Assignment 8: Time Series Analysis

Nargis Taraki

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

```r
# Check your working directory
getwd()  # This will show your current working directory
```

```
## [1] "/home/guest/EDE_Fall2024"
```

```r
# Load necessary libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## 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 error
```

```
library(lubridate)
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(dplyr)
library(here)
```

```
## here() starts at /home/guest/EDE_Fall2024
```

```
# Set ggplot theme
mytheme <- theme_gray(base_size = 12) +
  theme(axis.title = element_text(color = "black"),
        legend.position = "right",
        legend.title = element_text(color = "black"),
        plot.title = element_text(hjust = 0.5))

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.

```
# Importing raw data
Grainger2010 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2010_raw.csv'),
                          stringsAsFactors = TRUE)
Grainger2011 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2011_raw.csv'),
                          stringsAsFactors = TRUE)
Grainger2012 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2012_raw.csv'),
                          stringsAsFactors = TRUE)
Grainger2013 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2013_raw.csv'),
                          stringsAsFactors = TRUE)
```

```r
Grainger2014 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2014_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2015 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2015_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2016 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2016_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2017 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2017_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2018 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2018_raw.csv'),
                        stringsAsFactors = TRUE)
Grainger2019 <-
  read.csv(here('Data','Raw','Ozone_TimeSeries','EPAair_O3_GaringerNC2019_raw.csv'),
                        stringsAsFactors = TRUE)

# Creating a single dataframe using the bind_rows function
GaringerOzone <- bind_rows(Grainger2010, Grainger2011, Grainger2012, Grainger2013,
                           Grainger2014, Grainger2015, Grainger2016, Grainger2017,
                           Grainger2018, Grainger2019)

dim(GaringerOzone)
```

```
## [1] 3589   20
```

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 Using the as.Date function to set date column as date class
GaringerOzone$Date <- mdy(GaringerOzone$Date)

# 4  Wrangling to select Date, Daily.Max.8.hour.Ozone.Concentration, and DAILY_AQI_VALUE
GaringerOzone <- GaringerOzone %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration,DAILY_AQI_VALUE)

# 5  Creating daily dataset using as.data.frame(seq())
start_date <- as.Date("2010-01-01")
```

```r
end_date <- as.Date("2019-12-31")
DailyGaringerOzone <-
  as.data.frame(seq(start_date, end_date, by = "days"))
colnames(DailyGaringerOzone) <- "Date"

# 6  Using the left_join function to combine two datasets
GaringerOzone <- left_join(DailyGaringerOzone,GaringerOzone, by = "Date")
dim(GaringerOzone)
```

