ENV 790.30 - Time Series Analysis for Energy Data | Spring 2022

Assignment 5 - Due date 02/28/22

Colin Lee

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 project open the first thing you will do is change "Student Name" on line 3 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. Rename the pdf file such that it includes your first and last name (e.g., "LuanaLima_TSA_A05_Sp22.Rmd"). Submit this pdf using Sakai.

R packages needed for this assignment are listed below. 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(xlsx)
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.1.2

## 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 1.3.1 --

## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
## v purrr   0.3.4

## -- 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()
```

Decomposing Time Series

Consider the same data you used for A04 from the spreadsheet "Table_10.1_Renewable_Energy_Production_and_Consump
The data comes from the US Energy Information and Administration and corresponds to the January 2021 Monthly Energy Review.

```
#Importing data set - using xlsx package
energy_data <- read.xlsx(file="/Users/colinlee/Documents/Duke/Spring 2022/ENV790/ENV790_TimeSeriesAnaly

#Now let's extract the column names from row 11 only
read_col_names <- read.xlsx(file="/Users/colinlee/Documents/Duke/Spring 2022/ENV790/ENV790_TimeSeriesAna

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

```
##        Month Wood Energy Production Biofuels Production
## 1 1973-01-01                129.630       Not Available
## 2 1973-02-01                117.194       Not Available
## 3 1973-03-01                129.763       Not Available
## 4 1973-04-01                125.462       Not Available
## 5 1973-05-01                129.624       Not Available
## 6 1973-06-01                125.435       Not Available
##   Total Biomass Energy Production Total Renewable Energy Production
## 1                         129.787                           403.981
## 2                         117.338                           360.900
## 3                         129.938                           400.161
## 4                         125.636                           380.470
## 5                         129.834                           392.141
## 6                         125.611                           377.232
##   Hydroelectric Power Consumption Geothermal Energy Consumption
## 1                         272.703                         1.491
## 2                         242.199                         1.363
## 3                         268.810                         1.412
## 4                         253.185                         1.649
## 5                         260.770                         1.537
## 6                         249.859                         1.763
```

```
##   Solar Energy Consumption Wind Energy Consumption Wood Energy Consumption
## 1            Not Available           Not Available                 129.630
## 2            Not Available           Not Available                 117.194
## 3            Not Available           Not Available                 129.763
## 4            Not Available           Not Available                 125.462
## 5            Not Available           Not Available                 129.624
## 6            Not Available           Not Available                 125.435
##   Waste Energy Consumption Biofuels Consumption
## 1                    0.157        Not Available
## 2                    0.144        Not Available
## 3                    0.176        Not Available
## 4                    0.174        Not Available
## 5                    0.210        Not Available
## 6                    0.176        Not Available
##   Total Biomass Energy Consumption Total Renewable Energy Consumption
## 1                          129.787                            403.981
## 2                          117.338                            360.900
## 3                          129.938                            400.161
## 4                          125.636                            380.470
## 5                          129.834                            392.141
## 6                          125.611                            377.232
```

