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

Assignment 5 - Due date 02/18/25

Mazhar Bhuyan

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.\

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_excel(
  "./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`
```

```
#head(Energy_Data)

col_names <- read_excel(
  "./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) <- col_names

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!

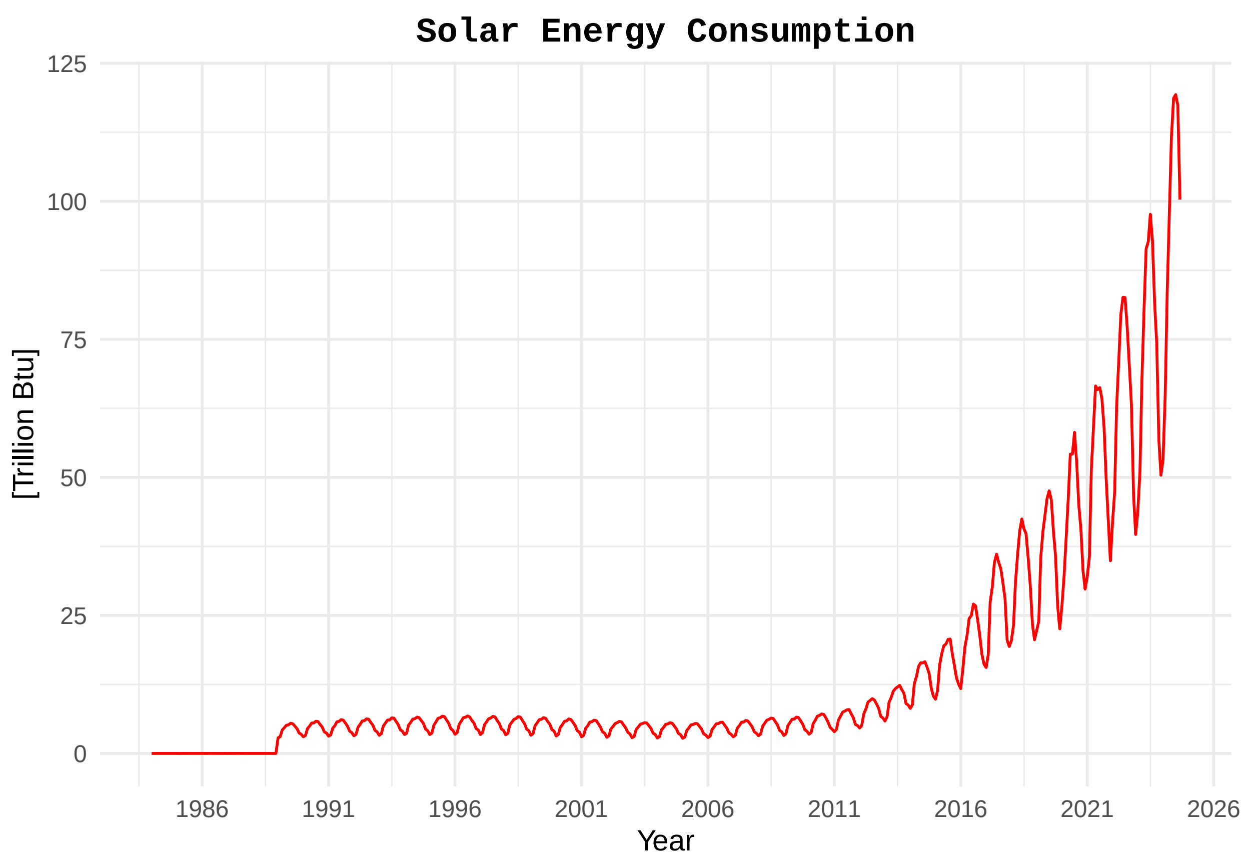
```
Sol_Wind <- Energy_Data %>%
  select(Month, `Solar Energy Consumption`, `Wind Energy Consumption`) %>%
  mutate(
    `Solar Energy Consumption` = na_if(`Solar Energy Consumption`, "Not Available"),
    # Convert "Not Available" to NA
    `Wind Energy Consumption` = na_if(`Wind Energy Consumption`, "Not Available"),
    `Solar Energy Consumption` = as.numeric(`Solar Energy Consumption`),
    `Wind Energy Consumption` = as.numeric(`Wind Energy Consumption`),
    Date = ymd(Month)
  ) %>%
  drop_na(`Solar Energy Consumption`, `Wind Energy Consumption`)
head(Sol_Wind)
```

```
## # A tibble: 6 x 4
##   Month               Solar Energy Consumpti~1 Wind Energy Consumpt~2 Date
##   <dttm>                                 <dbl>                  <dbl> <date>
## 1 1984-01-01 00:00:00                    0                      0     1984-01-01
## 2 1984-02-01 00:00:00                    0                      0.001 1984-02-01
## 3 1984-03-01 00:00:00                    0.001                  0.001 1984-03-01
## 4 1984-04-01 00:00:00                    0.001                  0.002 1984-04-01
## 5 1984-05-01 00:00:00                    0.002                  0.003 1984-05-01
## 6 1984-06-01 00:00:00                    0.003                  0.002 1984-06-01
## # i abbreviated names: 1: ‘Solar Energy Consumption‘,
## #   2: ‘Wind Energy Consumption‘
```