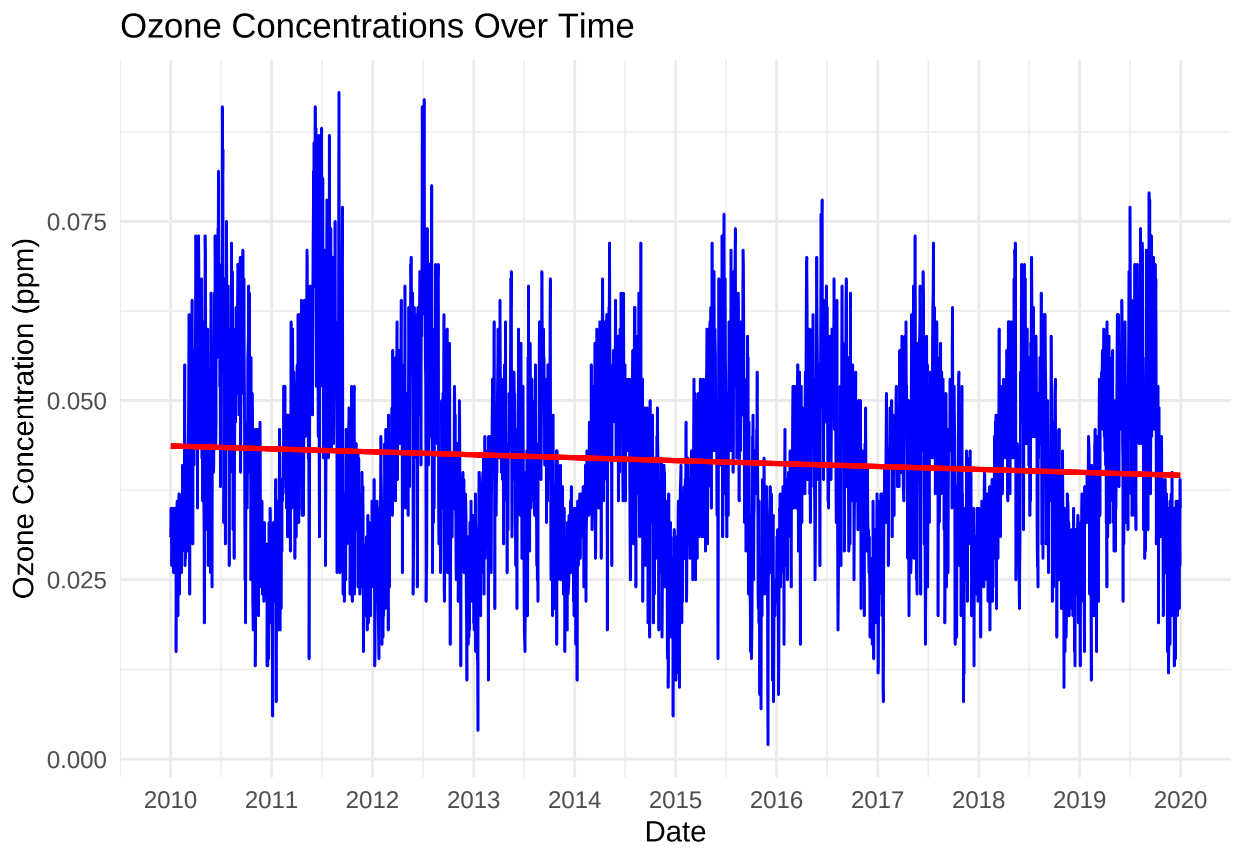
```
## [1] 3652    3
```

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 Load necessary libraries

library(ggplot2)
library(dplyr)

# Filtering out rows with missing ozone concentration values (NA)
filtered_data <- GaringerOzone %>%
  filter(!is.na(Daily.Max.8.hour.Ozone.Concentration))

# Checking if there are any valid data points left
if (nrow(filtered_data) > 0) {
  # Creating a line plot with ggplot2
  Ozone_Time <- ggplot(filtered_data, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
    geom_line(color = "blue") +  # Line for actual concentrations
    geom_smooth(method = "lm", se = FALSE, color = "red") +  # Linear trend line
    labs(
      title = "Ozone Concentrations Over Time",
      x = "Date",
      y = "Ozone Concentration (ppm)"
    ) +
    scale_x_date(date_labels = "%Y", date_breaks = "1 year") +  # Format x-axis for year
    theme_minimal()  # Use a minimal theme for better clarity

  # Print the plot
  print(Ozone_Time)
} else {
  cat("No valid data points for the plot.")
}
```

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



Answer: The plot shows a slightly decreasing trend in ozone concentrations from 2010 to 2020, indicating improving air quality over time. Additionally, there are noticeable seasonal fluctuations, with higher ozone levels typically observed in the warmer months and lower levels in the colder months.

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 Load the necessary package for interpolation
library(zoo)

# Use linear interpolation to fill missing ozone concentration values
GaringerOzone <- GaringerOzone %>%
  mutate(Daily.Max.8.hour.Ozone.Concentration =
           zoo::na.approx(Daily.Max.8.hour.Ozone.Concentration))

# Check for filled data
summary(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration)
```

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

Answer: We used linear interpolation to fill in missing ozone data because it connects the nearest data points with a straight line, reflecting the natural changes in ozone levels. Piecewise constant interpolation would ignore daily variations, while spline interpolation might introduce artificial fluctuations. Linear interpolation provides a realistic estimate of ozone concentrations on days without measurements.

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  Create the GaringerOzone.monthly data frame with year and month columns
GaringerOzone.monthly <- GaringerOzone %>%
  mutate(
    Year = year(Date),
    Month = month(Date)
  ) %>%
  group_by(Year, Month) %>%
  summarize(
    MeanOzone = mean(Daily.Max.8.hour.Ozone.Concentration, na.rm = TRUE)
  ) %>%
  ungroup()
```

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

```
# Step 2: Create a new Date column set to the first day of each month (for graphing purposes)
GaringerOzone.monthly <- GaringerOzone.monthly %>%
  mutate(
    Date = as.Date(paste(Year, Month, "01", sep = "-"))
  )

