ENV 790.30 - Time Series Analysis for Energy Data | Spring 2023

Assignment 5 - Due date 02/27/23

Student Name

Directions

You should open the .rmd file corresponding to this assignment on RStudio. The file is available on our class repository on Github. And to do so you will need to fork our repository and link it to your RStudio.

Once you have the file open on your local machine the first thing you will do is rename the file such that it includes your first and last name (e.g., "LuanaLima_TSA_A05_Sp23.Rmd"). Then change "Student Name" on line 4 with your name.

Then you will start working through the assignment by **creating code and output** that answer each question. Be sure to use this assignment document. Your report should contain the answer to each question and any plots/tables you obtained (when applicable).

When you have completed the assignment, **Knit** the text and code into a single PDF file. Submit this pdf using Sakai.

R packages needed for this assignment: "xlsx" or "readxl", "ggplot2", "forecast","tseries", and "Kendall". Install these packages, if you haven't done yet. Do not forget to load them before running your script, since they are NOT default packages.\

```
#Load/install required package here
library(forecast)
```

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

```
library(tseries)
library(ggplot2)
library(Kendall)
library(lubridate)
```

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

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

```
library(tidyverse)  #load this package so yon clean the data frame using pipes
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
```

```
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 1.0.0
## v purrr   1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
```

```
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

```
library(readxl)
```

Decomposing Time Series

Consider the same data you used for A04 from the spreadsheet "Table_10.1_Renewable_Energy_Production_and_Consumptio

```
#Importing data set - using xlsx package

#Importing data set without change the original file using read.xlsx
energy_data <- read_excel(path="../Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
## * `` -> `...13`
## * `` -> `...14`
```

```
#Now let's extract the column names from row 11
read_col_names <- read_excel(path="../Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Sou
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
## * `` -> `...13`
## * `` -> `...14`
```

```
colnames(energy_data) <- read_col_names
head(energy_data)
```

```
## # A tibble: 6 x 14
##   Month               Wood Ene~1 Biofu~2 Total~3 Total~4 Hydro~5 Geoth~6 Solar~7
##   <dttm>                   <dbl> <chr>     <dbl>   <dbl>   <dbl>   <dbl> <chr>
## 1 1973-01-01 00:00:00       130. Not Av~    130.    404.    273.    1.49 Not Av~
## 2 1973-02-01 00:00:00       117. Not Av~    117.    361.    242.    1.36 Not Av~
## 3 1973-03-01 00:00:00       130. Not Av~    130.    400.    269.    1.41 Not Av~
## 4 1973-04-01 00:00:00       125. Not Av~    126.    380.    253.    1.65 Not Av~
## 5 1973-05-01 00:00:00       130. Not Av~    130.    392.    261.    1.54 Not Av~
## 6 1973-06-01 00:00:00       125. Not Av~    126.    377.    250.    1.76 Not Av~
## # ... with 6 more variables: `Wind Energy Consumption` <chr>,
## #   `Wood Energy Consumption` <dbl>, `Waste Energy Consumption` <dbl>,
## #   `Biofuels Consumption` <chr>, `Total Biomass Energy Consumption` <dbl>,
## #   `Total Renewable Energy Consumption` <dbl>, and abbreviated variable names
## #   1: `Wood Energy Production`, 2: `Biofuels Production`,
## #   3: `Total Biomass Energy Production`,
## #   4: `Total Renewable Energy Production`, ...
```

```
nobs=nrow(energy_data)
nvar=ncol(energy_data)
```

Q1

For this assignment you will work only with the following columns: Solar Energy Consumption and Wind Energy Consumption. Create a data frame structure with these two time series only and the Date column. Drop the rows with *Not Available* and convert the columns to numeric. You can use filtering to eliminate the initial rows or convert to numeric and then use the drop_na() function. If you are familiar with pipes for data wrangling, try using it!

```
#create data frame
energy_solar_wind <- data.frame(energy_data$Month, energy_data$`Solar Energy Consumption`,energy_data$`

#Remove rows
energy_solar_wind2 <- energy_solar_wind[-c(1:132),]

#Convert to numeric
energy_solar_wind2$energy_data..Solar.Energy.Consumption.<-  as.numeric(energy_solar_wind2$energy_data.
energy_solar_wind2$energy_data..Wind.Energy.Consumption.<- as.numeric(energy_solar_wind2$energy_data..W