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

Q1

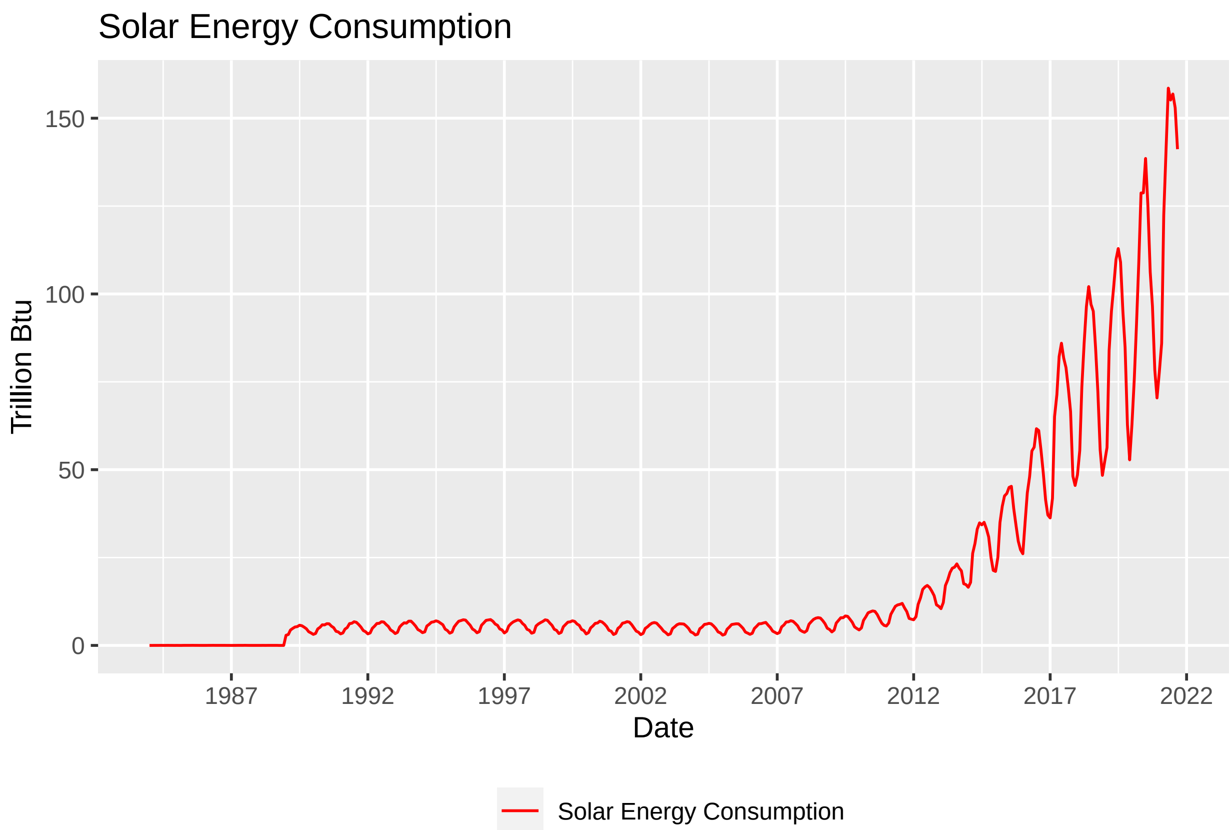
For this assignment you will work only with the following columns: Solar Energy Consumption and Wind Energy Consumption. Create a data frame structure with these two time series only and the Date column. Drop the rows with *Not Available* and convert the columns to numeric. You can use filtering to eliminate 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!

```
t <- c(1:nrow(energy_data))

solar_wind_data <- energy_data[,8:9] %>%
  cbind("Month" = ((energy_data[,1])))


for (i in t) {
  if ((solar_wind_data[i,1]) == "Not Available")
  solar_wind_data[i,1] = NA

  if ((solar_wind_data[i,2]) == "Not Available")
  solar_wind_data[i,2] = NA
}


clean_solar_wind <- na.omit(solar_wind_data)
```