# View the resulting data frame
head(GaringerOzone.monthly)
```

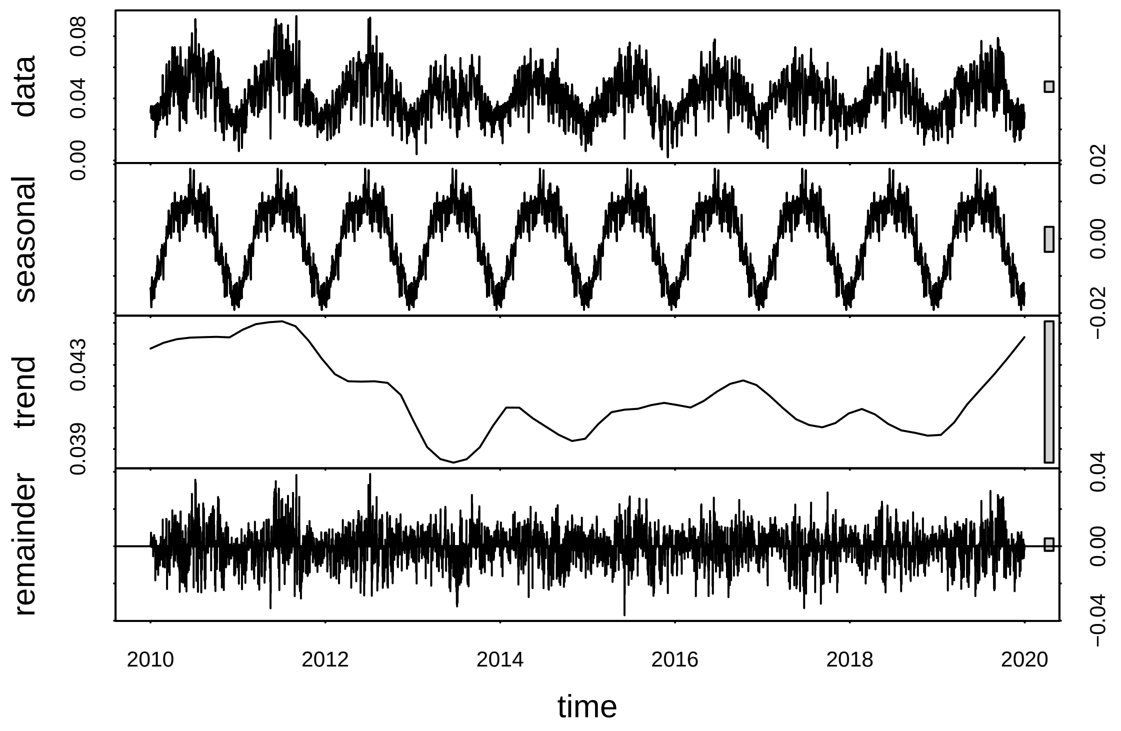
```
## # A tibble: 6 x 4
##    Year Month MeanOzone Date
##   <dbl> <dbl>     <dbl> <date>
## 1  2010     1    0.0305 2010-01-01
## 2  2010     2    0.0345 2010-02-01
## 3  2010     3    0.0446 2010-03-01
## 4  2010     4    0.0556 2010-04-01
## 5  2010     5    0.0466 2010-05-01
## 6  2010     6    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.

