ENV 790.30 - Time Series Analysis for Energy Data | Spring 2024

Assignment 5 - Due date 02/13/24

Yilun Zhu

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

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr   1.1.4     v stringr 1.5.1
## v forcats 1.0.0     v tibble  3.2.1
## v purrr   1.0.2     v tidyr   1.3.1
## v readr   2.1.5
```

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 December 2023 Monthly Energy Review.

```
#Importing data set - using xlsx package
#Directly change Not available to NA
#Since I don't know how to simply make NA strings to NA
energy_data <- read.csv(file="./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.cs
#startRow is equivalent to skip on read.table

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

Q1

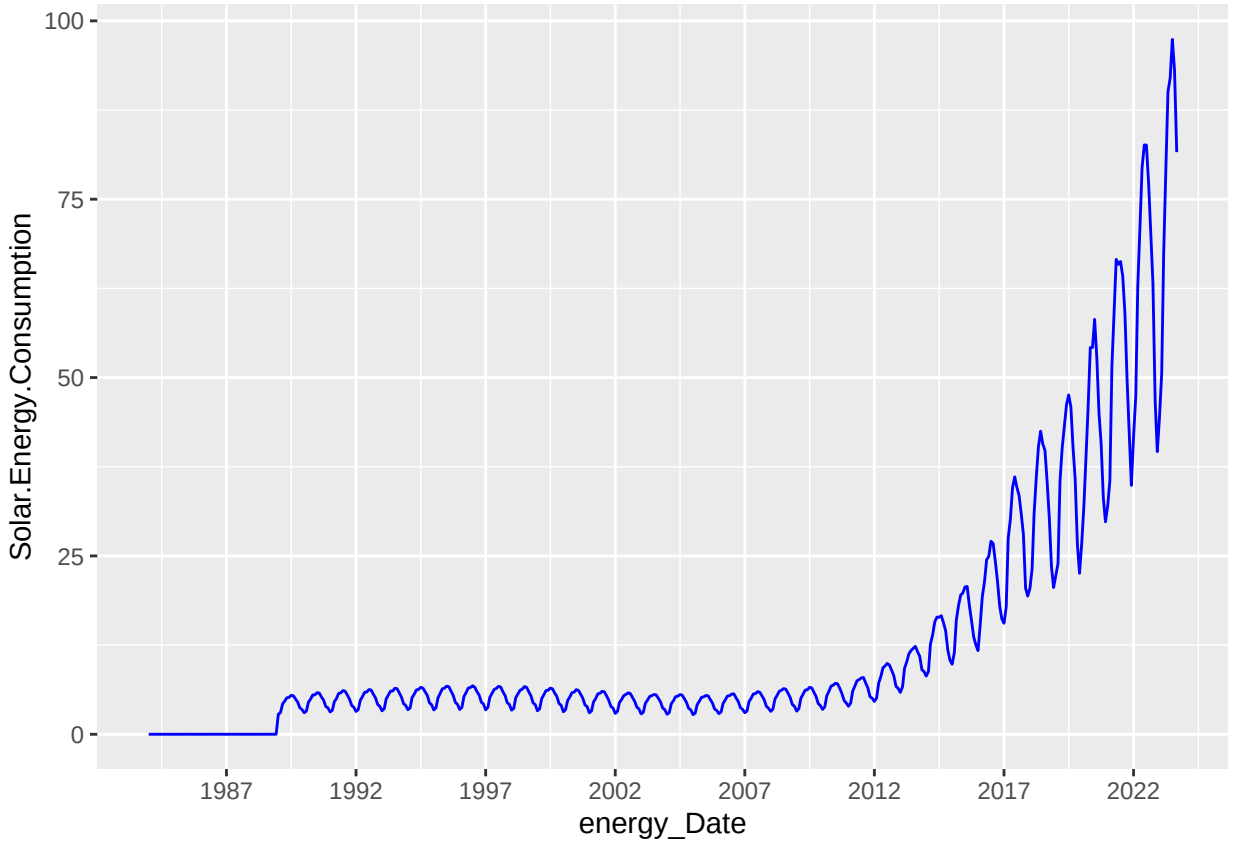
For this assignment you will work only with the following columns: Solar Energy Consumption and Wind Energy Consumption. Create a data frame structure with these two time series only and the Date column. Drop the rows with *Not Available* and convert the columns to numeric. You can use filtering to eliminate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

```
#extract two column
Twocol <- energy_data[,8:9]
#extract date column and transfer it to date value
energy_Date <- ym(energy_data[,1])
#Bind them to make a new data frame
Workdata <- cbind(energy_Date,Twocol)
nvar=ncol(Workdata)

#If didn't use na.strings in read.csv, use this from naniar package
#Workdata <- Workdata %>% replace_with_na(replace = list(Solar.Energy.Consumption = "Not Available", Wi

Cleandata <- na.omit(Workdata)
```