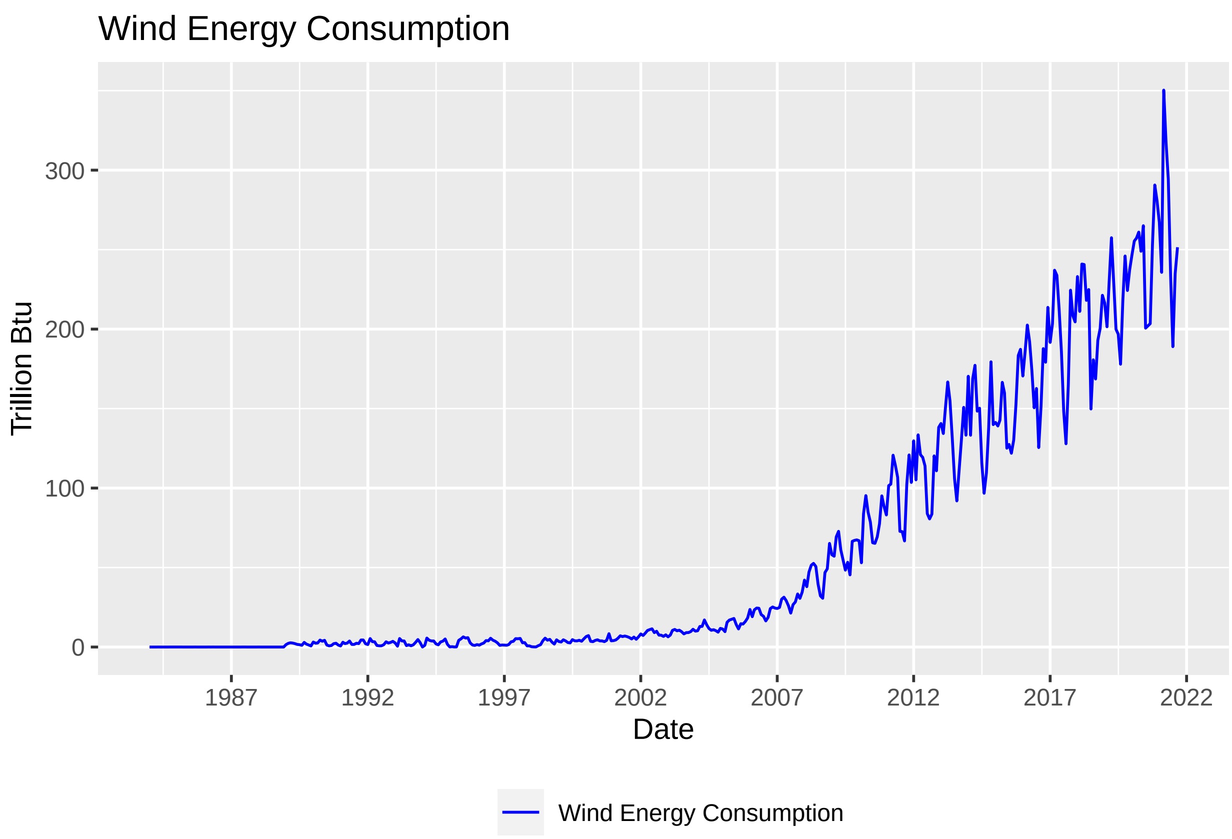
Q2

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")")*

```
ggplot(clean_solar_wind) +
  geom_line(aes(x = as.Date(clean_solar_wind[,3],origin = "1984-01-00"), y = (as.numeric(clean_solar_wi
  labs(y = "Trillion Btu",
            x = "Date",
            title = "Solar Energy Consumption",color="") +
  scale_color_manual(values = c("1" = "red"),
                              labels=c("Solar Energy Consumption")) +
  theme(legend.position = "bottom") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



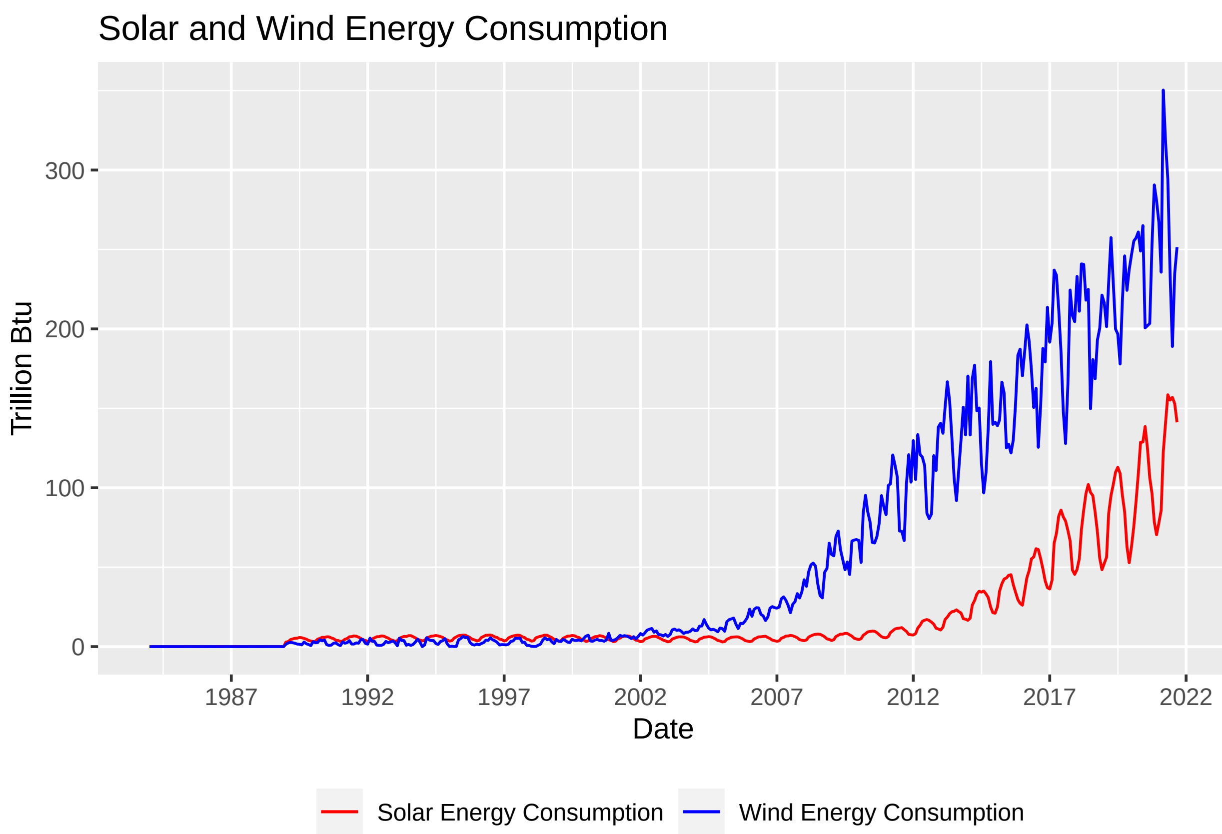

