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

Assignment 5 - Due date 02/27/23

John Rooney

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 you 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()
```

```r
library(cowplot)
```

```
##
## Attaching package: 'cowplot'
##
## The following object is masked from 'package:lubridate':
##
##     stamp
```

Decomposing Time Series

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

```r
#Importing data set -

energy.data <- read.csv(file="./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source-E
```

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!

```r
energy.data$Solar.Energy.Consumption <- as.numeric(as.character(energy.data$Solar.Energy.Consumption))
```

```
## Warning: NAs introduced by coercion
```

```r
energy.data$Wind.Energy.Consumption <- as.numeric(as.character(energy.data$Wind.Energy.Consumption))
```

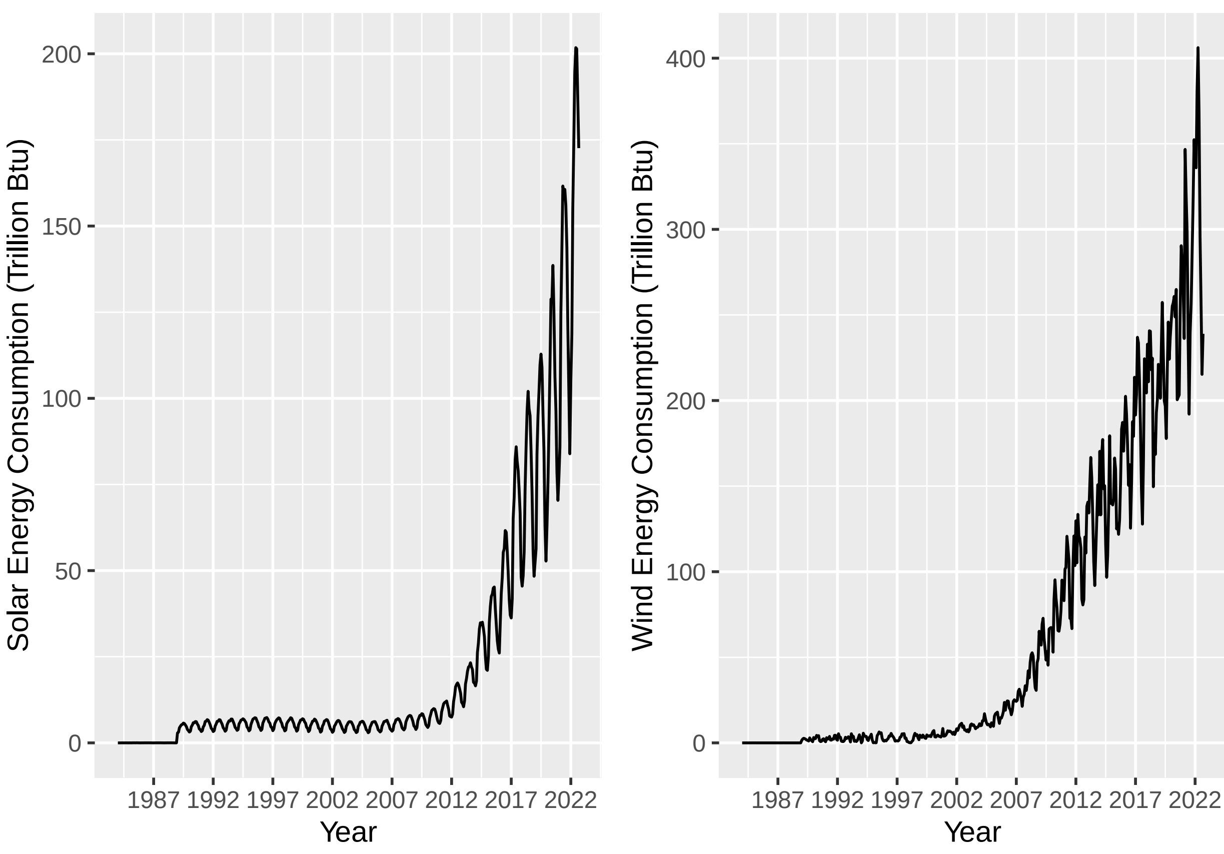
```
## Warning: NAs introduced by coercion
```

