Date))
f_year <- year(last(GaringerOzone$Date))
s_month <- month(first(GaringerOzone$Date))
f_month <- month(last(GaringerOzone$Date))
s_day <- yday(first(GaringerOzone$Date))
f_day <- yday(last(GaringerOzone$Date))

# 1. Generate the daily time series object
GaringerOzone.daily.ts <- ts(
  GaringerOzone.clean$Daily.Max.8.hour.Ozone.Concentration.clean,
  start = c(s_year, s_month),
  end = c(f_year, f_month),
  frequency = 365
```

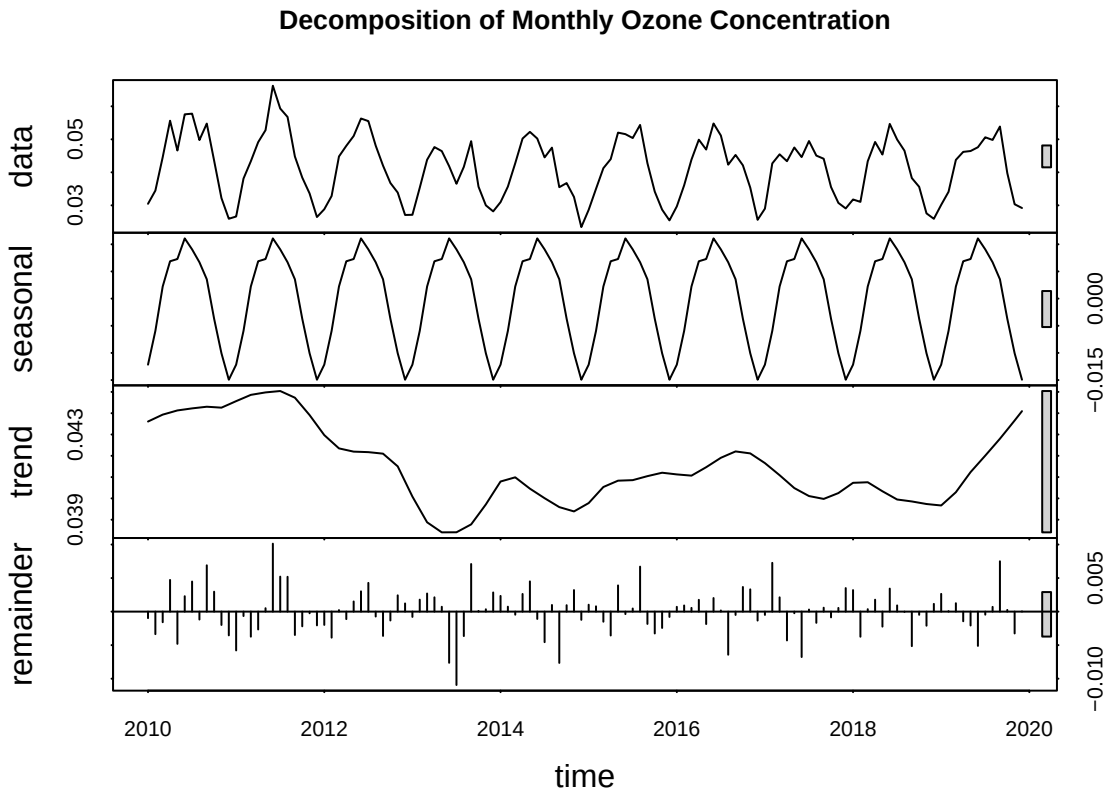
```
)

# 2. Generate the monthly time series object using aggregated monthly data
GaringerOzone.monthly.ts <- ts(
  GaringerOzone.monthly$mean_ozone,
  start = c(s_year, s_month),
  end = c(f_year, f_month),
  frequency = 12
)

# Display the created time series
GaringerOzone.daily.ts
GaringerOzone.monthly.ts
```

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

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

# Plot the components of the daily decomposition
plot(GaringerOzone.daily.decomp, main = "Decomposition of Daily Ozone Concentration")
```



```r
# Decompose the monthly time series
GaringerOzone.monthly.decomp <- stl(GaringerOzone.monthly.ts, s.window = "periodic")

# Plot the components of the monthly decomposition
plot(GaringerOzone.monthly.decomp, main = "Decomposition of Monthly Ozone Concentration")
```



12. Run a monotonic trend analysis for the monthly Ozone series. In this case the seasonal Mann-Kendall is most appropriate; why is this?

```r
# Perform the seasonal Mann-Kendall test on the monthly time series
ozone_trend_test <- Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)

# Display the test results
summary(ozone_trend_test)
```

