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

Assignment 5 - Due date 02/13/24

Vincient Whatley

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.\

```r
#Load/install required package here
library(forecast)
```

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

```r
library(tseries)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.3.2
```

```r
library(ggplot2)
library(Kendall)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
library(tidyverse)  #load this package so yon clean the data frame using pipes
```

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

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library("xlsx")
```

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.

```r
#Importing data set - using xlsx package
energy_data <- read.xlsx(file="./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.x



#Now let's extract the column names from row 11 only
read_col_names <- read.xlsx("./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xls

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                           219.839
## 2                         117.338                           197.330
## 3                         129.938                           218.686
## 4                         125.636                           209.330
## 5                         129.834                           215.982
## 6                         125.611                           208.249
##   Hydroelectric Power Consumption Geothermal Energy Consumption
## 1                          89.562                         0.490
## 2                          79.544                         0.448
## 3                          88.284                         0.464
## 4                          83.152                         0.542
## 5                          85.643                         0.505
## 6                          82.060                         0.579
##   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                            219.839
## 2                          117.338                            197.330
## 3                          129.938                            218.686
## 4                          125.636                            209.330
## 5                          129.834                            215.982
## 6                          125.611                            208.249
```

```r
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!

```r
#Filter for Solar and Wind Collumns
#Removing unnecessary columns, renaming columns
energy_data <- energy_data[,c(1, 8, 9)]
colnames(energy_data)=c("Date",  "Solar Energy Consumption", "Wind Energy Consumption")

#Convert columns to numeric and replace "Not Available" with NA
energy_data <- energy_data %>%
  mutate(`Solar Energy Consumption` = if_else(`Solar Energy Consumption` == "Not     Available", NA_real
    `Wind Energy Consumption` = if_else(`Wind Energy Consumption`=="Not            Available", NA_real_
     drop_na()
```

```
## Warning: There were 2 warnings in `mutate()`.
## The first warning was:
## i In argument: `Solar Energy Consumption = if_else(...)`.
## Caused by warning in `if_else()`:
## ! NAs introduced by coercion
## i Run `dplyr::last_dplyr_warnings()` to see the 1 remaining warning.
```

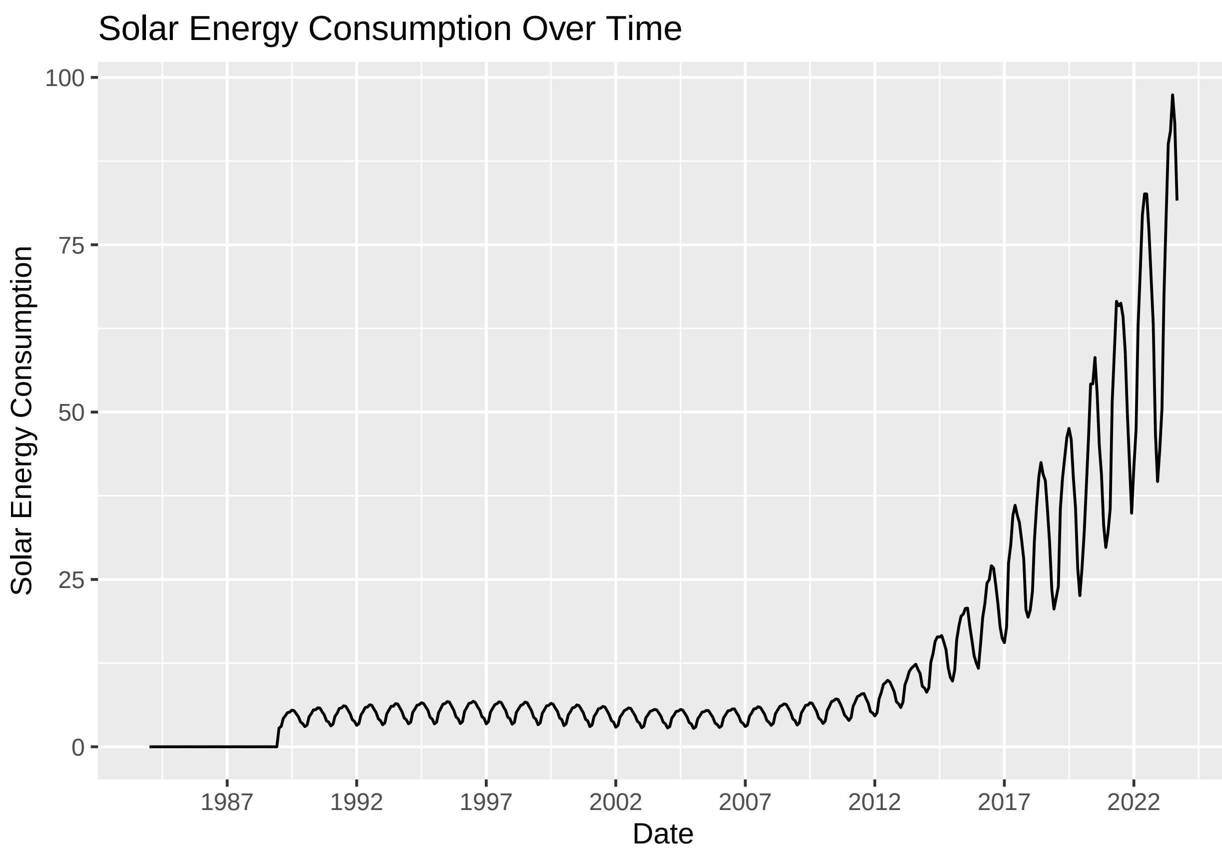
```r
#Converting to time series objects
energy_data$`Solar Energy Consumption` <- ts(energy_data$`Solar Energy Consumption`, frequency = 1)
energy_data$`Wind Energy Consumption` <- ts(energy_data$`Wind Energy Consumption`, frequency = 1)
```

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")")`

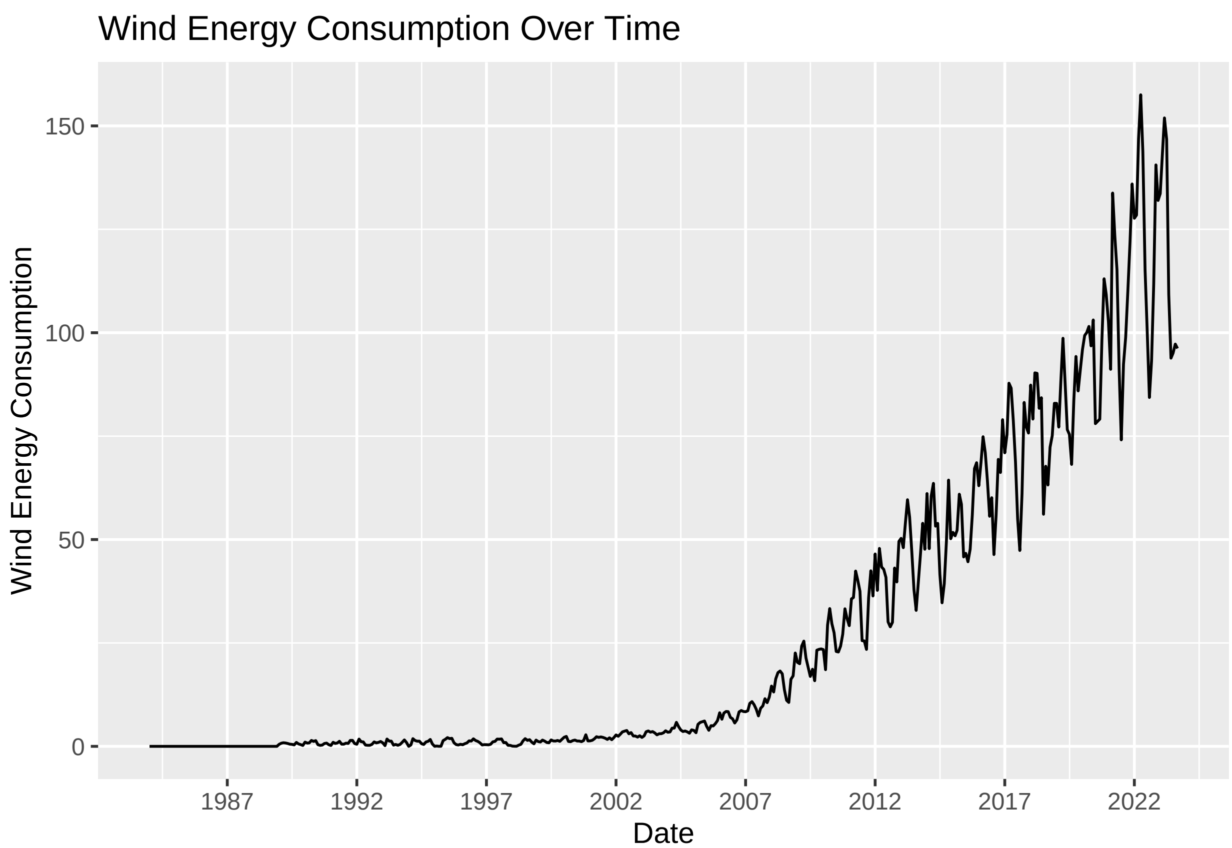
```r
#Plot for Solar Energy Consumption
ggplot(energy_data, aes(x = Date, y = `Solar Energy Consumption`)) +
  geom_line() +
  ylab("Solar Energy Consumption") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  ggtitle("Solar Energy Consumption Over Time")
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```




