ENV 797 - Time Series Analysis for Energy and Environment Applications | Spring 2025

Assignment 5 - Due date 02/18/25

Yuqi Yang

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_Sp25.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)
library(tseries)
library(ggplot2)
library(Kendall)
library(lubridate)
library(tidyverse)  #load this package so yon clean the data frame using pipes
#library(xlsx)
library(readxl)
```

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
#energy_data <- read.xlsx(file="./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source

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

```
#Importing data set
energy_data <- read_excel(
path="../Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xlsx",
skip = 12,
sheet="Monthly Data",
col_names=FALSE)
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
## * `` -> `...13`
## * `` -> `...14`
```

```
read_col_names <- read_excel(
path="../Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xlsx",
skip = 10,
n_max = 1,
sheet="Monthly Data",
col_names=FALSE)
```

```
## 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 Energy Production` `Biofuels Production`
```

```
##   <dttm>                                <dbl> <chr>
## 1 1973-01-01 00:00:00                   130. Not Available
## 2 1973-02-01 00:00:00                   117. Not Available
## 3 1973-03-01 00:00:00                   130. Not Available
## 4 1973-04-01 00:00:00                   125. Not Available
## 5 1973-05-01 00:00:00                   130. Not Available
## 6 1973-06-01 00:00:00                   125. Not Available
## # i 11 more variables: 'Total Biomass Energy Production' <dbl>,
## #   'Total Renewable Energy Production' <dbl>,
## #   'Hydroelectric Power Consumption' <dbl>,
## #   'Geothermal Energy Consumption' <dbl>, 'Solar Energy Consumption' <chr>,
## #   'Wind Energy Consumption' <chr>, 'Wood Energy Consumption' <dbl>,
## #   'Waste Energy Consumption' <dbl>, 'Biofuels Consumption' <chr>,
## #   'Total Biomass Energy Consumption' <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!

```
energy_clean <- energy_data %>%
  select(1, 8, 9) %>%
  mutate(across(2:3, as.numeric)) %>%
  mutate(Month = as.Date(Month)) %>%
  drop_na() %>%
  as.data.frame()
```

```
## Warning: There were 2 warnings in 'mutate()'.
## The first warning was:
## i In argument: 'across(2:3, as.numeric)'.
## Caused by warning:
## ! NAs introduced by coercion
## i Run 'dplyr::last_dplyr_warnings()' to see the 1 remaining warning.
```

```
str(energy_clean)
```

```
## 'data.frame':    489 obs. of  3 variables:
##  $ Month                   : Date, format: "1984-01-01" "1984-02-01" ...
##  $ Solar Energy Consumption: num  0 0 0.001 0.001 0.002 0.003 0.001 0.003 0.003 0.002 ...
##  $ Wind Energy Consumption : num  0 0.001 0.001 0.002 0.003 0.002 0.002 0.001 0.002 0.003 ...
```

```
head(energy_clean)
```

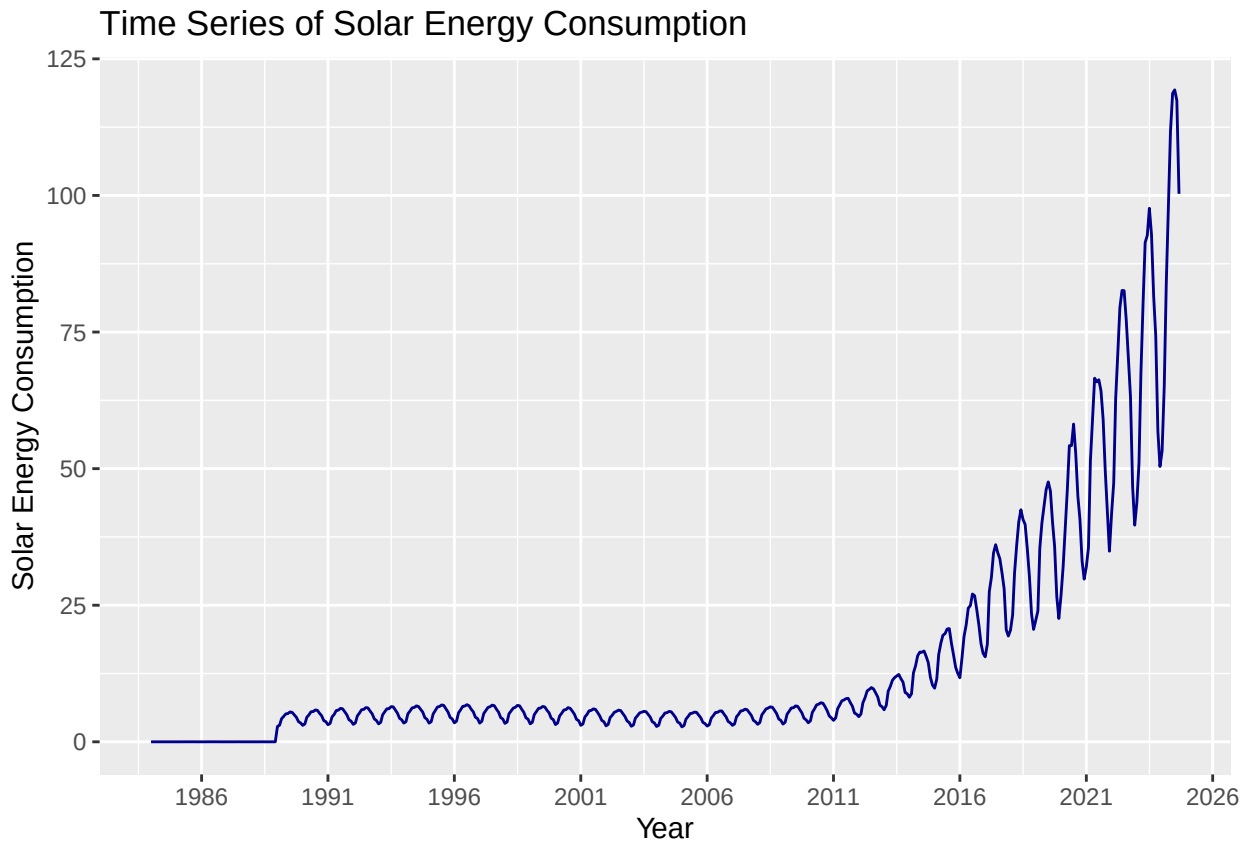
```
##        Month Solar Energy Consumption Wind Energy Consumption
## 1 1984-01-01                   0.000                   0.000
```

```
## 2 1984-02-01                0.000                0.001
## 3 1984-03-01                0.001                0.001
## 4 1984-04-01                0.001                0.002
## 5 1984-05-01                0.002                0.003
## 6 1984-06-01                0.003                0.002
```

Q2

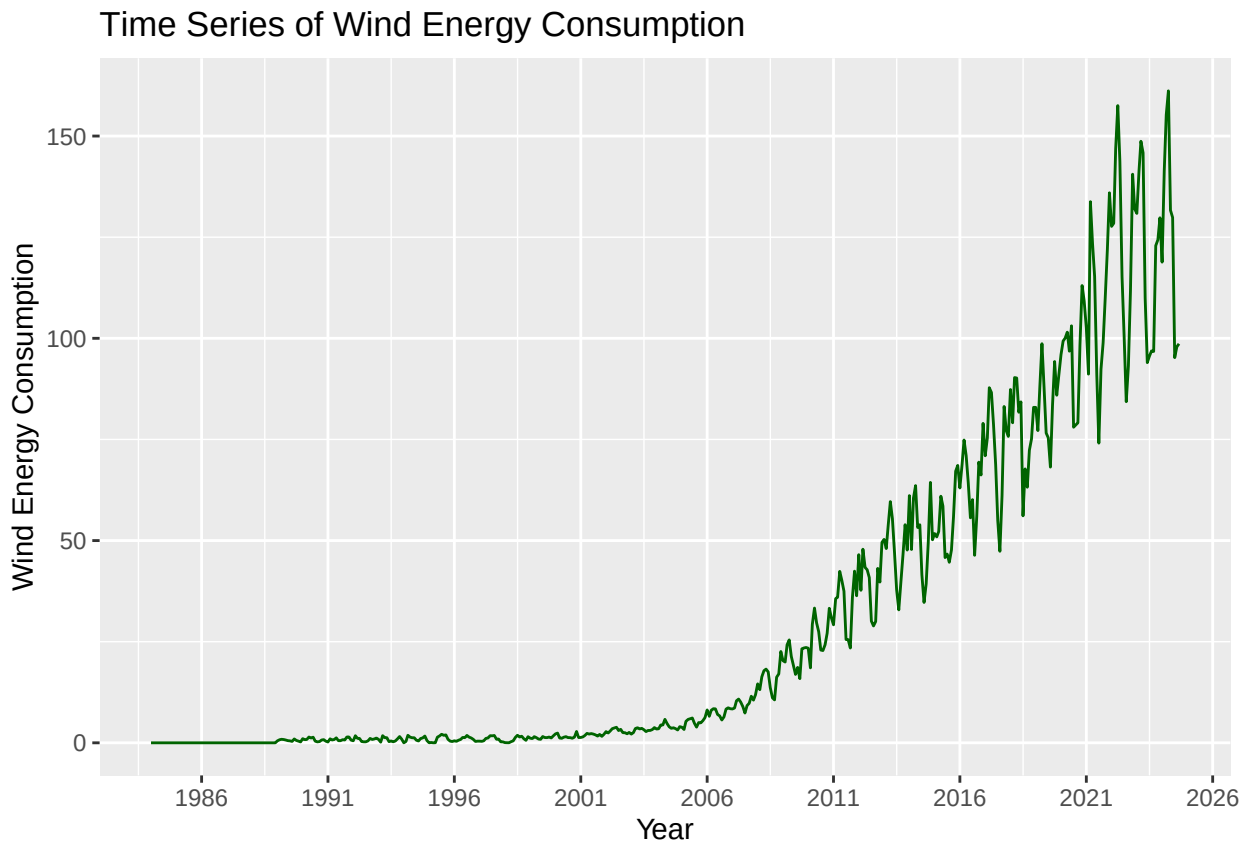
Plot the Solar and Wind energy consumption over time using ggplot. Plot each series on a separate graph. No need to add legend. Add informative names to the y axis using `ylab()`. Explore the function `scale_x_date()` on ggplot and see if you can change the x axis to improve your plot. Hint: use `scale_x_date(date_breaks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

```
#Plot the Solar Energy Consumption
ggplot(energy_clean, aes(x = Month, y = `Solar Energy Consumption`)) +
       geom_line(color = "darkblue") +
       ggtitle("Time Series of Solar Energy Consumption") +
       xlab("Year") +
       ylab("Solar Energy Consumption") +
       scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```




```
#Plot the Wind Energy Consumption
ggplot(energy_clean, aes(x = Month, y = `Wind Energy Consumption`)) +
       geom_line(color = "darkgreen") +
       ggtitle("Time Series of Wind Energy Consumption") +
```

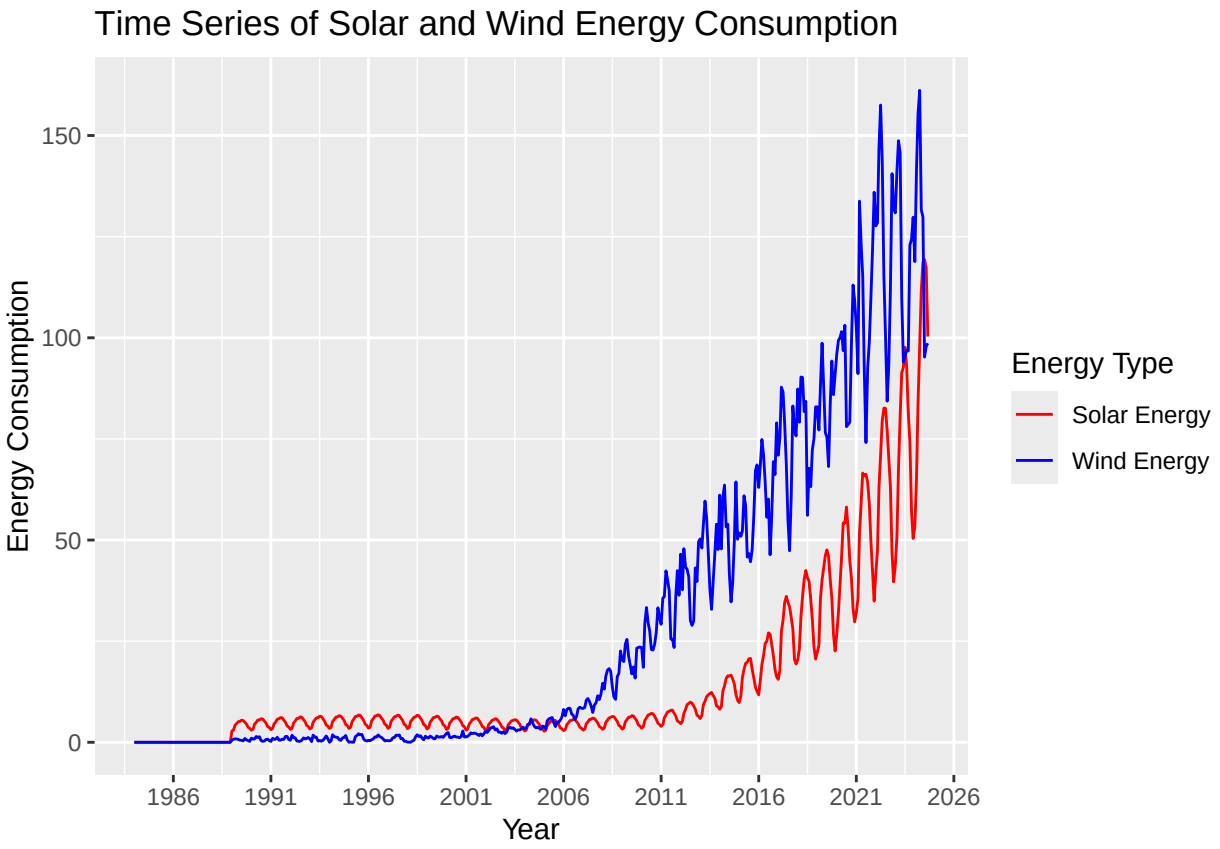
```
      xlab("Year") +
      ylab("Wind Energy Consumption") +
      scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```



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.

```
ggplot(energy_clean, aes(x = Month)) +
  geom_line(aes(y = `Solar Energy Consumption`, color = "Solar Energy")) +
  geom_line(aes(y = `Wind Energy Consumption`, color = "Wind Energy")) +
  labs(
    title = "Time Series of Solar and Wind Energy Consumption",
    x = "Year",
    y = "Energy Consumption"
  ) +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  scale_color_manual(
    name = "Energy Type",
    values = c("Solar Energy" = "red", "Wind Energy" = "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.

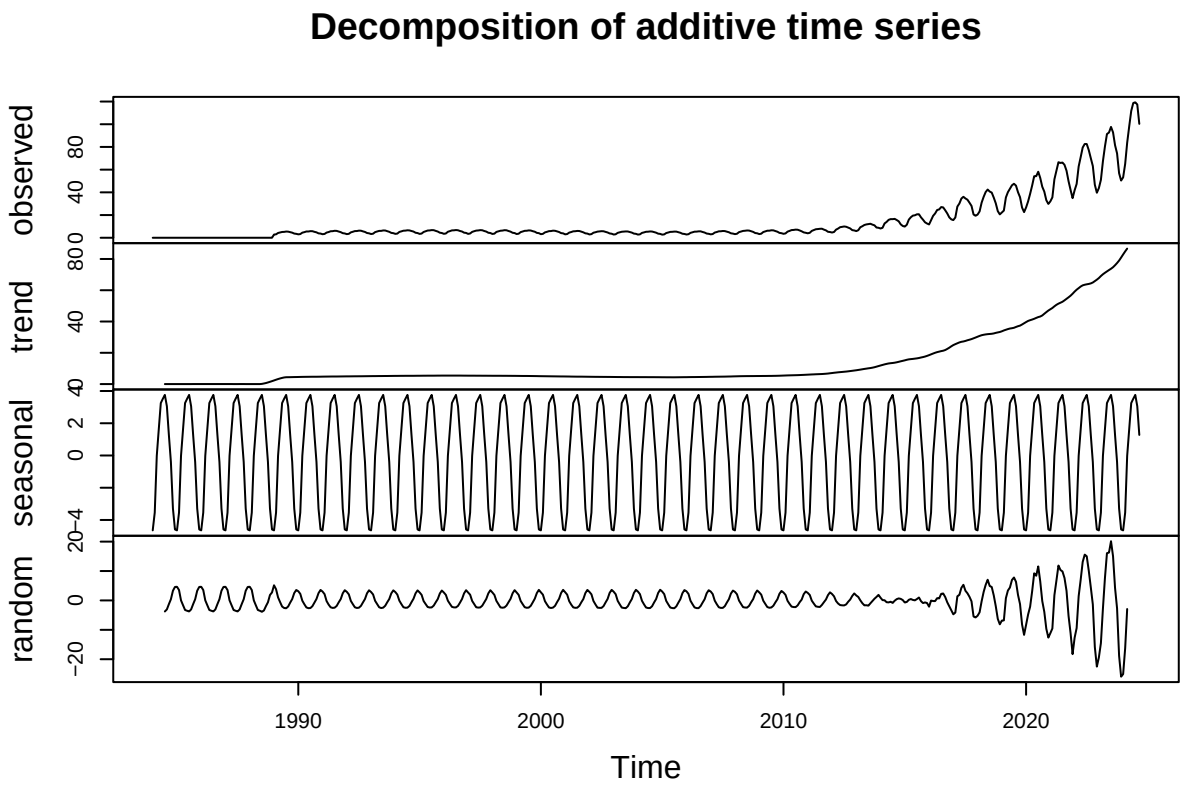
Additional info on `decompose()`.

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.

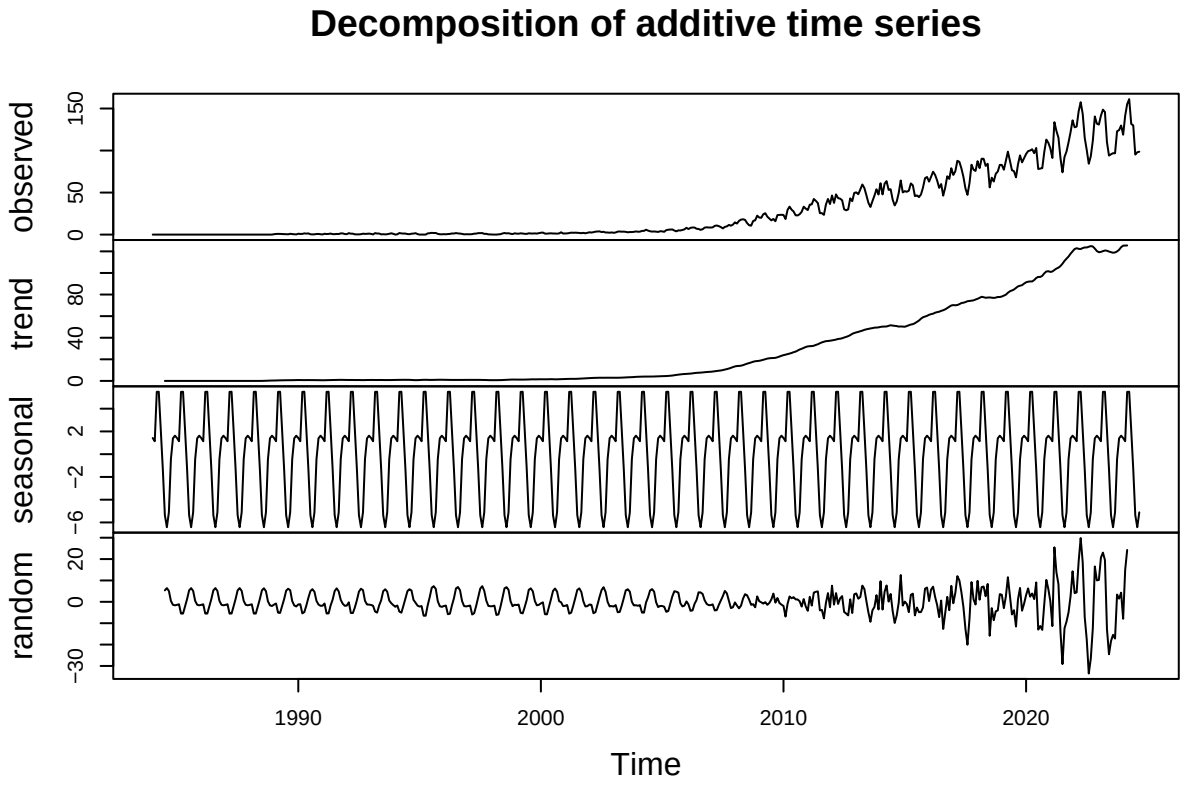
Q4

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
#Transform into time series objects
solar_ts <- ts(energy_clean[,2],start = c(1984,1),frequency = 12)
wind_ts <- ts(energy_clean[,3],start = c(1984,1),frequency = 12)

#Decomposing the time series -alternative
solar_decomp_a <- decompose(solar_ts, type = "additive")
wind_decomp_a <- decompose(wind_ts, type = "additive")

#Plot the decomposition results
plot(solar_decomp_a)
```




```r
plot(wind_decomp_a)
```

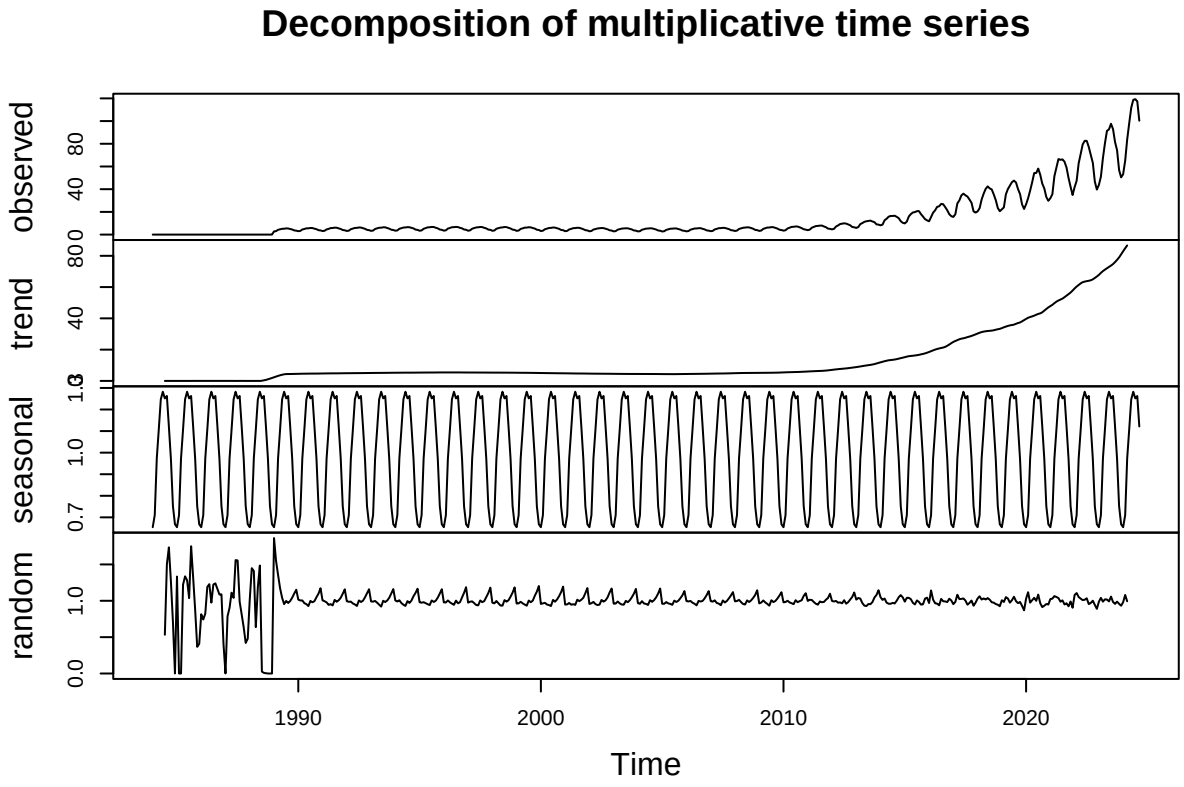


Answer:The **trend components** of both series show a clear growth pattern over time. Solar energy consumption is relatively flat from around 1990 to 2010 and then increases. Wind energy consumption remains almost constant until the early 2000s, followed by a rapid rise.
For solar energy consumption, the **random component** consistently exhibits seasonality, indicating that some seasonal effects are not fully removed. For wind energy consumption, the random component shows seasonality in early years, but the pattern becomes more irregular over time.

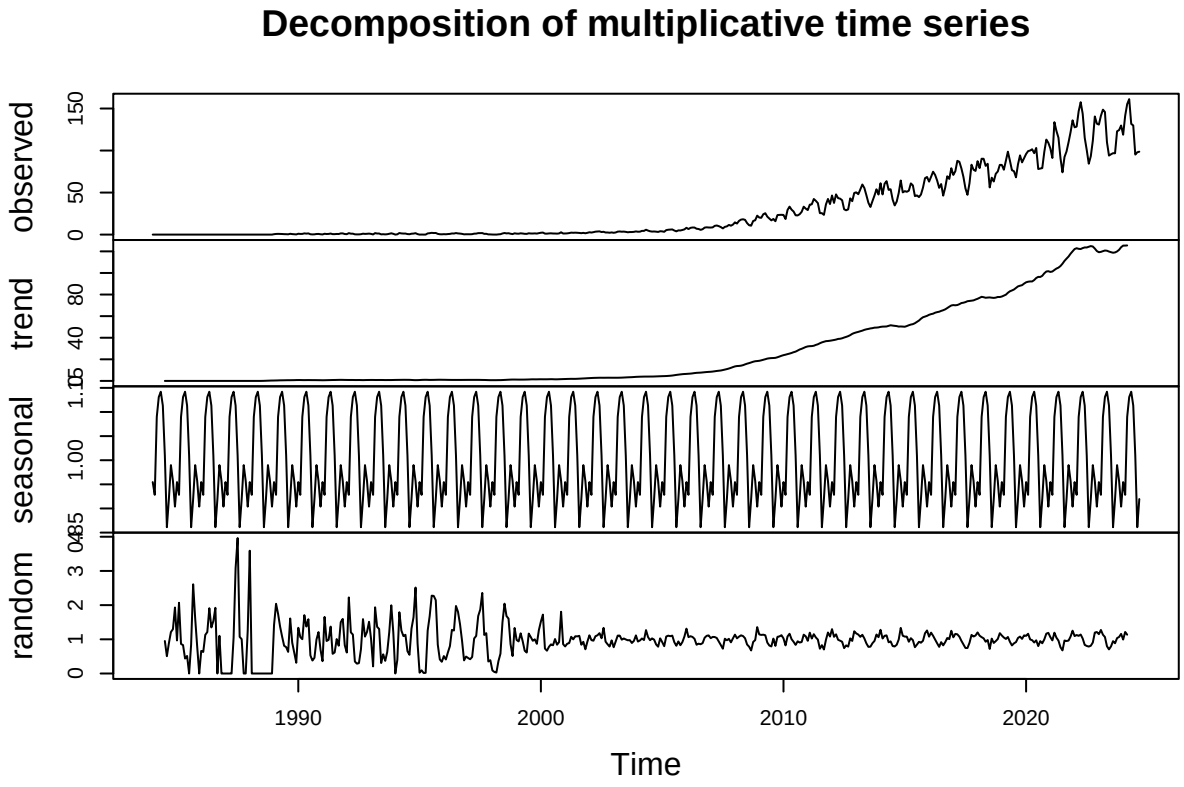
Q5

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?

```
#Decomposing the time series -multiplicative
solar_decomp_m <- decompose(solar_ts, type = "multiplicative")
wind_decomp_m <- decompose(wind_ts, type = "multiplicative")

#Plot the decomposition results
plot(solar_decomp_m)
```




```
plot(wind_decomp_m)
```



Answer:The multiplicative decomposition has largely reduced the seasonality in the random component for both solar and wind energy consumption. However, in the solar energy consumption, there are still some seasonal trends present between 1990 and 2010, while that of wind energy does not exhibit any distinct seasonal patterns.

Q6

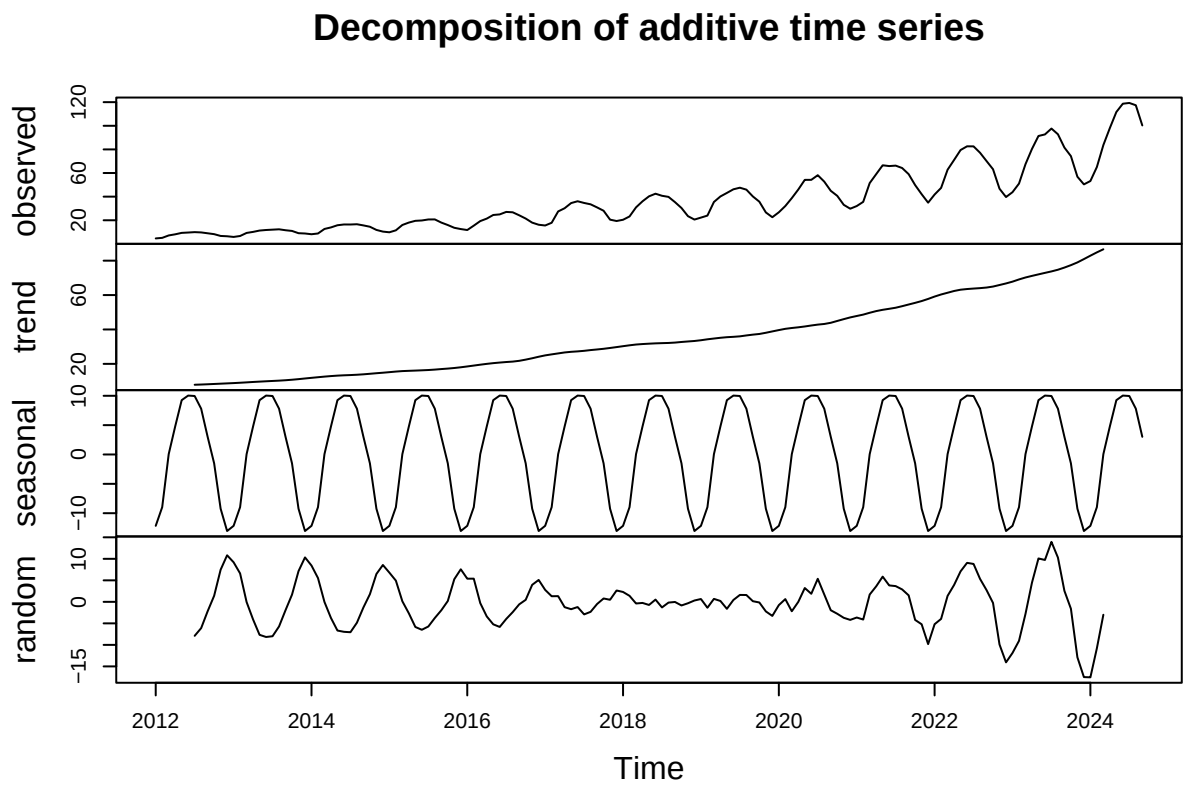
When fitting a model to this data, do you think you need all the historical data? Think about the data from 90s and early 20s. Are there any information from those years we might need to forecast the next six months of Solar and/or Wind consumption. Explain your response.

Answer:I think we do not need all historical data for forecasting the next six months of Solar and/or Wind consumption.
Because both solar and wind energy consumption were relatively flat from 1984 to early 2000s but has a strong upward trend after 2010. This is likely due to the dramatic changes in technology, policy, and markets in recent years, leading to a structural shift in energy consumption, especially the emphasis on renewable energy. Recent data is more relevant for forecasting future trends, while the earlier data (from the 90s and early 2000s) may bias the model towards historical patterns that are no longer relevant. Also, the decomposition plots show that seasonal cycles remain consistent, so the data from recent years is sufficient to capture seasonality.
Therefore, using recent data (past 5–10 years) for short-term forecasting (next six months) is the most relevant and effective approach, improving both accuracy and efficiency.

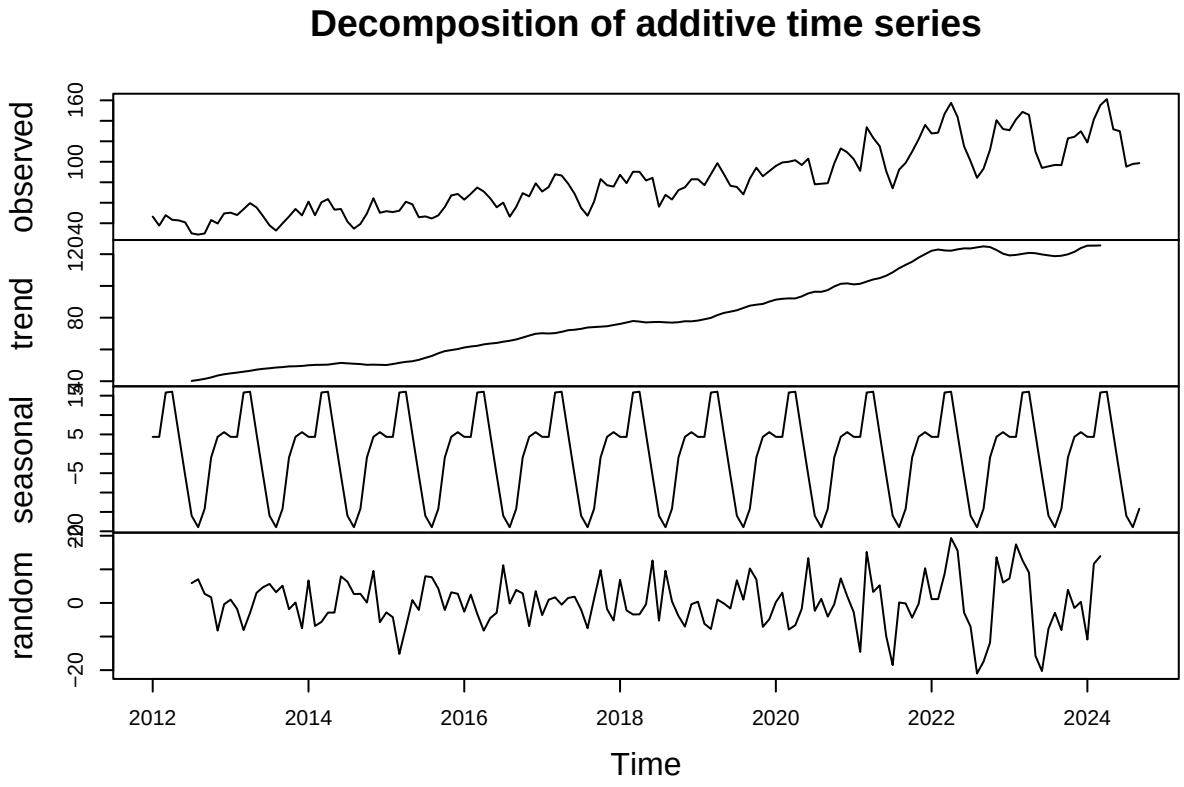
Q7

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.

```
energy_recent <- energy_clean %>%
  filter(year(Month) >= 2012)

#Transform into time series objects
energy_recent_ts <- ts(energy_recent[,2:3],start = c(2012,1),frequency = 12)

#Decomposing the time series -alternative
solar_recent_decomp_a <- decompose(energy_recent_ts[,1], type = "additive")
wind_recent_decomp_a <- decompose(energy_recent_ts[,2], type = "additive")

#Plot the decomposition results
plot(solar_recent_decomp_a)
```




```
plot(wind_recent_decomp_a)
```



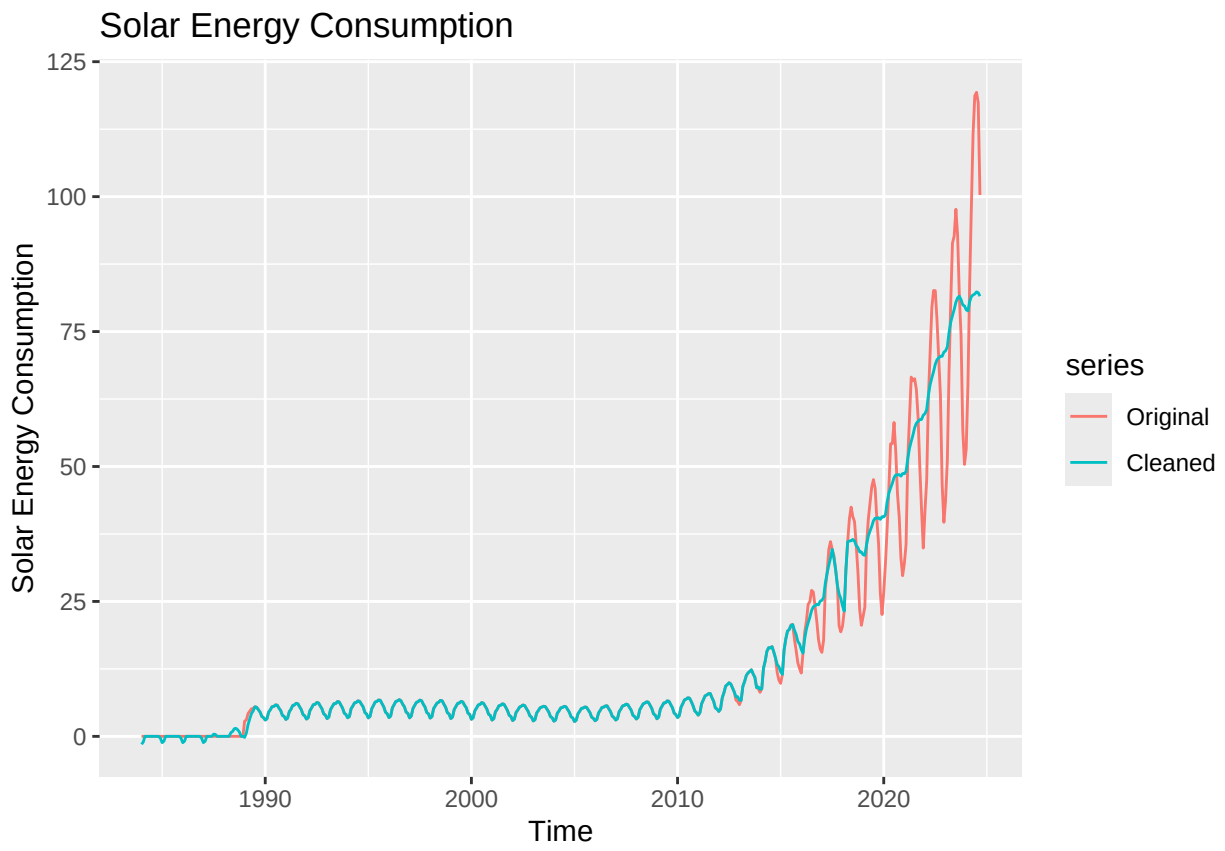
Answer:For **trend component**, both solar and wind energy consumption have been steadily increasing since 2012, showing a consistent upward trend.
For **seasonal component**, both solar and wind energy exhibit strong annual seasonality, but the seasonal magnitude remains constant, which aligns with the additive decomposition assumption.
For **random component**, solar energy residuals still show clear patterns, while wind energy residuals seem more random but appear to retain some structure, indicating that the additive model may not have fully captured seasonality.

Identify and Remove outliers

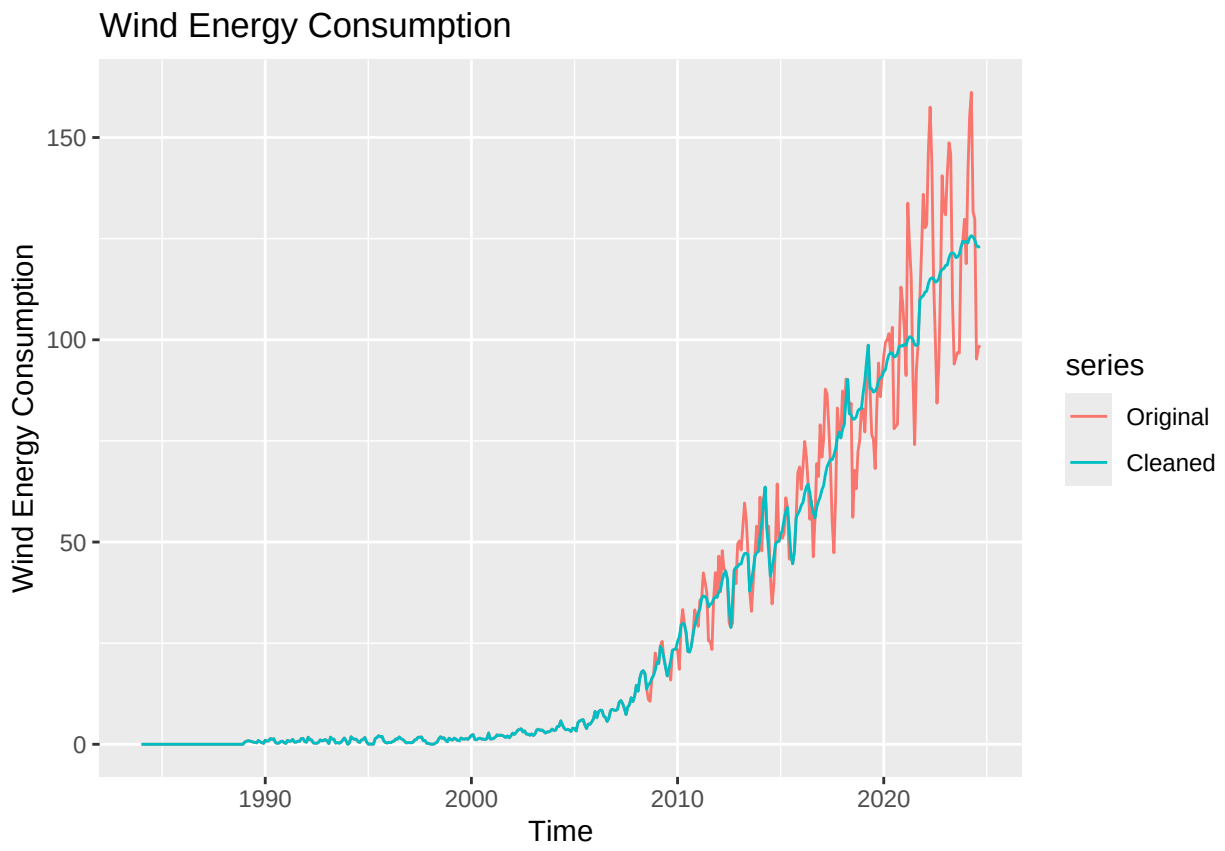
Q8

Apply the `tsclean()` to both series from Q4. Did the function removed any outliers from the series? Hint: Use `autoplot()` to check if there is difference between cleaned series and original series.

```
#Remove outliers
solar_clean <- tsclean(solar_ts)
wind_clean <- tsclean(wind_ts)

autoplot(cbind(Original = solar_ts, Cleaned = solar_clean)) +
  ggtitle("Solar Energy Consumption") +
  ylab("Solar Energy Consumption")
```




```
autoplot(cbind(Original = wind_ts, Cleaned = wind_clean)) +
  ggtitle("Wind Energy Consumption") +
  ylab("Wind Energy Consumption")
```

