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

Assignment 5 - Due date 02/27/23

Kassie Huang

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()
```

Decomposing Time Series

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

```
#Importing data set - using xlsx package
library(readxl)
energy_data <- read_excel("./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xlsx
```

```
## 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 only
read_col_names <- read_excel("./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xl
```

```
## 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 the data frame
df_data <- data.frame(energy_data$Month, as.numeric(energy_data$`Solar Energy Consumption`),as.numeric(
```

```
## Warning in data.frame(energy_data$Month, as.numeric(energy_data$`Solar Energy
## Consumption`), : NAs introduced by coercion

## Warning in data.frame(energy_data$Month, as.numeric(energy_data$`Solar Energy
## Consumption`), : NAs introduced by coercion
```

```
colnames(df_data) <- c('Date','Solar','Wind')

#inspect the NAs
sum(is.na(df_data$Solar))
```

```
## [1] 132
```

```
sum(is.na(df_data$Wind))
```

```
## [1] 120
```

```
#drop the NA rows
cleaned_df <- drop_na(df_data)
#or: cleaned_df <- filter(df_data, !is.na(Solar) & !is.na(Wind))

#inspect the NAs again
sum(is.na(cleaned_df$Solar))
```

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

```
sum(is.na(cleaned_df$Wind))
```

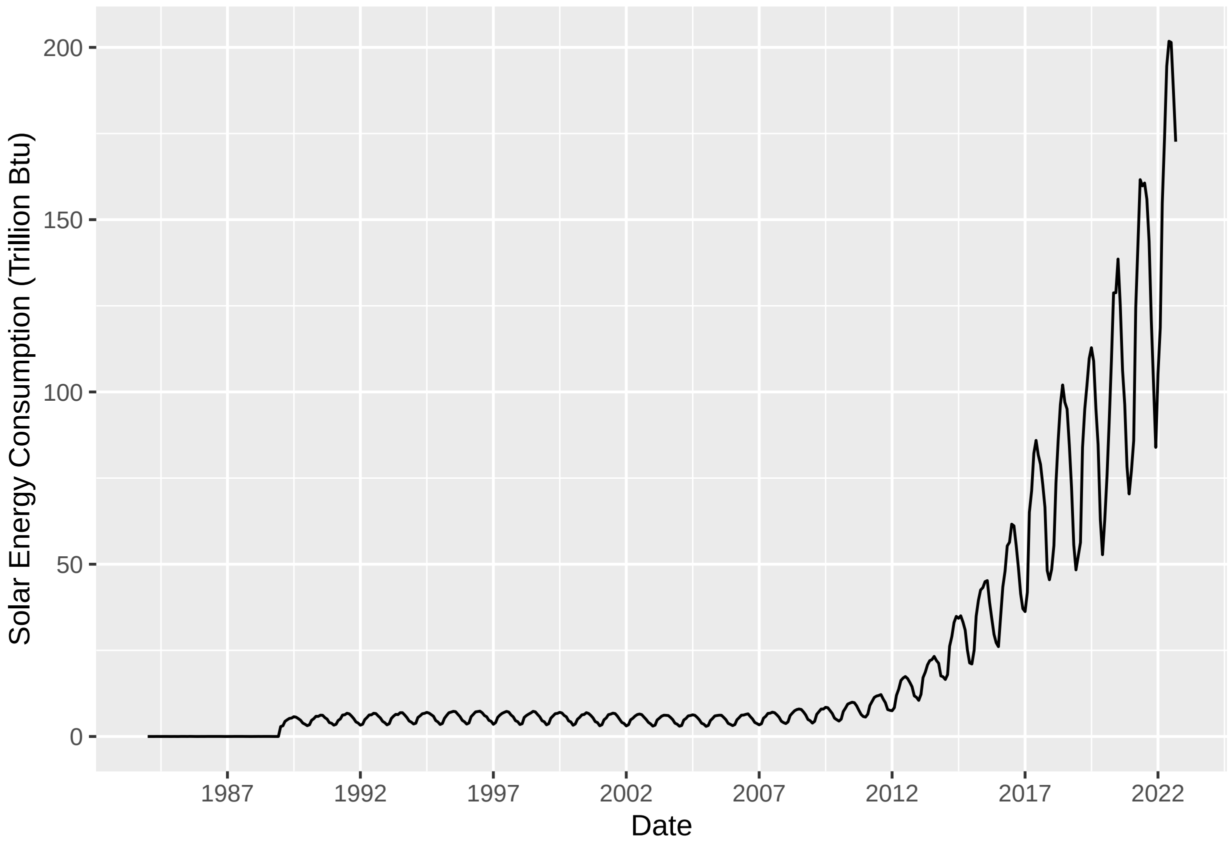
```
## [1] 0
```

Q2

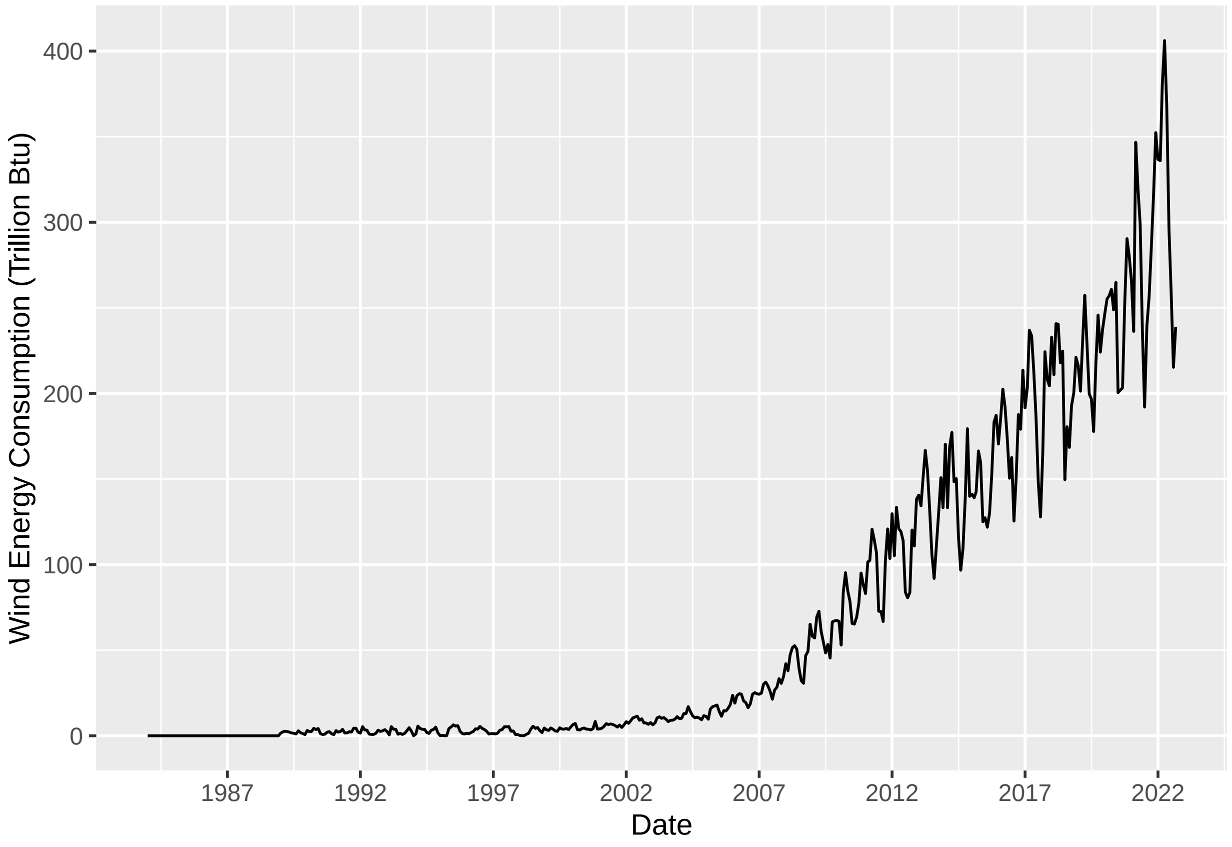
Plot the Solar and Wind energy consumption over time using ggplot. Plot each series on a separate graph. No need to add legend. Add informative names to the y axis using `ylab()`. Explore the function scale_x_date() on ggplot and see if you can change the x axis to improve your plot. Hint: use *scale_x_date(date_breaks = "5 years", date_labels = "%Y")")*

```
#convert to time series data
cleaned_df$Date <- ymd(cleaned_df$Date)

#plot the data
print(ggplot(data = cleaned_df, aes(x=Date))+
        geom_line(aes(y=Solar))+
        ylab("Solar Energy Consumption (Trillion Btu)")+
        scale_x_date(date_breaks = "5 years", date_labels = "%Y"))
```



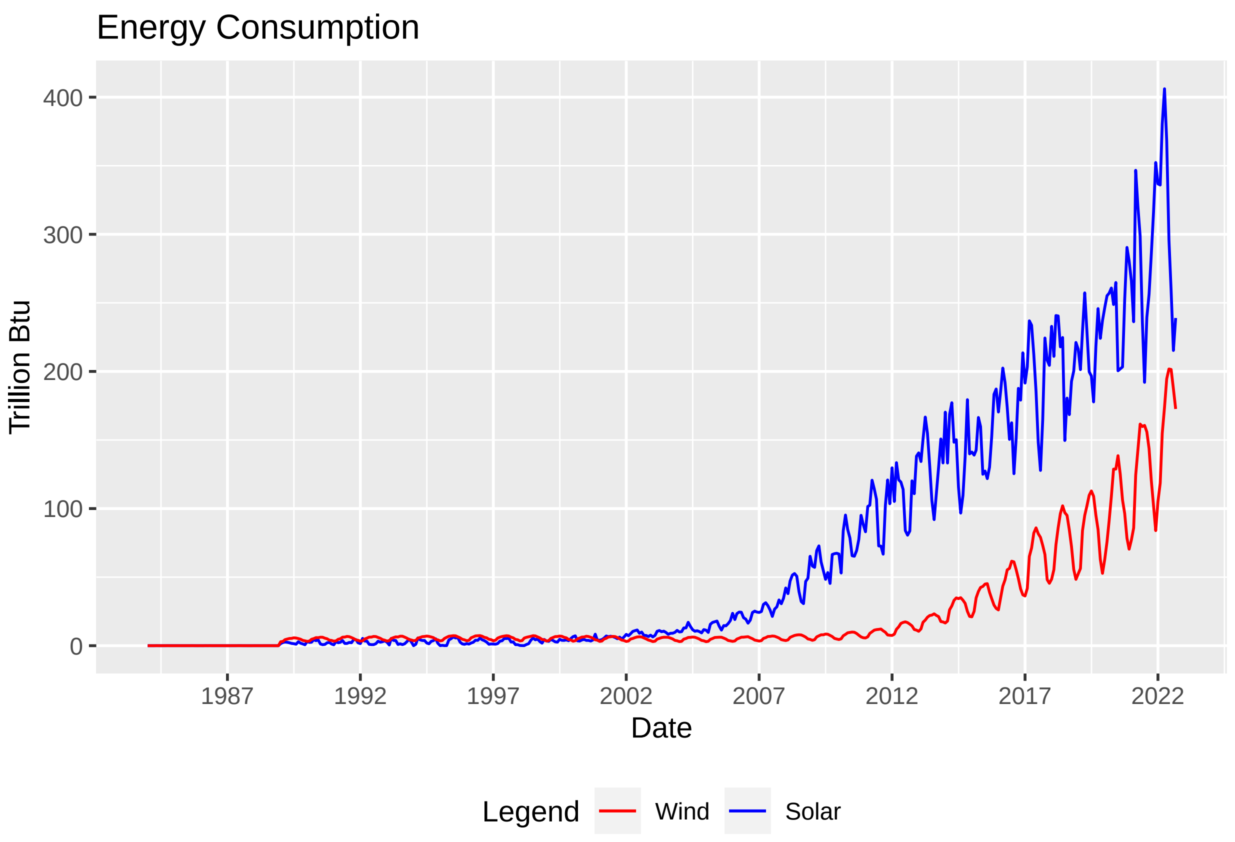
```
print(ggplot(data = cleaned_df, aes(x=Date))+
        geom_line(aes(y=Wind))+
        ylab("Wind Energy Consumption (Trillion Btu)")+
        scale_x_date(date_breaks = "5 years", date_labels = "%Y"))
```



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.

```
print(ggplot(data = cleaned_df)+
        geom_line(aes(x = Date, y = Wind, color = "Wind")) +
        geom_line(aes(x = Date, y = Solar, color = "Solar")) +
        ylab("Trillion Btu")+
        ggtitle('Energy Consumption')+
        scale_x_date(date_breaks = "5 years", date_labels = "%Y")+
        scale_color_manual(name='Legend',
                           values=c("Wind"='blue',"Solar"='red'),
                           labels=c('Wind','Solar'))+
        theme(legend.position = "bottom")
      )
```

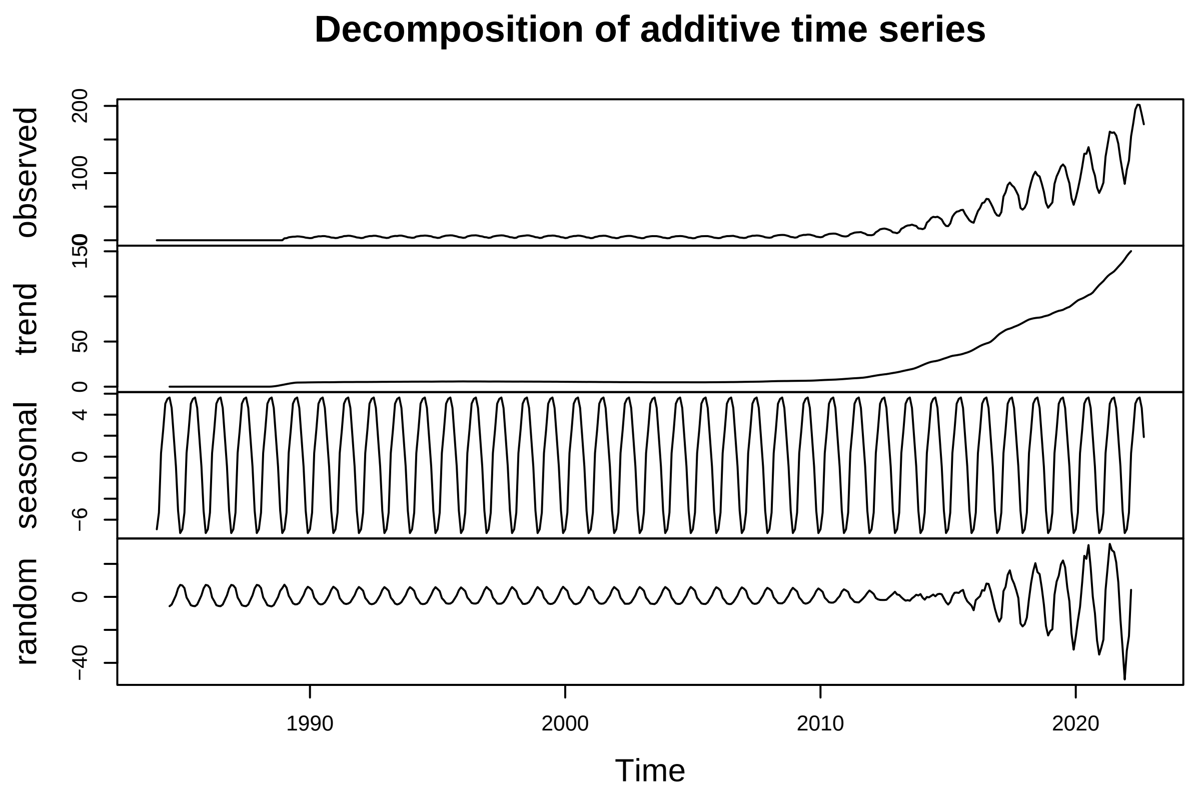


Q3

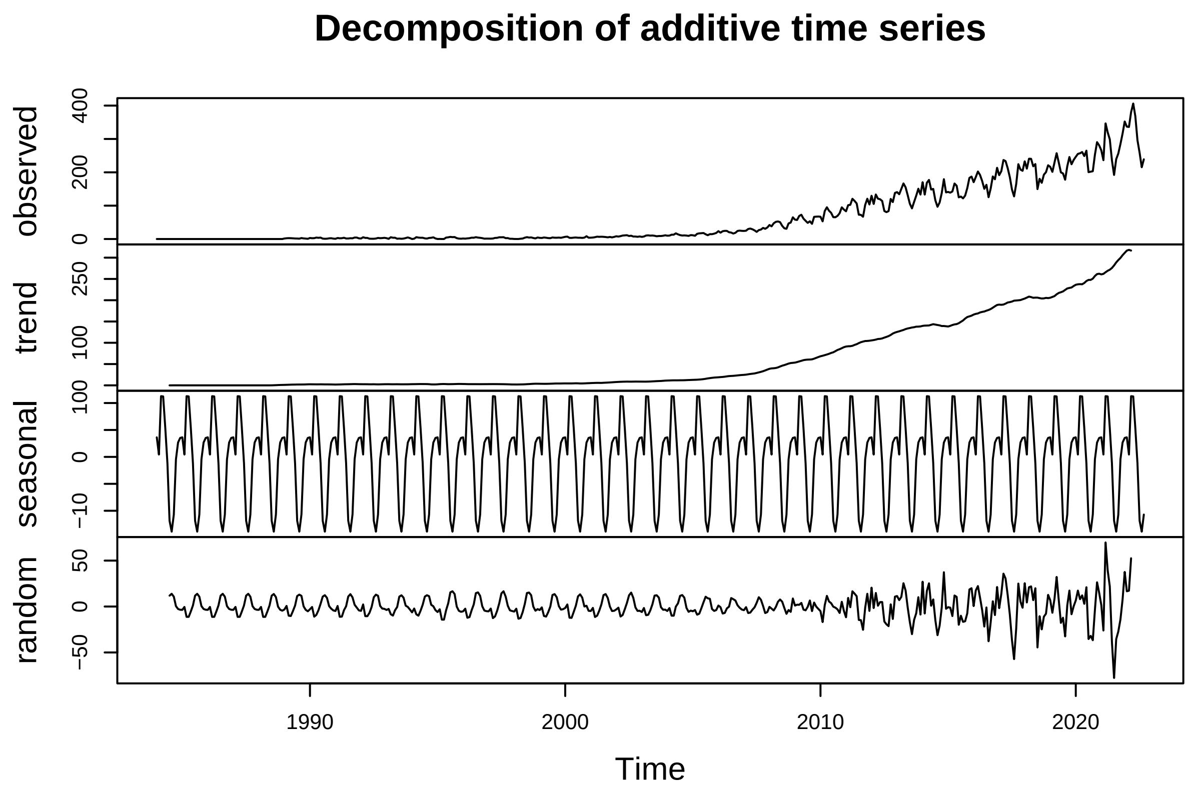
Transform wind and solar series into a time series object and apply the decompose function on them using the additive option, i.e., `decompose(ts_data, type = "additive")`. What can you say about the trend component? What about the random component? Does the random component look random? Or does it appear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

```
ts_solar <- ts(cleaned_df$Solar,frequency = 12,start=c(1984,1))
ts_wind <- ts(cleaned_df$Wind,frequency = 12,start=c(1984,1))

decompose_solar_add <- decompose(ts_solar, type = "additive")
plot(decompose_solar_add)
```



```
decompose_wind_add <- decompose(ts_wind, type = "additive")
plot(decompose_wind_add)
```



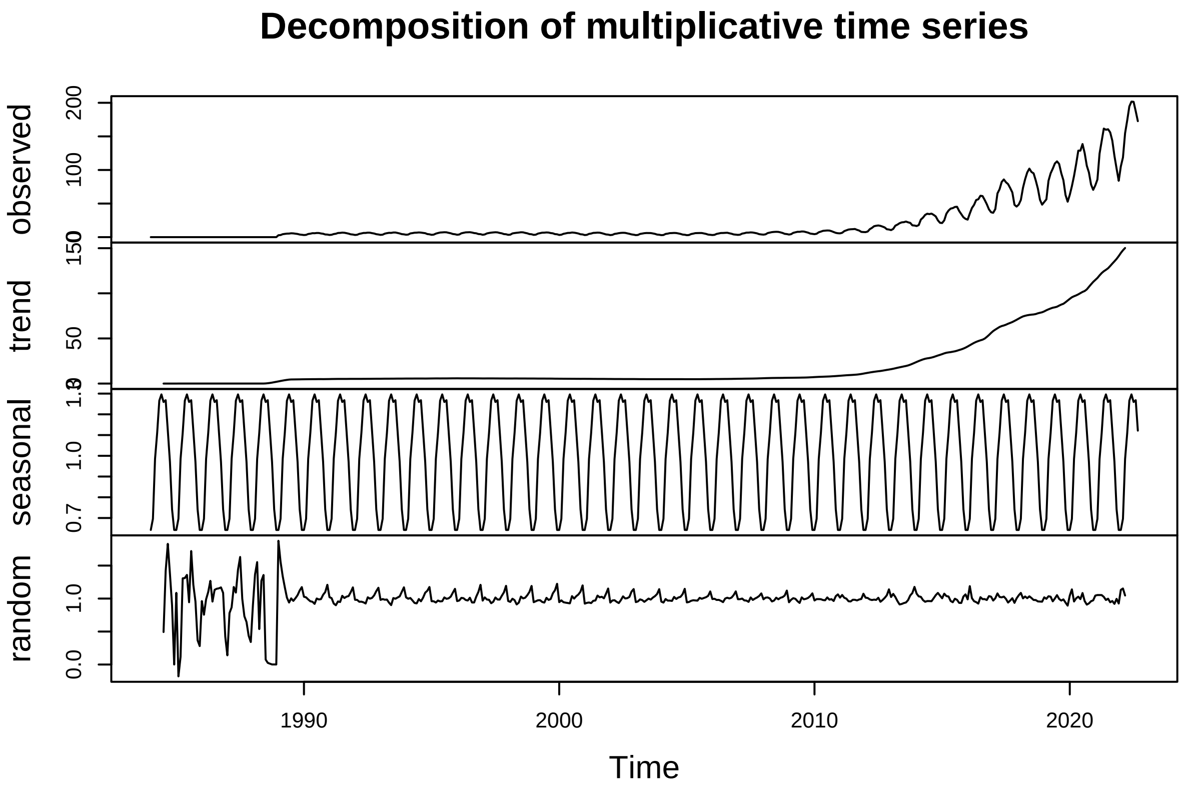
Trend Component: Both the wind and solar series remained stable until 2005, when they began to increase.

Random component: The random component of the solar series doesn't look "random", but follows a very regular seasonal wave pattern. The magnitude of the wave slowly decreased from the beginning until ~2015, and then began to increase. The random component of the wind series also appears to be seasonal, with the magnitude decreasing slowly until 2010 and increasing rapidly thereafter.

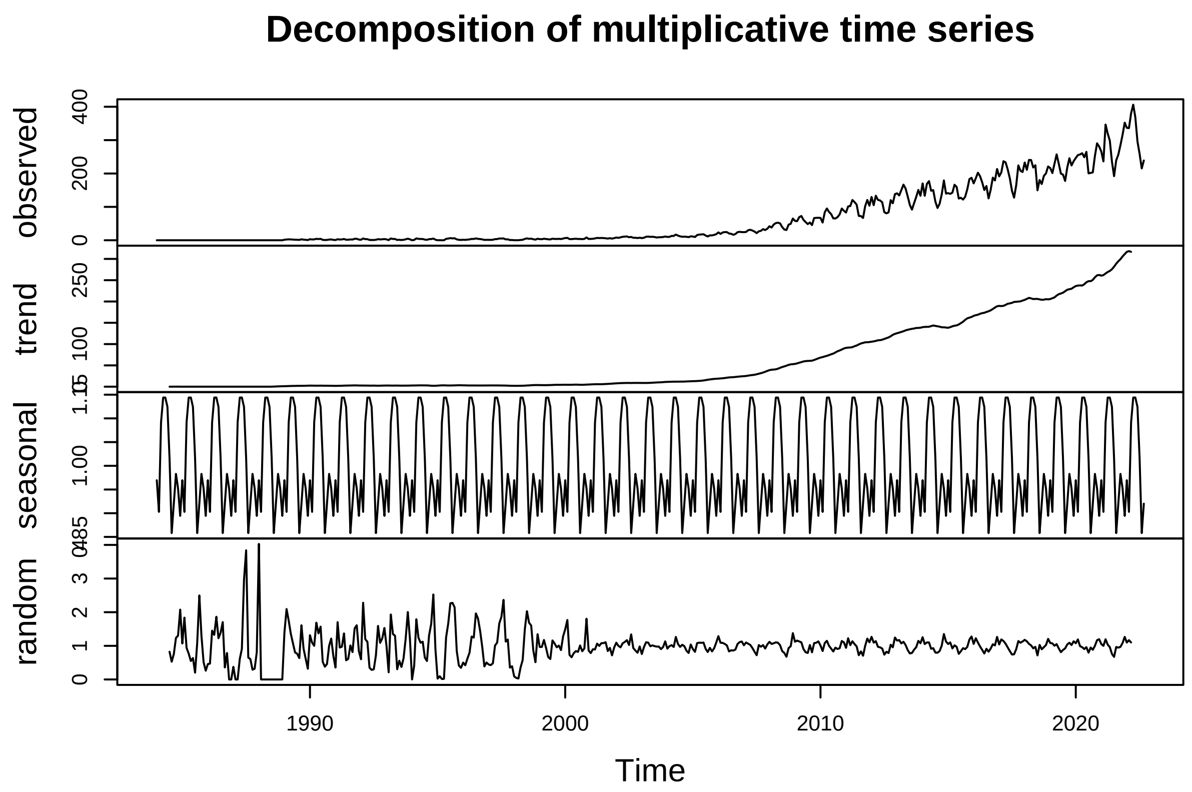
Q4

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?

```r
decompose_solar_mul <- decompose(ts_solar, type = "multiplicative")
plot(decompose_solar_mul)
```



```r
decompose_wind_mul <- decompose(ts_wind, type = "multiplicative")
plot(decompose_wind_mul)
```



For the solar series, its random component still doesn't look "random": it now fluctuates around 1 instead of 0, and the magnitude of the fluctuation decreased dramatically around 1990, remaining fairly stable from then on. The wind series shows a similar pattern: its random component fluctuates around 1, and the fluctuation was less pronounced after 2000.

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: No, I don't think we need all the historical data. Because both the technology and society's perception of wind and solar have changed, and that will have a big impact on the market dynamics and therefore on their consumption. If we were only forecasting consumption for the next six months, it would make more sense to only look at the last few years.

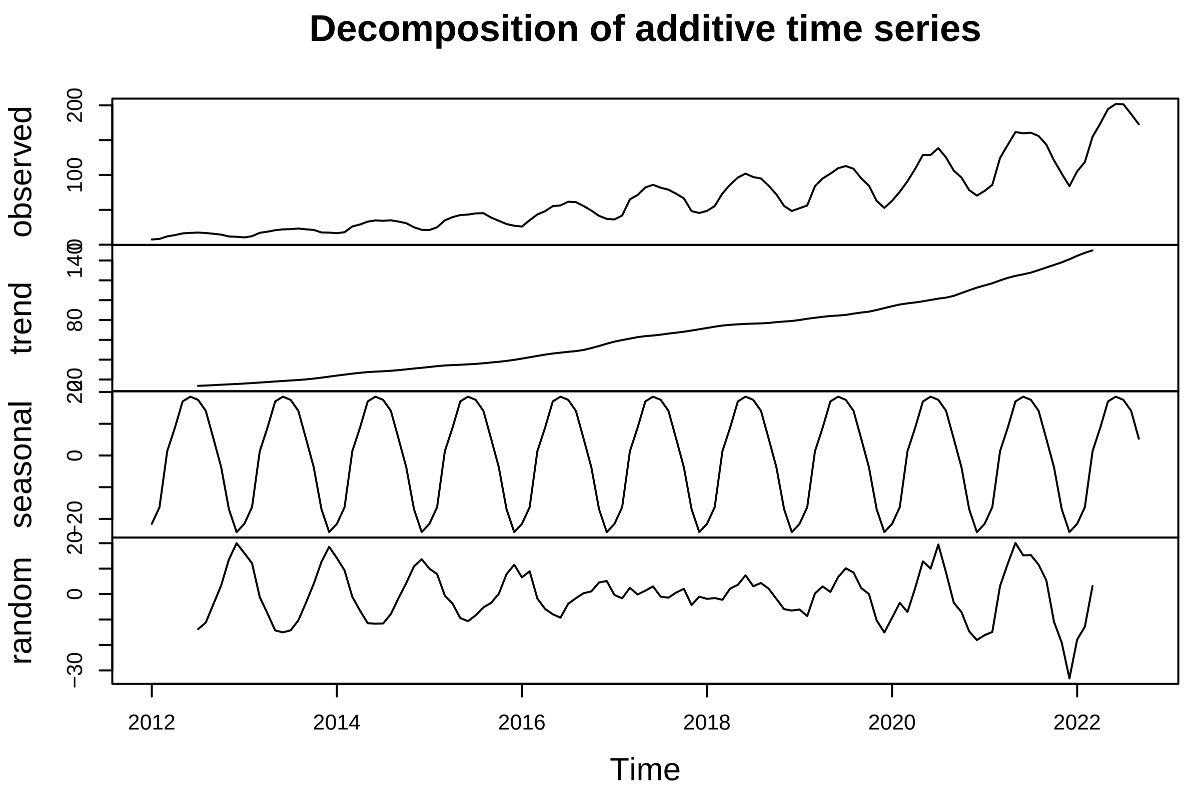
Q6

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.

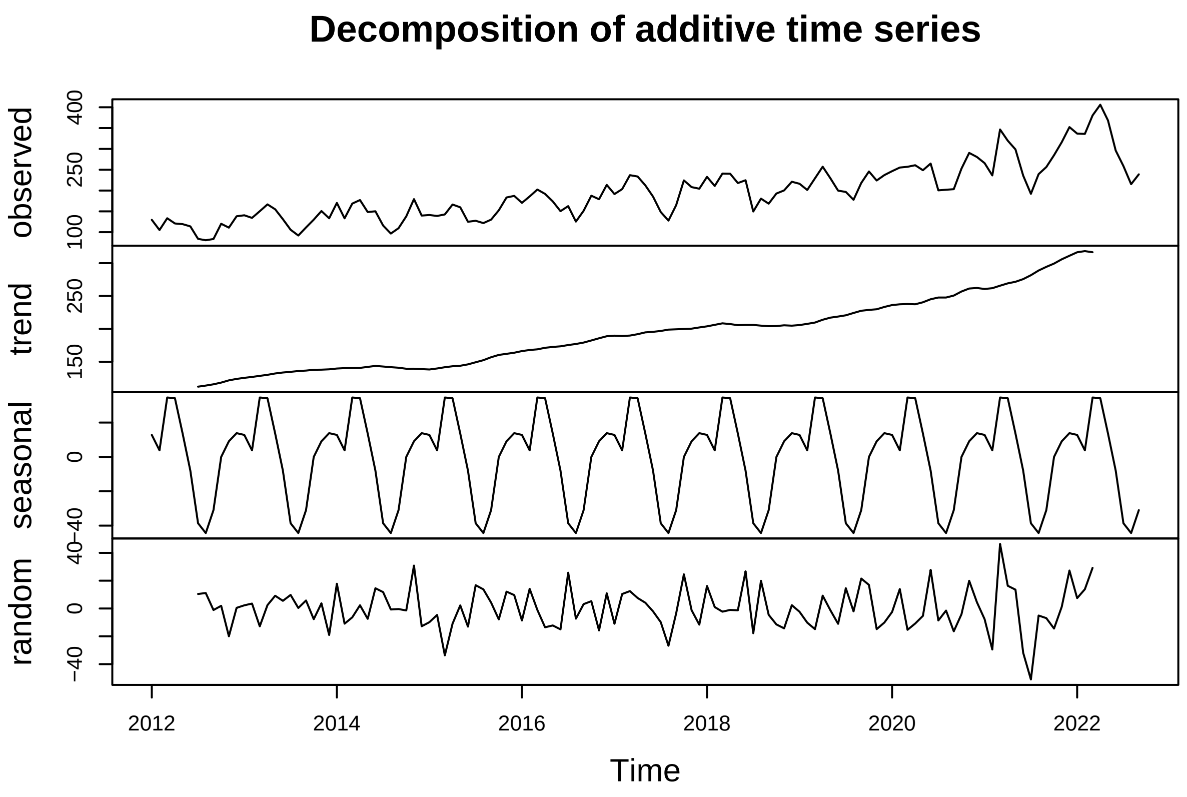
```r
#filter only data after 2012
new_df <- filter(cleaned_df,year(Date)>=2012)
new_ts_solar <- ts(new_df$Solar,frequency = 12,start=c(2012,1,1))
new_ts_wind <- ts(new_df$Wind,frequency = 12,start=c(2012,1,1))

#try additive
```

```
new_decompose_solar_add <- decompose(new_ts_solar, type = "additive")
plot(new_decompose_solar_add)
```



```
new_decompose_wind_add <- decompose(new_ts_wind, type = "additive")
plot(new_decompose_wind_add)
```



Answer: For the solar series, the additive model was not a good fit because the random component doesn't look random and its level has a clear changing decreasing-and-then-increasing pattern over time. For the wind series, the random component looks random.