```r
#Plot for Wind Energy Consumption
ggplot(energy_data, aes(x = Date, y = `Wind Energy Consumption`)) +
  geom_line() +
  ylab("Wind Energy Consumption") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  ggtitle("Wind Energy Consumption Over Time")
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```

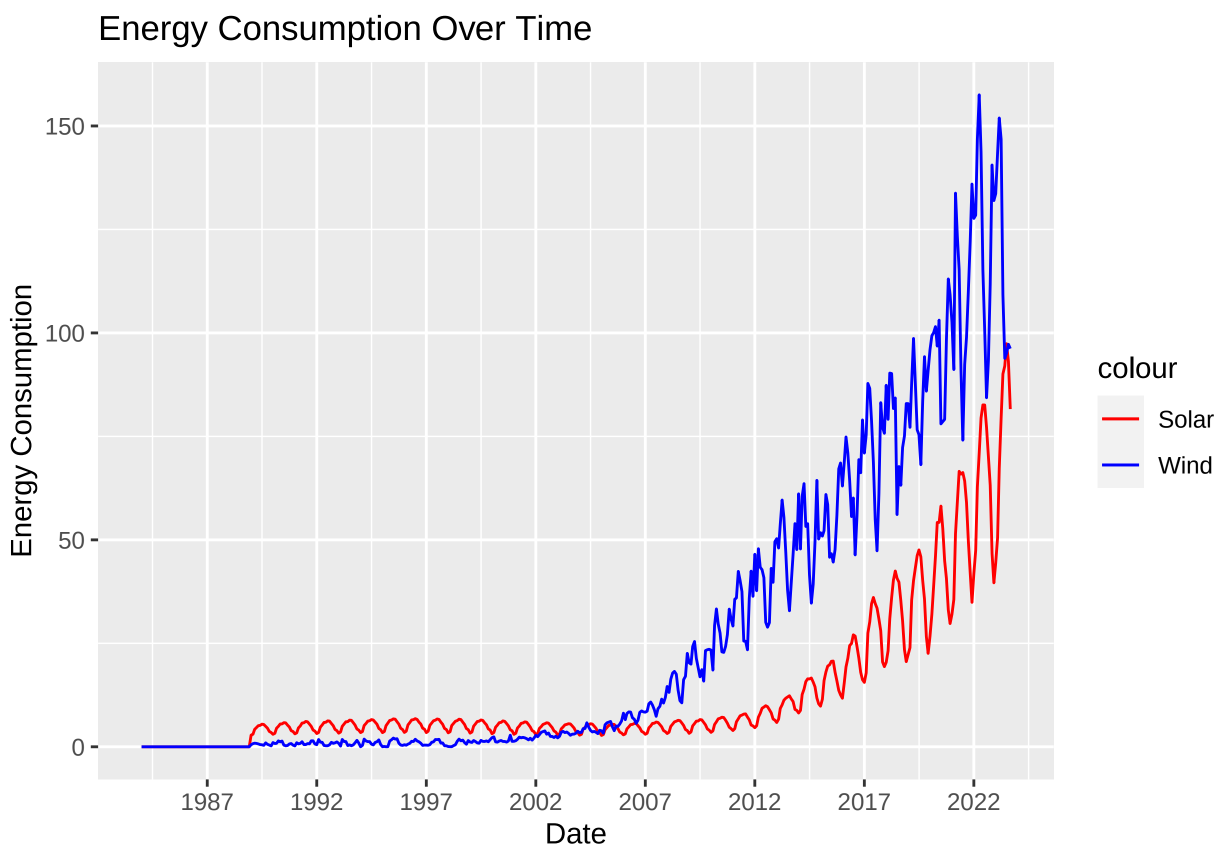


Q3

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.

```
ggplot(energy_data, aes(x=Date))+
  geom_line(aes(y=`Solar Energy Consumption`, color="Solar"))+
  geom_line(aes(y=`Wind Energy Consumption`, color="Wind"))+
  scale_color_manual(values=c("Solar"="red", "Wind"="blue"))+
  ylab("Energy Consumption")+
  scale_x_date(date_breaks="5 years", date_labels="%Y")+
  ggtitle("Energy Consumption Over Time")
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```



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.

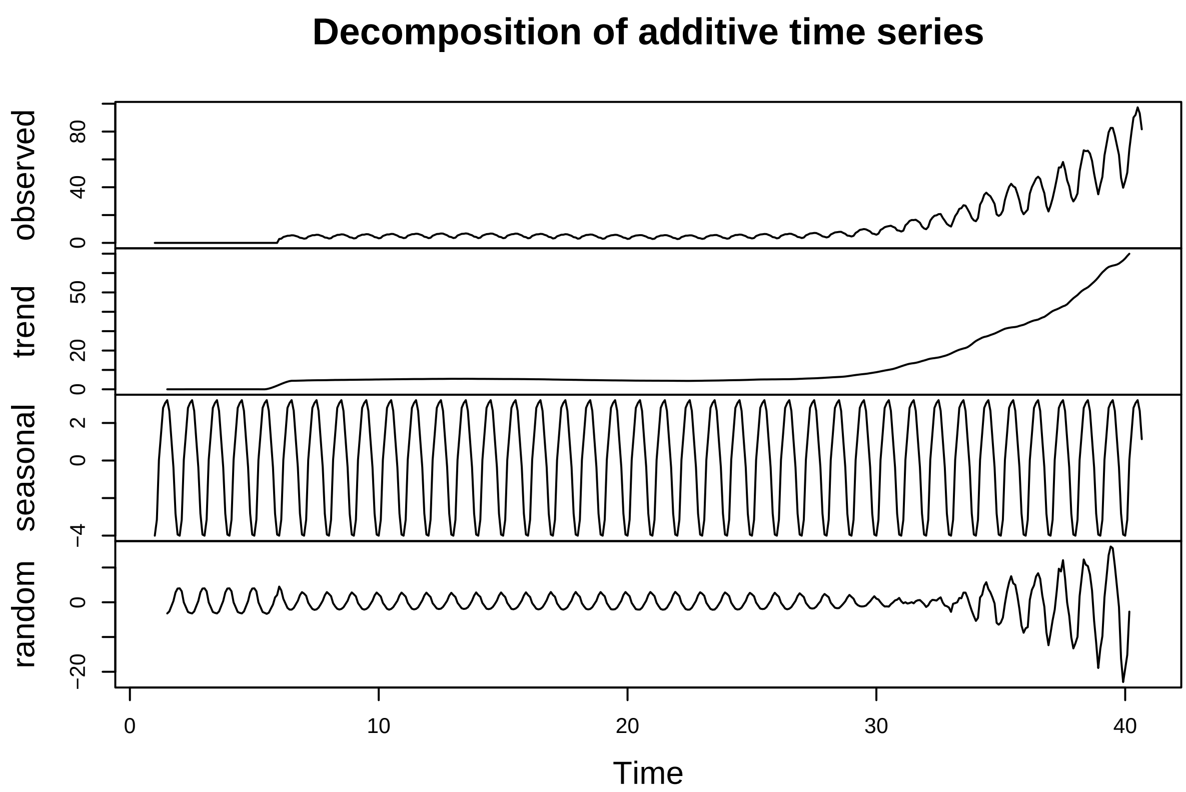
Additional info on `decompose()`.

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.