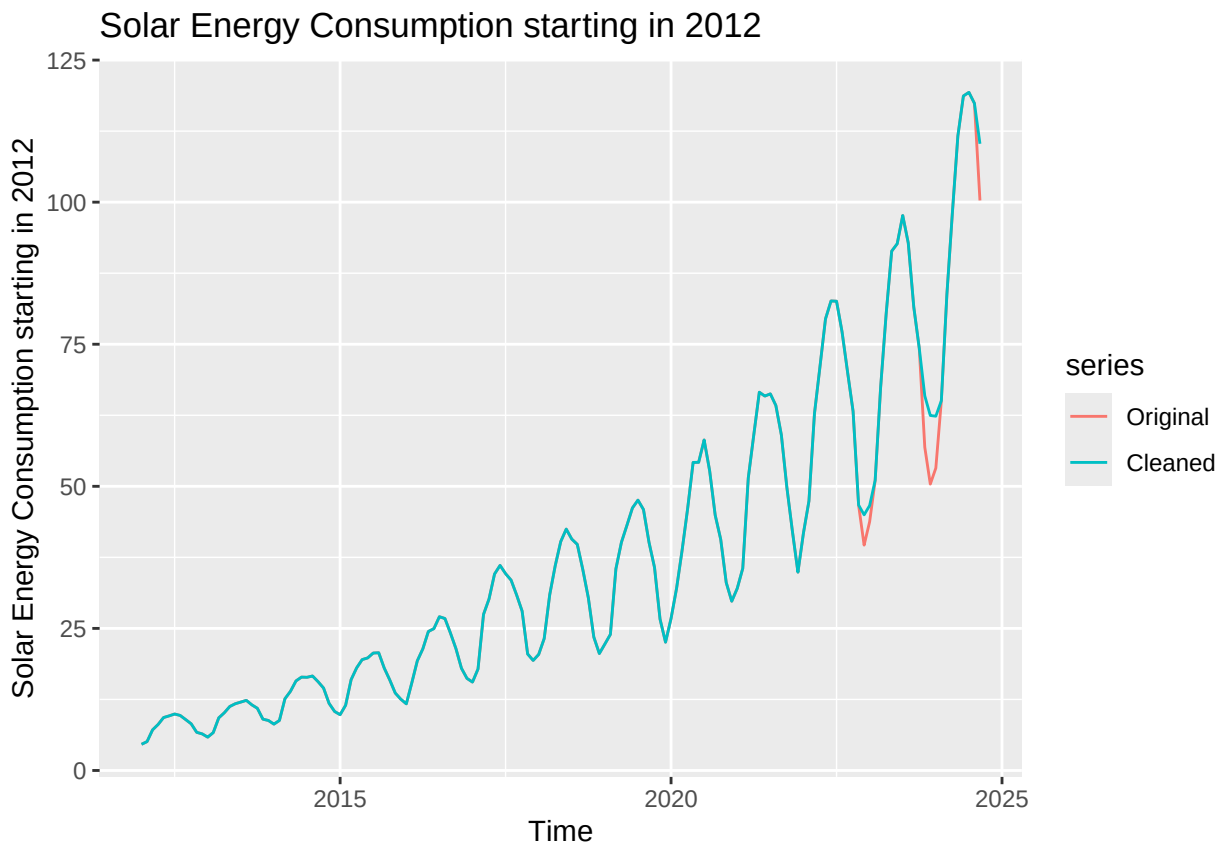


Answer:Yes, this function removed outliers from both series, smoothing extreme spikes while preserving the trend and seasonality, especially in recent years.

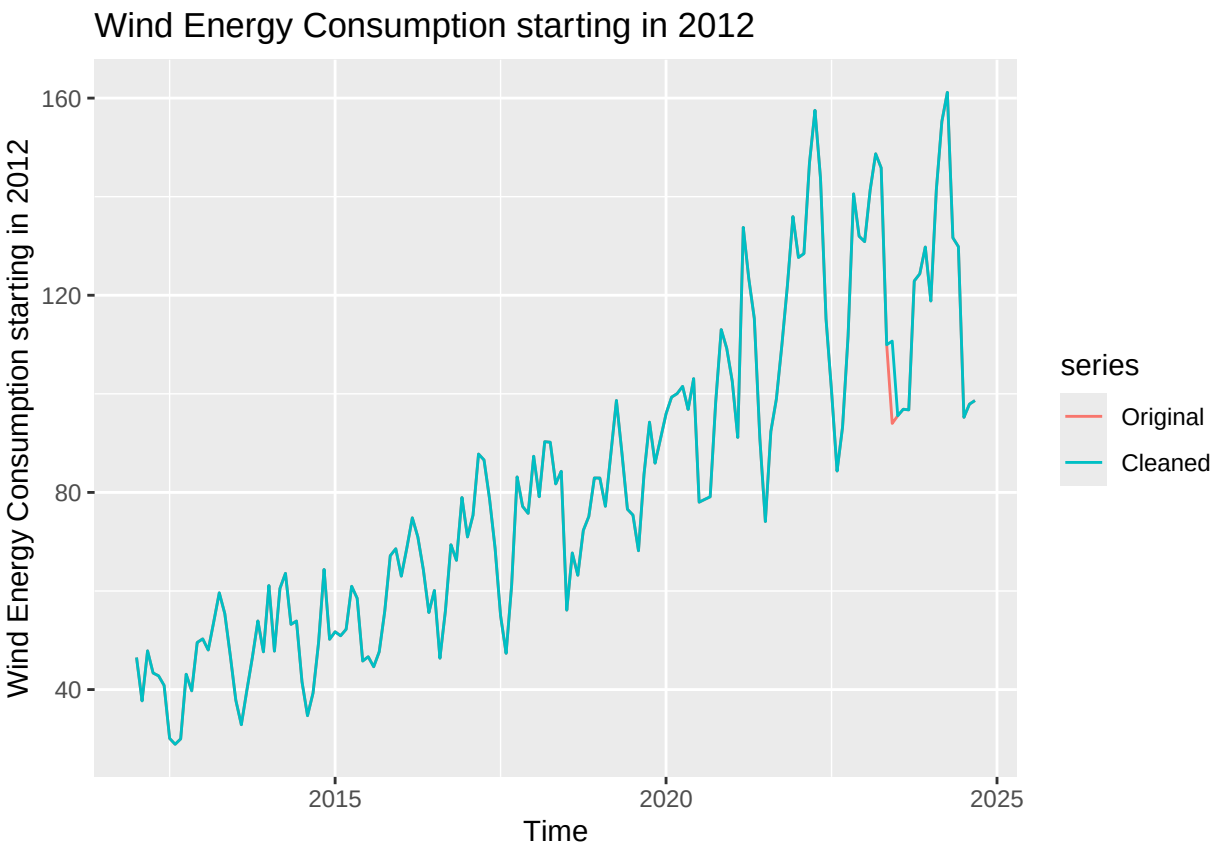
Q9

Redo number Q8 but now with the time series you created on Q7, i.e., the series starting in 2012. Using what `autoplot()` again what happened now?Did the function removed any outliers from the series?

```
#Remove outliers
solar_recent_clean <- tsclean(energy_recent_ts[,1])
wind_recent_clean <- tsclean(energy_recent_ts[,2])

autoplot(cbind(Original = energy_recent_ts[,1], Cleaned = solar_recent_clean)) +
  ggtitle("Solar Energy Consumption starting in 2012") +
  ylab("Solar Energy Consumption starting in 2012")
```




```
autoplot(cbind(Original = energy_recent_ts[,2], Cleaned = wind_recent_clean)) +
  ggtitle("Wind Energy Consumption starting in 2012") +
  ylab("Wind Energy Consumption starting in 2012")
```



Answer:For the data starting in 2012, the function only removed a few short-term outliers, with little effect on the long-term trend and seasonality.