```r
solar.wind.energy.data <- select(energy.data, Month, Solar.Energy.Consumption,Wind.Energy.Consumption)%
  drop_na(Solar.Energy.Consumption, Wind.Energy.Consumption)

solar.wind.energy.data$Month <- ym(solar.wind.energy.data$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")"*)

```
solar.plot<-ggplot(solar.wind.energy.data, aes(x=Month, y=Solar.Energy.Consumption))+
  geom_line()+
  xlab("Year")+
  ylab("Solar Energy Consumption (Trillion Btu)")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")

wind.plot<-ggplot(solar.wind.energy.data, aes(x=Month, y=Wind.Energy.Consumption))+
  geom_line()+
  xlab("Year")+
  ylab("Wind Energy Consumption (Trillion Btu)")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")

plot_grid(solar.plot, wind.plot, nrow=1, align = "h")
```

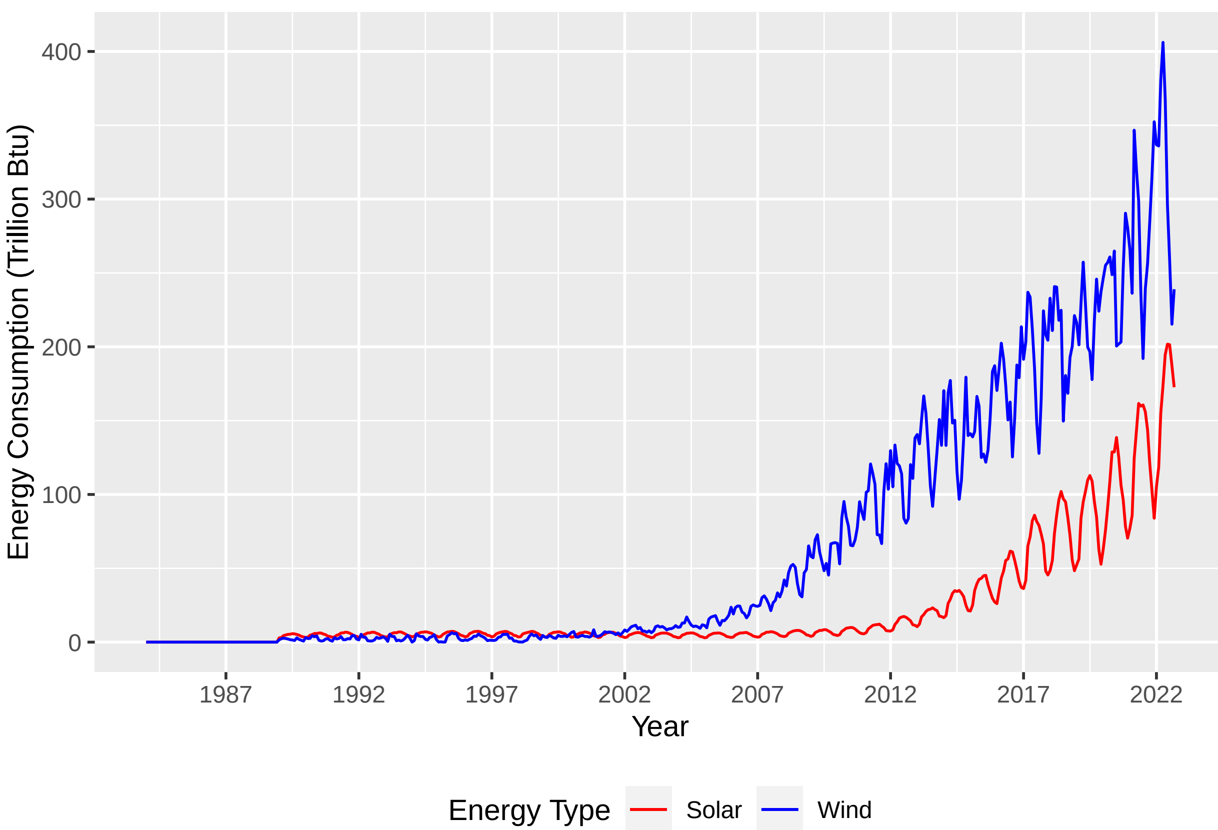


Q3

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.