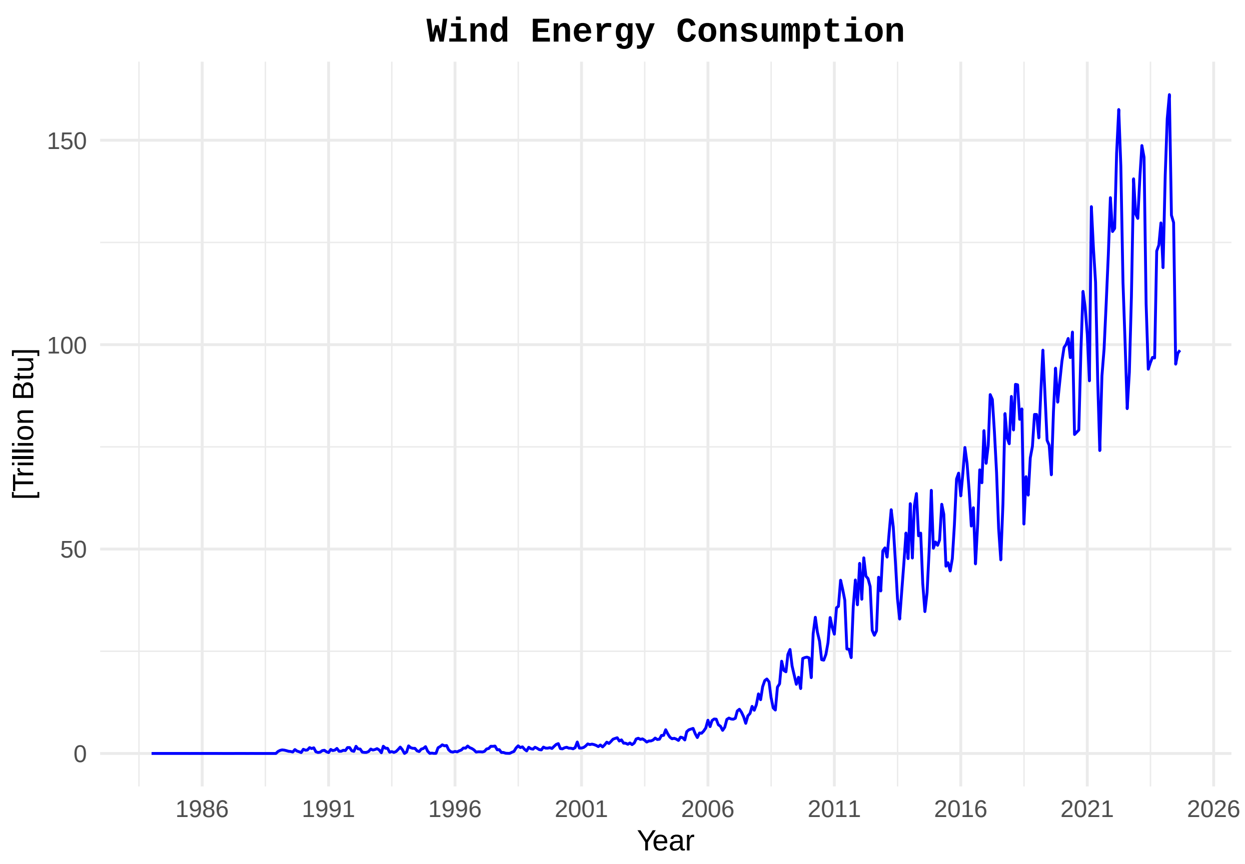
Q2

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(Sol_Wind, aes(x = Date, y = `Solar Energy Consumption`)) +
  geom_line(color = "red") +  # Color should be outside aes()
  xlab("Year") +
  ylab("[Trillion Btu]") +
  ggtitle("Solar Energy Consumption")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  theme_minimal()+
  theme(plot.title = element_text(
    family = "Courier",
    hjust = 0.5,
    face = "bold"))
```