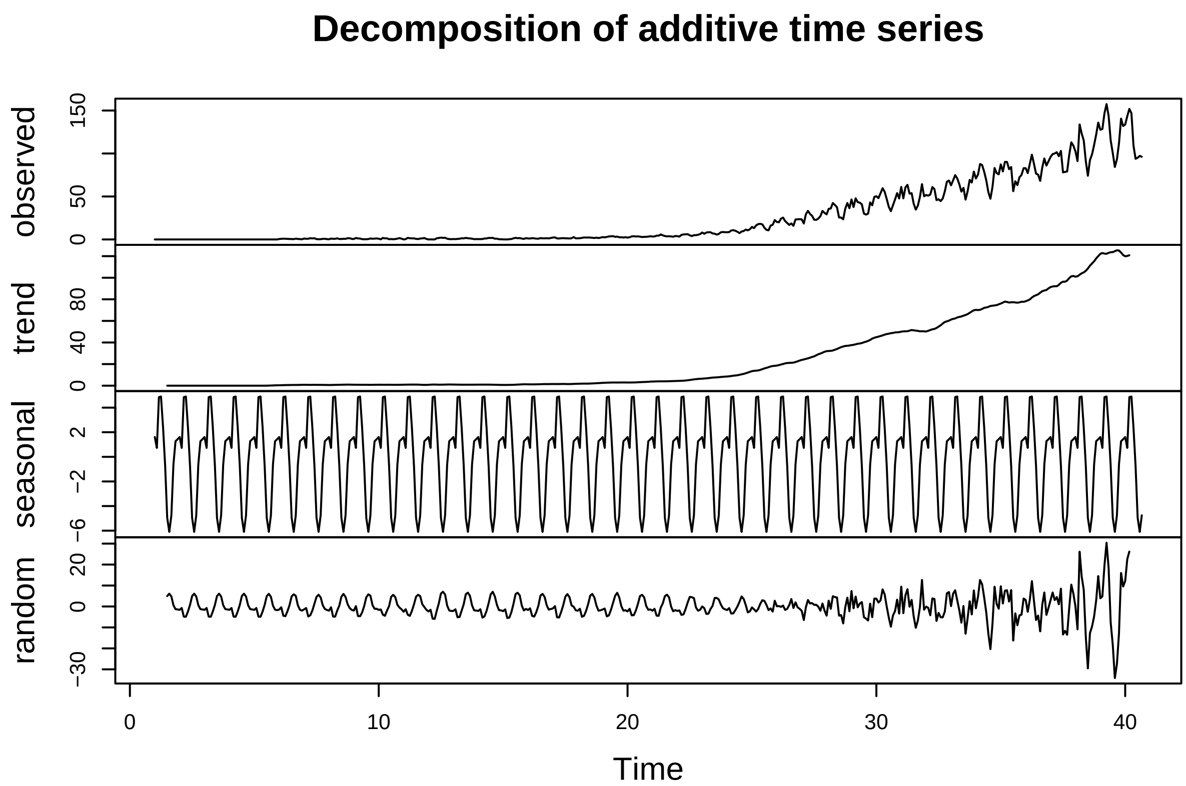
Q4

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?

```
#Convert "Solar Energy Consumption" and "Wind Energy Consumption" to time series objects

solarTs<- ts(energy_data$`Solar Energy Consumption`,frequency=12)
SolarDecomposed <-decompose(solarTs,type ="additive")

plot(SolarDecomposed)
```




```
windTs <-ts(energy_data$`Wind Energy Consumption`,frequency=12)
WindDecomposed <-decompose(windTs,type="additive")

plot(WindDecomposed)
```

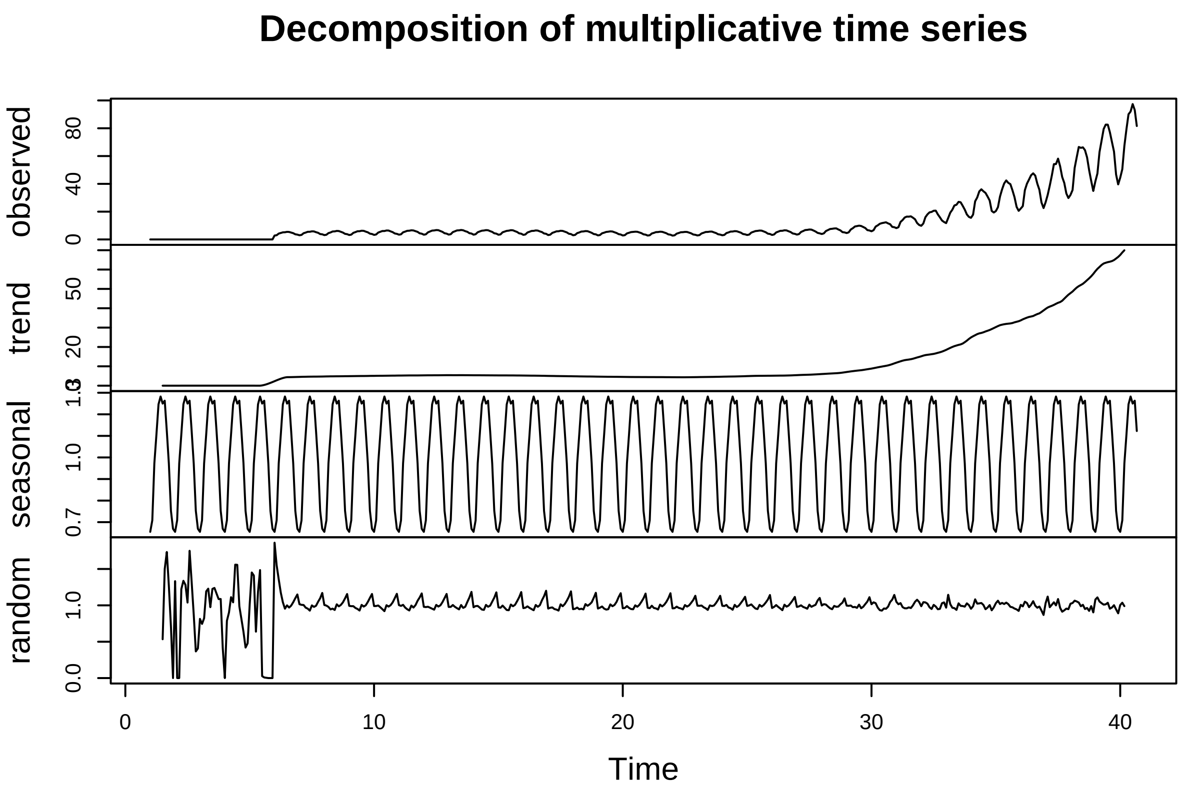


> Answer:For solar data and wind data they both demostrate an exponentilay potive trend which peaks towards the end of the data set. For both data sets though the flucuations only occur at a single time period in the data. While the rest remain sysmtreic suggesting that their that there is still some seasonality in the data set that wasnt yet removed.

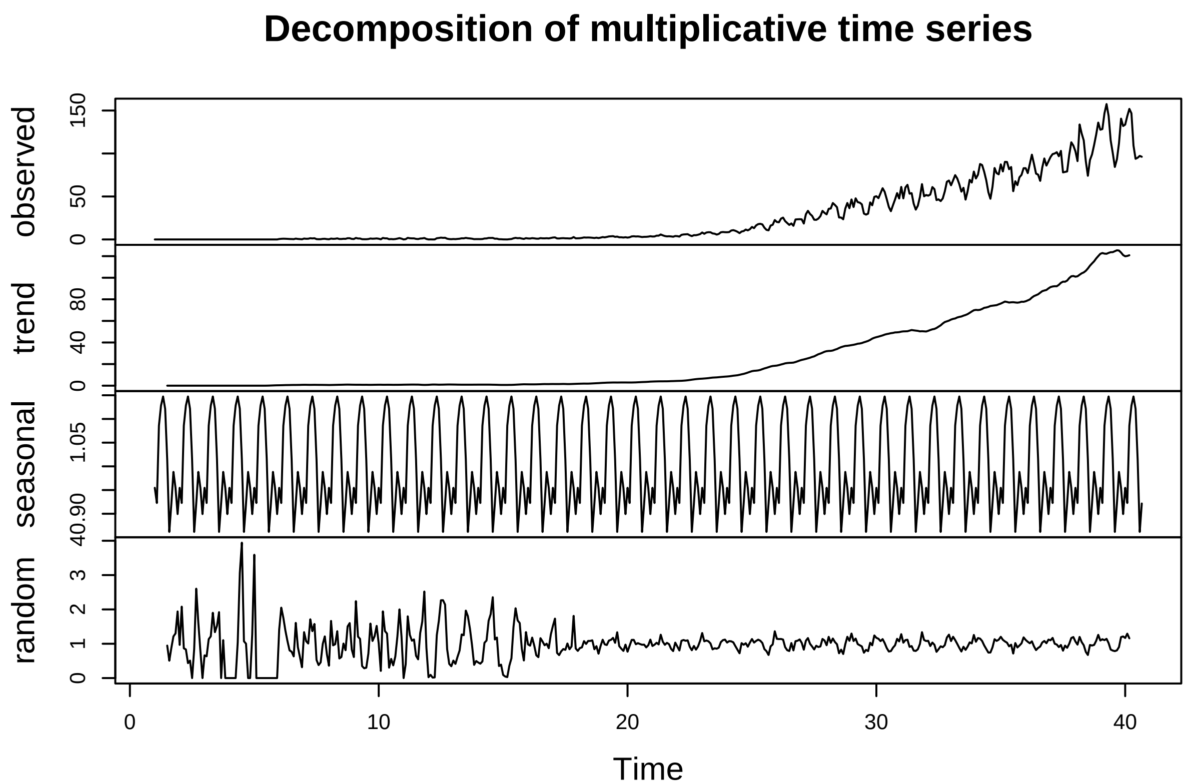
Q5

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?