#change column names
energy_solar_wind2 <- energy_solar_wind2 %>%
       rename("Month" = 1 , "Solar Energy Consumption" = 2, "Wind Energy Consumption" = 3)

head(energy_solar_wind2)
```

```
##          Month Solar Energy Consumption Wind Energy Consumption
## 133 1984-01-01                  -0.001                   0.000
## 134 1984-02-01                   0.001                   0.002
## 135 1984-03-01                   0.002                   0.002
## 136 1984-04-01                   0.003                   0.006
## 137 1984-05-01                   0.007                   0.008
## 138 1984-06-01                   0.010                   0.006
```

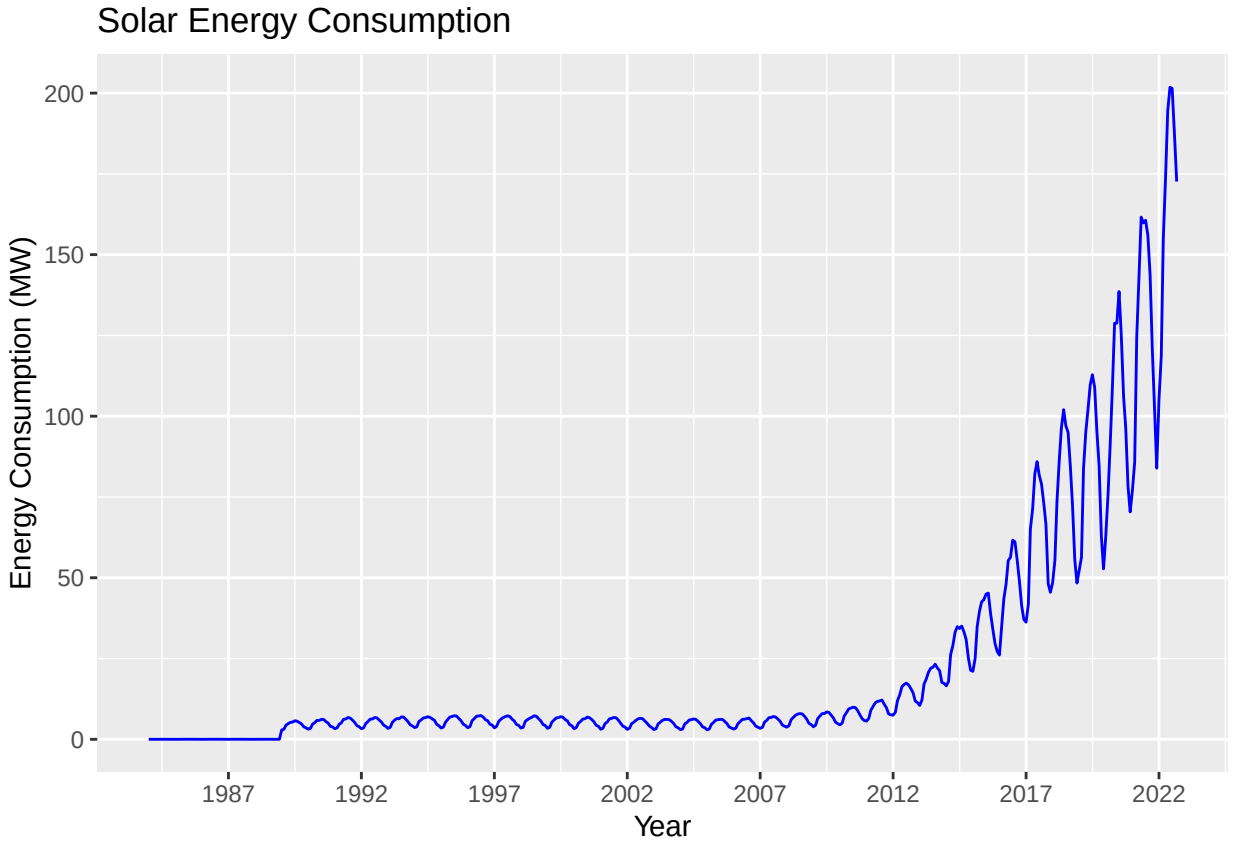
Q2

Plot the Solar and Wind energy consumption over time using ggplot. Plot each series on a separate graph. No need to add legend. Add informative names to the y axis using `ylab()`. Explore the function scale_x_date()

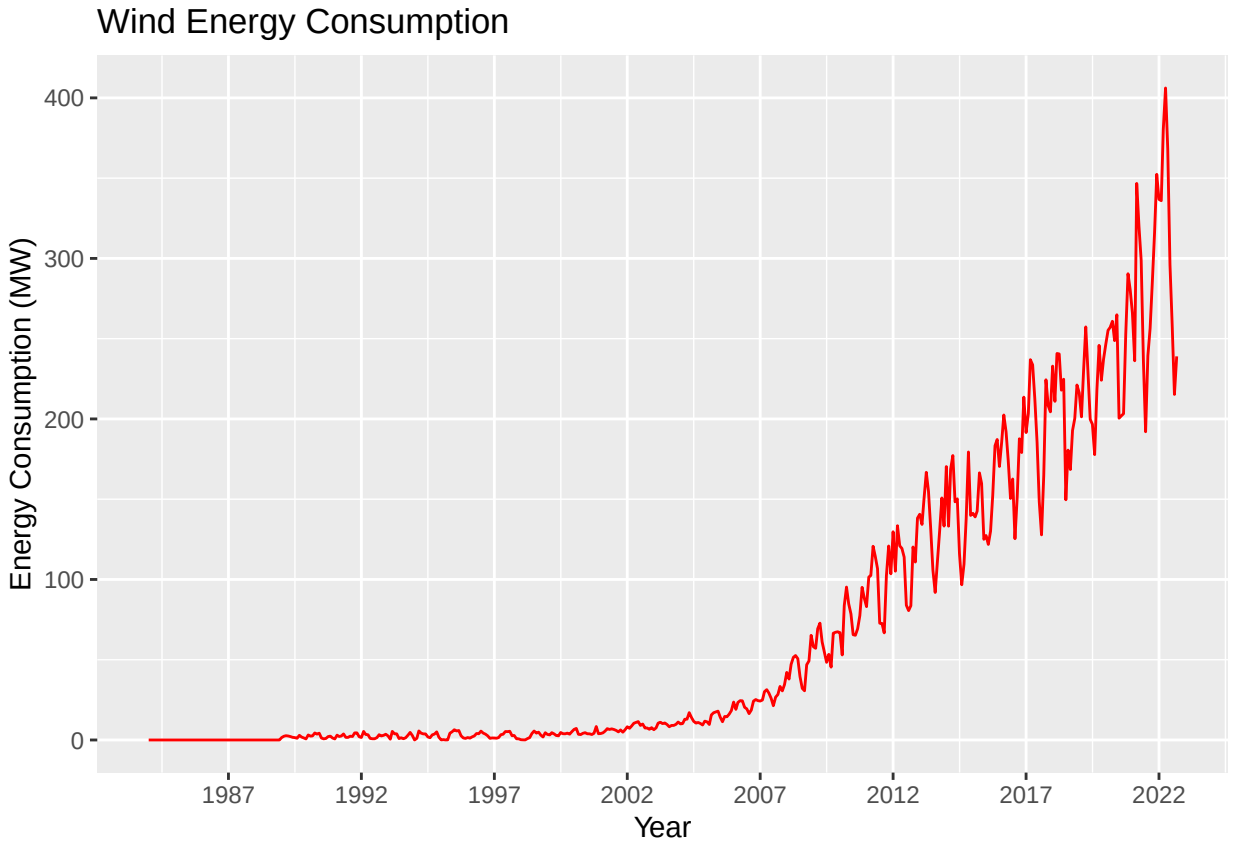
on ggplot and see if you can change the x axis to improve your plot. Hint: use *scale_x_date(date_breaks = "5 years", date_labels = "%Y")")*