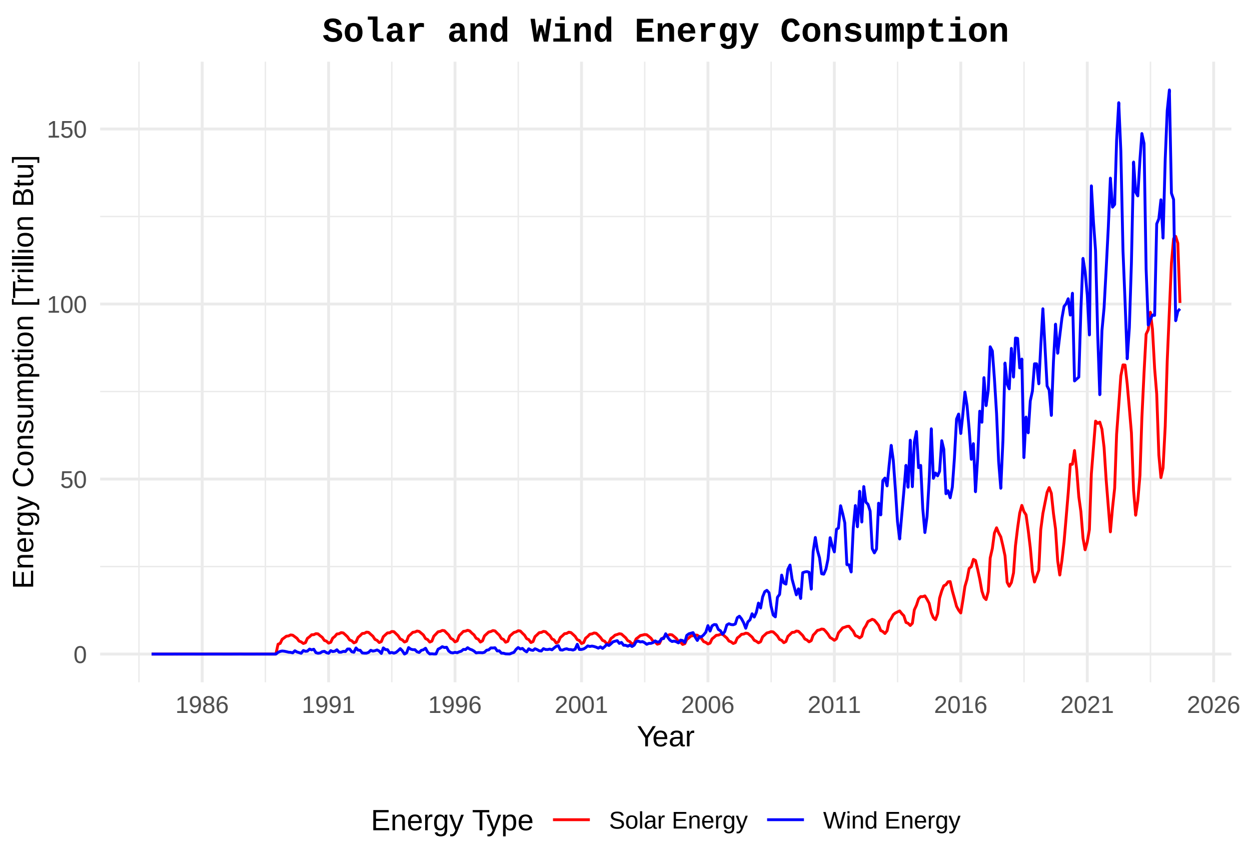
```r
ggplot(Sol_Wind, aes(x = Date, y = `Wind Energy Consumption`)) +
  geom_line(color = "blue") +  # Color should be outside aes()
  xlab("Year") +
  ylab("[Trillion Btu]") +
  ggtitle("Wind Energy Consumption")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  theme_minimal()+
  theme(plot.title = element_text(
    family = "Courier",
    hjust = 0.5,
    face = "bold"))
```



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.

