Assignment 7: Time Series Analysis

Shivani Kuckreja

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on time series analysis.

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, submit the completed exercise (PDF file) to the dropbox in Sakai. Add your last name into the file name (e.g., "Fay_A07_TimeSeries.Rmd") prior to submission.

The completed exercise is due on Monday, March 14 at 7:00 pm.

Set up

1. Set up your session:

- Check your working directory
- Load the tidyverse, lubridate, zoo, and trend packages
- Set your ggplot theme

```
#1
getwd()
```

```
## [1] "/Users/shivanikuckreja/OneDrive - Wellesley College/Duke/Spring 2022 Classes/Environmental Data
```

```
#install.packages("tidyverse")
#install.packages("lubridate")
#install.packages("trend")
#install.packages("zoo")
#install.packages("Kendall")
#install.packages("tseries")
#install.packages("contrib.url")

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(trend)
library(zoo)
```

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

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

```
library(Kendall)
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
# Set theme
mytheme <- theme_classic(base_size = 14) +
  theme(axis.text = element_text(color = "blue"),
        legend.position = "top")
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.

```
#2

EPAair2010 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2010_raw.csv", stringsAsFactors

EPAair2011 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2011_raw.csv", stringsAsFactors

EPAair2012 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2012_raw.csv", stringsAsFactors

EPAair2013 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2013_raw.csv", stringsAsFactors

EPAair2014 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2014_raw.csv", stringsAsFactors

EPAair2015 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2015_raw.csv", stringsAsFactors

EPAair2016 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2016_raw.csv", stringsAsFactors

EPAair2017 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2017_raw.csv", stringsAsFactors

EPAair2018 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2018_raw.csv", stringsAsFactors
```

```
EPAair2019 <- read.csv("./Data/Raw/Ozone_TimeSeries/EPAair_O3_GaringerNC2019_raw.csv", stringsAsFactors

#Combine all datasets into a single dataframe named `GaringerOzone` of
#3589 observation and 20 variables.

GaringerOzone <- rbind(EPAair2010, EPAair2011, EPAair2012, EPAair2013,
                       EPAair2014, EPAair2015, EPAair2016, EPAair2017,
                       EPAair2018, EPAair2019)
```

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')

# 4
GaringerOzone_Wrangled <- GaringerOzone %>%
  select(Date, Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
summary(GaringerOzone_Wrangled)
```

```
##       Date            Daily.Max.8.hour.Ozone.Concentration DAILY_AQI_VALUE
##  Min.   :2010-01-01   Min.   :0.00200                      Min.   :  2.00
##  1st Qu.:2012-07-03   1st Qu.:0.03200                      1st Qu.: 30.00
##  Median :2015-01-04   Median :0.04100                      Median : 38.00
##  Mean   :2015-01-01   Mean   :0.04163                      Mean   : 41.57
##  3rd Qu.:2017-07-02   3rd Qu.:0.05100                      3rd Qu.: 47.00
##  Max.   :2019-12-31   Max.   :0.09300                      Max.   :169.00
```

```
sum(is.na(GaringerOzone_Wrangled))
```

```
## [1] 0
```

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

# 6
GaringerOzone <- left_join(Days, GaringerOzone_Wrangled)
```

```
## Joining, 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?