```
ggplot(clean_solar_wind) +
  geom_line(aes(x = as.Date(clean_solar_wind[,3],origin = "1984-01-00"), y = (as.numeric(clean_solar_wi
  labs(y = "Trillion Btu",
            x = "Date",
            title = "Wind Energy Consumption",color="") +
  scale_color_manual(values = c("2" = "blue"),
                              labels=c("Wind Energy Consumption")) +
  theme(legend.position = "bottom") +
  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 142-149 of the file `05_Lab_OutliersMissingData_Solution` 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.

```
ggplot(clean_solar_wind) +
  geom_line(aes(x = as.Date(clean_solar_wind[,3],origin = "1984-01-00"), y = (as.numeric(clean_solar_wi
  geom_line(aes(x = as.Date(clean_solar_wind[,3],origin = "1984-01-00"), y = (as.numeric(clean_solar_wi
  labs(y = "Trillion Btu",
            x = "Date",
            title = "Solar and Wind Energy Consumption",color="") +
  scale_color_manual(values = c("1" = "red", "2" = "blue"),
                                labels=c("Solar Energy Consumption", "Wind Energy Consumption")) +
  theme(legend.position = "bottom") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



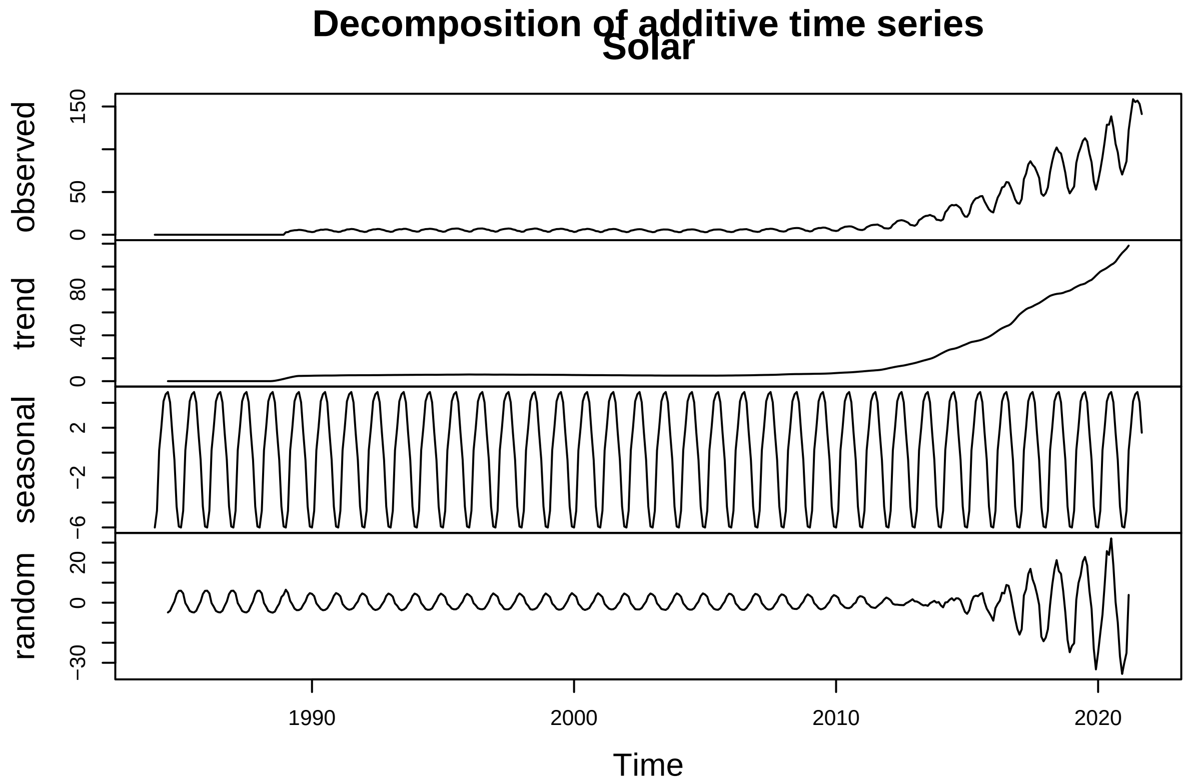
Q4

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?

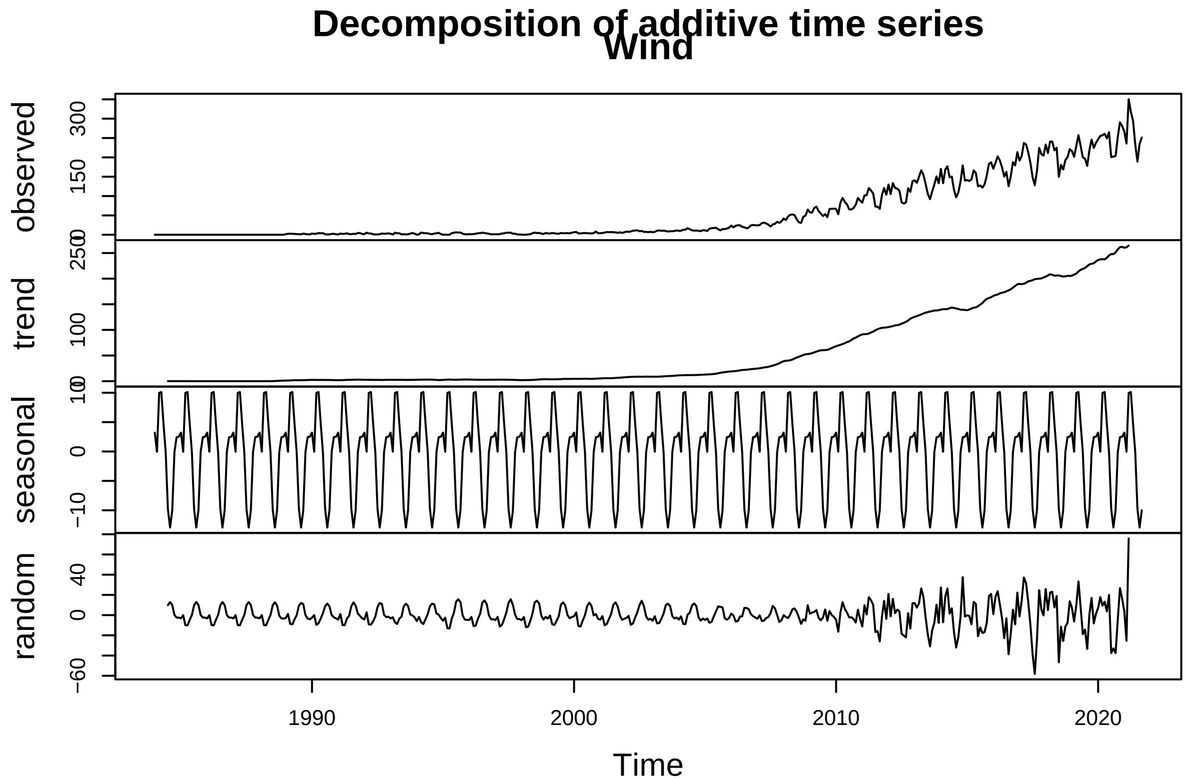
For the trend component for both solar and wind, there is a clearly increasing trend. However, this trend does not start (is not very easily visible) until the late 2000s for wind and mid 2010s for solar.

For the random component for both solar and wind, we can see an increasing level of randomness over time. The random component does actually look like it has some sort of repetition that resembles seasonality.

```
#ts_solar_wind <- ts(as.numeric(clean_solar_wind[,1:2]), start = c(1984,1), frequency = 12)

ts_solar <- ts(as.numeric(clean_solar_wind[,1]),start = c(1984,1), frequency = 12)
ts_wind <- ts(as.numeric(clean_solar_wind[,2]),start = c(1984,1), frequency = 12)
decompose_solar <- decompose(ts_solar,type = "additive")
decompose_wind <- decompose(ts_wind,type = "additive")


plot(decompose_solar)
title(main = "Solar")
```




```
plot(decompose_wind)
title(main = "Wind")
```

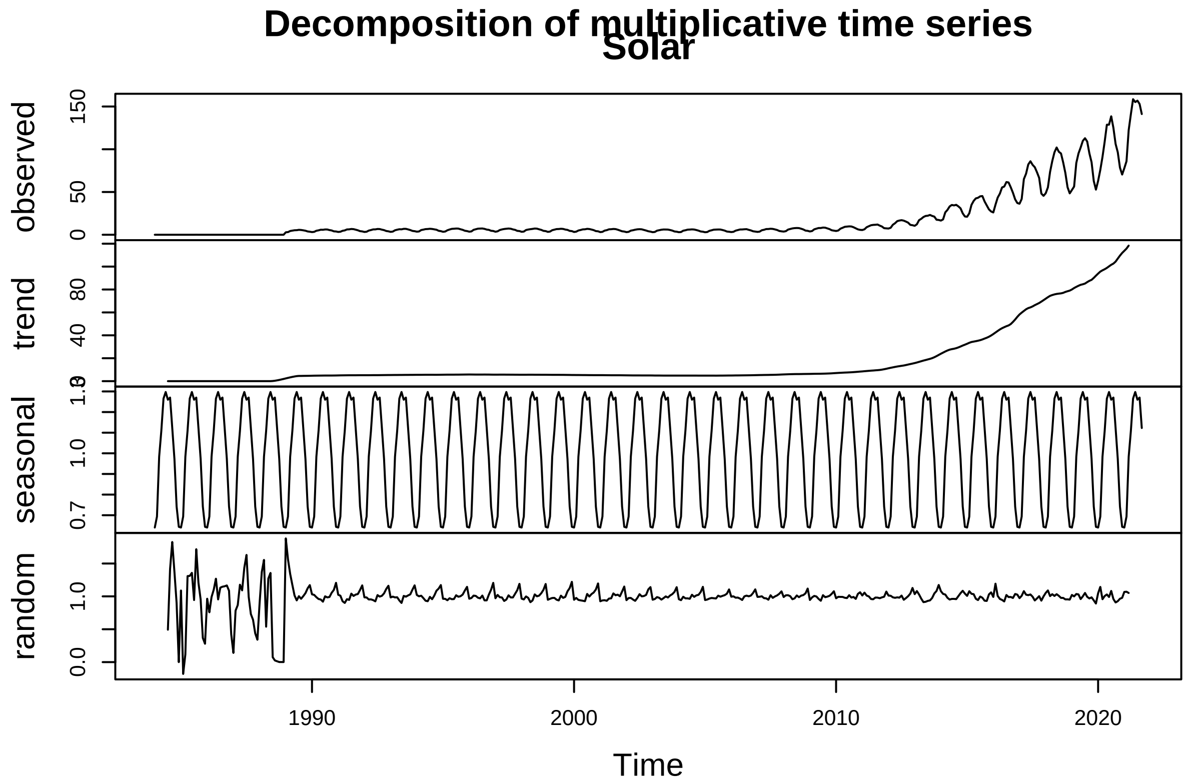


Q5

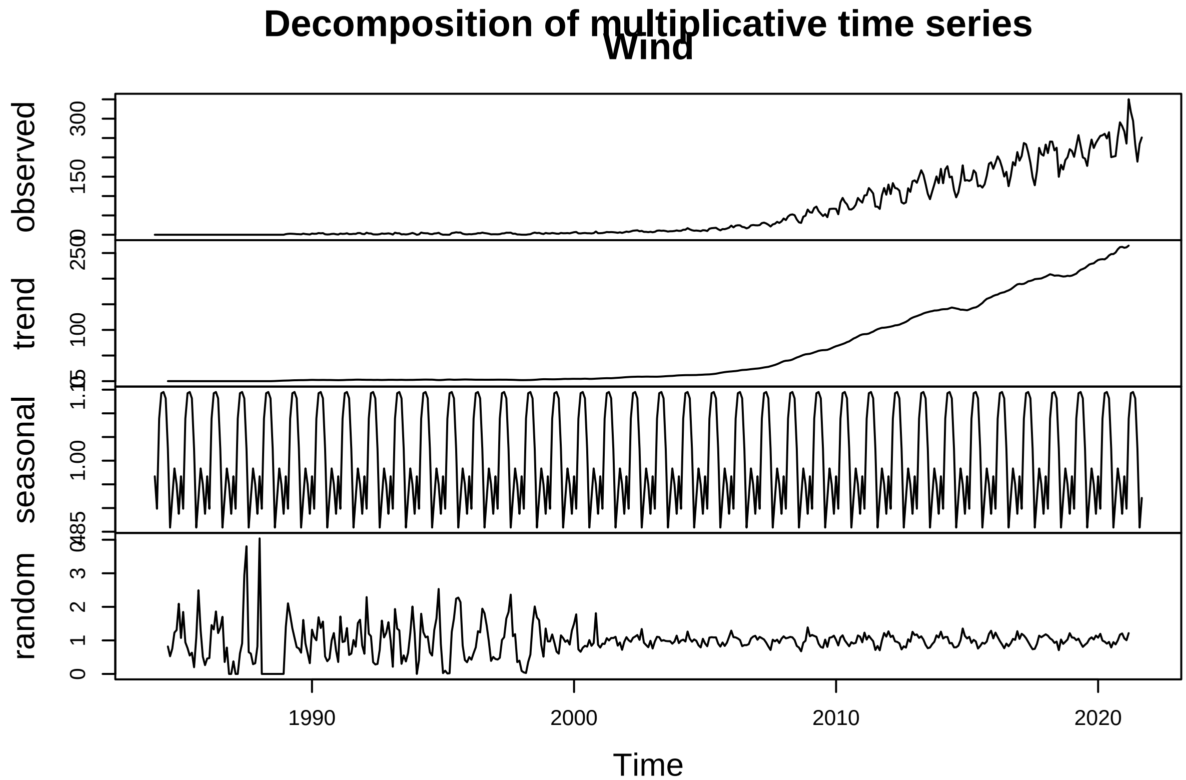
Use the decompose function again but now change the type of the seasonal component from additive to multiplicative. What happened to the random component this time?

The random component is now much more random until about 1990 for solar and early 2000s for wind, where they do not follow the seasonal pattern it did in the additive decomposition. However, after those years, there still appears to be some small patterning, not as obvious as with the addititive series.