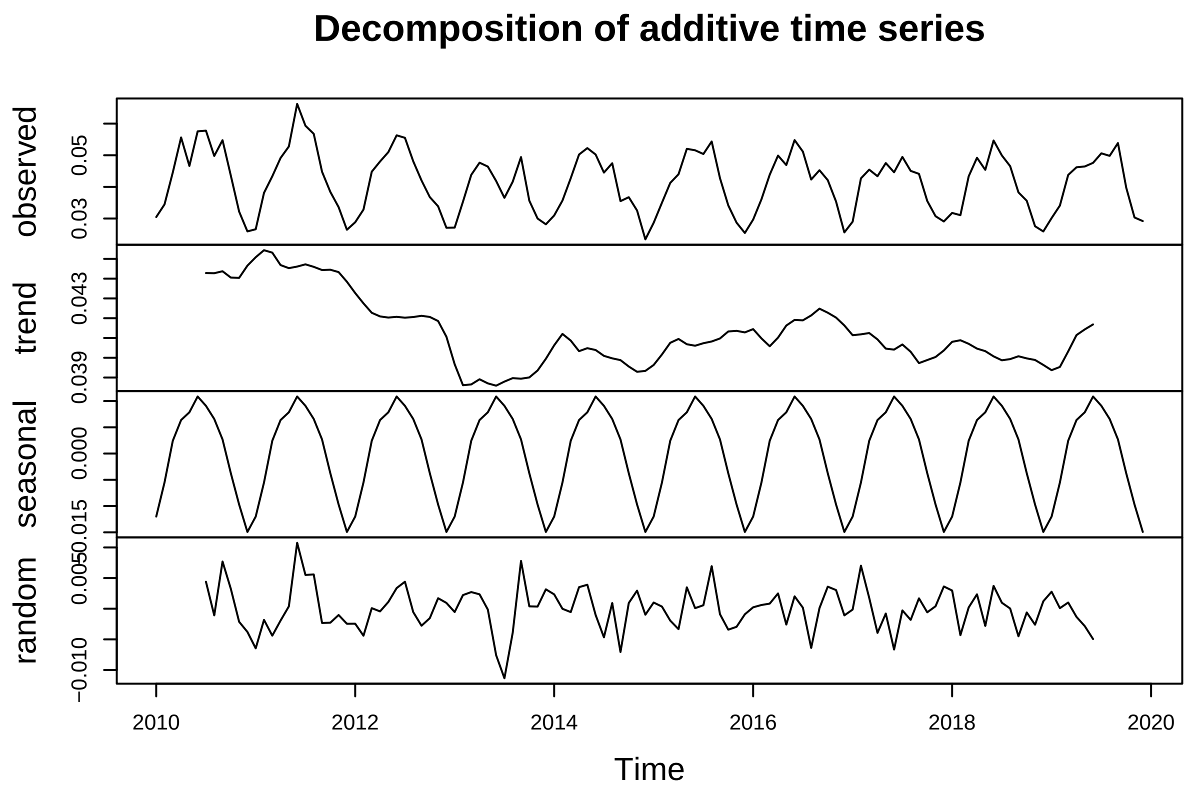
```r
#10
GaringerOzone.daily.ts <- ts(
  GaringerOzone$Daily.Max.8.hour.Ozone.Concentration,
  start = c(2010, 1),
  end = c(2019, 365),
  frequency = 365
)

# 2.The monthly data spans from 2010-01 to 2019-12, with a frequency of 12 for monthly data
GaringerOzone.monthly.ts <- ts(
  GaringerOzone.monthly$MeanOzone,
  start = c(2010, 1),
  end = c(2019, 12),
  frequency = 12
)
```

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

```r
#11
# Decomposing the daily time series
GaringerOzone.daily.decomp <- stl(GaringerOzone.daily.ts, s.window = "periodic")

# Plotting the components (default settings usually handle layout well)
plot(GaringerOzone.daily.decomp)
```



```r
# Decomposing the monthly time series
GaringerOzone.monthly.decomp <- decompose(GaringerOzone.monthly.ts)

# Plotting the components
plot(GaringerOzone.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?

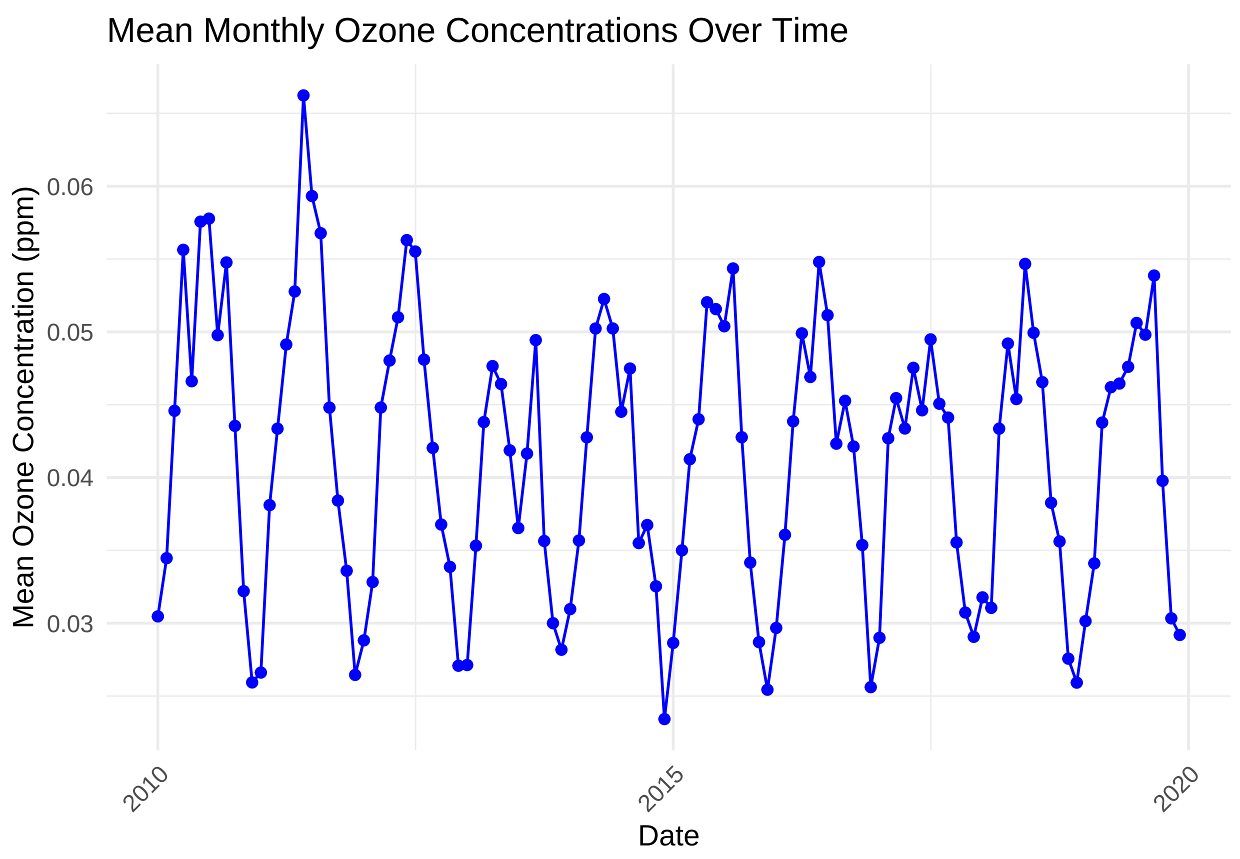
```r
#12#
# Load the trend package
library(trend)