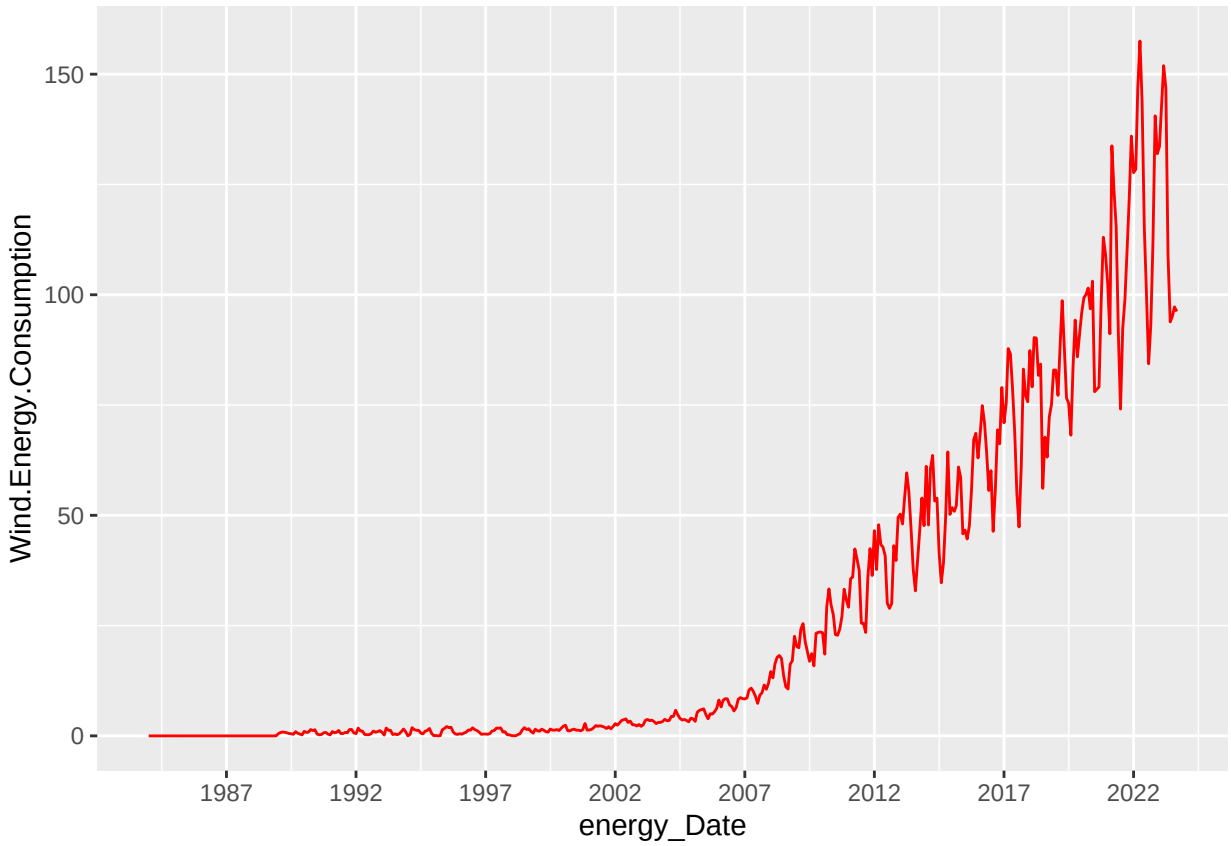
Q2

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(Cleandata, aes(x= energy_Date))+
  geom_line(aes(y=Cleandata[,2]),color = "blue")+
  ylab("Solar.Energy.Consumption")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



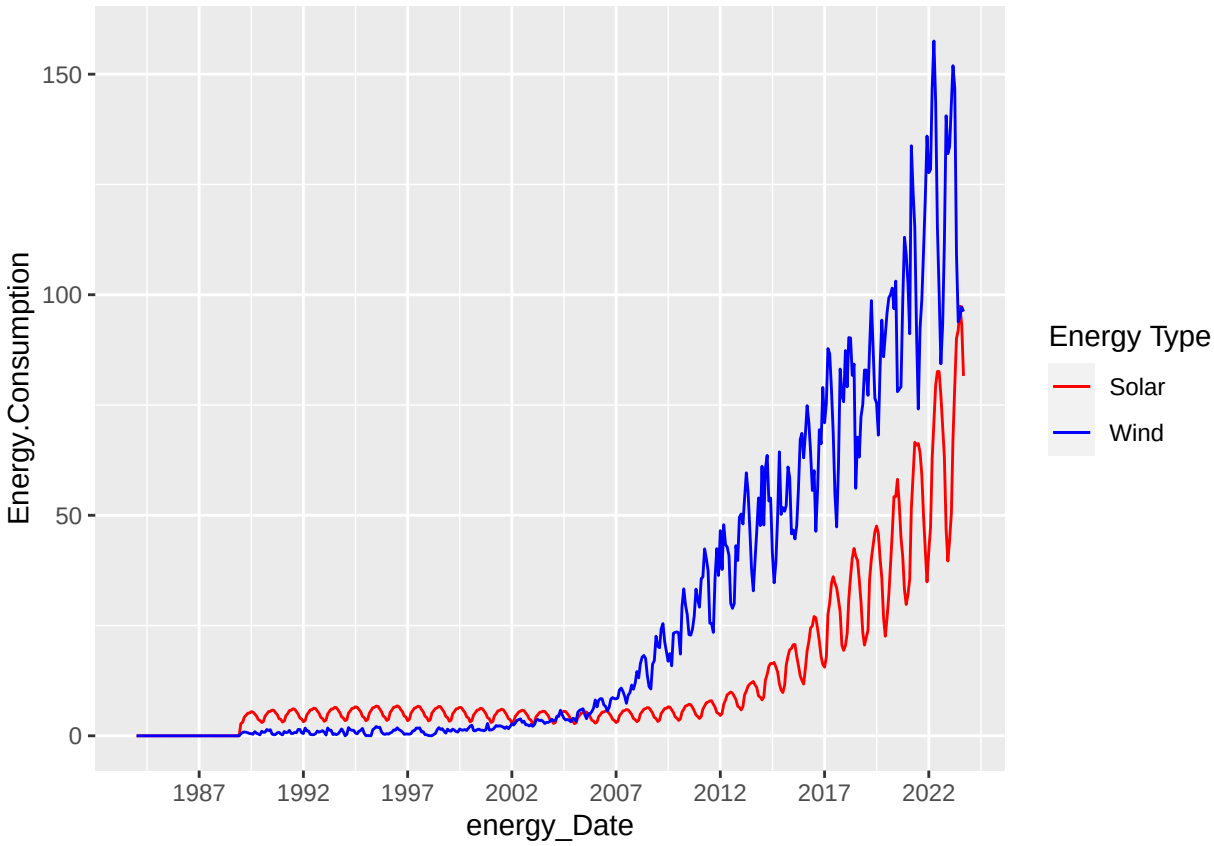

```
ggplot(Cleandata, aes(x= energy_Date))+
  geom_line(aes(y=Cleandata[,3]),color = "red")+
  ylab("Wind.Energy.Consumption")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



Q3

Now plot both series in the same graph, also using ggplot(). Use function `scale_color_manual()` 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()` to set x axis breaks every 5 years.