```r
ggplot(Sol_Wind, aes(x = Date)) +
  geom_line(aes(y = `Solar Energy Consumption`,
                color = "Solar Energy"))+
  geom_line(aes(y = `Wind Energy Consumption`,
                color = "Wind Energy")) +
  xlab("Year") +
  ylab("Energy Consumption [Trillion Btu]") +
  ggtitle("Solar and Wind Energy Consumption")+
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  scale_color_manual(
    values = c("Solar Energy" = "red", "Wind Energy" = "blue"),
    name = "Energy Type"
  ) +
  theme_minimal() +
  theme(legend.position = "bottom")+
  theme(plot.title = element_text(hjust = 0.5,
                                  face = "bold",
                                  family = "Courier"))
```



Adding breaks in every five years provides a clear picture that after 2006 both energy consuptions have shooted up significantly.

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.

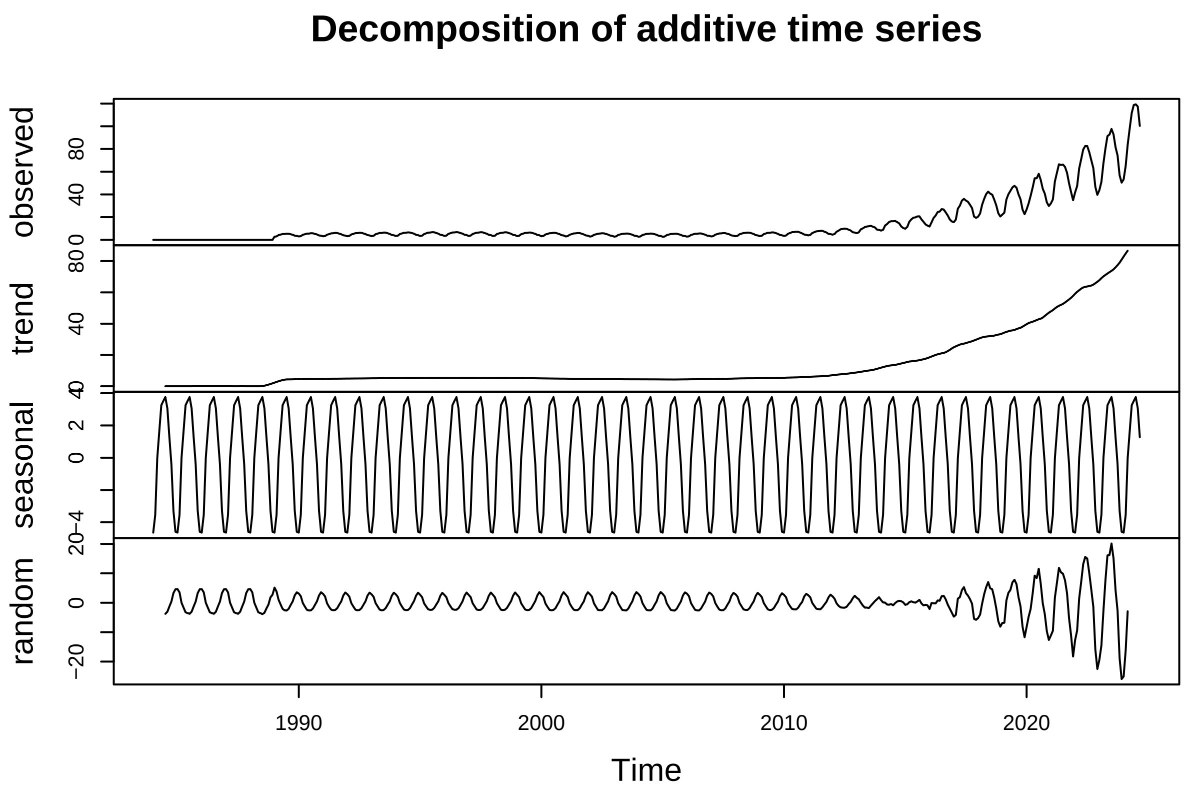
Additional info on `decompose()`.

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.

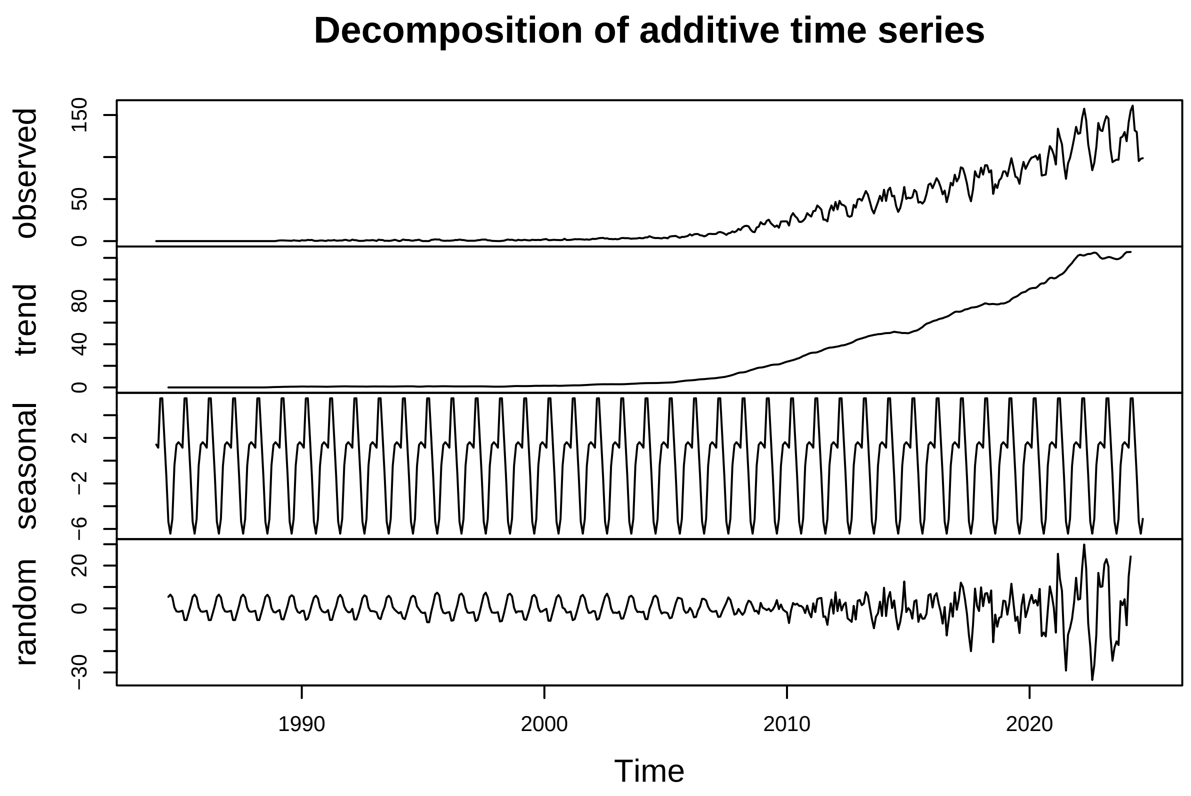
Q4

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
solar_ts <- ts(Sol_Wind$`Solar Energy Consumption`,
               start = c(1984,1),
               frequency = 12)

wind_ts <- ts(Sol_Wind$`Wind Energy Consumption`,
              start = c(1984,1),
              frequency = 12)


# Additive Decomposition
solar_decomp_ad <- decompose(solar_ts, type = "additive")
wind_decomp_ad <- decompose(wind_ts, type = "additive")

# Plotting decomposition for Solar Energy
plot(solar_decomp_ad)
```



```r
# Plotting decomposition for Wind Energy
plot(wind_decomp_ad)
```