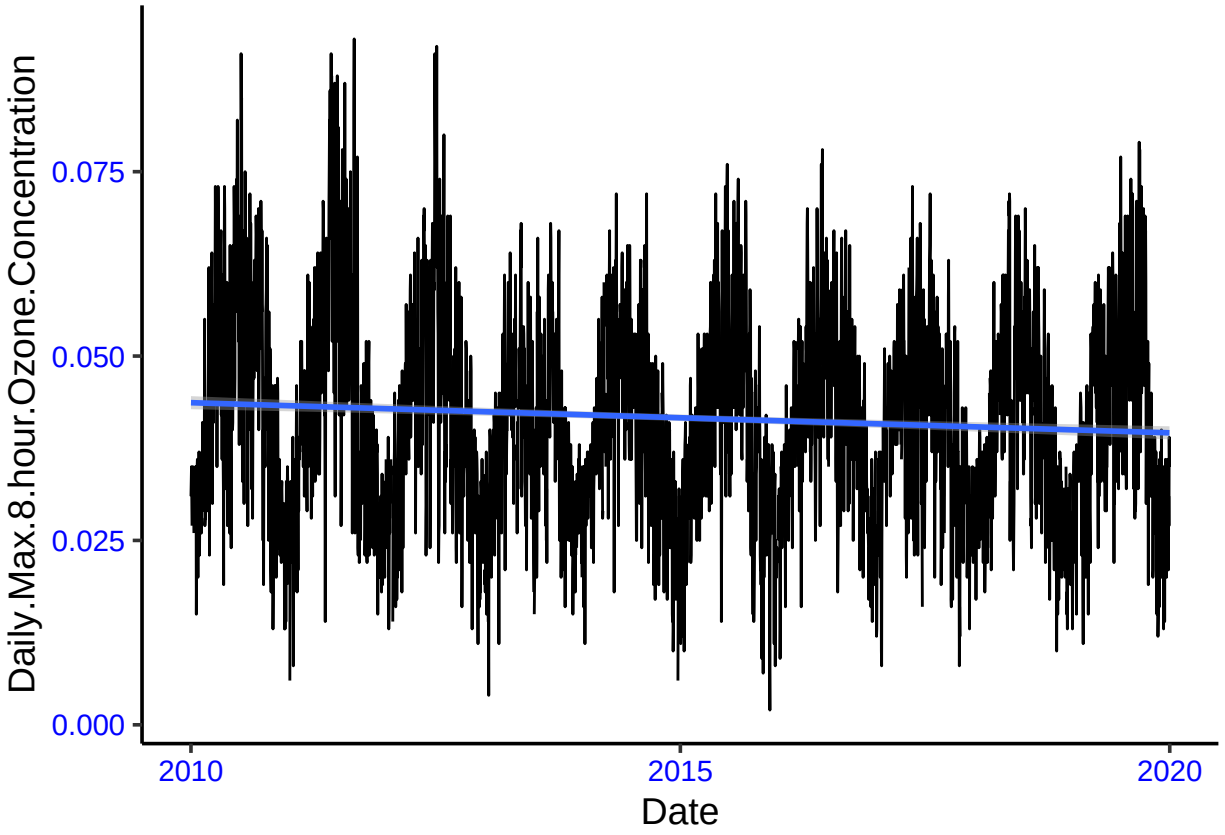
```
#7
summary(GaringerOzone$Daily.Max.8.hour.Ozone.Concentration)
```

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

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

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

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



Answer: Yes, the plot does suggest a cyclical trend and slightly donward-facing trend in ozone concentration overtime.

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
summary(GaringerOzone)
```

```
##       Date            Daily.Max.8.hour.Ozone.Concentration DAILY_AQI_VALUE
##  Min.   :2010-01-01   Min.   :0.00200                      Min.   :  2.00
##  1st Qu.:2012-07-01   1st Qu.:0.03200                      1st Qu.: 30.00
##  Median :2014-12-31   Median :0.04100                      Median : 38.00
##  Mean   :2014-12-31   Mean   :0.04163                      Mean   : 41.57
##  3rd Qu.:2017-07-01   3rd Qu.:0.05100                      3rd Qu.: 47.00
##  Max.   :2019-12-31   Max.   :0.09300                      Max.   :169.00
##                       NA's   :63                           NA's   :63
```

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

Answer: We used linear interpolation because the data pattern is known. We could "connect the dots"–Any missing data are assumed to fall between the previous and next measurement, with a straight line drawn between the known points determining the values of the interpolated data on any given date.

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.monthly <-
  GaringerOzone_MissingDaily %>%
  mutate(Month=month(Date),
  Year=year(Date)) %>%
  mutate (Date=my(paste0(Month, "-", Year))) %>%
  dplyr::group_by(Date, Month, Year) %>%
  dplyr::summarize(mean_Ozone=mean(Daily.Max.8.hour.Ozone.Concentration)) %>%
  select(mean_Ozone, Date)
```