```
#Here since color is defined 'Solar', so it is in aes()
#Instead of "aes(y=xxx),color='solar' (which is what we did before)
ggplot(Cleandata, aes(x= energy_Date))+
  geom_line(aes(y=Cleandata[,2],color = "Solar"))+
  geom_line(aes(y=Cleandata[,3],color = "Wind"))+
  ylab("Energy.Consumption")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")+
  scale_color_manual(name = "Energy Type",
                     breaks = c("Solar","Wind"),
                     values = c("Solar"= "red","Wind" = "blue"))
```



Decomposing the time series

The stats package has a function called decompose(). This function only take time series object. As the name says the decompose function will decompose your time series into three components: trend, seasonal and random. This is similar to what we did in the previous script, but in a more automated way. The random component is the time series without seasonal and trend component.

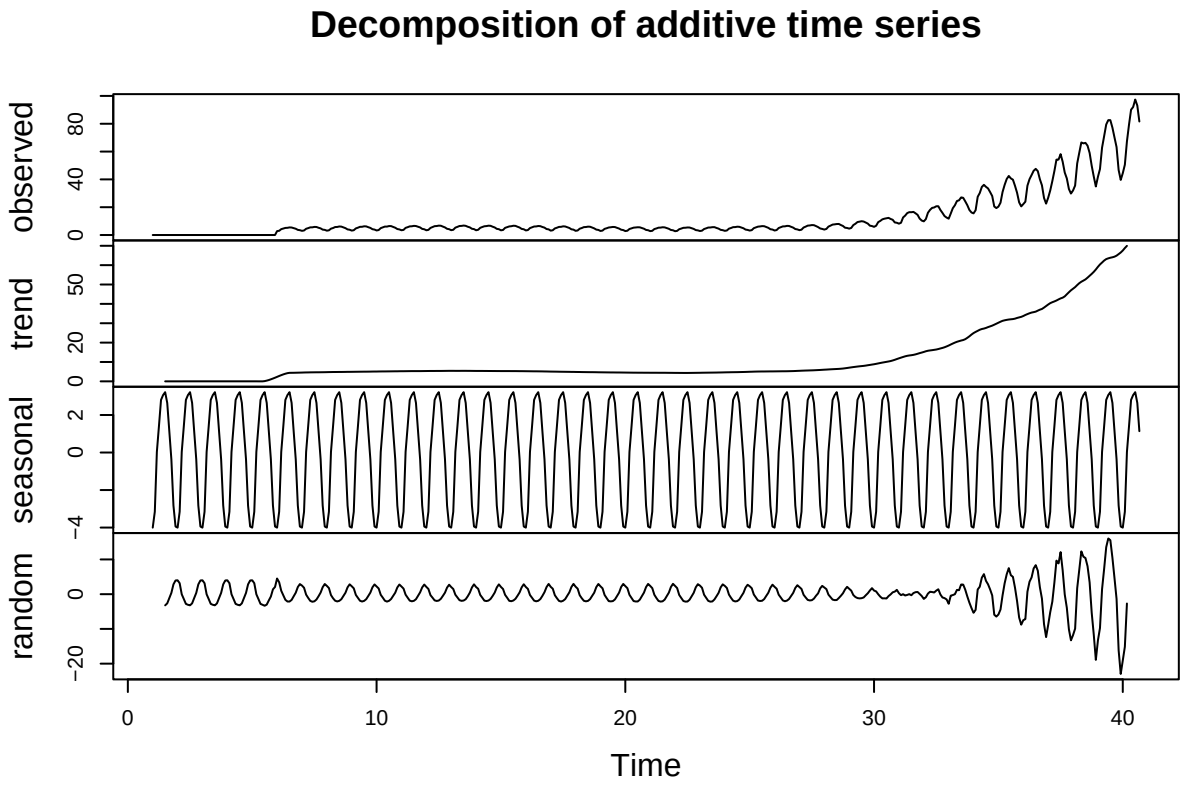
Additional info on `decompose()`.

1) You have two options: alternative and multiplicative. Multiplicative models exhibit a change in frequency over time.
2) The trend is not a straight line because it uses a moving average method to detect trend.
3) The seasonal component of the time series is found by subtracting the trend component from the original data then grouping the results by month and averaging them.
4) The random component, also referred to as the noise component, is composed of all the leftover signal which is not explained by the combination of the trend and seasonal component.

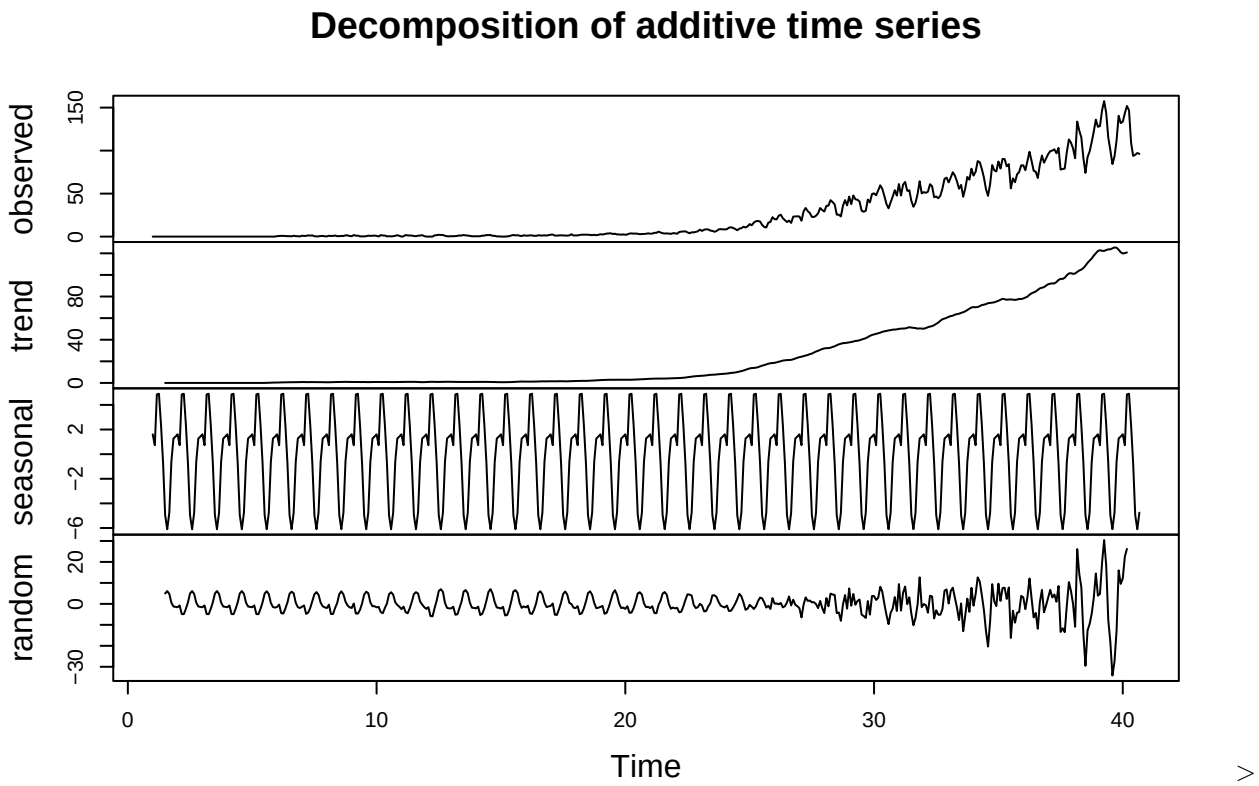
Q4

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?