```
SolarDecomposed2 <-decompose(solarTs,type="multiplicative")
plot(SolarDecomposed2)
```




```r
WindDecomposed2 <-decompose(windTs, type="multiplicative")
plot(WindDecomposed2)
```



Answer:For the solar data we see some apparent randomeness towards the beginning and end of the dataset. While the middle still displays seasonality. As for the wind data A significant portion shows some erratic flucuations displaying that we might captured the random component. But there is also a slight seasonality that should still be investigated further.

Q6

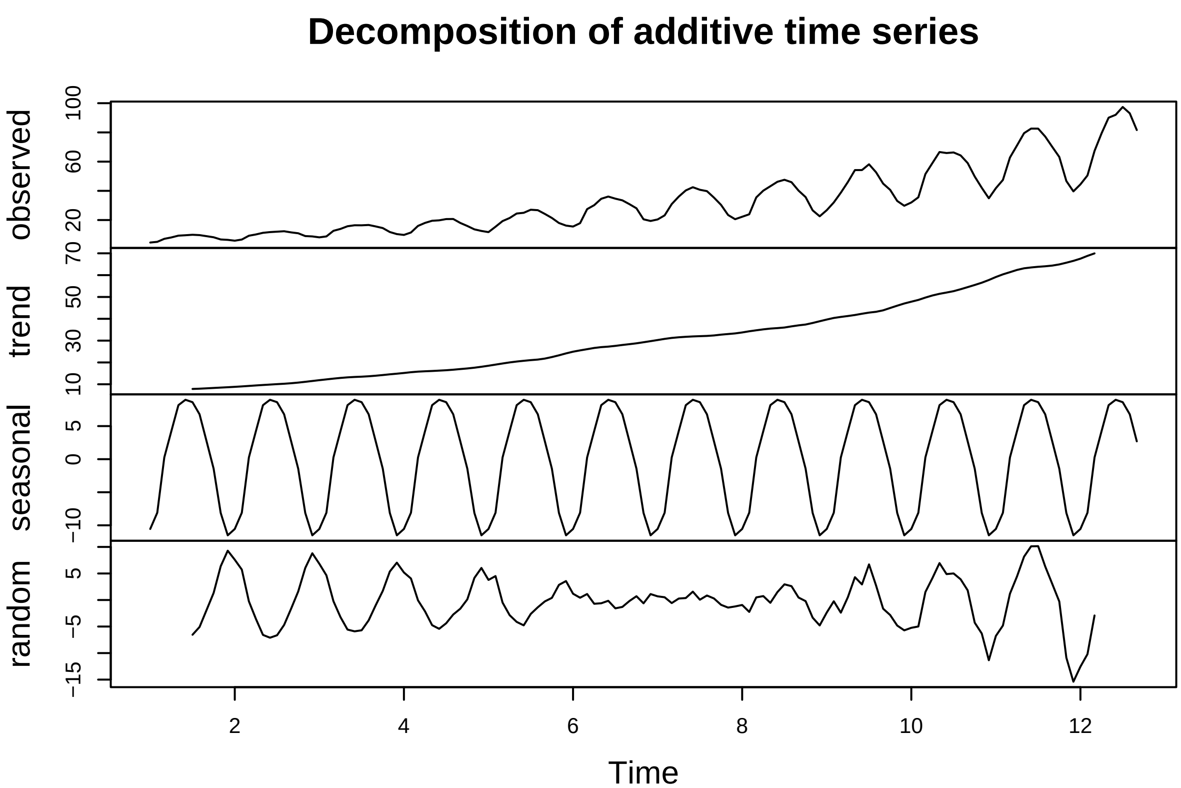
When fitting a model to this data, do you think you need all the historical data? Think about the data from 90s and early 20s. Are there any information from those years we might need to forecast the next six months of Solar and/or Wind consumption. Explain your response.

Answer: Not necessarily, at itmes weather event that occured in those years could be what is messing up the data. When looking at this data we see that it is not starting to change a flucate even more than before.

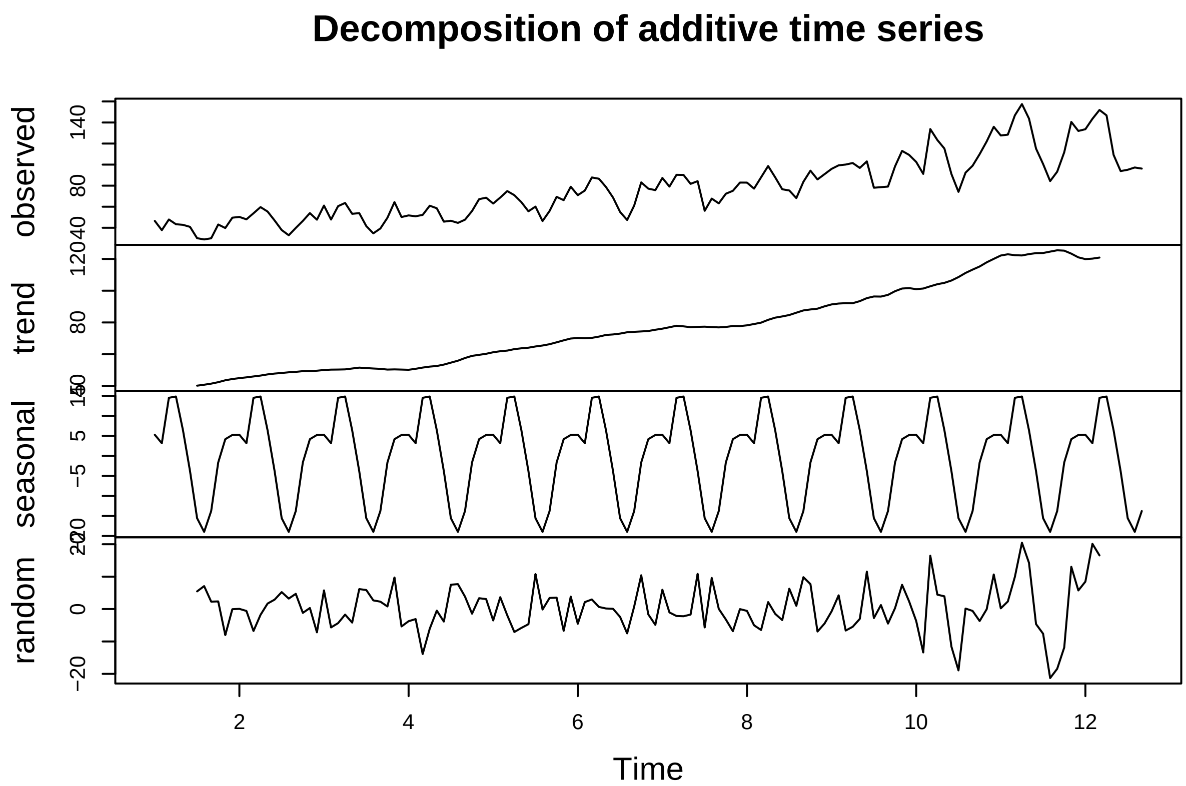
Q7

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.