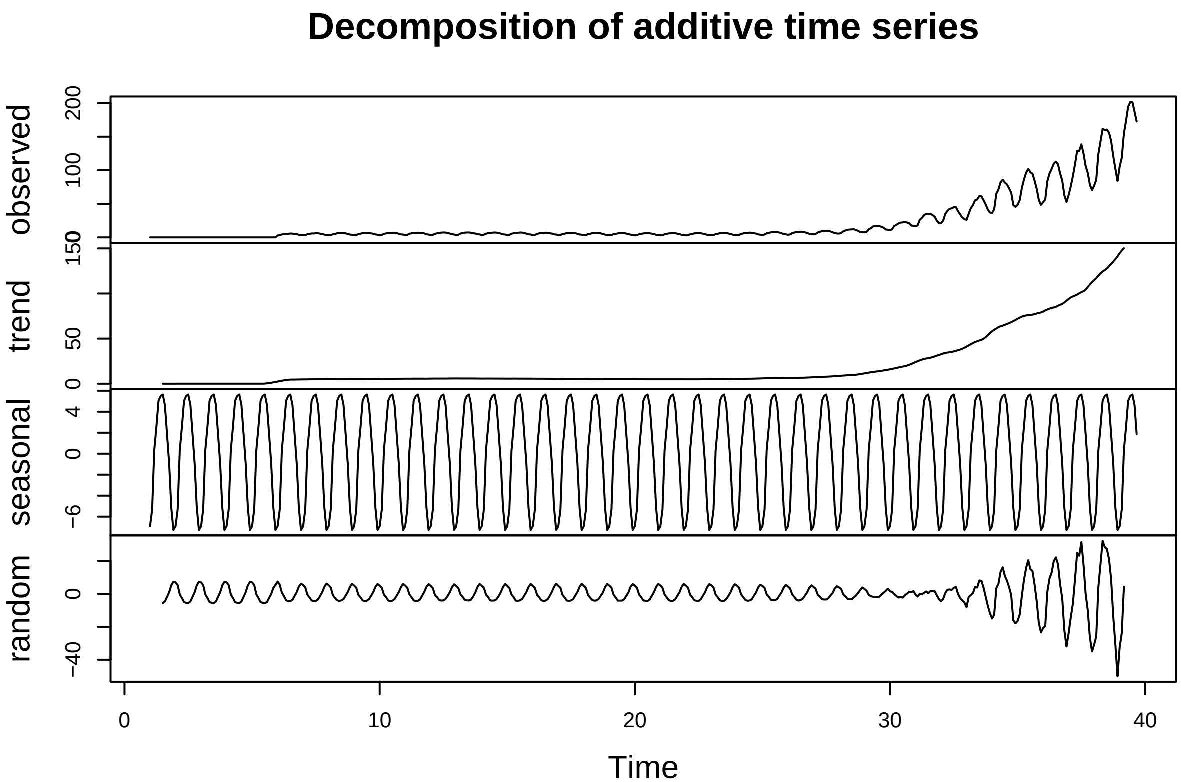
```
energy.plot<-ggplot(solar.wind.energy.data) +
  geom_line(aes(x = Month, y = Solar.Energy.Consumption, color = "Solar")) +
  geom_line(aes(x = Month, y = Wind.Energy.Consumption, color = "Wind")) +
  labs(x="Year", y= "Energy Consumption (Trillion Btu)", fill = "Energy Type") +
  theme(legend.position = "bottom")  +
  scale_color_manual(values = c("Solar" = "red", "Wind" = "blue"),
                     labels=c("Solar", "Wind")) +
  guides(color = guide_legend(title = "Energy Type")) +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")

energy.plot
```