```
Date <- as_date(energy_solar_wind2$Month, tz=NULL)

#Solar
ggplot(data = energy_solar_wind2, aes(x=Date, y=energy_solar_wind2[,2]))+
  geom_line(color="blue")+
  ylab("Energy Consumption (MW)")+
  xlab("Year")+
  ggtitle("Solar Energy Consumption")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```




```
#Wind
ggplot(data = energy_solar_wind2, aes(x=Date, y=energy_solar_wind2[,3]))+
  geom_line(color="red")+
  ylab("Energy Consumption (MW)")+
  xlab("Year")+
  ggtitle("Wind Energy Consumption")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```

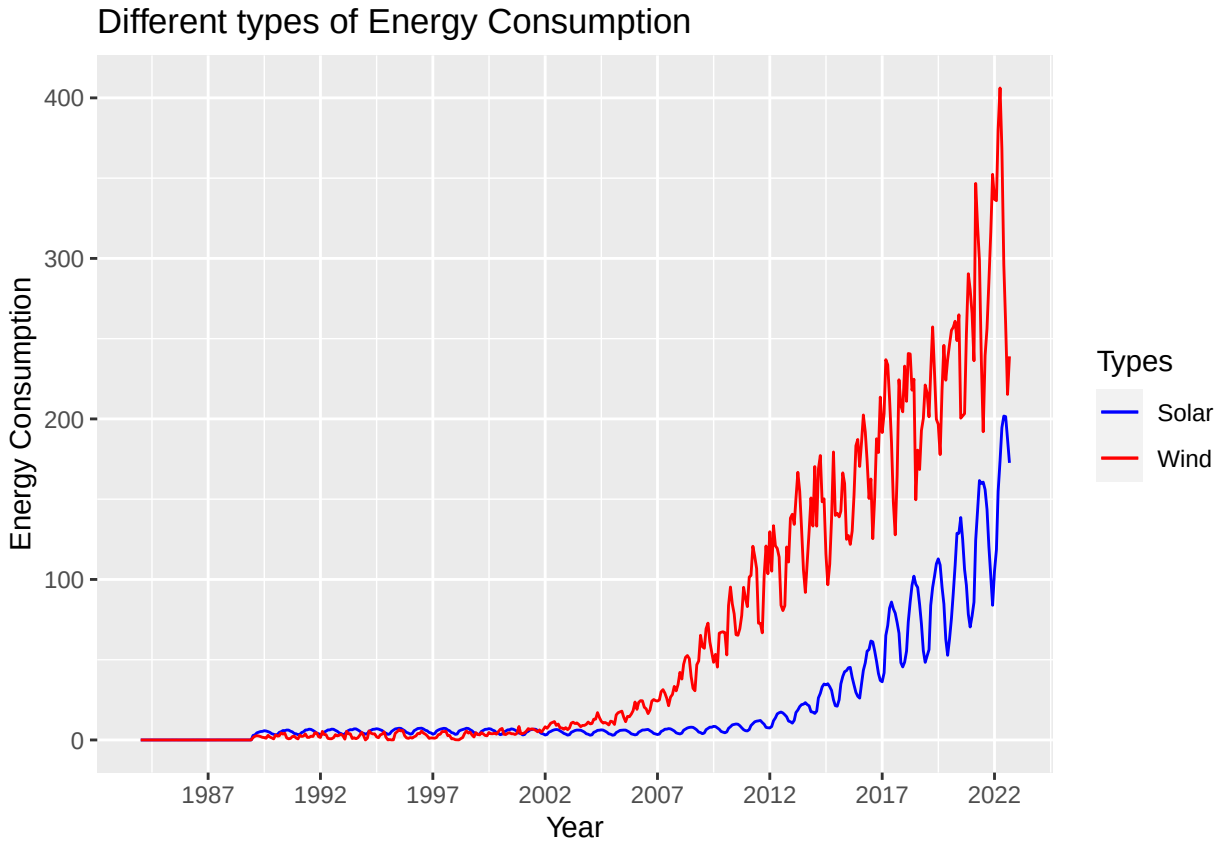


Q3

Now plot both series in the same graph, also using ggplot(). Look at lines 141-148 of the file `M4_OutliersMissingData_Part2_Complete.Rmd` to learn how to manually add a legend to ggplot. Make the solar energy consumption red and wind energy consumption blue. Add informative name to the y axis using `ylab("Energy Consumption)`. And use function scale_x_date() again to improve x axis.