```r
#Filter to start at January 2012
energydataNEW<-energy_data %>%
  filter(year(Date) >= 2012)

#Convert filtered data to time series object
#Apply decompose function with additive option
solarTsNEW<- ts(energydataNEW$`Solar Energy Consumption`,frequency=12)
solardecomposed2012 <- decompose(solarTsNEW, type="additive")
plot(solardecomposed2012)
```




```r
windTsNEW<- ts(energydataNEW$`Wind Energy Consumption`, frequency=12)
winddecomposed2012<- decompose(windTsNEW,type="additive")
plot(winddecomposed2012)
```



Answer:Solar data demostrates more randomeness than previous plots, particullary the middile section, but still displays some seasonality. While, Wind data now displays the randomenss that we were looking for.Though the levels are diffrent this diffrence is what helped us remove the seasonality suggesting that what we needed to remove was within the years that we removed.

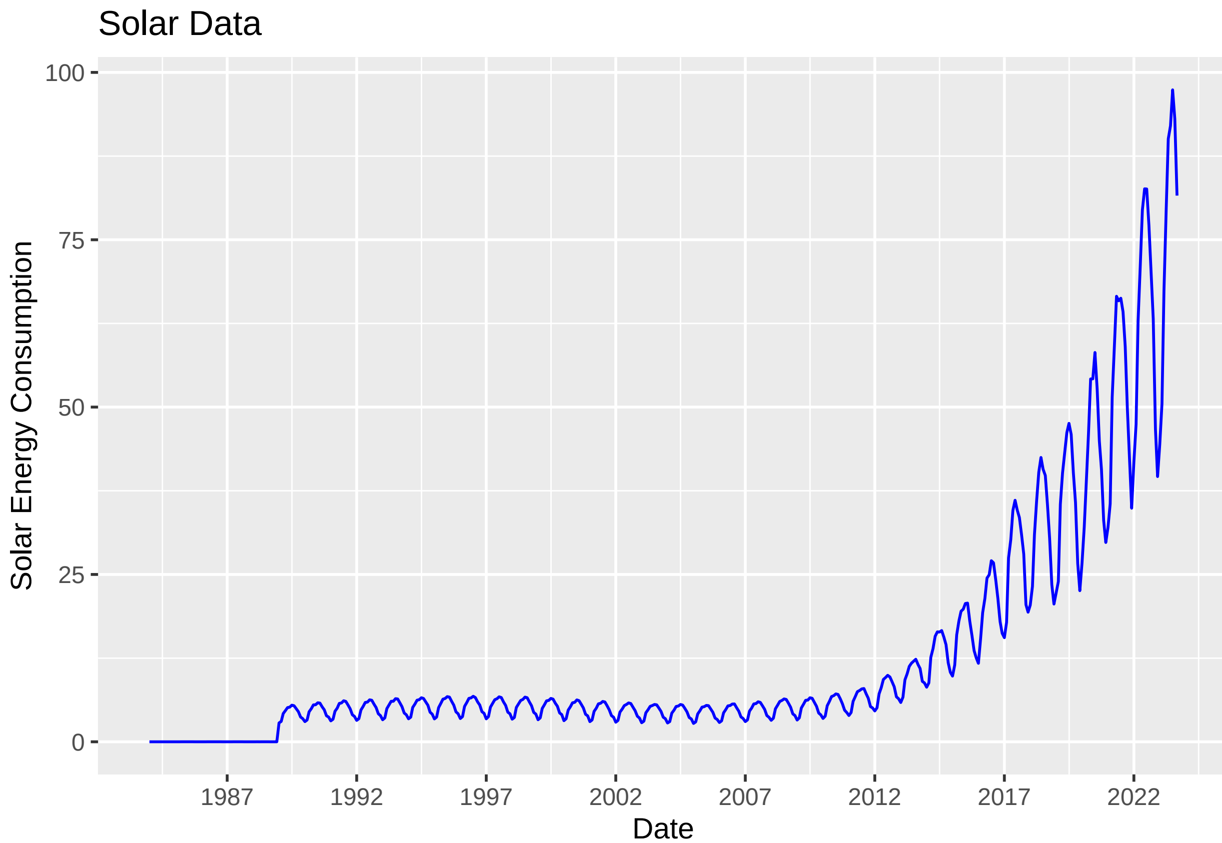
Identify and Remove outliers

Q8

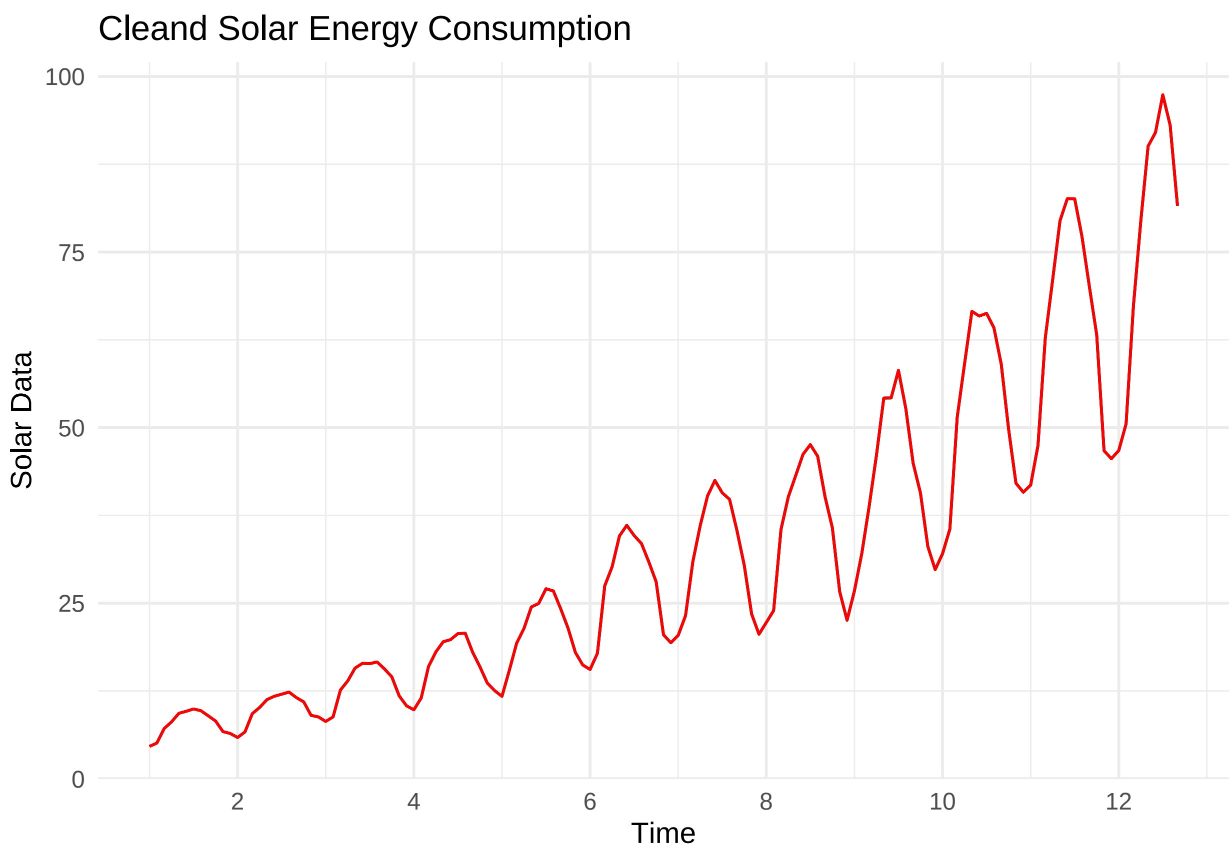
Apply the `tsclean()` to both series from Q7. Did the function removed any outliers from the series? Hint: Use `autoplot()` to check if there is difference between cleaned series and original series.

```
#Apply tsclean() to data
solar_cleaned <- tsclean(solarTsNEW)
wind_cleaned <- tsclean(windTsNEW)


ggplot(energy_data, aes(x=Date,y=`Solar Energy Consumption`))+
  geom_line(color="blue")+
  ylab("Solar Energy Consumption")+
  scale_x_date(date_breaks="5 years", date_labels="%Y")+
  ggtitle("Solar Data")
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```

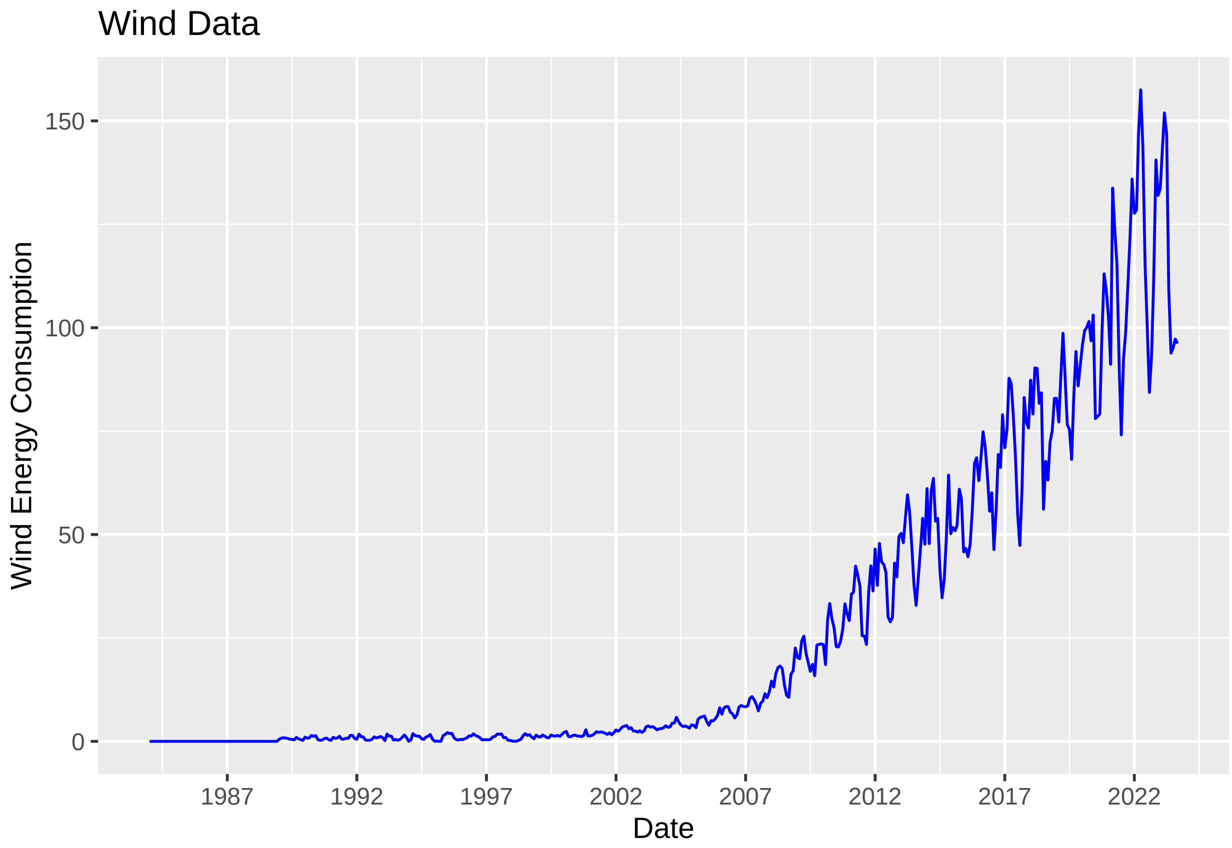



```r
#plot using autoplot
autoplot(solar_cleaned)+
  geom_line(color="red")+
  ggtitle("Cleand Solar Energy Consumption") +
  theme_minimal() +
  ylab("Solar Data")
```

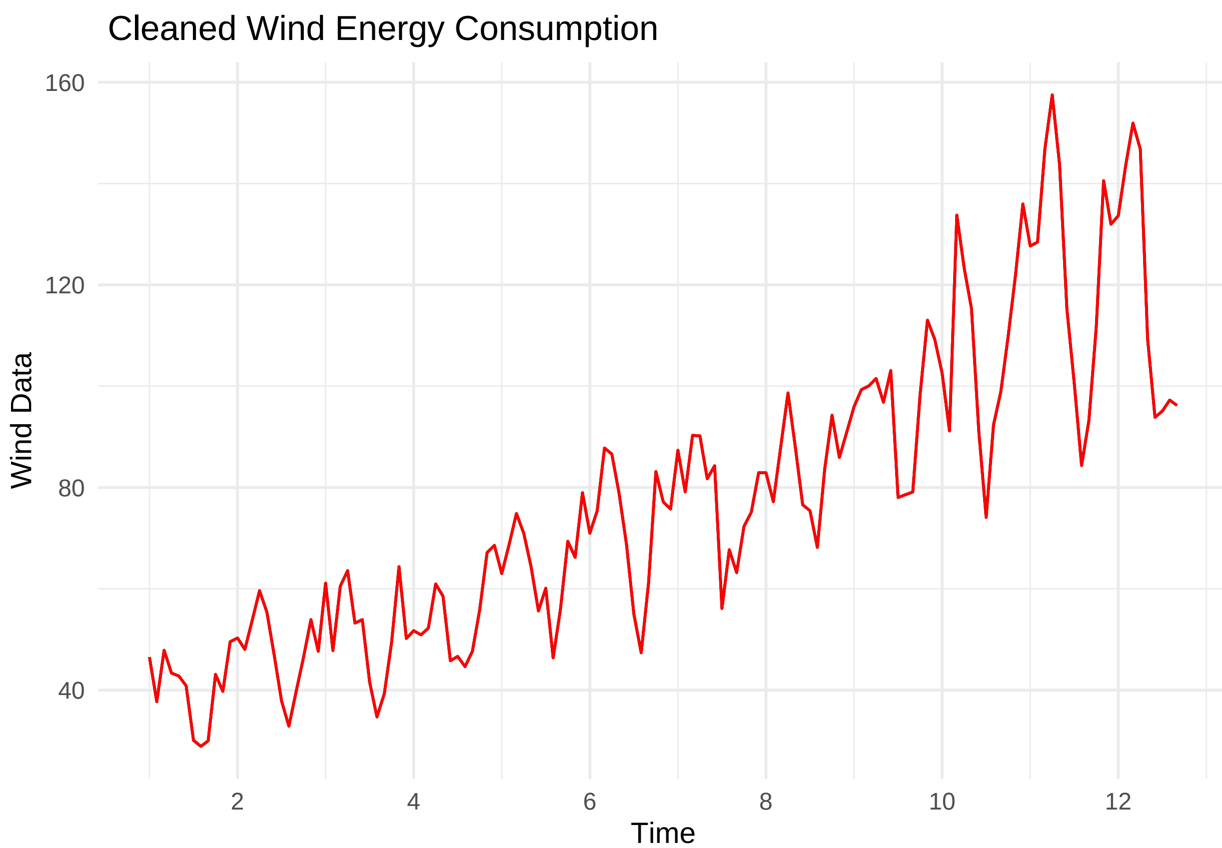



```
ggplot(energy_data, aes(x = Date, y = `Wind Energy Consumption`)) +
  geom_line(color = "blue") +
  ylab("Wind Energy Consumption") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  ggtitle("Wind Data")
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```




```r
#Plot using  autoplot
autoplot(wind_cleaned) +
  geom_line(color = "red") +
  ggtitle(" Cleaned Wind Energy Consumption") +
  theme_minimal() +
  ylab("Wind Data")
```

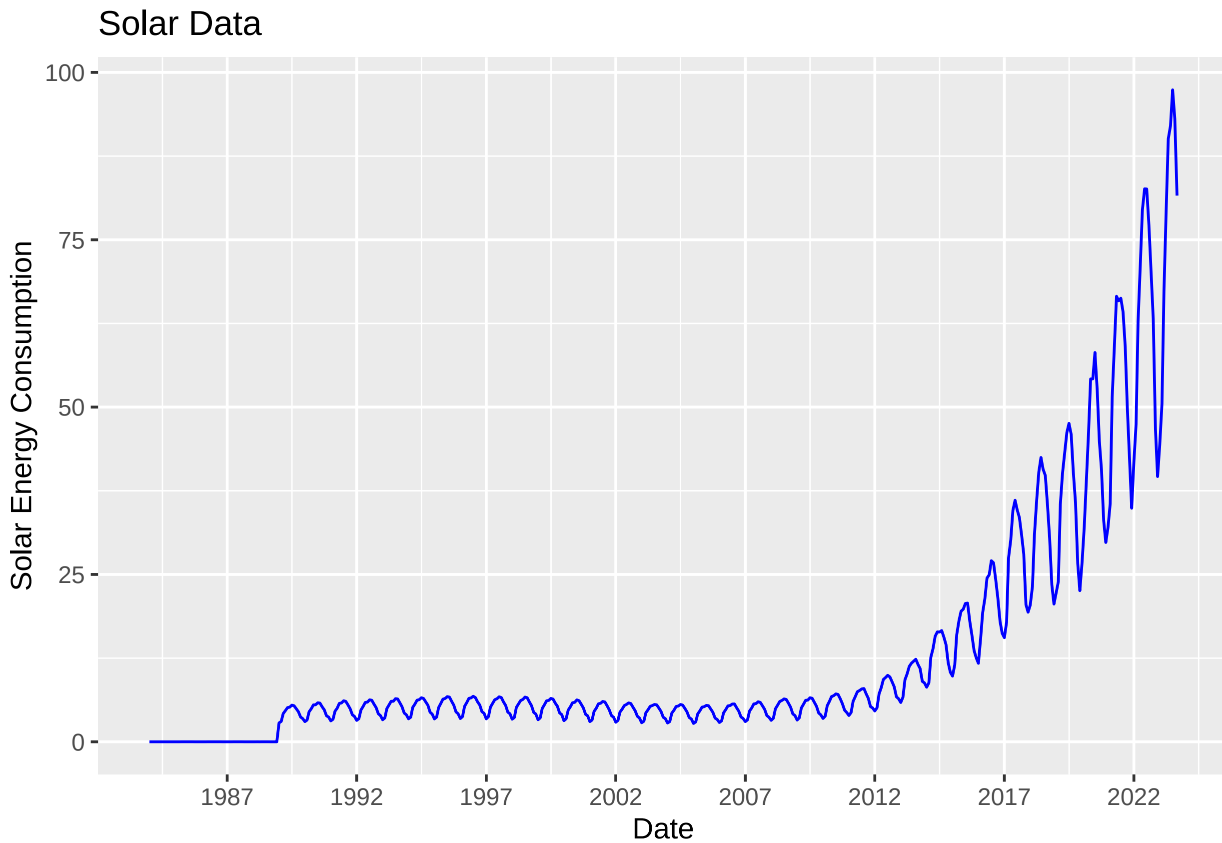


Q9

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?

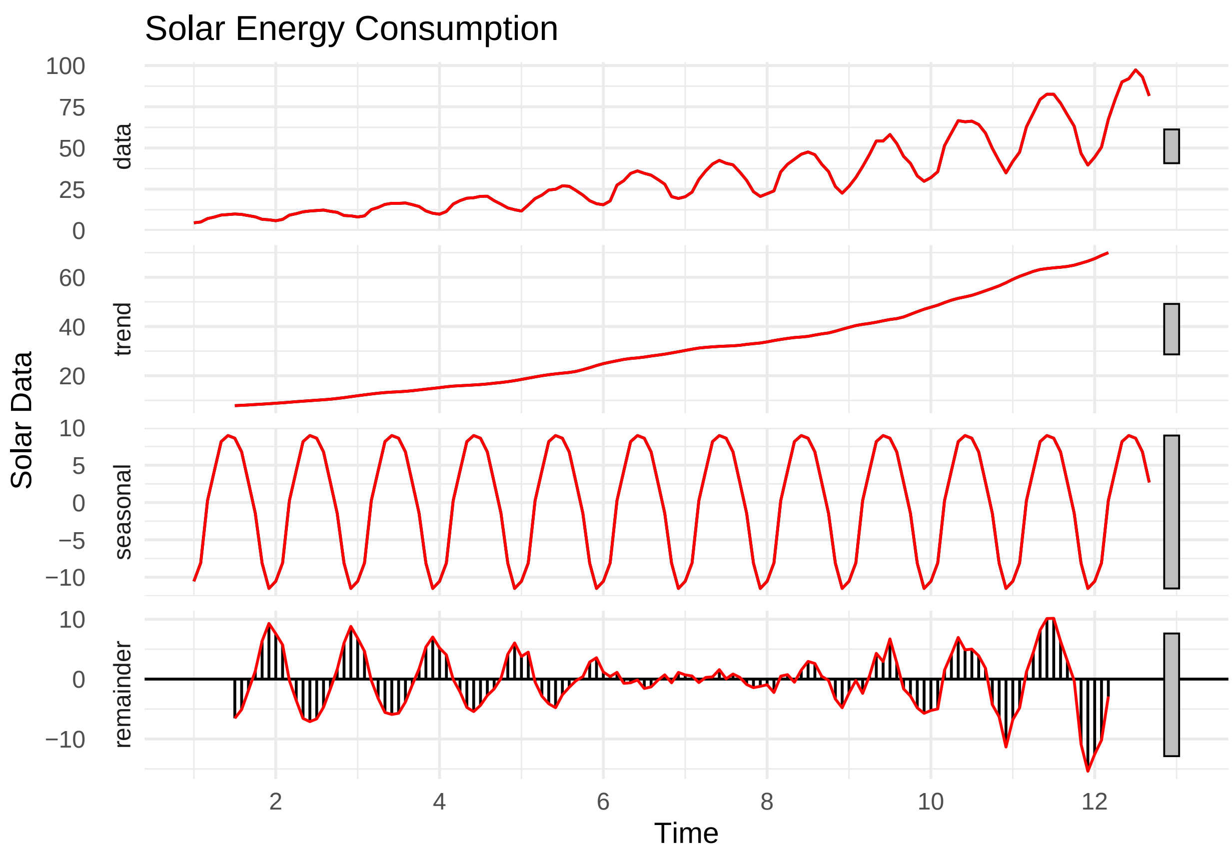
```
ggplot(energy_data, aes(x = Date, y = `Solar Energy Consumption`)) +
  geom_line(color = "blue") +
  ylab("Solar Energy Consumption") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  ggtitle("Solar Data")
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```