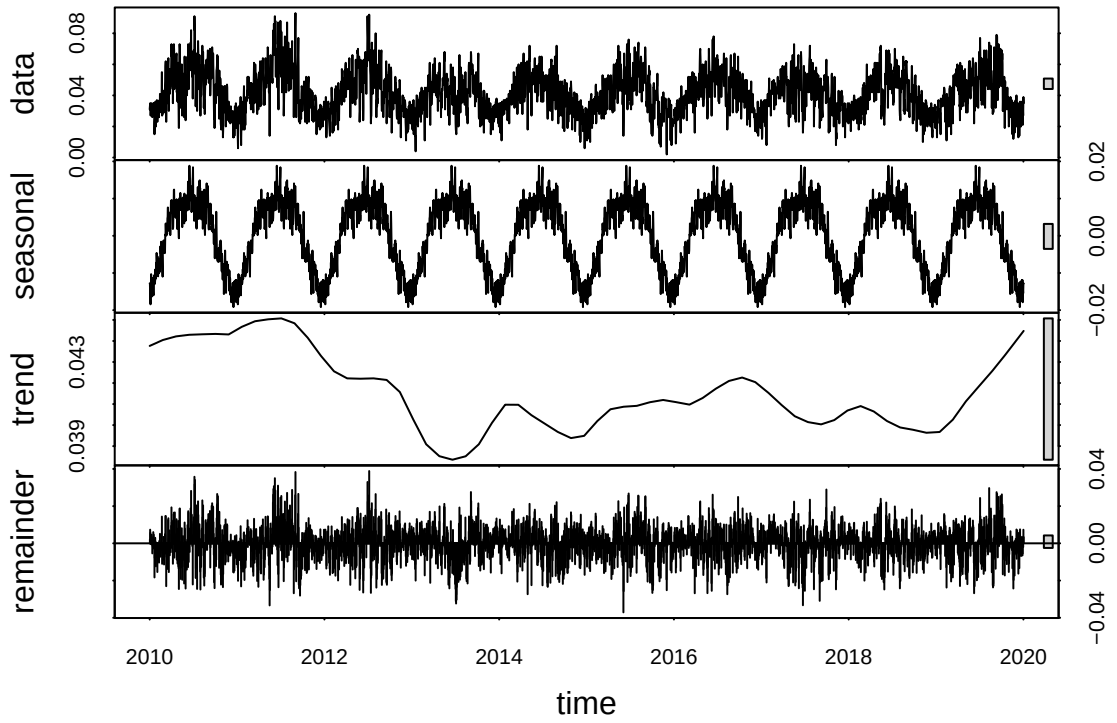
```
## `summarise()` has grouped output by 'Date', 'Month'. You can override using the
## `.groups` argument.
## Adding missing grouping variables: `Month`
```

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.

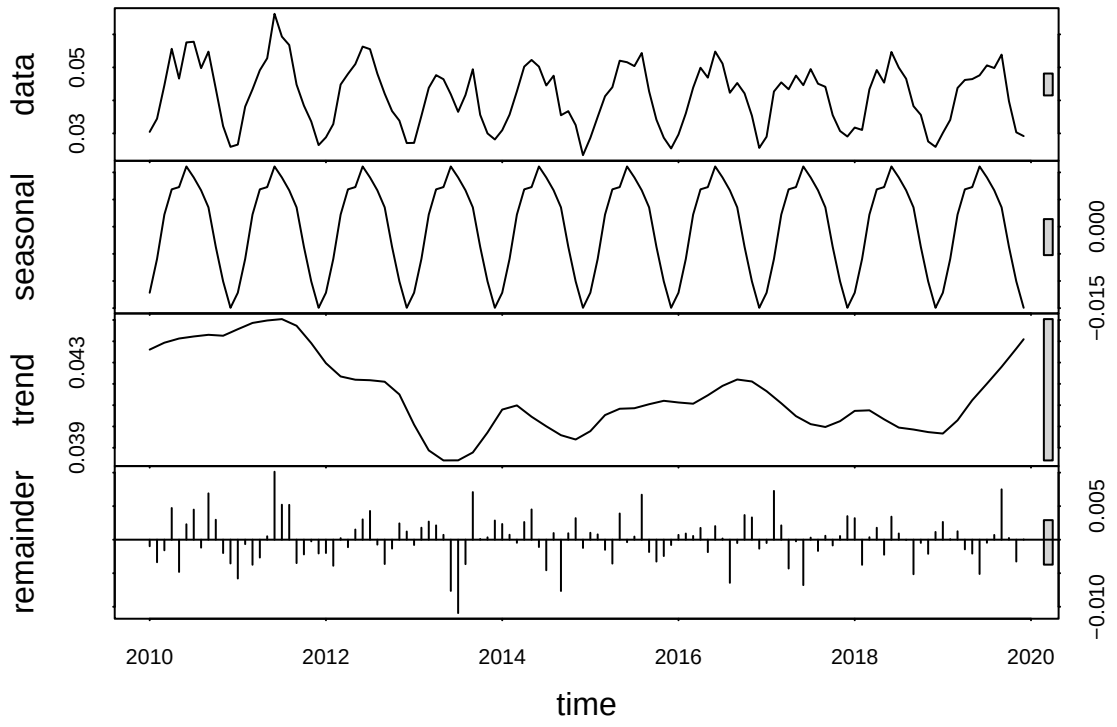
```
#10
GaringerOzone.daily.ts <-
  ts(GaringerOzone_MissingDaily$Daily.Max.8.hour.Ozone.Concentration,
  start=c(2010,1),
   frequency=365)

GaringerOzone.monthly.ts <-
 ts(GaringerOzone.monthly$mean_Ozone, start=c(2010,1),
    frequency=12)
```

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

```r
#11
DailyDecomposed <- stl(GaringerOzone.daily.ts, s.window = "periodic")
plot(DailyDecomposed)
```




```r
MonthlyDecomposed <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
plot(MonthlyDecomposed)
```