```
Date <- as_date(energy_solar_wind2$Month, tz=NULL)

ggplot(data = energy_solar_wind2, aes(x=Date, y=energy_solar_wind2[,2]))+
  geom_line(aes(x=Date, y=energy_solar_wind2[,2], color="Solar"))+
  geom_line(aes(x=Date, y=energy_solar_wind2[,3], color="Wind"))+
  ylab("Energy Consumption")+
  xlab("Year")+
  ggtitle("Different types of Energy Consumption")+
  scale_color_manual(name = "Types", values= c("Solar" = "blue", "Wind" = "red"),
                     labels=c("Solar", "Wind"))+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



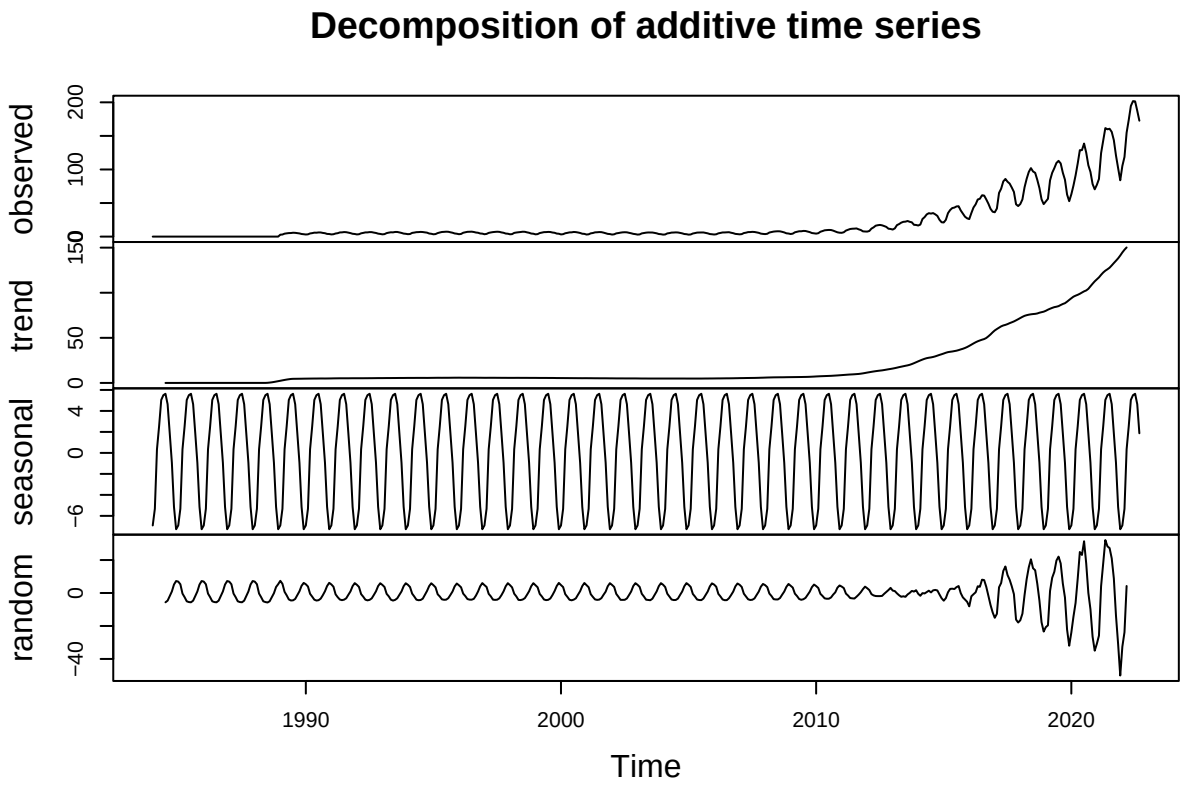
Q3

Transform wind and solar series into a time series object and apply the decompose function on them using the additive option, i.e., `decompose(ts_data, type = "additive")`. What can you say about the trend component? What about the random component? Does the random component look random? Or does it appear to still have some seasonality on it?

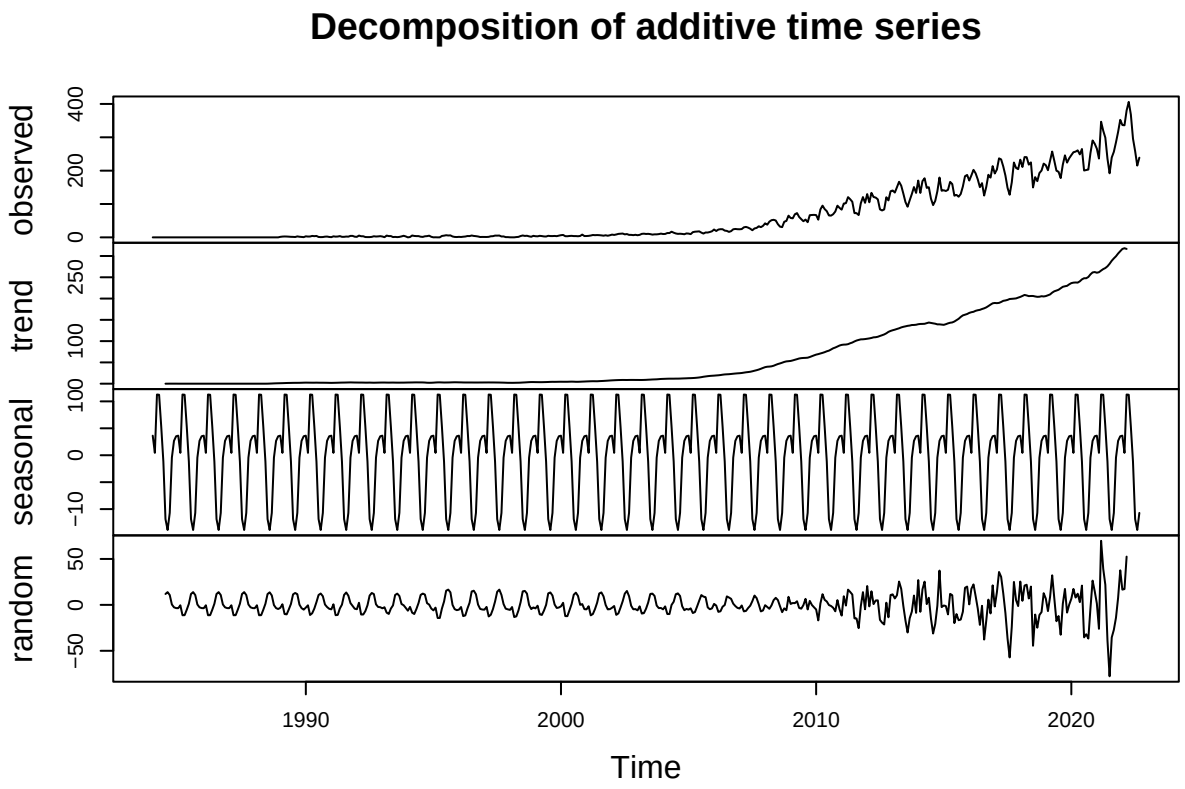
There is an increasing trend in the trend component for both solar and wind energy consumption. For the random component, both graphs show regular movements/constant patterns at the beginning. The pattern starts fluctuate since 2015 for solar energy consumption and since 2008 for wind energy consumption. Therefore, the random component appears to have some seasonality on it at the beginning, but look random in the later years.

```
ts_energy <- ts(energy_solar_wind2[,2:3], start = c(1984,1), frequency =12)

#Solar
decompose_ts_energy_solar <- decompose(ts_energy[,1], type ="additive")
plot(decompose_ts_energy_solar)
```




```
#Wind
decompose_ts_energy_wind <- decompose(ts_energy[,2], type = "additive")
plot(decompose_ts_energy_wind)
```