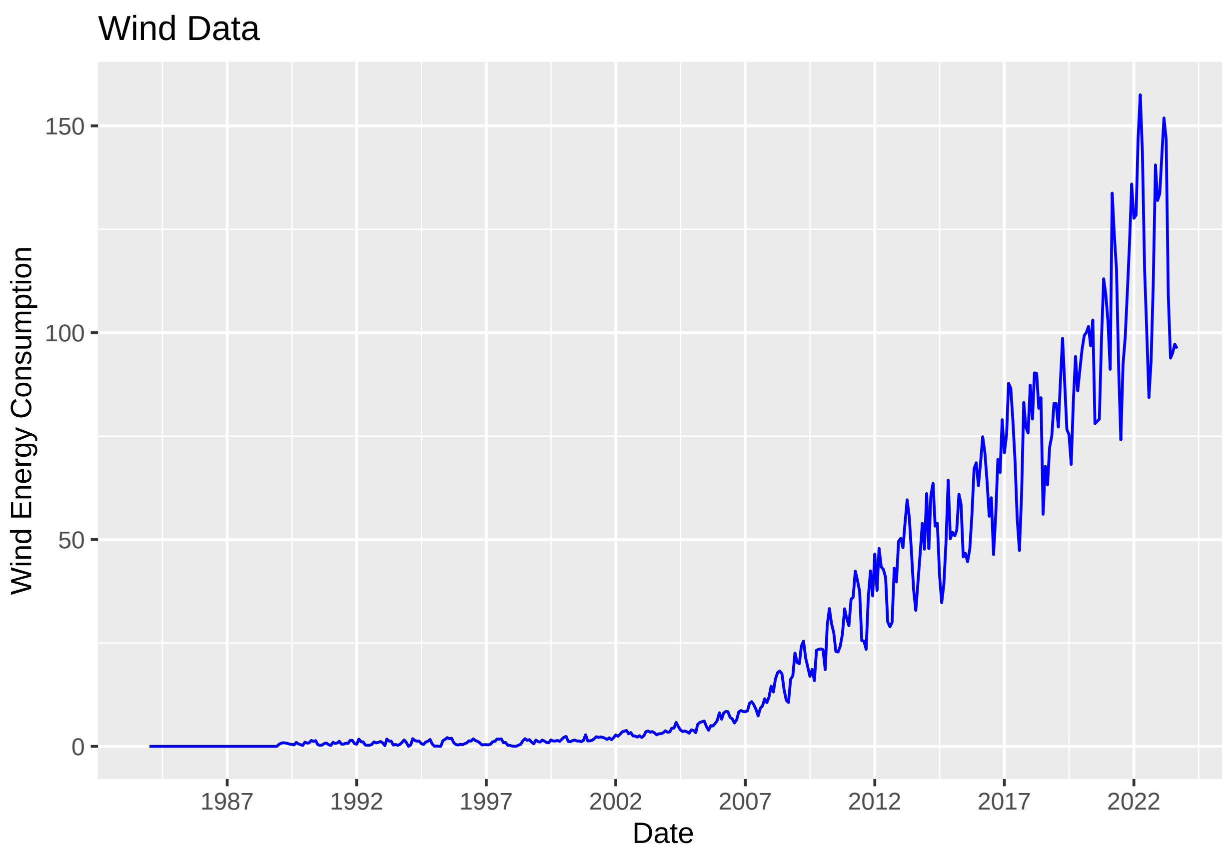

```r
#Plot using autoplot
autoplot(solardecomposed2012) +
  geom_line(color = "red") +
  ggtitle("Solar Energy Consumption") +
  theme_minimal() +
  ylab("Solar Data")
```

```
## Warning: Removed 6 rows containing missing values (‘geom_line()‘).
```



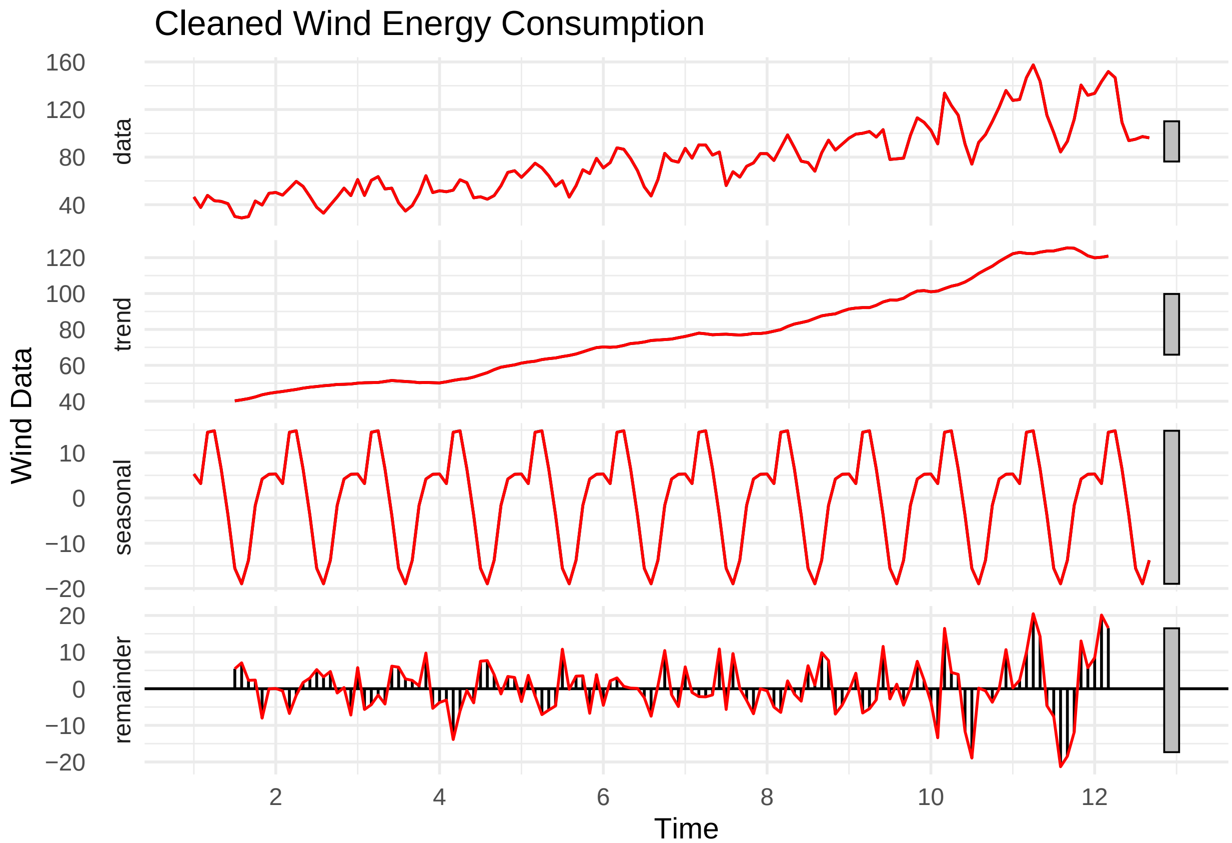

```r
ggplot(energy_data, aes(x = Date, y = `Wind Energy Consumption`)) +
  geom_line(color = "blue") +
  ylab("Wind Energy Consumption") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  ggtitle("Wind Data")
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```




```r
#Plot using autoplot
autoplot(winddecomposed2012) +
  geom_line(color = "red") +
  ggtitle(" Cleaned Wind Energy Consumption") +
  theme_minimal() +
  ylab("Wind Data")
```

```
## Warning: Removed 6 rows containing missing values (‘geom_line()‘).
```



Answer:We dont see much change between the two except that solar data have started to present a wave like function with seasonality. While wind data became increasingly more random with the data that was filtered for less years.