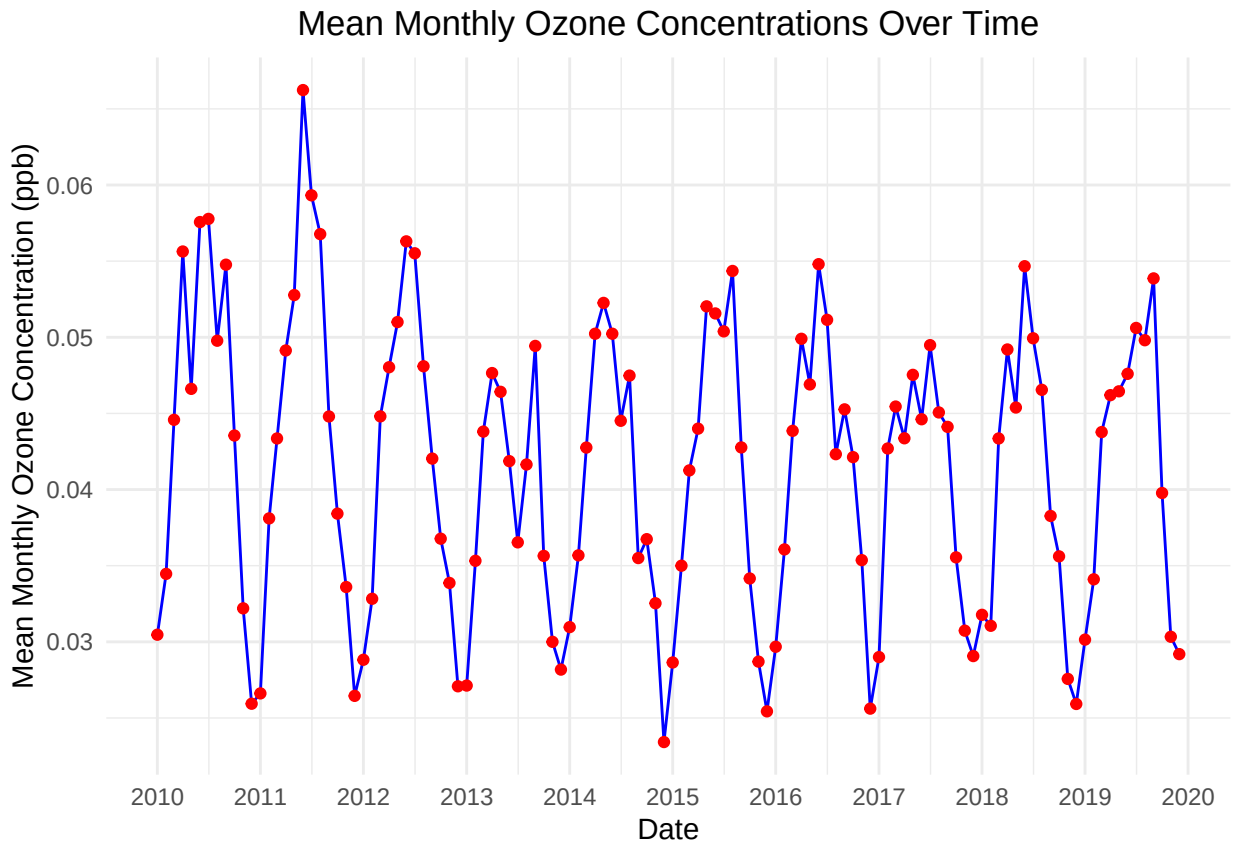
```
## Score =  -77 , Var(Score) = 1499
## denominator =  539.4972
## tau = -0.143, 2-sided pvalue =0.046724
```

Answer: The seasonal Mann-Kendall test is most appropriate for the monthly ozone series because it accounts for seasonality—periodic fluctuations in data that occur at regular intervals, such as monthly or annually. Ozone levels often vary with environmental factors like temperature and sunlight, creating predictable seasonal patterns. A standard Mann-Kendall test, which detects trends without assuming a specific data distribution, might misinterpret these regular seasonal

effects as part of a long-term trend. The seasonal Mann-Kendall test, however, analyzes each season separately, allowing for an accurate detection of any underlying trend while properly accounting for recurring seasonal variations.

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.

```
# Assuming `GaringerOzone.monthly` is a data frame with columns `Date` and `mean_ozone`
# Make sure `Date` is a Date object representing the first day of each month

ggplot(GaringerOzone.monthly, aes(x = Date, y = mean_ozone)) +
  geom_line(color = "blue") +          # Line layer
  geom_point(color = "red") +          # Point layer
  labs(
    title = "Mean Monthly Ozone Concentrations Over Time",
    x = "Date",
    y = "Mean Monthly Ozone Concentration (ppb)" # adjust the unit as appropriate
  ) +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y") + # One tick per year
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5) # Center the title
  )
```



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: To determine if ozone concentrations have changed over the 2010s at this station, the plot reveals a significant downward trend in mean monthly ozone levels throughout the decade. This trend, observed in the plot and confirmed by the seasonal Mann-Kendall test (tau = -0.143, p-value = 0.0467), indicates that ozone concentrations decreased overall across the 2010s. Seasonal patterns show periodic peaks in summer and lows in winter, but despite these fluctuations, the general trend points toward a reduction in ozone levels, suggesting potential impacts from changes in atmospheric conditions over this period.

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.

```
# Decompose the time series
decomposed_ts <- stl(GaringerOzone.monthly.ts, s.window = "periodic")

# Extract the seasonal component
seasonal_component <- decomposed_ts$time.series[, "seasonal"]

# Subtract the seasonal component from the original time series
non_seasonal <- GaringerOzone.monthly.ts - seasonal_component

#plot(GaringerOzone.monthly.ts)
#plot(non_seasonal)
```

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.

```
library(Kendall)
# Perform the seasonal Mann-Kendall test on the monthly time series
ozone_trend_test2 <- MannKendall(non_seasonal)

# Display the test results
summary(ozone_trend_test2)
```

```
## Score =  -1179 , Var(Score) = 194365.7
## denominator =  7139.5
## tau = -0.165, 2-sided pvalue =0.0075402
```

    Answer: The comparison between the non-seasonal and seasonal Mann-Kendall tests shows that both indicate a statistically significant downward trend in ozone levels, but the seasonal test provides a more conservative result. The non-seasonal test, with a higher score, variance, and stronger significance (p-value ~ 0.007), suggests a stronger trend; however, it doesn't account for seasonal fluctuations, potentially overestimating the trend's significance. The seasonal Mann-Kendall test, with a smaller score and a p-value closer to 0.05, captures the trend more precisely by adjusting for seasonality, making it a more reliable choice for time series with seasonal patterns.