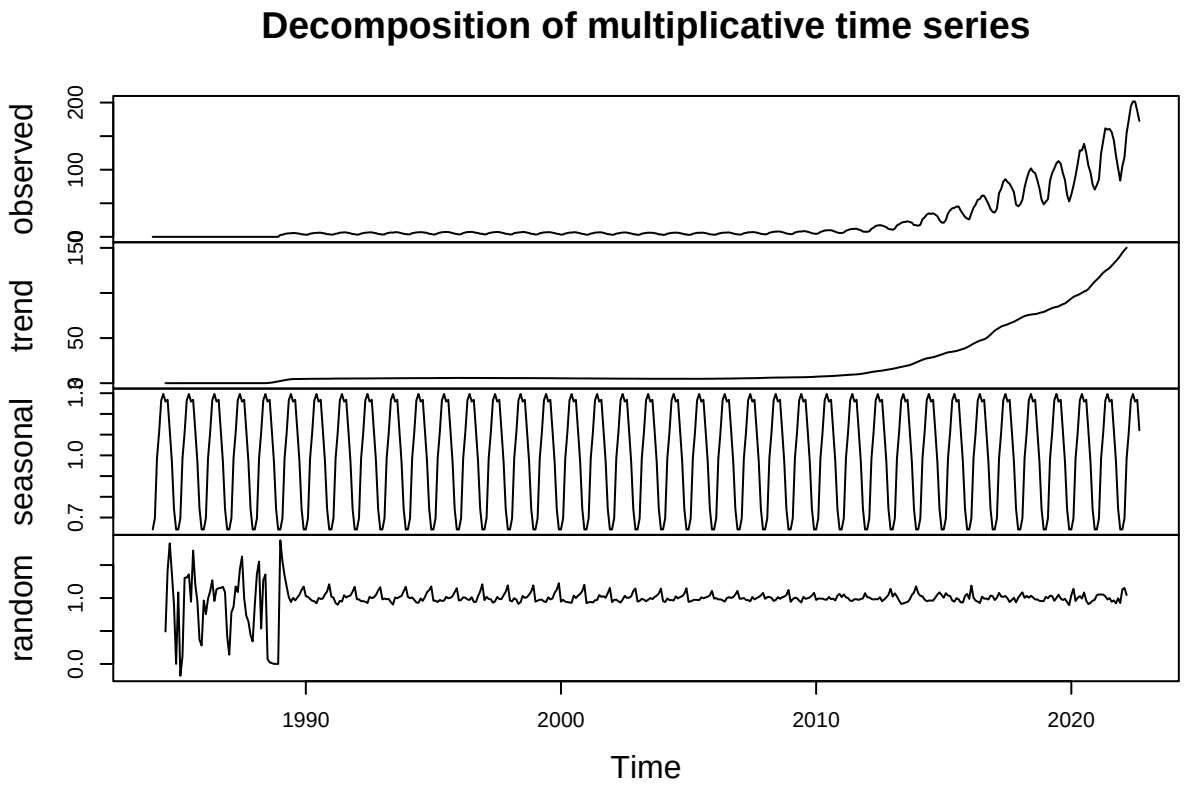


Q4

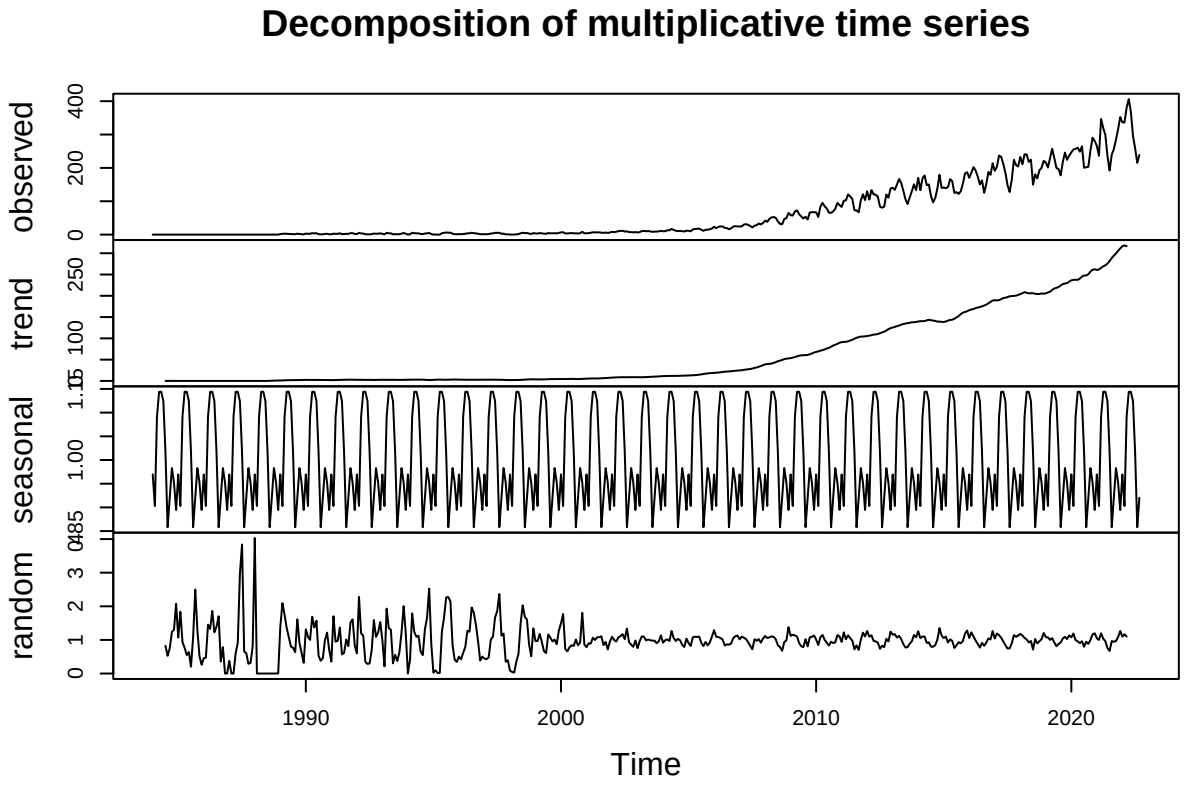
Use the decompose function again but now change the type of the seasonal component from additive to multiplicative. What happened to the random component this time?

