Assignment 8: Time Series Analysis

Andi Mujollari

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/EDA_Fall2022"
```

```
#install.packages("tidyverse")
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)
#install.packages("zoo")
library(zoo)
```

```
##
## Attaching package: 'zoo'
##
```

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

```
#install.packages("trend")
library(trend)
#install.packages("here")
library(here)
```

```
## here() starts at /home/guest/EDA_Fall2022
```

```
# Here I set the ggplot theme
theme_set(theme_classic())
```

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

#setwd("C:/Users/andimujollari/Desktop/EDE-Fall2023/Data/Raw/Ozone_TimeSeries")
#here("./Users/andimujollari/Desktop/EDE-Fall2023/Data/Raw/Ozone_TimeSeries")

# Here is the list of file names to import
GaringerFiles = list.files(path = "./Data/Raw/Ozone_TimeSeries/", pattern="*.csv", full.names=TRUE)
GaringerFiles
```

```
##  [1] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2010_raw.csv"
##  [2] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2011_raw.csv"
##  [3] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2012_raw.csv"
##  [4] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2013_raw.csv"
##  [5] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2014_raw.csv"
##  [6] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2015_raw.csv"
##  [7] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2016_raw.csv"
##  [8] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2017_raw.csv"
##  [9] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2018_raw.csv"
## [10] "./Data/Raw/Ozone_TimeSeries//EPAair_O3_GaringerNC2019_raw.csv"
```

```
GaringerOzone <- GaringerFiles %>%
  plyr::ldply(read.csv)


# Here I check the dimensions of the combined dataframe
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.

```
# 3

GaringerOzone$Date <- as.Date(GaringerOzone$Date, format = "%m/%d/%Y")

class(GaringerOzone$Date)
```

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

```
# 4

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

str(GaringerOzone)
```

```
## '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

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

# Here I create a "Days" dataframe with a complete sequence of dates
start_date <- as.Date("2010-01-01")
end_date <- as.Date("2019-12-31")
Days <- as.data.frame(seq(from = start_date, to = end_date, by = "day"))
colnames(Days) <- "Date"