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
GaringerOzone_Monthly_Trend1 <-
  Kendall::SeasonalMannKendall(GaringerOzone.monthly.ts)
summary(GaringerOzone_Monthly_Trend1)
```

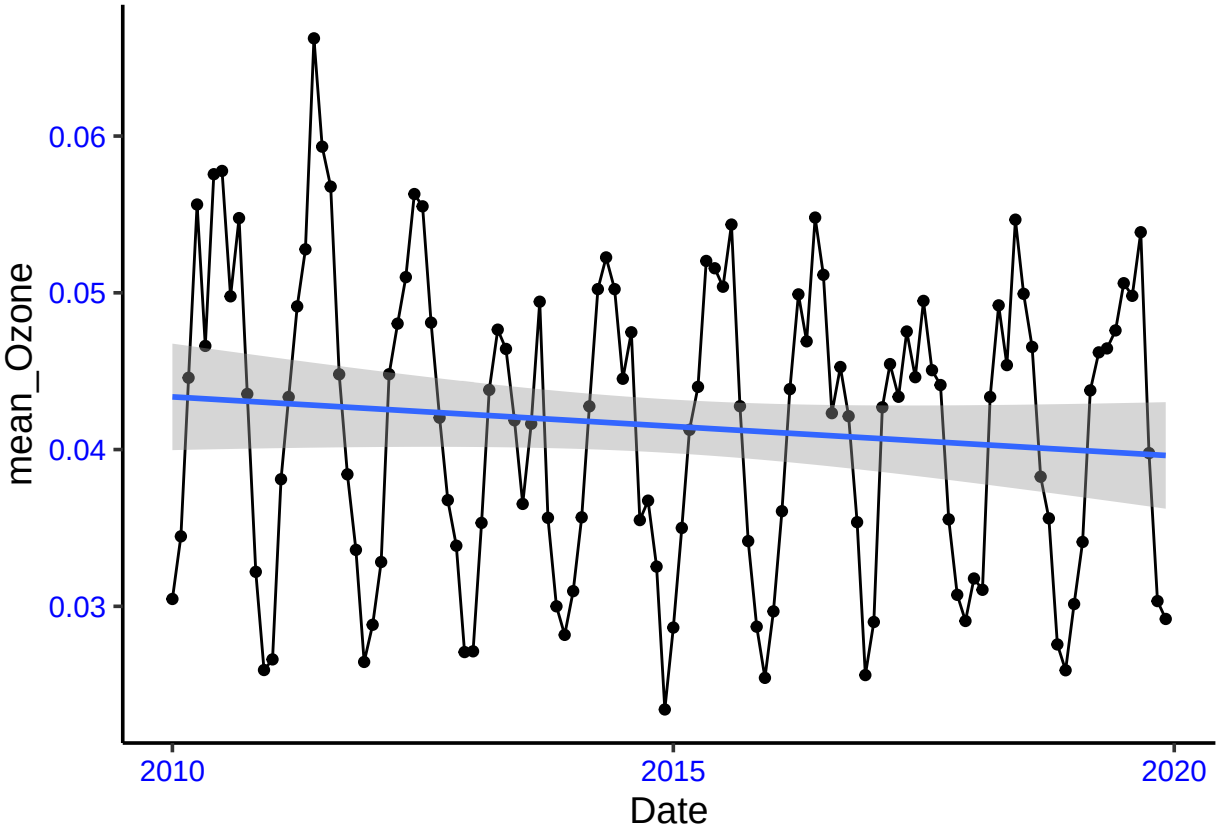
```
## Score =  -77 , Var(Score) = 1499
## denominator =  539.4972
## tau = -0.143, 2-sided pvalue =0.046724
```

Answer: In this case, the Seasonal Mann-Kendallis most appropriate because the stl function determined that we have seasonal data.

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
Mean_Monthly_Ozone <-
  ggplot(GaringerOzone.monthly, aes(x=Date, y=mean_Ozone)) +
  geom_point()+
  geom_line()+
  geom_smooth(method = lm)
    print(Mean_Monthly_Ozone)
```

```
## `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: Code not running

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

GaringerOzone_Monthly_NonSeasonal <-
  mutate(GaringerOzone_Monthly_NonSeasonal_Components,
         Observed = GaringerOzone.monthly$mean_Ozone,
           Date = GaringerOzone.monthly$Date,
Ozone = trend+remainder) %>%
select(Ozone, Observed, Date)


#16

GaringerOzone_Monthly_NonSeasonal.ts <-
 ts(GaringerOzone_Monthly_NonSeasonal$Ozone, start=c(2010,1),
   frequency=12)

GaringerOzone_Monthly_Trend3 <-
  Kendall::MannKendall(GaringerOzone_Monthly_NonSeasonal.ts)
summary(GaringerOzone_Monthly_Trend3)
```

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

Answer: