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

Assignment 5 - Due date 02/28/22

Yu Hai

## 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(readxl)
```

```
## Warning:   'readxl' R 4.1.2
```

```
library(forecast)
```

```
## Warning:   'forecast' R 4.1.2
```

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

```
library(tseries)
```

```
## Warning:   'tseries' R 4.1.2
```

```
library(ggplot2)
library(Kendall)
```

```
## Warning:   'Kendall' R 4.1.2
```

```
library(lubridate)
```

```
##
##     '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  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     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(path="./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xlsx",start
↪  = 13,sheetIndex=1) #startRow is equivalent to skip on read.table
energy_data<-read_excel(path="./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xl
```

```
## New names:
## * `` -> ...1
## * `(Trillion Btu)` -> `(Trillion Btu)...2`
## * `(Trillion Btu)` -> `(Trillion Btu)...3`
## * `(Trillion Btu)` -> `(Trillion Btu)...4`
## * `(Trillion Btu)` -> `(Trillion Btu)...5`
## * ...
```

```
head(energy_data)
```

```
## # A tibble: 6 x 14
##   ...1                `(Trillion Btu)...2` `(Trillion Btu)...3` `(Trillion Btu)~
##   <dttm>                             <dbl> <chr>                           <dbl>
## 1 1973-01-01 00:00:00                 130. Not Available                    130.
## 2 1973-02-01 00:00:00                 117. Not Available                    117.
## 3 1973-03-01 00:00:00                 130. Not Available                    130.
## 4 1973-04-01 00:00:00                 125. Not Available                    126.
## 5 1973-05-01 00:00:00                 130. Not Available                    130.
## 6 1973-06-01 00:00:00                 125. Not Available                    126.
## # ... with 10 more variables: (Trillion Btu)...5 <dbl>,
## #   (Trillion Btu)...6 <dbl>, (Trillion Btu)...7 <dbl>,
## #   (Trillion Btu)...8 <chr>, (Trillion Btu)...9 <chr>,
## #   (Trillion Btu)...10 <dbl>, (Trillion Btu)...11 <dbl>,
## #   (Trillion Btu)...12 <chr>, (Trillion Btu)...13 <dbl>,
## #   (Trillion Btu)...14 <dbl>
```

```
#Now let's extract the column names from row 11 only
read_col_names <-
 ↪  read_xlsx(path="./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xlsx",range
 ↪  = "A11:N11")

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

```
## # A tibble: 6 x 14
##   `logical(0)`        `logical(0)` `logical(0)`  `logical(0)` `logical(0)`
##   <dttm>                     <dbl> <chr>                <dbl>        <dbl>
## 1 1973-01-01 00:00:00         130. Not Available         130.         404.
## 2 1973-02-01 00:00:00         117. Not Available         117.         361.
## 3 1973-03-01 00:00:00         130. Not Available         130.         400.
## 4 1973-04-01 00:00:00         125. Not Available         126.         380.
## 5 1973-05-01 00:00:00         130. Not Available         130.         392.
## 6 1973-06-01 00:00:00         125. Not Available         126.         377.
## # ... with 9 more variables: logical(0) <dbl>, logical(0) <dbl>,
## #   logical(0) <chr>, logical(0) <chr>, logical(0) <dbl>, logical(0) <dbl>,
## #   logical(0) <chr>, logical(0) <dbl>, logical(0) <dbl>
```

```
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!

```
raw_RE_data<-read_excel(path="./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xl
raw_RE_data
```

```
## # A tibble: 586 x 14
##    Month               `Wood Energy Prod~ `Biofuels Product~ `Total Biomass Ene~
##    <dttm>              <chr>              <chr>              <chr>
##  1 NA                  (Trillion Btu)     (Trillion Btu)     (Trillion Btu)
##  2 1973-01-01 00:00:00 129.63             Not Available      129.787
##  3 1973-02-01 00:00:00 117.194            Not Available      117.338
##  4 1973-03-01 00:00:00 129.763            Not Available      129.938
##  5 1973-04-01 00:00:00 125.462            Not Available      125.636
##  6 1973-05-01 00:00:00 129.624            Not Available      129.834
##  7 1973-06-01 00:00:00 125.435            Not Available      125.611
##  8 1973-07-01 00:00:00 129.616            Not Available      129.787
##  9 1973-08-01 00:00:00 129.734            Not Available      129.918
## 10 1973-09-01 00:00:00 125.603            Not Available      125.782
## # ... with 576 more rows, and 10 more variables:
## #   Total Renewable Energy Production <chr>,
## #   Hydroelectric Power Consumption <chr>, Geothermal Energy Consumption <chr>,
## #   Solar Energy Consumption <chr>, Wind Energy Consumption <chr>,
## #   Wood Energy Consumption <chr>, Waste Energy Consumption <chr>,
## #   Biofuels Consumption <chr>, Total Biomass Energy Consumption <chr>,
## #   Total Renewable Energy Consumption <chr>
```

```
sub_RE_data <- raw_RE_data[-c(1),8:9]
RE_data <- cbind(raw_RE_data[-c(1),1],sub_RE_data[,])
RE_data$`Solar Energy Consumption`<- as.numeric(RE_data$`Solar Energy Consumption`)
```

```
## Warning:       NA
```

```
RE_data$`Wind Energy Consumption`<- as.numeric(RE_data$`Wind Energy Consumption`)
```

```
## Warning:       NA
```

```
sum(is.na(RE_data$`Solar Energy Consumption`))
```

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

```
clean_RE_data <- na.omit(RE_data)
head(clean_RE_data)
```

```
##           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
```

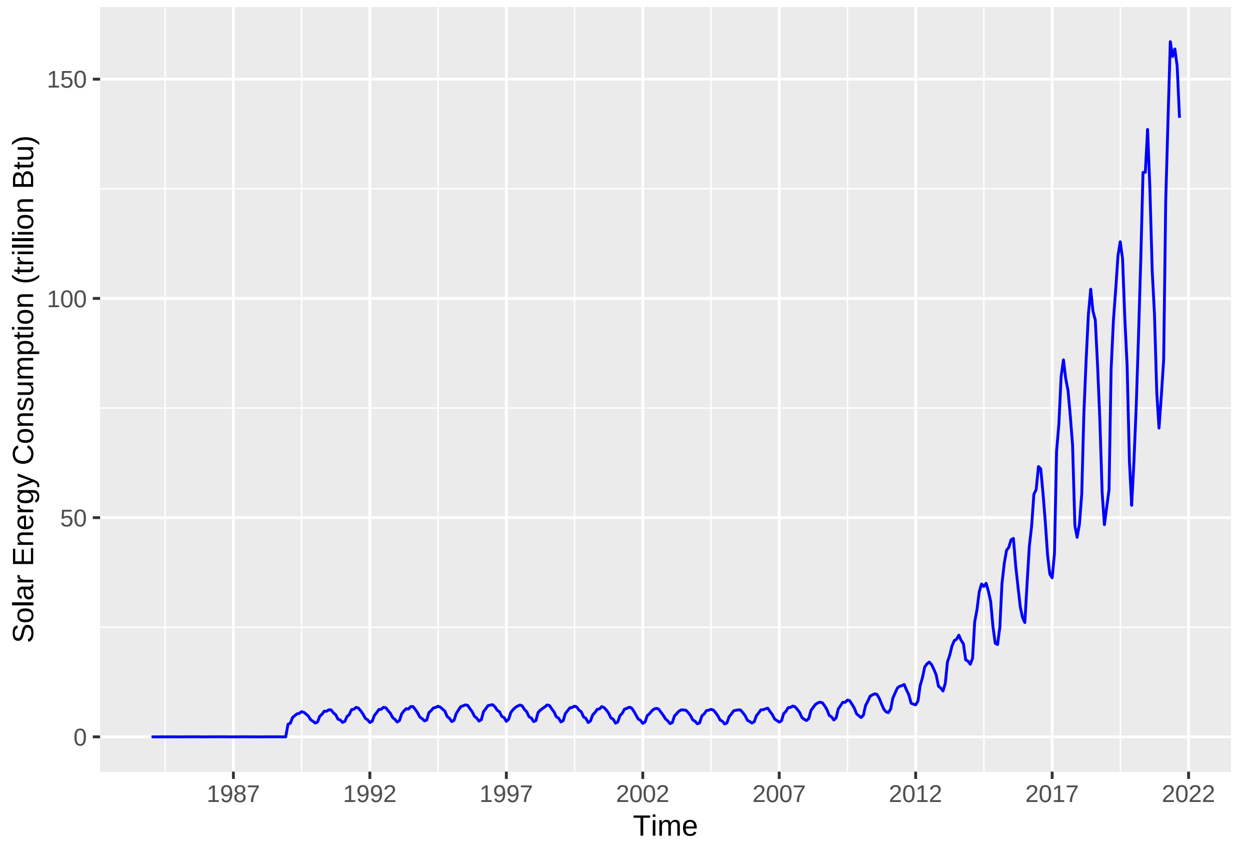
```
clean_RE_data$Month<-as.Date(clean_RE_data$Month)
```

### Q2

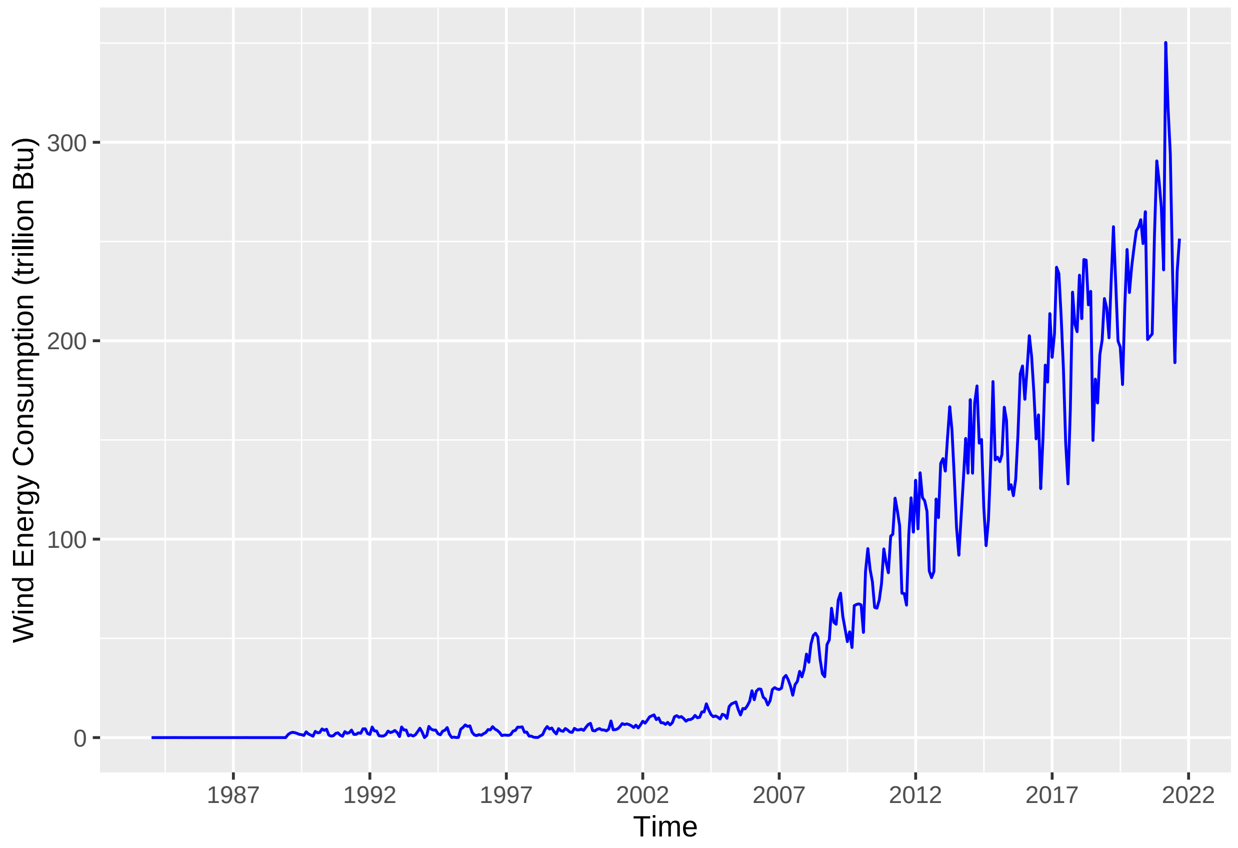
Plot the Solar and Wind energy consumption over time using ggplot. Plot each series on a separate graph. No need to add legend. Add informative names to the y axis using `ylab()`. Explore the function scale_x_date() on ggplot and see if you can change the x axis to improve your plot. Hint: use *scale_x_date(date_breaks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

```
par(mfrow=c(1,2))

ggplot(clean_RE_data, aes(x=Month, y=clean_RE_data[,2])) +
            geom_line(color="blue") +
            ylab("Solar Energy Consumption (trillion Btu)")+
            xlab("Time")+
            scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



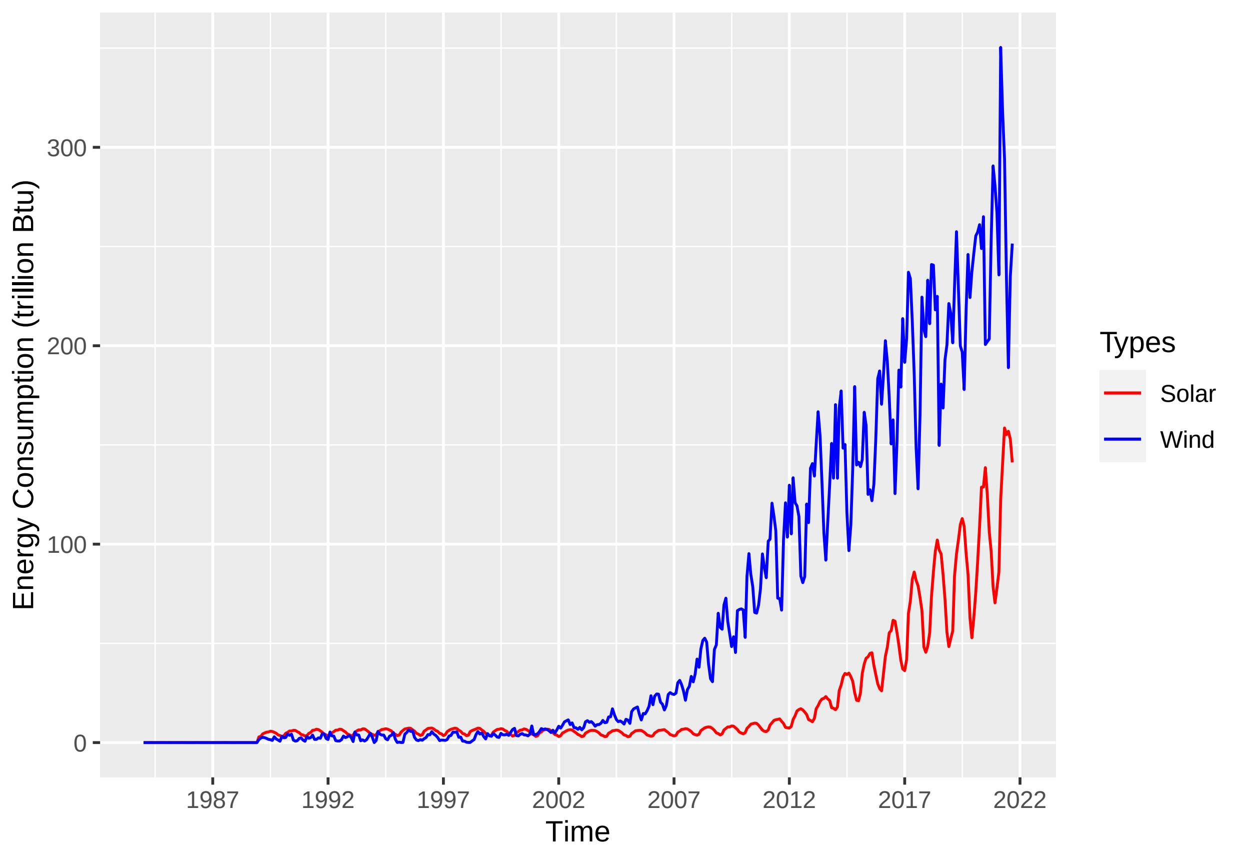
```
ggplot(clean_RE_data, aes(x=Month, y=clean_RE_data[,3])) +
            geom_line(color="blue") +
            ylab("Wind Energy Consumption (trillion Btu)")+
            xlab("Time")+
            scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



**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_RE_data, aes(x=Month,y=clean_RE_data[,2],show.legend=TRUE)) +
            ylab("Energy Consumption (trillion Btu)") +
            xlab("Time")+
            geom_line(aes(y=clean_RE_data[,2],color="Solar"))+
            geom_line(aes(y=clean_RE_data[,3],color="Wind"))+
  scale_color_manual(name = "Types", breaks=c("Solar","Wind"),
                     values = c("Solar" = "red",
                                "Wind" = "blue"))+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



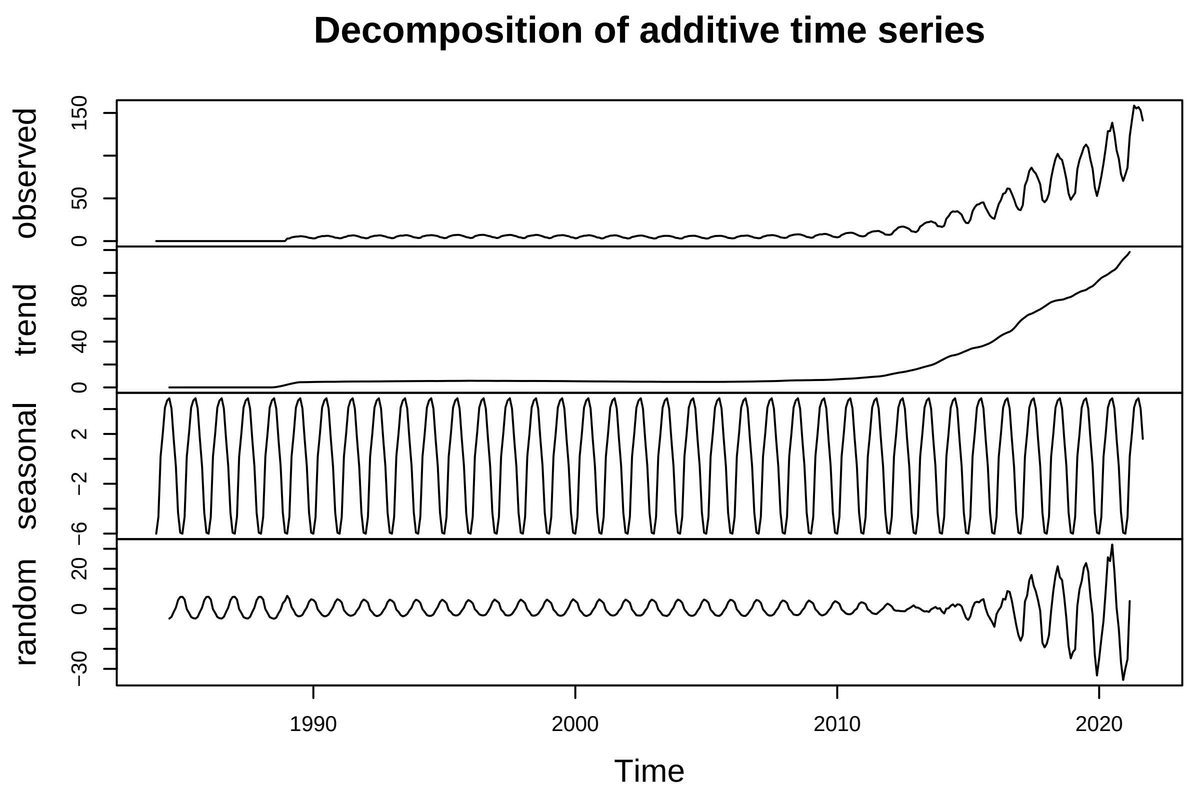
**Q3**

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?

```
#solar
ts_solar_data <- ts(RE_data$`Solar Energy Consumption`,frequency=12,start=c(1973, 1),
 ↪  end=c(2021, 09))
ts_solar_data<-na.omit(ts_solar_data)
head(ts_solar_data)
```

```
##          Jan    Feb    Mar    Apr    May    Jun
## 1984 -0.001  0.001  0.002  0.003  0.007  0.010
```

```
decompose_solar_data=decompose(ts_solar_data,"additive")
plot(decompose_solar_data)
title(sub="solar")
```



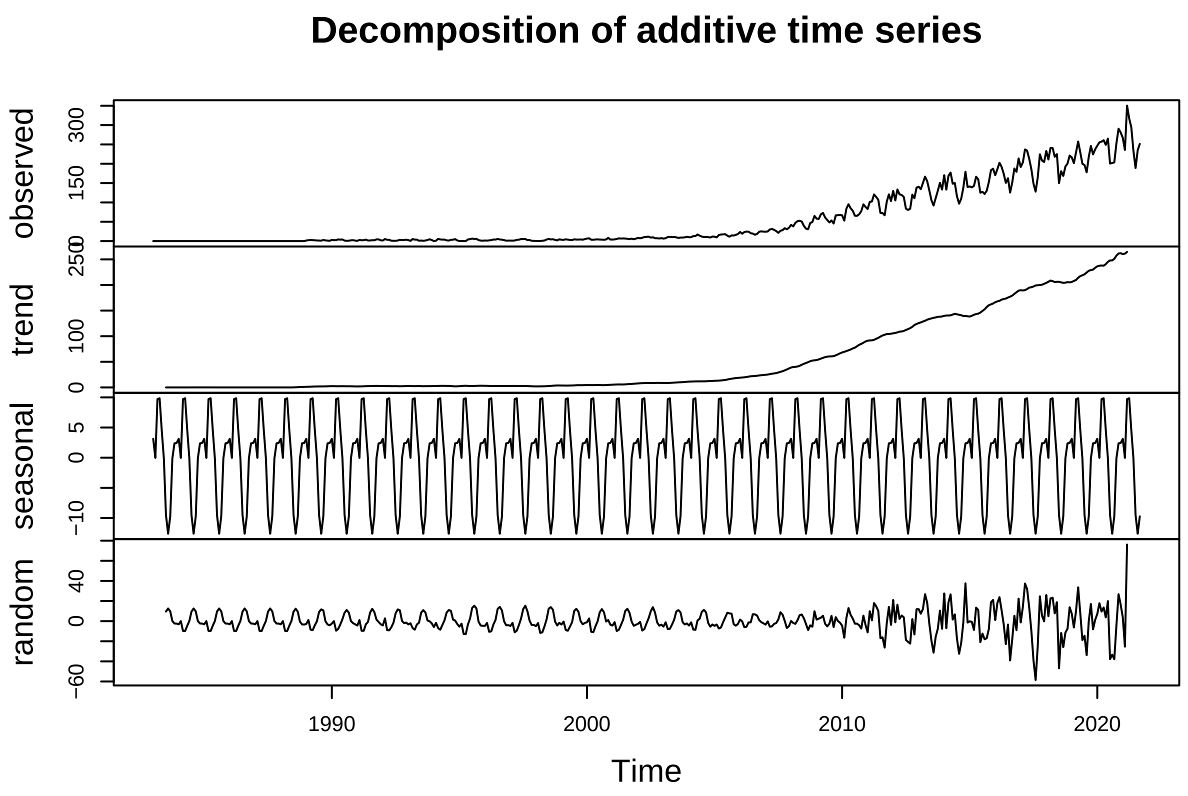
solar

For solar energy consumption, the trend component shows an increasing trend, but it is not linear (slope becomes much steeper since 2010). The random component doesn't seems random, and it appears to still have seasonality on it as there is periodic fluctuation throughout the time of observations (the fluctuation becomes stronger after 2015). Thus, the additive decomposition may not be the appropriate method for this time series.

```
#wind
ts_wind_data <- ts(RE_data$`Wind Energy Consumption`,frequency=12,start=c(1973, 1),
↪  end=c(2021, 09))
ts_wind_data<-na.omit(ts_wind_data)
head(ts_solar_data)
```

```
##         Jan    Feb    Mar    Apr    May    Jun
## 1984 -0.001  0.001  0.002  0.003  0.007  0.010
```

```
decompose_wind_data=decompose(ts_wind_data,"additive")
plot(decompose_wind_data)
title(sub="wind")
```



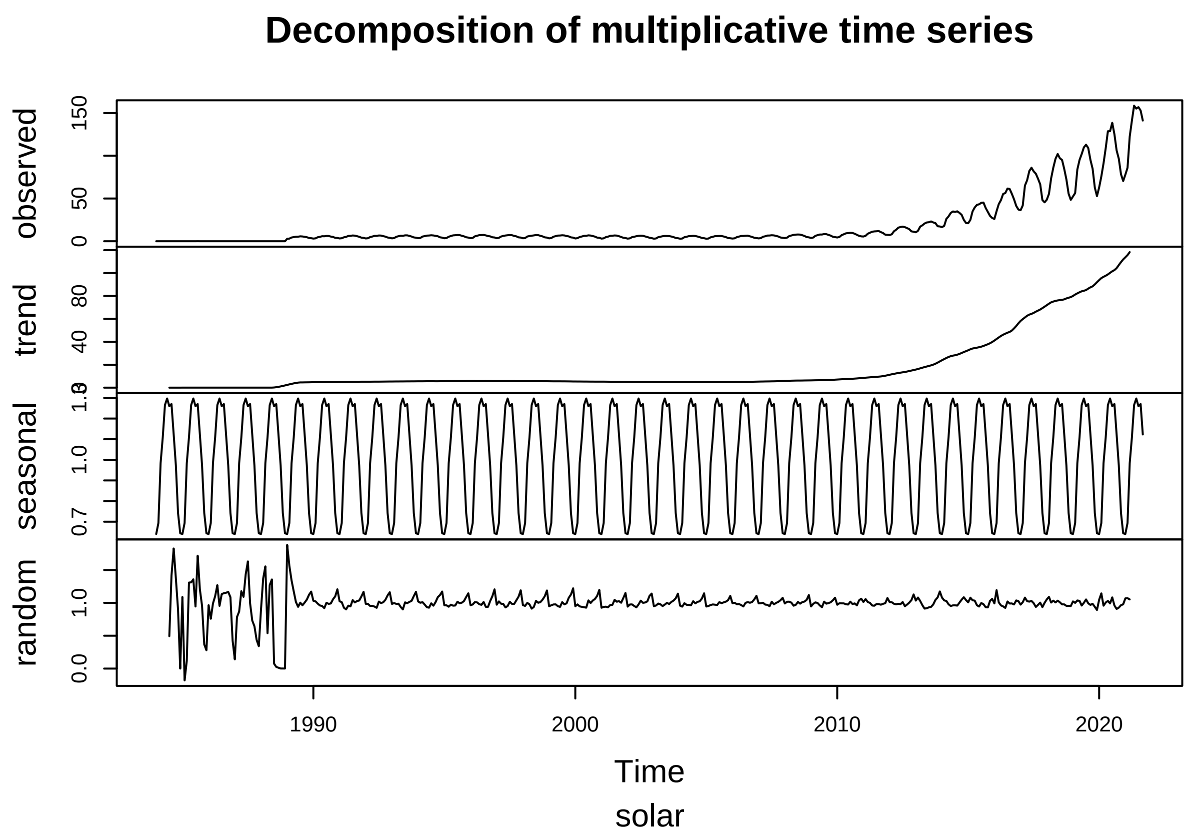
wind

Similar to solar, the trend component of wind energy consumption shows a non-linear, increasing trend. The random component shows seasonality for observation before 2010. After 2010, the random component seems random but with greater fluctuations.

## Q4

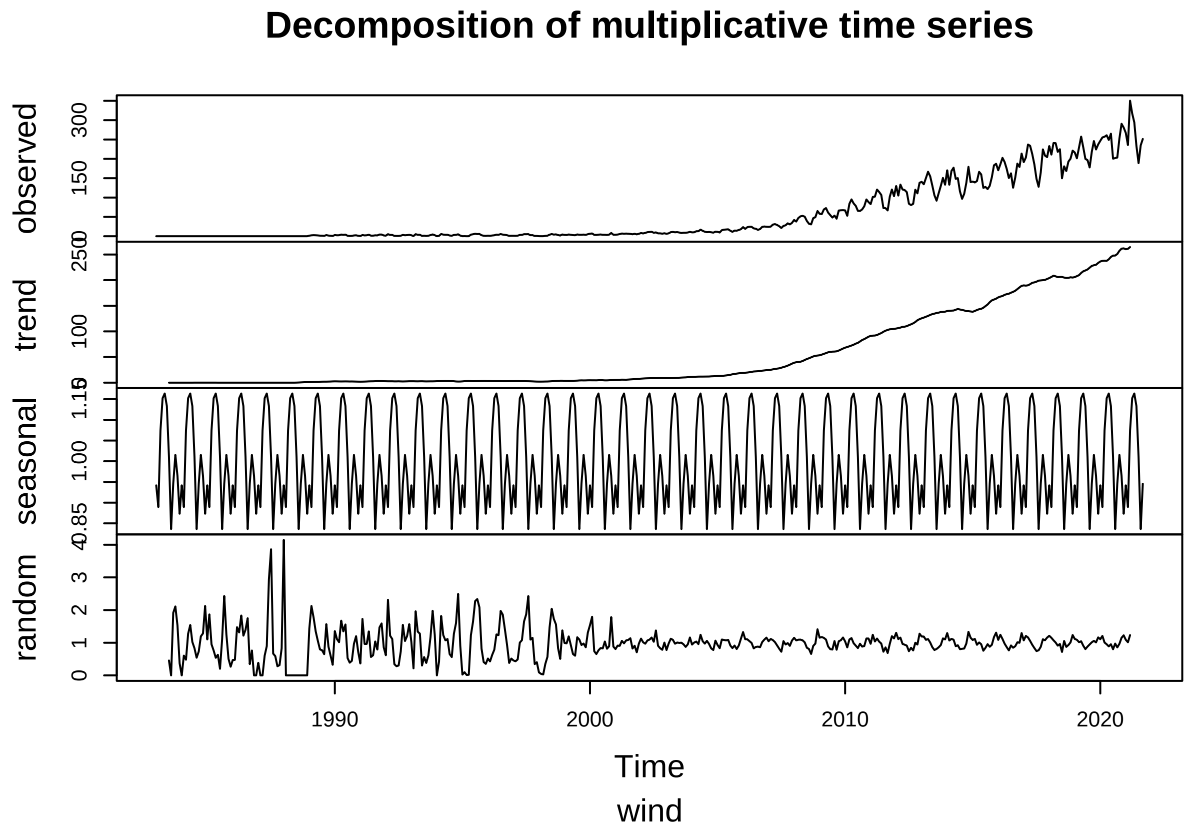
Use the decompose function again but now change the type of the seasonal component from additive to multiplicative. What happened to the random component this time?

```r
#solar
decompose_solar_data_2=decompose(ts_solar_data,"multiplicative")
plot(decompose_solar_data_2)
title(sub="solar")
```



For solar energy consumption, compared to the additive decomposition, the random component shows no seasonality for observations before 1990, but there is still inconsistency of the pattern of fluctuations between the observations before and after 1990. The random component of observations after 1990 still shows seasonality, but the seasonality gets weaker after 2005.

```
#wind
decompose_wind_data_2=decompose(ts_wind_data,"multiplicative")
plot(decompose_wind_data_2)
title(sub="wind")
```



Compared to the additive decomposition, the random component of wind energy consumption shows less seasonality, especially on observations before 2010. However, the random component of observations after 2010 shows a weak seasonality. The random component of observations before 2000 also shows a greater fluctuation compared to later observations.

**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. #level shift? but by defination (changes in concepts and definitions or complation methods of the survey population?) > Answer:No. The observations from 90s and early 20s show a totally different patterns than observaionts more recently (after 2005 for wind, after 2010 for solar), so we can suspect there is a permanent change in the time series level. If fitting a model is to forecast future situations, and considering the significnat technological development and adoptions of wind and solar overtime, it is very unlikely that future solar and wind consumption will follow the trends of observations from 90s and early 20s, so we only need to consider more recent data to fit a model and use it to forecast future solar and wind consumptions.

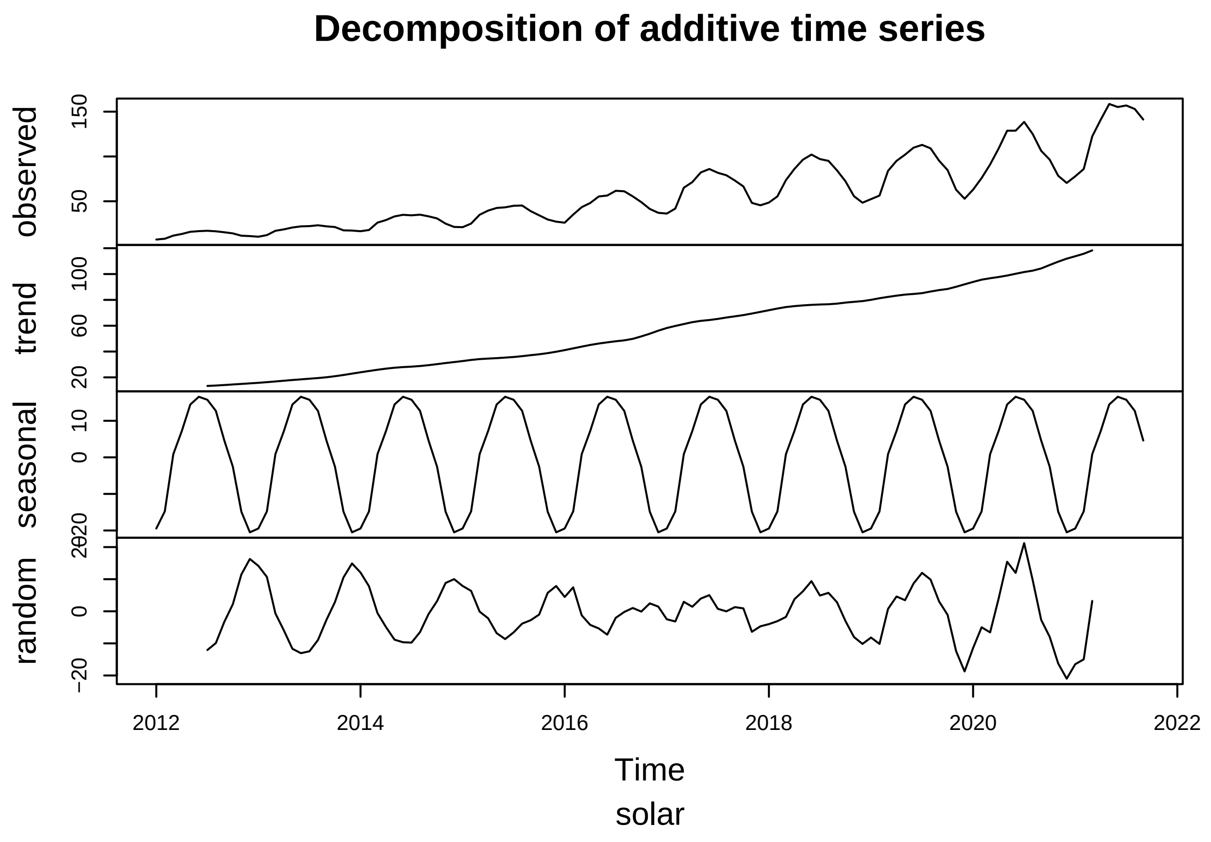
**Q6**

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

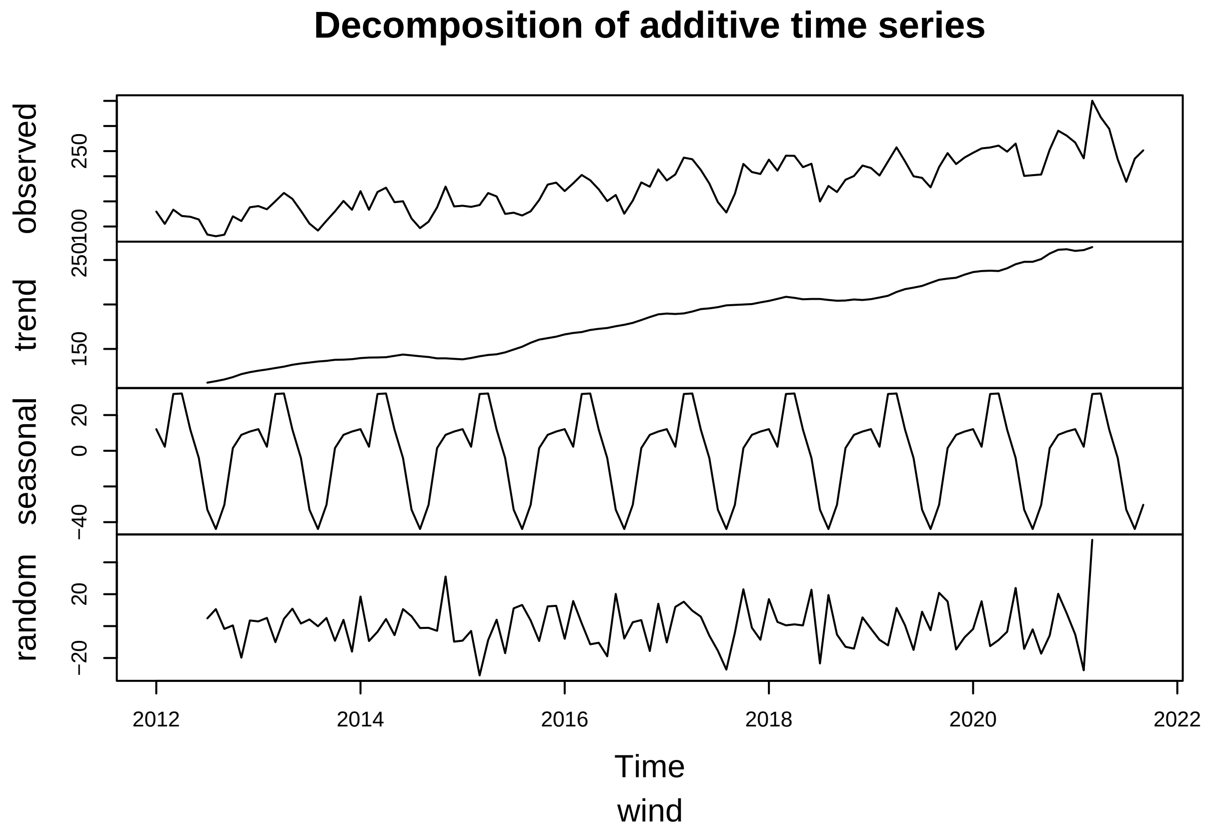
```
#solar
solar_data_3<-filter(clean_RE_data, as.numeric(year(Month)) >= 2012 )
head(solar_data_3)
```

```
##        Month Solar Energy Consumption Wind Energy Consumption
## 1 2012-01-01                    7.288                 129.726
## 2 2012-02-01                    8.165                 105.171
## 3 2012-03-01                   11.678                 133.476
## 4 2012-04-01                   13.478                 120.941
## 5 2012-05-01                   15.933                 119.336
## 6 2012-06-01                   16.651                 113.928
```

```
ts_solar_data_3<-ts(solar_data_3$`Solar Energy Consumption`, frequency=12,start=c(2012,
↪  1), end=c(2021, 09))
decompose_solar_data_3=decompose(ts_solar_data_3,"additive")
plot(decompose_solar_data_3)
title(sub="solar")
```



```
#wind
ts_wind_data_3<-ts(solar_data_3$`Wind Energy Consumption`, frequency=12,start=c(2012, 1),
↪  end=c(2021, 09))
decompose_wind_data_3=decompose(ts_wind_data_3,"additive")
plot(decompose_wind_data_3)
title(sub="wind")
```



Answer: For wind energy consumption after 2012, the random component seems random, while the random component of solar energy consumption after 2012 still shows seasonality. The reason might be that the seasonal component of solar energy consumption is not constant over time, but rather contains a long-term trend (e.g. climate change impact), so the additive decomposition cannot completely remove this type of seasonal component.