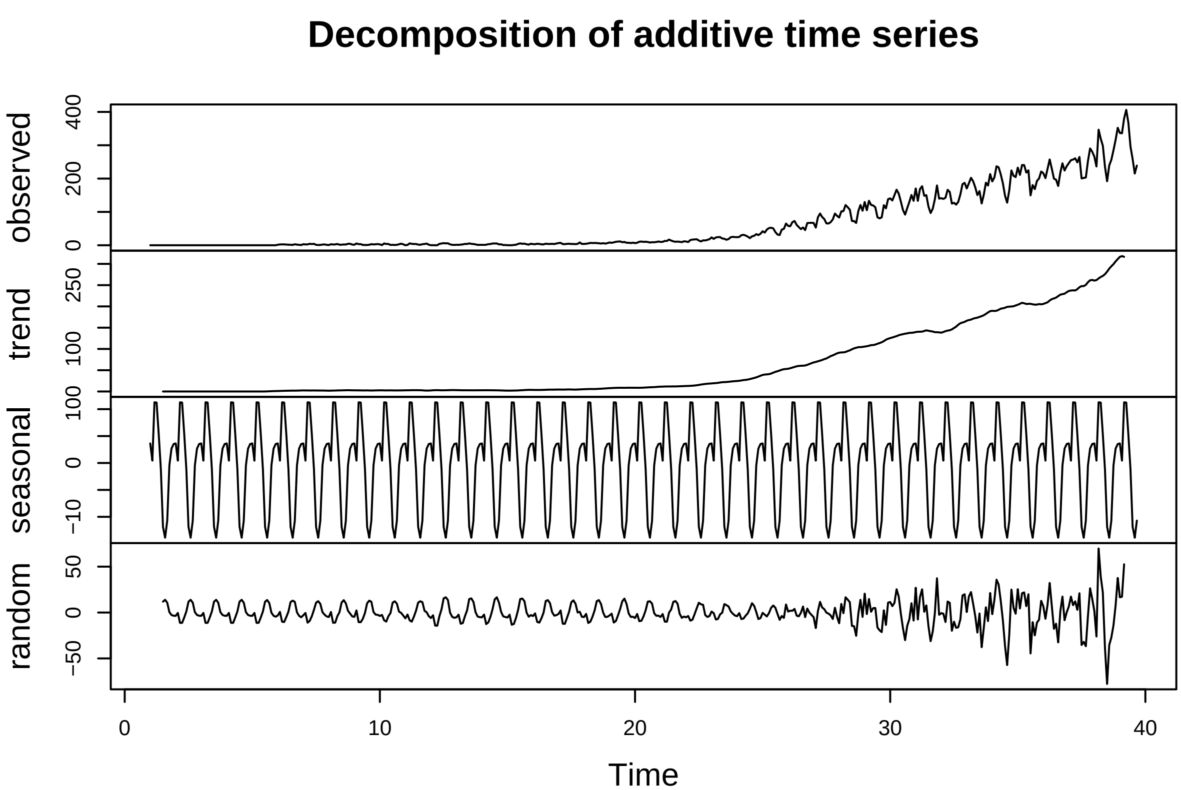
Q4

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?

```
ts.solar.energy.data<-ts(solar.wind.energy.data[,2], frequency=12)
decompose.solar.data<-decompose(ts.solar.energy.data, type = "additive")
plot(decompose.solar.data)
```




```
ts.wind.energy.data<-ts(solar.wind.energy.data[,3], frequency=12)
decompose.wind.data<-decompose(ts.wind.energy.data, type="additive")
plot(decompose.wind.data)
```