```
decompose_solar_m <- decompose(ts_solar,type = "multiplicative")
decompose_wind_m <- decompose(ts_wind,type = "multiplicative")


plot(decompose_solar_m)
title(main = "Solar")
```




```
plot(decompose_wind_m)
title(main = "Wind")
```



Q6

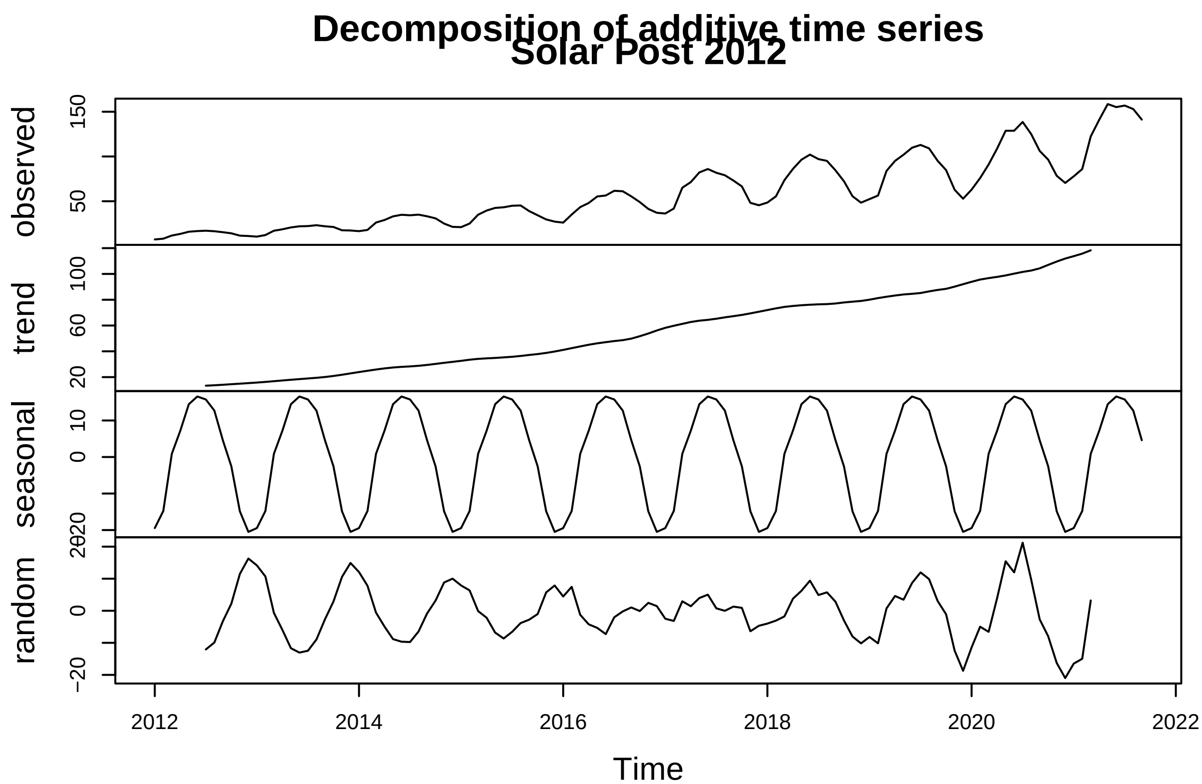
When fitting a model to this data, do you think you need all the historical data? Think about the data from 90s and early 20s. Are there any information from those years we might need to forecast the next six months of Solar and/or Wind consumption. Explain your response.

> Answer: I don't think we need all the historical data to forecast the next six months of solar and/or wind energy consumption. This is because for a lot of these earlier time periods, there was not a lot of renewable energy being generated due to small availability of the technology. Only in the last two decades has solar technology become much more available for commercial use where solar is more prevalent and we can more accurately forecast the future of solar growth. Growth before the mid-late 2010s is not too important because the technology was in very early phases. For wind, we could use a little more historical data going into the 2010s because the wind turbine technology was developed and advanced earlier.

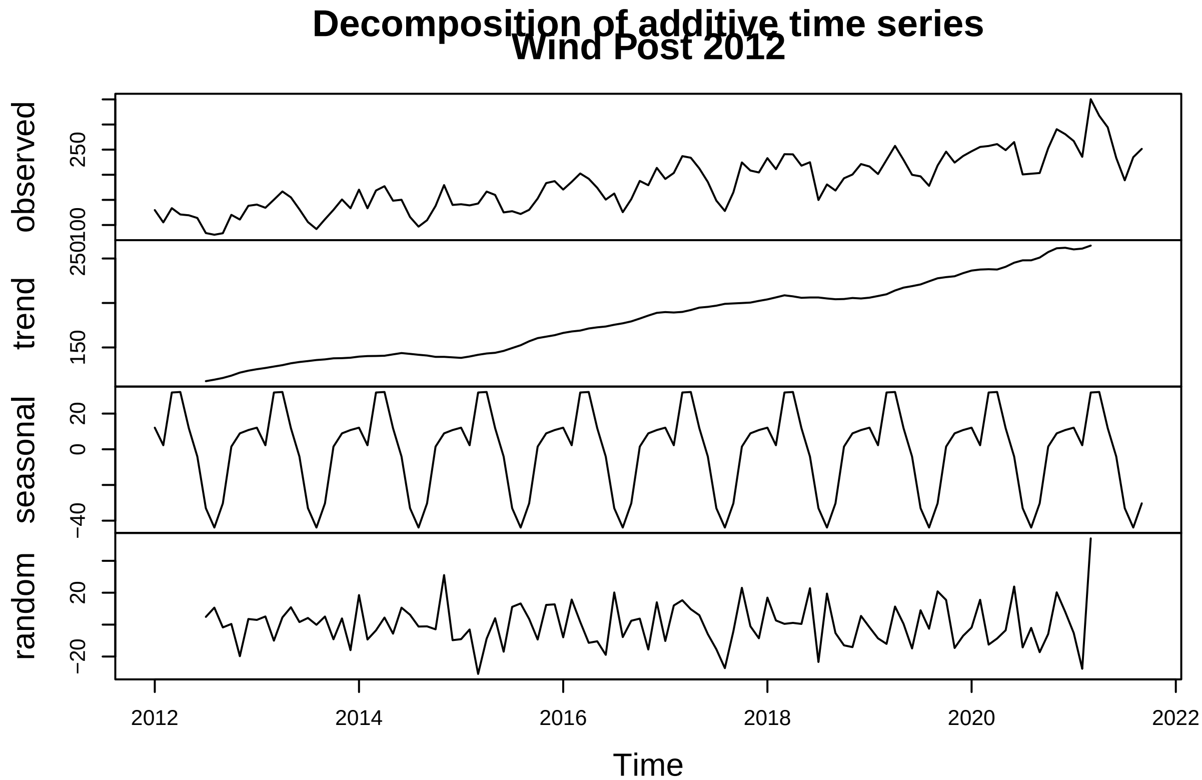
Q7

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 trying to remove the seasonal component and the challenge of trend on the seasonal component.

```r
solar_wind_new <- filter(clean_solar_wind, year(Month) >= 2012)

ts_solar_new <- ts(as.numeric(solar_wind_new[,1]),start = c(2012,1), frequency = 12)
ts_wind_new <- ts(as.numeric(solar_wind_new[,2]),start = c(2012,1), frequency = 12)
decompose_solar_new <- decompose(ts_solar_new,type = "additive")
decompose_wind_new <- decompose(ts_wind_new,type = "additive")


plot(decompose_solar_new)
title(main = "Solar Post 2012")
```




```r
plot(decompose_wind_new)
title(main = "Wind Post 2012")
```



Answer: The random component for solar looks somewhat seasonal, with some randomness between 2016 and 2018. However, the random component for wind is much more random in comparison. With the trend, we can see a more steady increase in the trend for both, instead of decades of flat/no trend before large increases. The difficulty of having an actually random component look random might have been due to the non-uniform trend in the original time series. Over time, the trend changes significantly, making removing it alongside the seasonal component to form the random component still results in some seasonal patterns. Another source for the problem may be a stochastic seasonal trend requiring differencing to get fully removed.