The random component has high fluctuation/random at the beginning for solar and wind energy consumption, and it has small fluctuation as time goes on. There is no obvious seasonality on the random component.

```
#Solar
decompose_ts_energy_solar <- decompose(ts_energy[,1], type ="multiplicative")
plot(decompose_ts_energy_solar)
```




```
#Wind
decompose_ts_energy_wind <- decompose(ts_energy[,2], type = "multiplicative")
plot(decompose_ts_energy_wind)
```



Q5

When fitting a model to this data, do you think you need all the historical data? Think about the data from 90s and early 20s. Are there any information from those years we might need to forecast the next six months of Solar and/or Wind consumption. Explain your response.

> Answer: I don't think we need all the historical data. We can see that there are a lack of data or the values are low from 90s for both solar and wind energy consumption. Therefore, it is not that significant to use these data (90s) to forecast the next six months of solar and/or wind consumption.

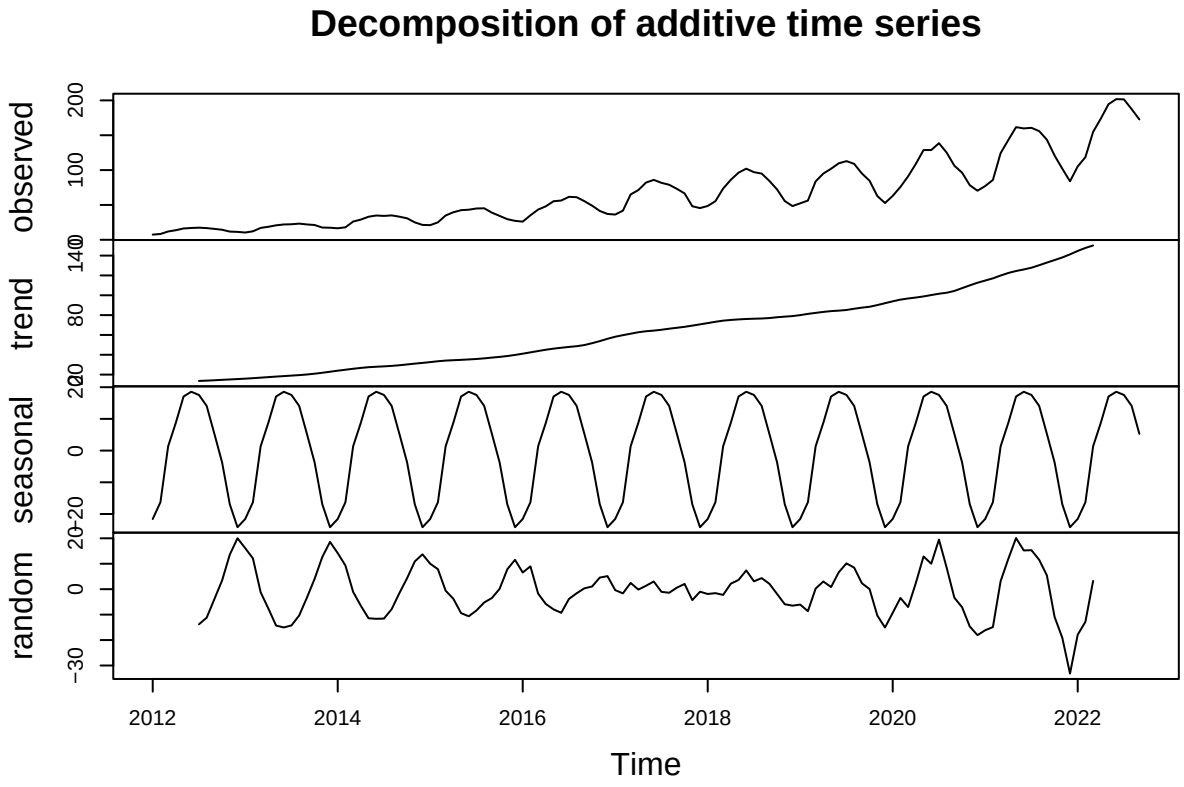
Q6