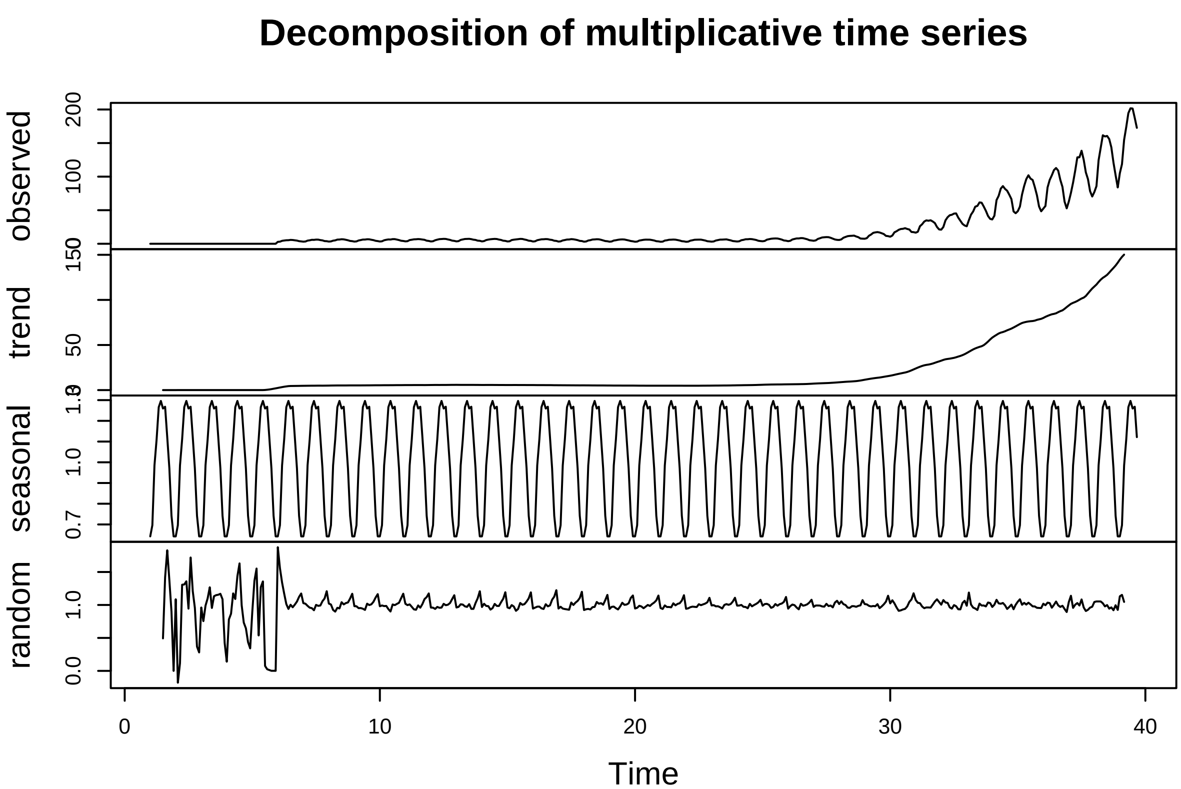
>

Answer: For both wind and solar, the trend components seem to track the observed series very well. The random components make sense to me up until about 27 or 28 years for wind, when the magnitude of each series becomes larger with some erratic behavior, and around 30 years for solar when it gradually increases. I am guessing this mirrors the rise in energy production by each type of technology. I would say there is clear seasonality to wind, and some (although less clear) seasanality to solar.