# Run the Seasonal Mann-Kendall test
result <- smk.test(GaringerOzone.monthly.ts)

# Print the results
print(result)
```

```
##
##  Seasonal Mann-Kendall trend test (Hirsch-Slack test)
##
## data:  GaringerOzone.monthly.ts
## z = -1.963, p-value = 0.04965
## alternative hypothesis: true S is not equal to 0
## sample estimates:
##    S varS
##  -77 1499
```

Answer: The Seasonal Mann-Kendall test is ideal for analyzing monthly ozone data because it accounts for seasonal variations, such as temperature changes and pollution sources. This allows for a clearer identification of trends over time. The test results show a statistically significant decreasing trend in monthly ozone concentrations, with a z-value of -1.963. The p-value of 0.046724 is below 0.05, indicating this trend is statistically significant at the 5% level, suggesting a real decline in ozone levels over the analyzed period.

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

library(ggplot2)

# Create a plot of mean monthly ozone concentrations over time
plot <- ggplot(GaringerOzone.monthly, aes(x = Date, y = MeanOzone)) +
  geom_point(color = "blue") +
  geom_line(color = "blue") +
  labs(
    title = "Mean Monthly Ozone Concentrations Over Time",
    x = "Date",
    y = "Mean Ozone Concentration (ppm)"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
# Print the plot
print(plot)
```



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 illustrates a slight but consistent decrease in ozone levels from July 2010 to February 2019. The Seasonal Mann-Kendall trend test results show a statistically significant decreasing trend in ozone concentrations (tau = -0.143, p-value = 0.046724). This finding suggests that air quality may be improving over time, but the p-value is close to the significance threshold of 0.05, indicating that further studies are necessary to confirm this trend and explore any underlying factors affecting 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.

```r
#15  Decomposing the time series
 GaringerOzone.monthly.ts_decomposed <- decompose(GaringerOzone.monthly.ts)
 # Extracting the seasonal component
 seasonal_component <- GaringerOzone.monthly.ts_decomposed$seasonal
 # Subtracting the seasonal component from the original time series
 GaringerOzone.monthly.ts_deseasonalized <- GaringerOzone.monthly.ts- seasonal_component
 #16
 # Running the Mann-Kendall test on the deseasonalized data
```

```r
mk_result <- mk.test(GaringerOzone.monthly.ts_deseasonalized)
print(mk_result)
```

```
##
##  Mann-Kendall trend test
##
## data:  GaringerOzone.monthly.ts_deseasonalized
## z = -2.6039, n = 120, p-value = 0.009216
## alternative hypothesis: true S is not equal to 0
## sample estimates:
##             S          varS           tau
## -1.149000e+03  1.943657e+05 -1.609356e-01
```

Answer:The Non-Seasonal Mann-Kendall test on the deseasonalized ozone data yielded a z-value of -2.6039, a p-value of 0.009216, and a tau of -0.1609356, indicating a statistically significant decreasing trend (p-value < 0.05) with moderate strength. In contrast, the Seasonal Mann-Kendall test on the original data showed a tau of -0.143 and a p-value of 0.046724, suggesting no significant trend. This comparison highlights that the deseasonalized data reveals a clearer significant downward trend in ozone levels that was not apparent with the seasonal component included.