Create a new time series object where historical data starts on January 2012. Hint: use `filter()` function so that you don't need to point to row numbers, .i.e, `filter(xxxx, year(Date) >= 2012 )`. Apply the decompose function `type=additive` to this new time series. Comment the results. Does the random component look random? Think about our discussion in class about seasonal components that depends on the level of the series.

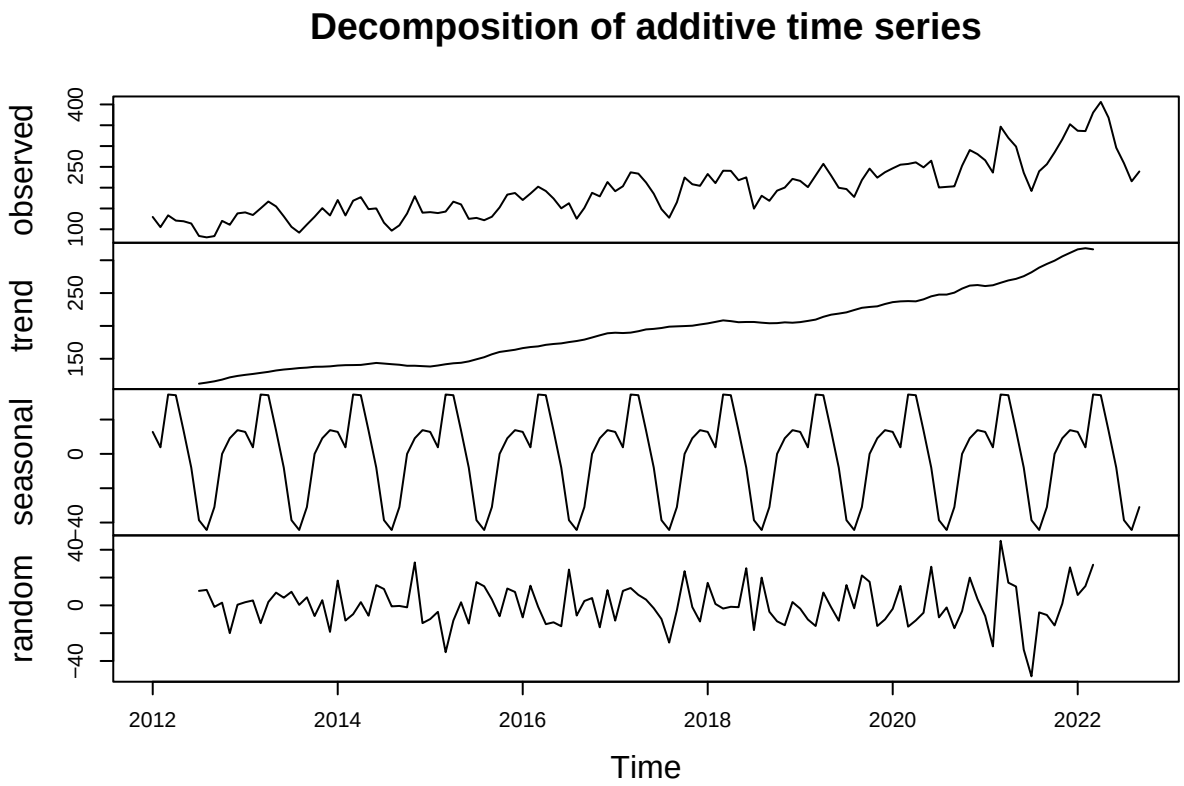
```
energy_2012 <- filter(energy_solar_wind2, year(Month) >= 2012)

#time series
ts_energy_2012 <- ts(energy_2012[,2:3], start = c(2012,1), frequency = 12)

#Solar
decompose_ts_energy_solar_2012 <- decompose(ts_energy_2012[,1], type ="additive")
plot(decompose_ts_energy_solar_2012)
```




```
#Wind
decompose_ts_energy_wind_2012 <- decompose(ts_energy_2012[,2], type = "additive")
plot(decompose_ts_energy_wind_2012)
```



Answer: We saw an increasing trend in the trend component and seasonality in the seasonal

component for both series. The random component seems random, and the magnitude of the seasonal fluctuations does not vary with the level of time series.