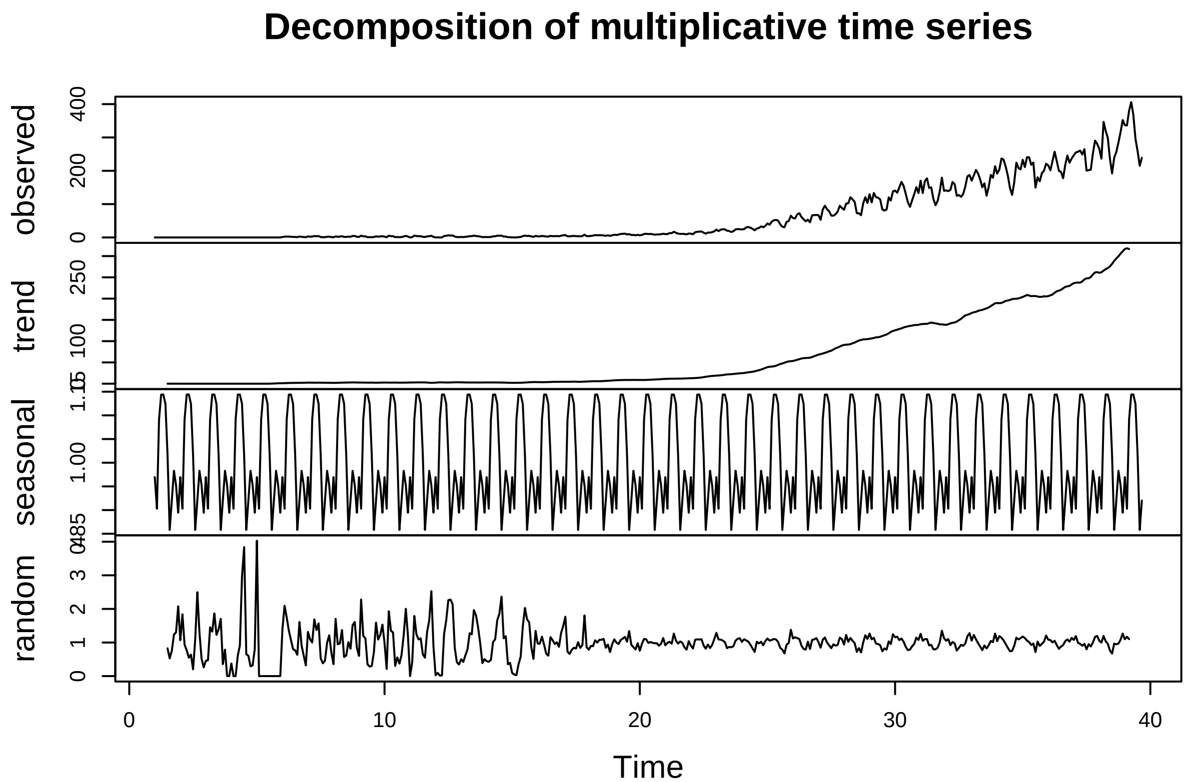
Q5

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?

```
decompose.solar.data2<-decompose(ts.solar.energy.data, type = "multiplicative")
plot(decompose.solar.data2)
```




```
decompose.wind.data2<-decompose(ts.wind.energy.data, type="multiplicative")
plot(decompose.wind.data2)
```



>

Answer: The erratic behavior in the random components are now at the beginning of the series. It seems less predictable in terms of magnitude for both wind and solar this time. I would guess this is because it is showing a lack of seasonality (and therefore less predicatability) at the beginning.

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.

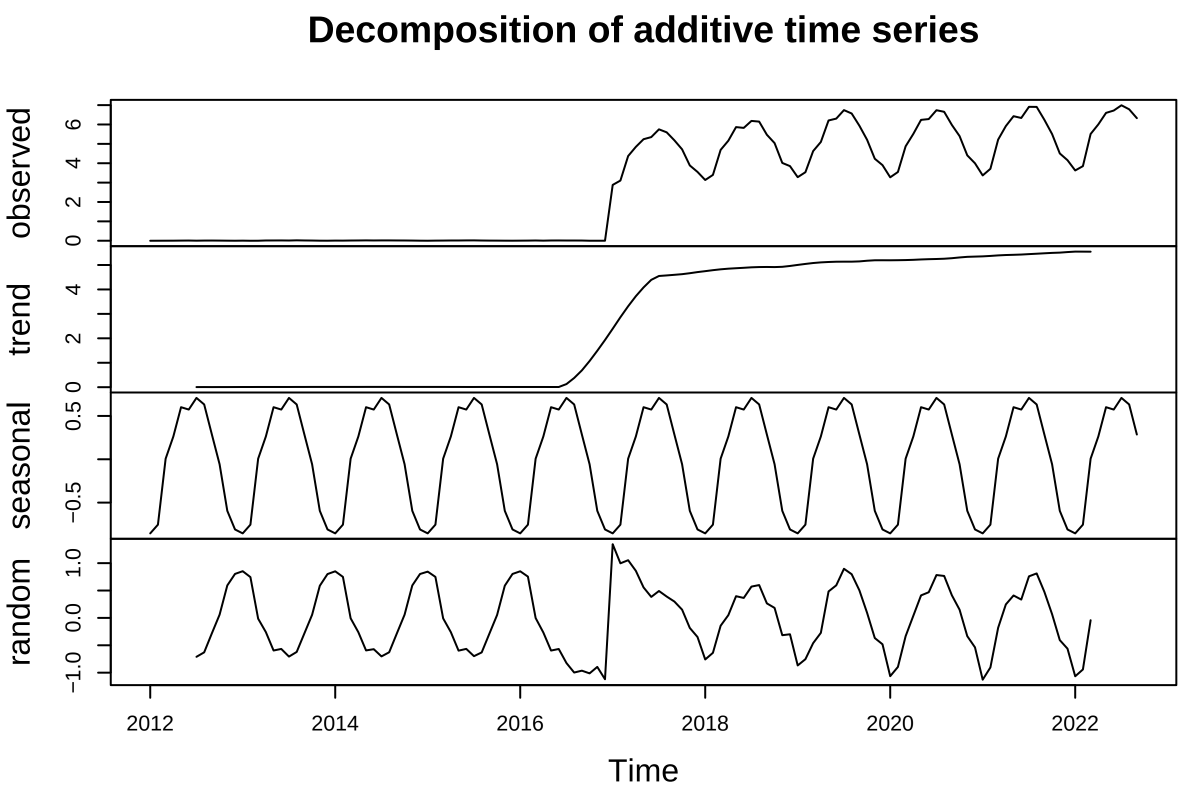
> Answer: In general, I think the more historical data, the better. With that said, the 90s and early aughts may be less helpful as production was much smaller than in recent years so I think there is less predictive power in the data from those years.