```r
ts_Work <- ts(Cleandata[,2:3],frequency = 12)
decompose_Solar=decompose(ts_Work[,1],"additive")
decompose_Wind=decompose(ts_Work[,2],"additive")
plot(decompose_Solar)
```




```r
plot(decompose_Wind)
```

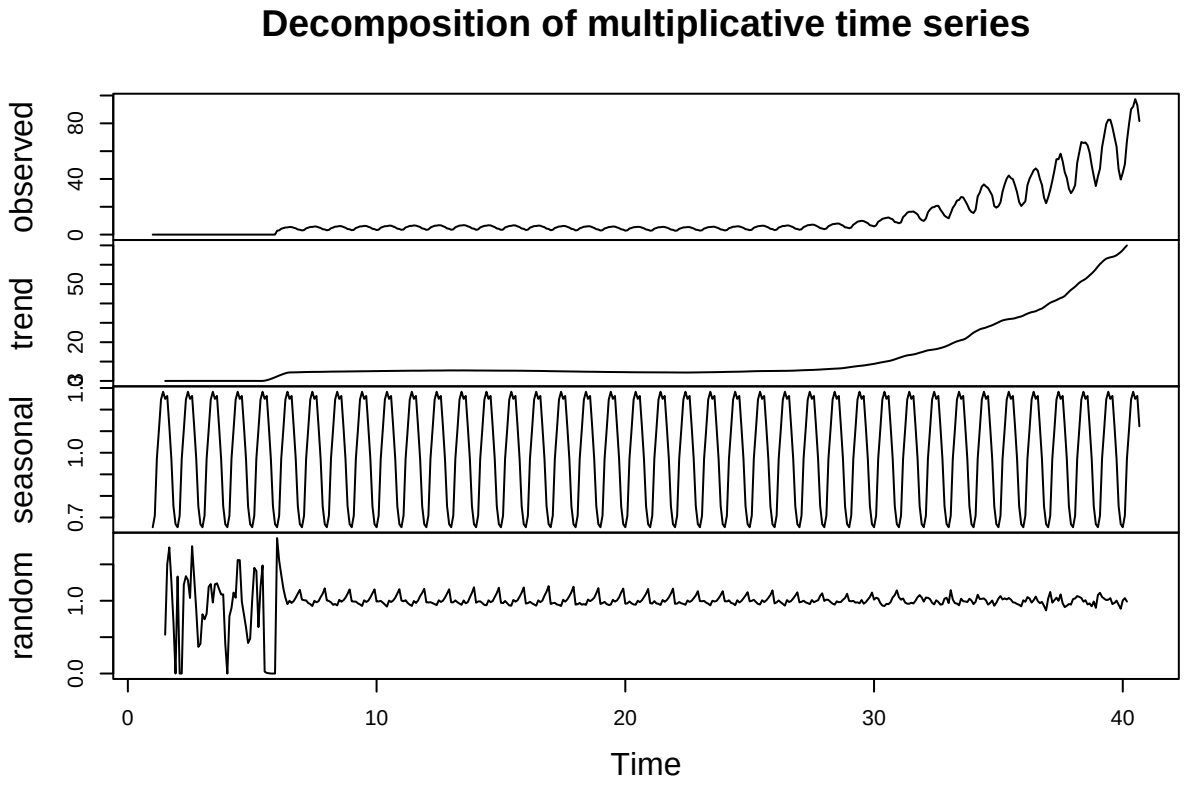


Answer: From the decomposed plot, it's obvious that both Solar and Wind energy consumption have seasonal trend. Solar has an increasing trend since about 30 years after 1984 and wind has an increasing trend since about 20 years after 1984. About random trend, both plots still show some seasonalities.

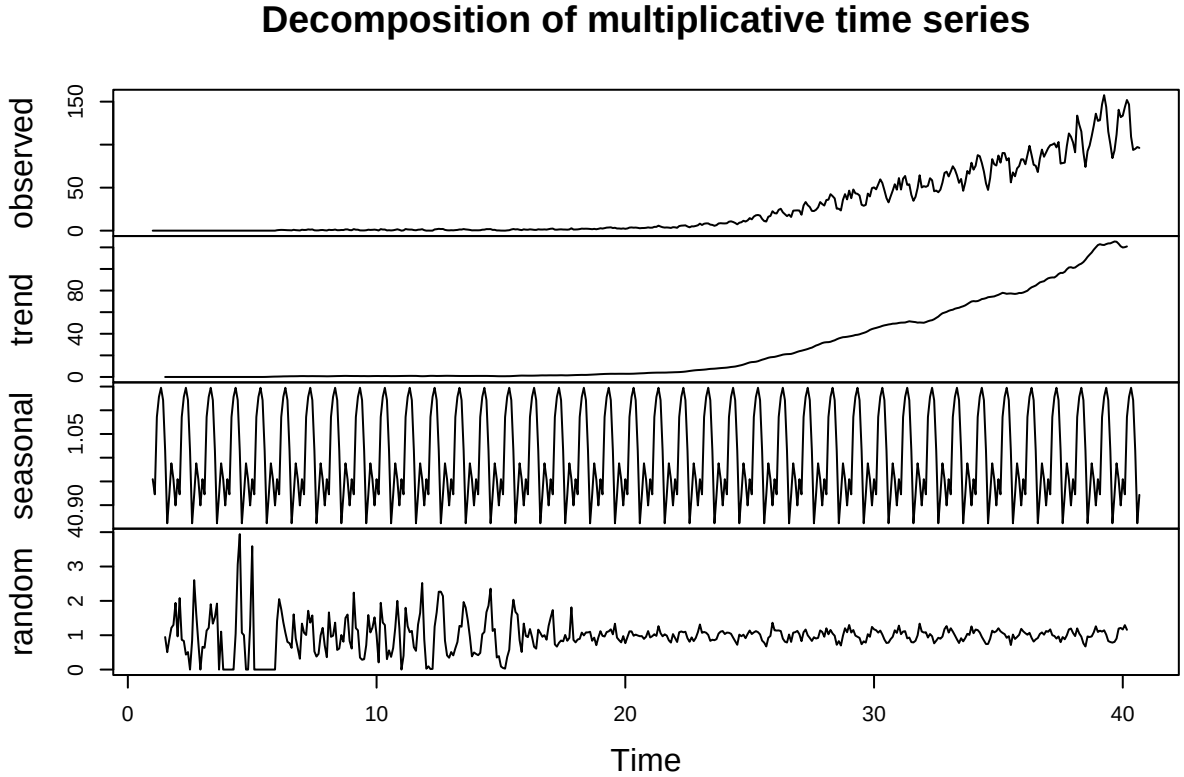
Q5

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=decompose(ts_Work[,1],"multiplicative")
decompose_Wind=decompose(ts_Work[,2],"multiplicative")
plot(decompose_Solar)
```




```
plot(decompose_Wind)
```



>

Answer: This time random components of both plots are much better but still have some seasonalities in some period.

Q6

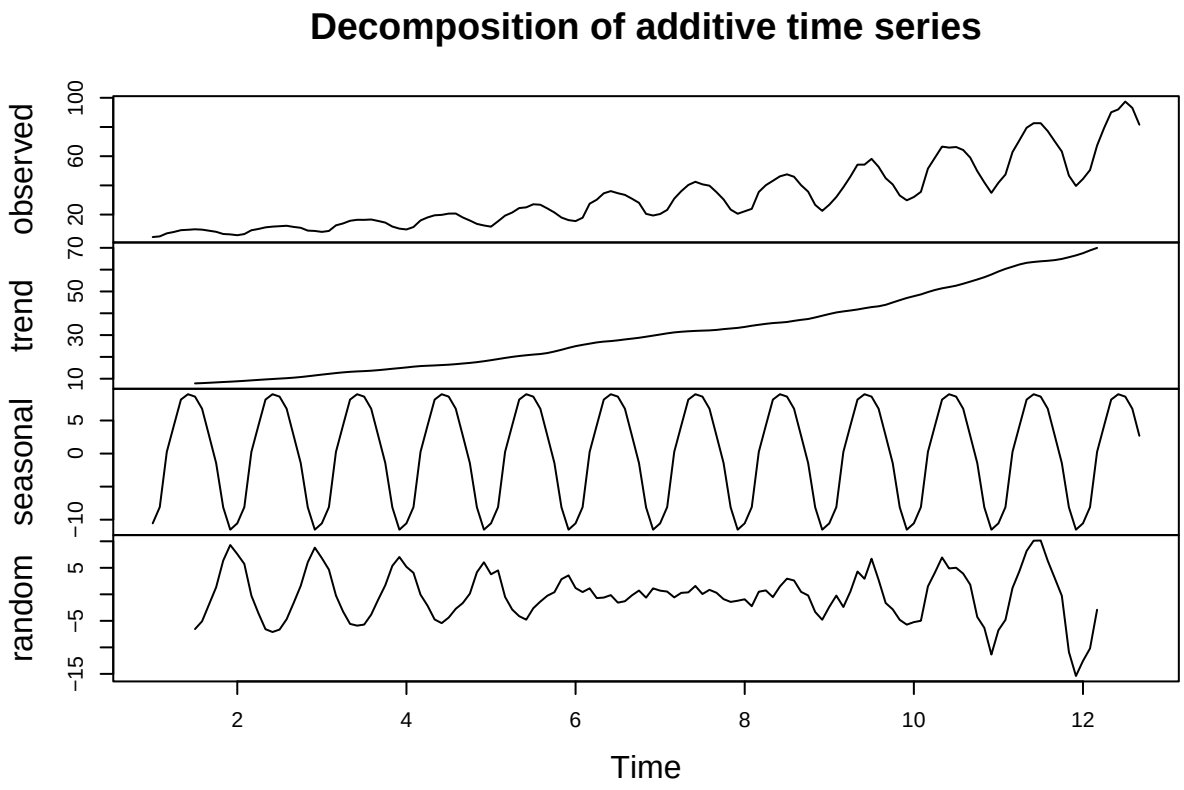
When fitting a model to this data, do you think you need all the historical data? Think about the data from 90s and early 20s. Are there any information from those years we might need to forecast the next six months of Solar and/or Wind consumption. Explain your response.

> Answer: Since there is a long period from 1984 to 10s that SEC and WEC follow flat trend, which is totally different when it comes to 2010s, we may not need all the historical data if we want to forecast future SEC and WEC. From data after 2010s, certainly including early 20s, we mainly need the new trend components in this 10 years, as well as old seasonal trend and random trend.

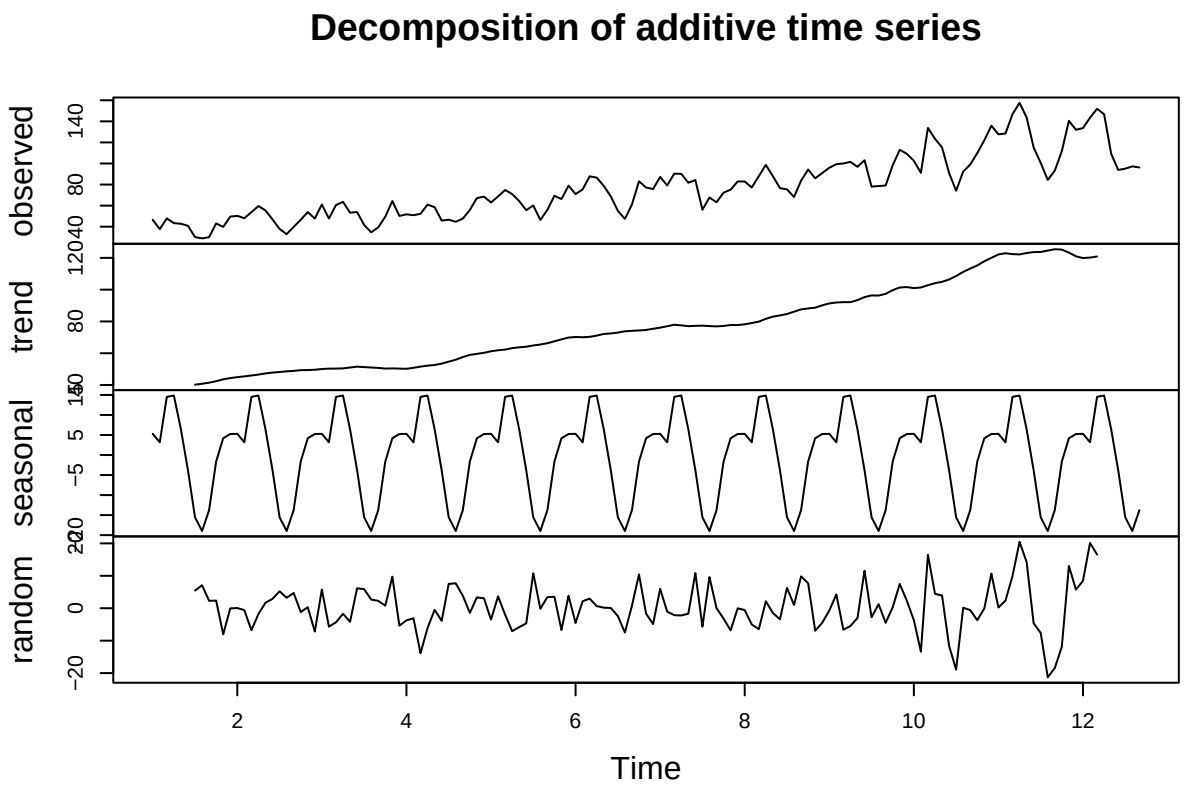
Q7

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.

```
data_a2012 <- dplyr::filter(Cleandata, year(energy_Date) >= 2012 )
ts_a2012 <- ts(data_a2012[,2:3],frequency = 12)

decompose_a2012Solar=decompose(ts_a2012[,1],"additive")
decompose_a2012Wind=decompose(ts_a2012[,2],"additive")
plot(decompose_a2012Solar)
```



```r
plot(decompose_a2012Wind)
```



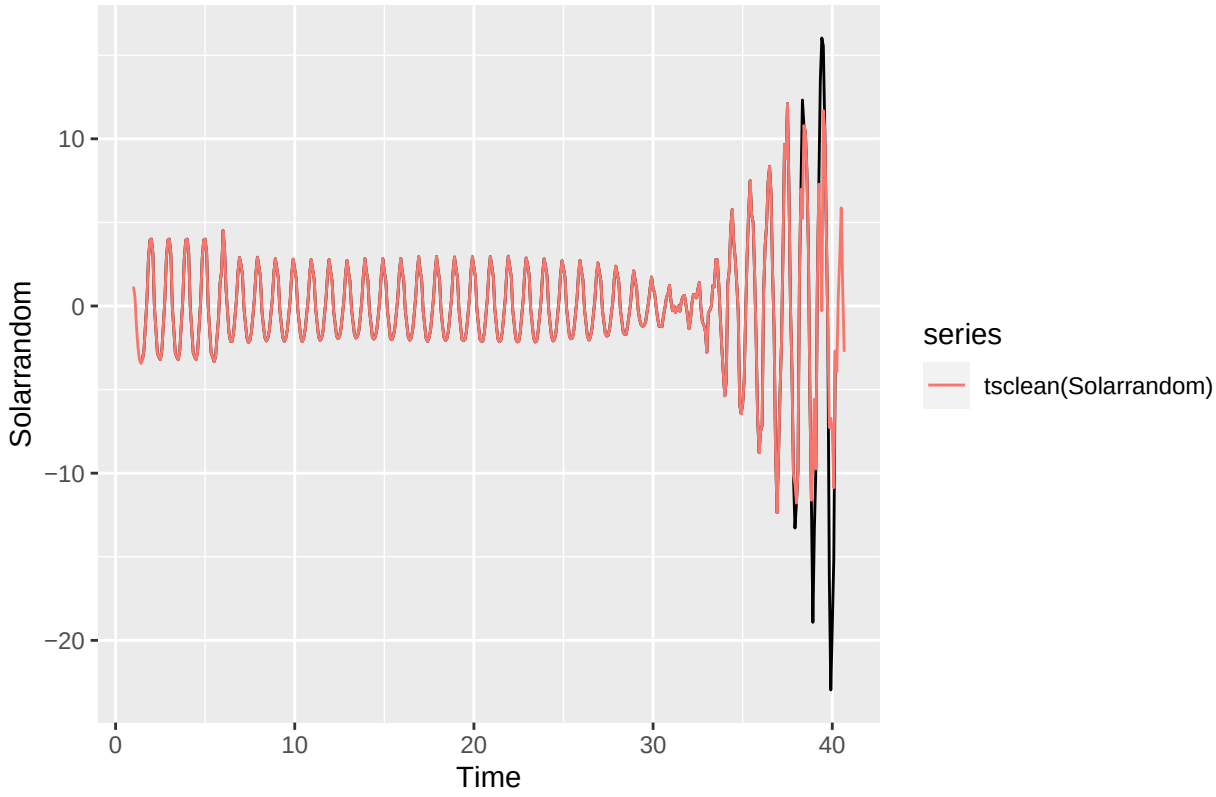
Answer: Random component of Wind is more random than before, but Solar still has partly seasonal trend in random component.

Identify and Remove outliers

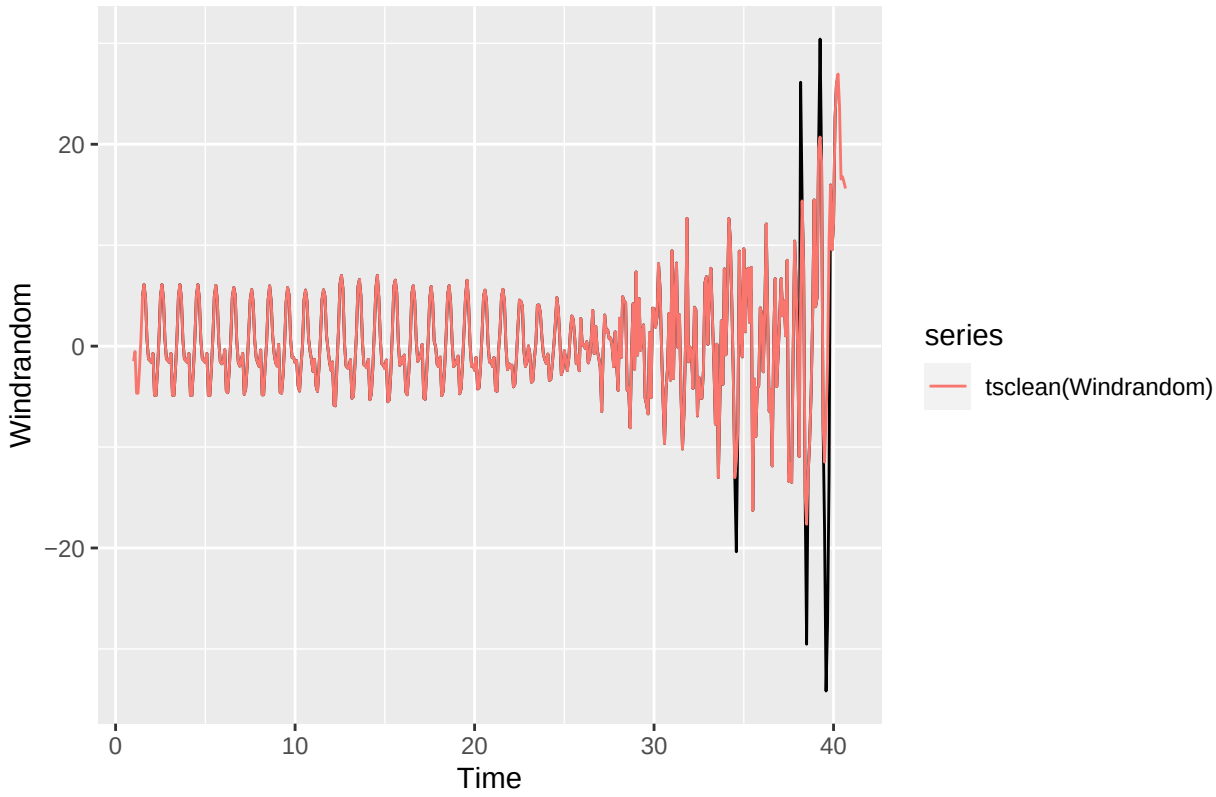
Q8

Apply the `tsclean()` to both series from Q7. Did the function removed any outliers from the series? Hint: Use `autoplot()` to check if there is difference between cleaned series and original series.

```r
ts_clean <- ts(Cleandata[,2:3], frequency = 12)
decompose_Solar=decompose(ts_clean[,1],"additive")
decompose_Wind=decompose(ts_clean[,2],"additive")
Solarrandom <- decompose_Solar$random
Windrandom <- decompose_Wind$random

autoplot(Solarrandom)+
  autolayer(tsclean(Solarrandom))
```




```
autoplot(Windrandom)+
  autolayer(tsclean(Windrandom))
```

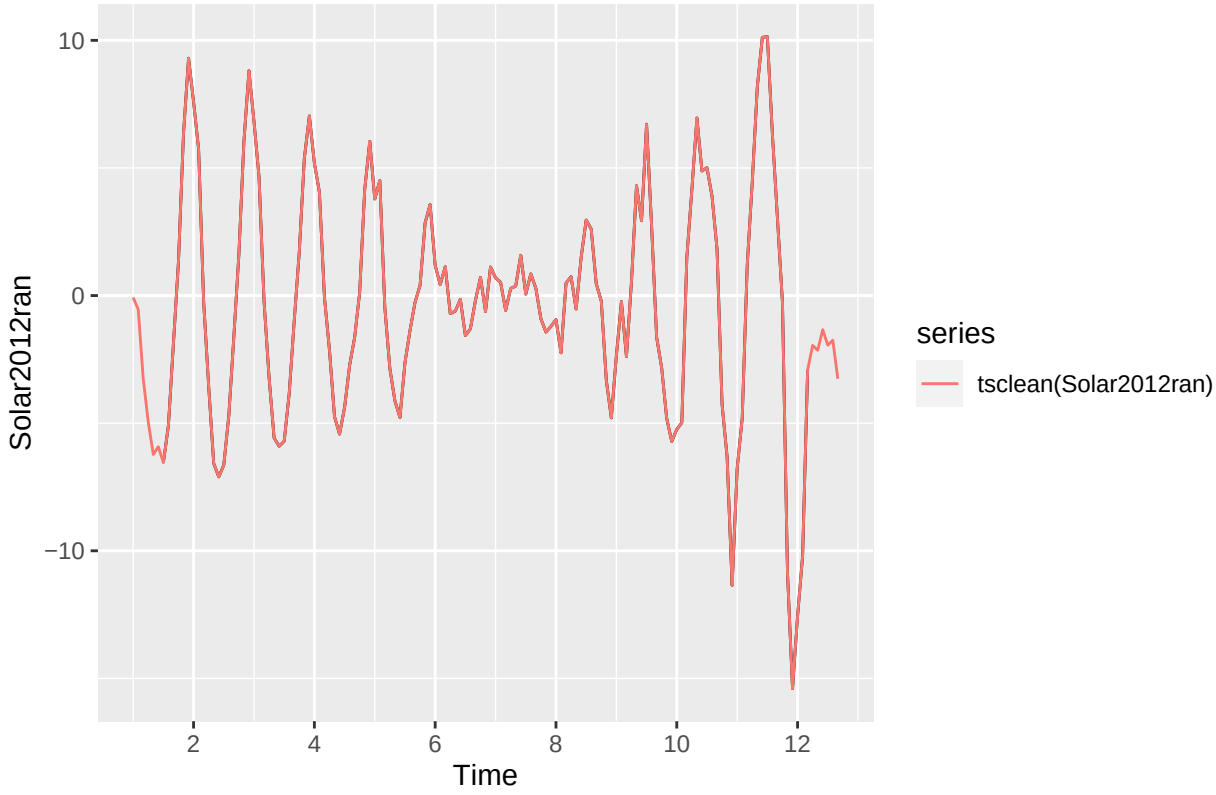


Answer: Outliers between 30 to 40 is removed.

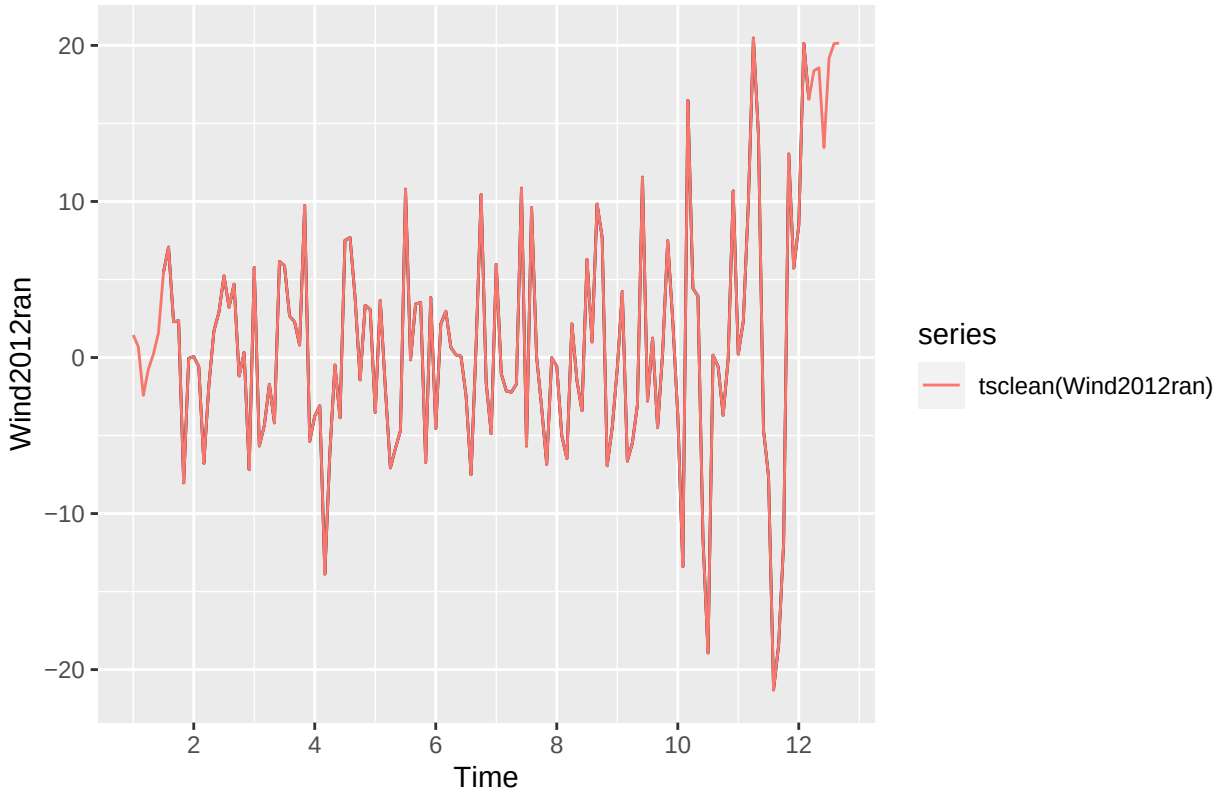
Q9

Redo number Q8 but now with the time series you created on Q7, i.e., the series starting in 2014. Using what `autoplot()` again what happened now?Did the function removed any outliers from the series?

```
Solar2012ran <- decompose_a2012Solar$random
Wind2012ran <- decompose_a2012Wind$random
CleanSolar2012 <- tsclean(Solar2012ran)
CleanWind2012 <- tsclean(Wind2012ran)

autoplot(Solar2012ran)+
  autolayer(tsclean(Solar2012ran))
```




```
autoplot(Wind2012ran)+
  autolayer(tsclean(Wind2012ran))
```



Answer: No outlier is removed. Since the period is changed, so some value in 1984-2024 is out of normal distribution but is fine in 2012-2024.