#  Fill in missing days
GaringerOzone <- Days %>%
  left_join(GaringerOzone, by = "Date")


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


# 6

# Combining the data frames
GaringerOzone <- left_join(Days, GaringerOzone, by = "Date")

# Checking the dimensions of the combined data frame
dim(GaringerOzone)
```

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

Visualize

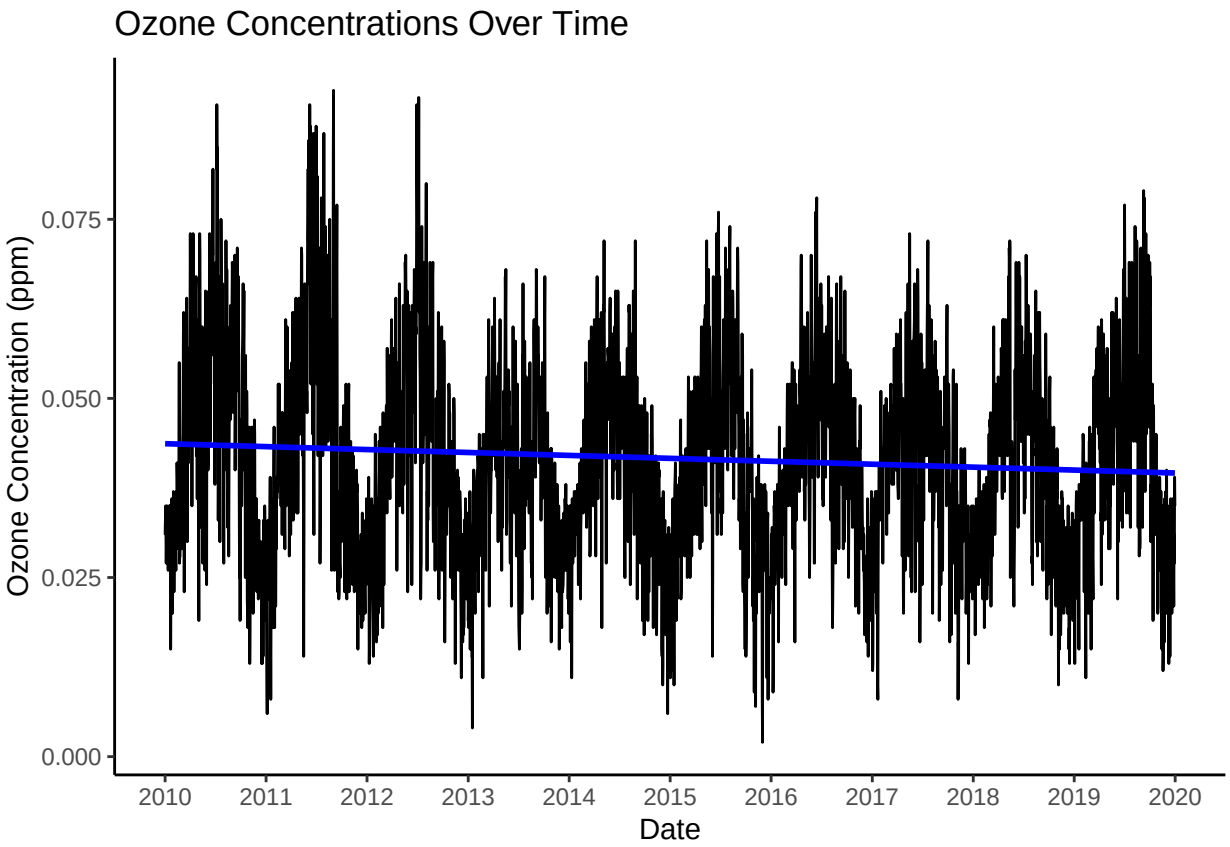
7. Create a line plot depicting ozone concentrations over time. In this case, we will plot actual concentrations in ppm, not AQI values. Format your axes accordingly. Add a smoothed line showing any linear trend of your data. Does your plot suggest a trend in ozone concentration over time?

```
#7
# 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
  ggplot(filtered_data, aes(x = Date, y = Daily.Max.8.hour.Ozone.Concentration)) +
    geom_line() +
    geom_smooth(method = "lm", se = FALSE, color = "blue") +
    labs(
      title = "Ozone Concentrations Over Time",
      x = "Date",
      y = "Ozone Concentration (ppm)"
    ) +
    scale_x_date(date_labels = "%Y", date_breaks = "1 year") +
    theme_classic()
} else {
  cat("No valid data points for the plot.")
}
```

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




```
print(plot)
```

```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x5616b5016888>
## <environment: namespace:base>
```

Answer: Upon analyzing the plot, it becomes evident that there is a gradual decrease in the trend over the past decade ranging from 2010 to 2020. Not only that, but we can also observe seasonal changes and variations within each of the years in question. It is apparent that certain months experience peaks while others show a decrease, which results in noticeable troughs. This information indicates that there are multiple factors impacting the trend, and it is important to consider all of them when making any projections or conclusions.

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

  GaringerOzone$Daily.Max.8.hour.Ozone.Concentration <- na.approx(GaringerOzone$Daily.Max.8.hour.Ozone.
```

Answer: Linear interpolation was chosen over piecewise constant and spline interpolation to fill in the missing ozone data because it accounts for the steady rate of change between data points, which aligns with the nature of ozone fluctuations. Piecewise constant interpolation does

not account for daily variations, while spline interpolation could introduce artificial swings and potentially misrepresent the true 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
GaringerOzone <- GaringerOzone %>%
  mutate(Year = year(Date), Month = month(Date))

GaringerOzone.monthly <- GaringerOzone %>%
  group_by(Year, Month) %>%
  summarize(Mean_Ozone = 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 <- 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.

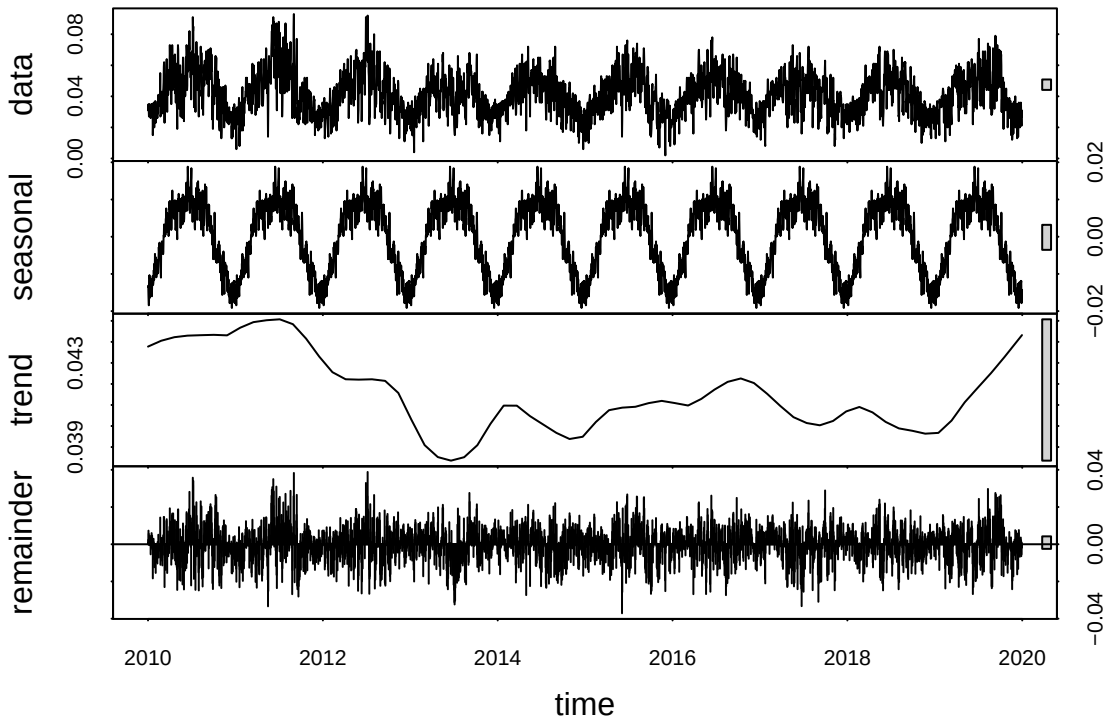
```
#10
# Daily Time Series
start_year_daily <- year(min(GaringerOzone$Date))
end_year_daily <- year(max(GaringerOzone$Date))
GaringerOzone.daily.ts <- ts(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration,
                             start = c(start_year_daily, 1),
                             end = c(end_year_daily, 365),
                             frequency = 365)

# Monthly Time Series
start_year_monthly <- min(GaringerOzone.monthly$Year)
end_year_monthly <- max(GaringerOzone.monthly$Year)
start_month_monthly <- min(GaringerOzone.monthly$Month[GaringerOzone.monthly$Year == start_year_monthly])
end_month_monthly <- max(GaringerOzone.monthly$Month[GaringerOzone.monthly$Year == end_year_monthly])
GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$Mean_Ozone,
                               start = c(start_year_monthly, start_month_monthly),
                               end = c(end_year_monthly, end_month_monthly),
                               frequency = 12)
```

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

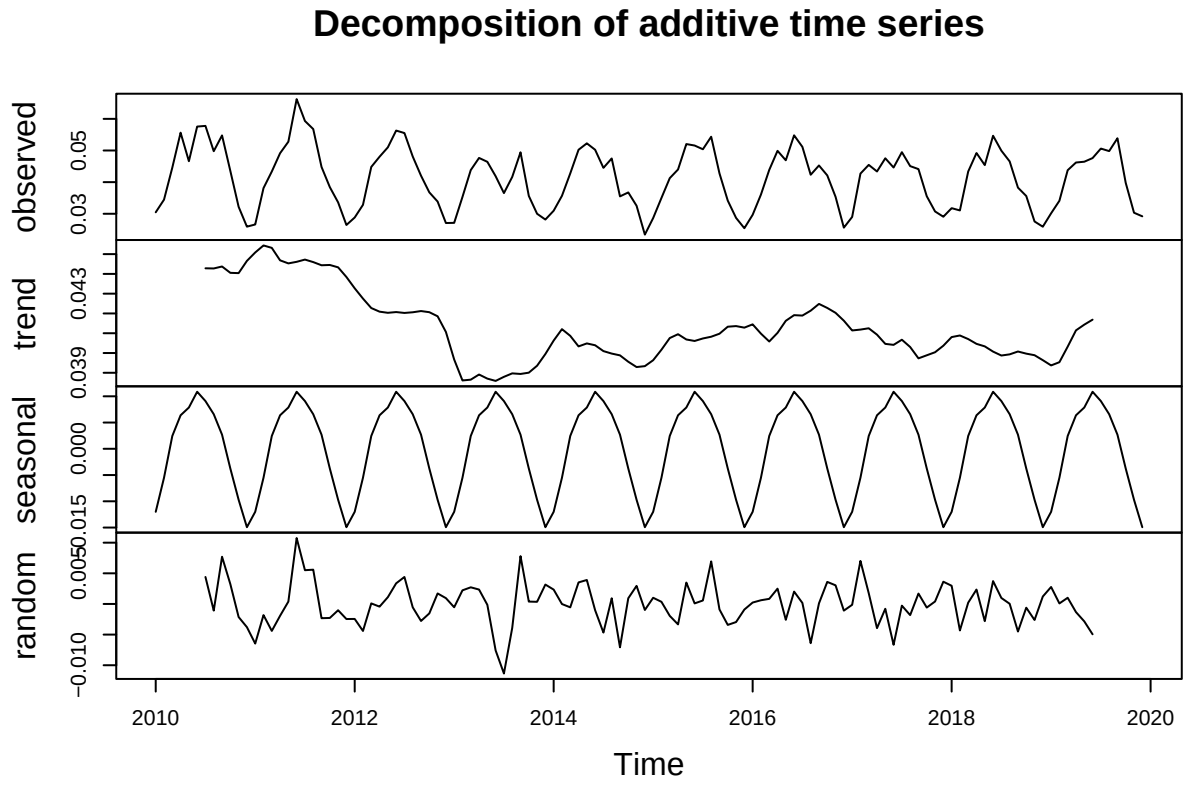
```
#11
# Decomposing the daily time series
GaringerOzone.daily.decomp <- stl(GaringerOzone.daily.ts, s.window = "periodic")

# Plotting the components
plot(GaringerOzone.daily.decomp)
```




```
# 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?

```
#12

# Install the 'trend' package if you haven't already
#install.packages("trend")

# Load the 'trend' package
library(trend)

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

# Output 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: To analyze trends in monthly ozone levels, the Seasonal Mann-Kendall test is the most suitable method. This test takes into account seasonal variations caused by various factors such as weather, which is crucial to accurately identify trends over time. The Seasonal Mann-Kendall test is specifically designed to address these seasonal cycles, which can otherwise mask or distort the true underlying trend in the data when using a standard Mann-Kendall test. Based on my test results, the monthly Ozone series shows a statistically significant decreasing trend at a 5% significance level, with a z-value of -1.963 and a p-value of 0.04965.

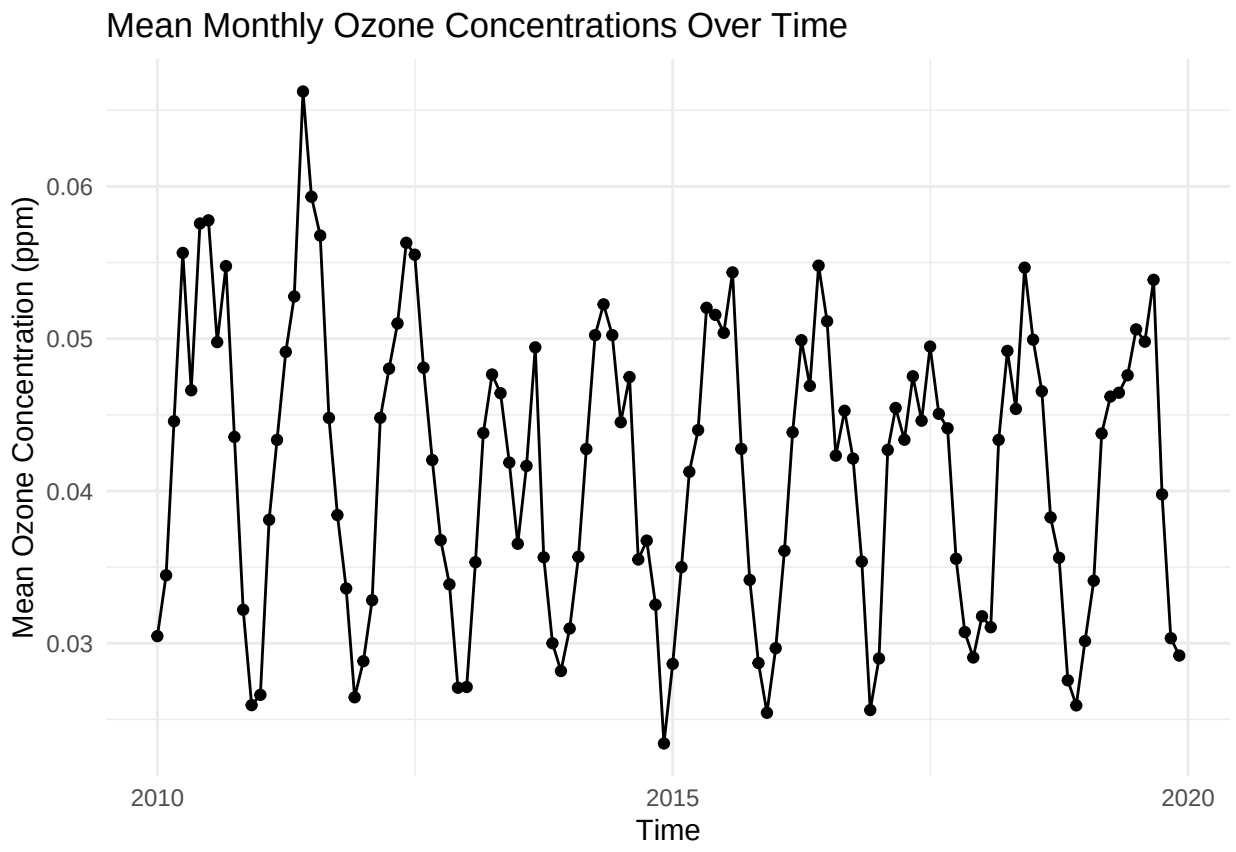
13. Create a plot depicting mean monthly ozone concentrations over time, with both a geom_point and a geom_line layer. Edit your axis labels accordingly.

```
# 13

library(ggplot2)

plot <- ggplot(GaringerOzone.monthly, aes(x = Date, y = Mean_Ozone)) +
  geom_point() +
  geom_line() +
  labs(
    title = "Mean Monthly Ozone Concentrations Over Time",
    x = "Time",
    y = "Mean Ozone Concentration (ppm)"
  ) +
  theme_minimal()

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: After analyzing the monthly mean ozone concentration data from July 2010 to February 2019 using the Seasonal Mann-Kendall trend test, it appears that there has been a consistent, albeit slight, decrease in ozone concentration each month when accounting for seasonal variation. The results indicate a statistically significant downward trend in ozone levels over this period (z = -1.963, p-value = 0.04965). However, it's important to note that the trend is marginally statistically significant, given that the p-value is close to the conventional threshold of 0.05.

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

# 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
mk_result <- mk.test(GaringerOzone.monthly.ts_deseasonalized)

# Printing the result
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: Non-Seasonal Mann-Kendall Test on Deseasonalized Data: The data suggests a notable decrease in deseasonalized ozone levels (p-value < 0.05), with a moderate trend strength (tau). Upon conducting the Seasonal Mann-Kendall Test on the original data, no season showed a significant trend, as evidenced by all p-values being greater than 0.05. However, upon comparing the results, it is evident that the deseasonalized data shows a significant downward trend that was not evident when analyzing the data with its seasonal component intact.