Q7

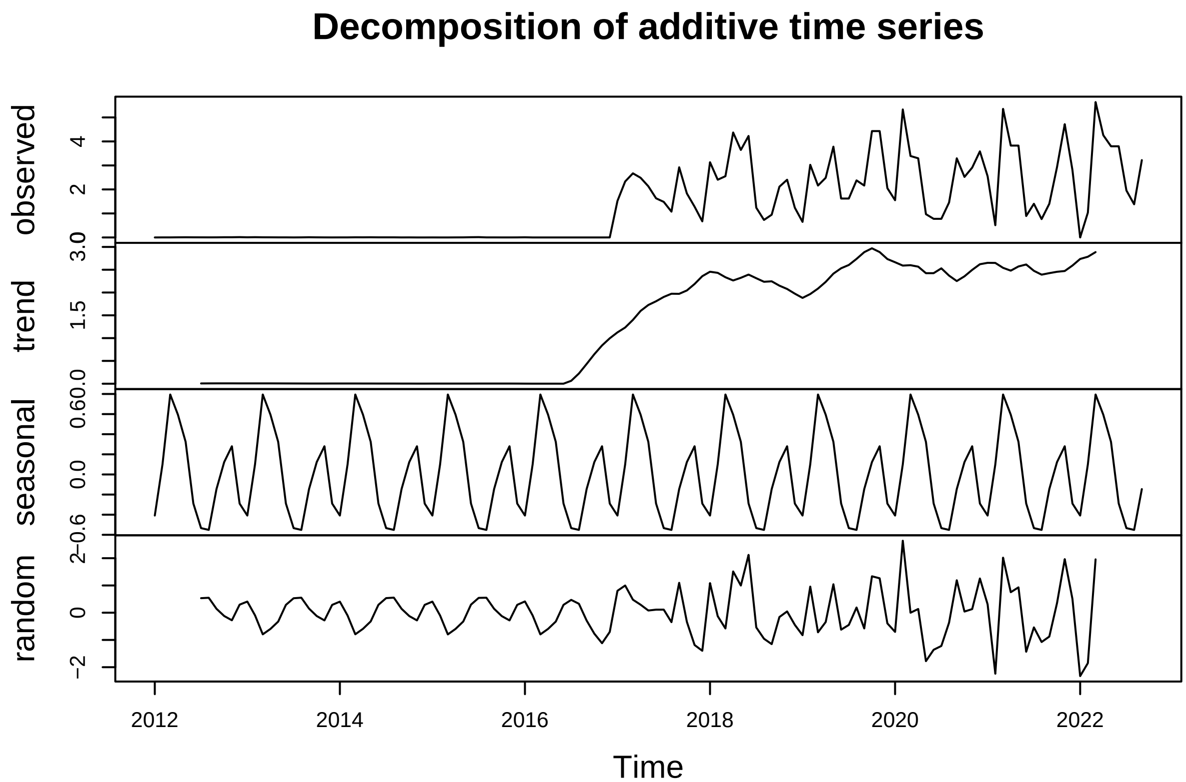
Create a new time series object where historical data starts on January 2012. Hint: use `filter()` function so that you don't need to point to row numbers, .i.e, `filter(xxxx, year(Date) >= 2012 )`. Apply the decompose function `type=additive` to this new time series. Comment the results. Does the random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

```
filtered.ts.solar.energy.data<-
  ts(solar.wind.energy.data[,2], start=c(2012,1), end=c(2022,9), frequency=12)

filtered.ts.wind.energy.data<-
  ts(solar.wind.energy.data[,3], start=c(2012,1), end=c(2022,9),frequency=12)
```

```
decompose.solar.data3<-decompose(filtered.ts.solar.energy.data, type = "additive")
plot(decompose.solar.data3)
```




```
decompose.wind.data3<-decompose(filtered.ts.wind.energy.data, type="additive")
plot(decompose.wind.data3)
```



Answer:The solar series looks quite seasonal, with the exception of a big jump that corresponds the the upward trend seen in the observed and trend components. Wind appears seasonal at first, then tracks the observed component starting around 2018.