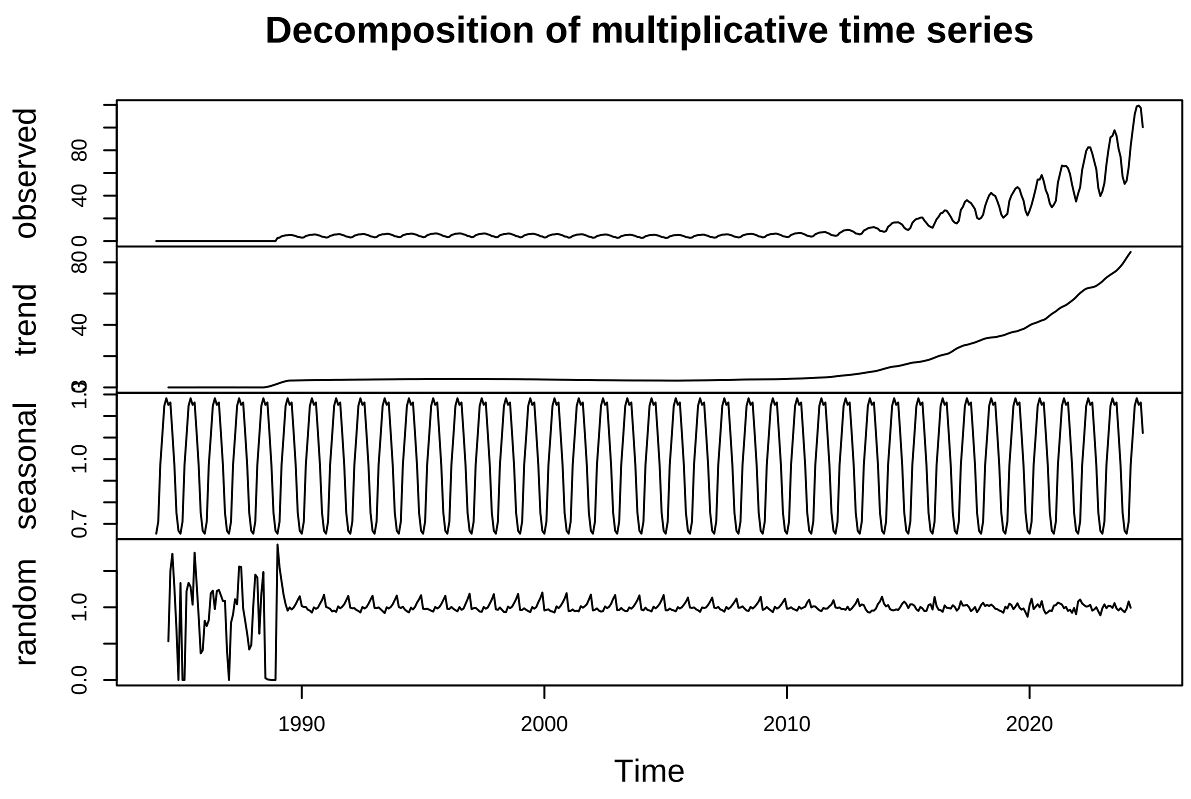


> The trend shows a steady increase in solar and wind energy consumption over time. This indicates long-term growth in renewable energy use. > The random component still has some patterns, meaning it is not entirely random. This suggests that some seasonal effects or external factors are not fully accounted for. > Since the random component should ideally have no visible patterns, this means the decomposition may not have completely removed seasonality.

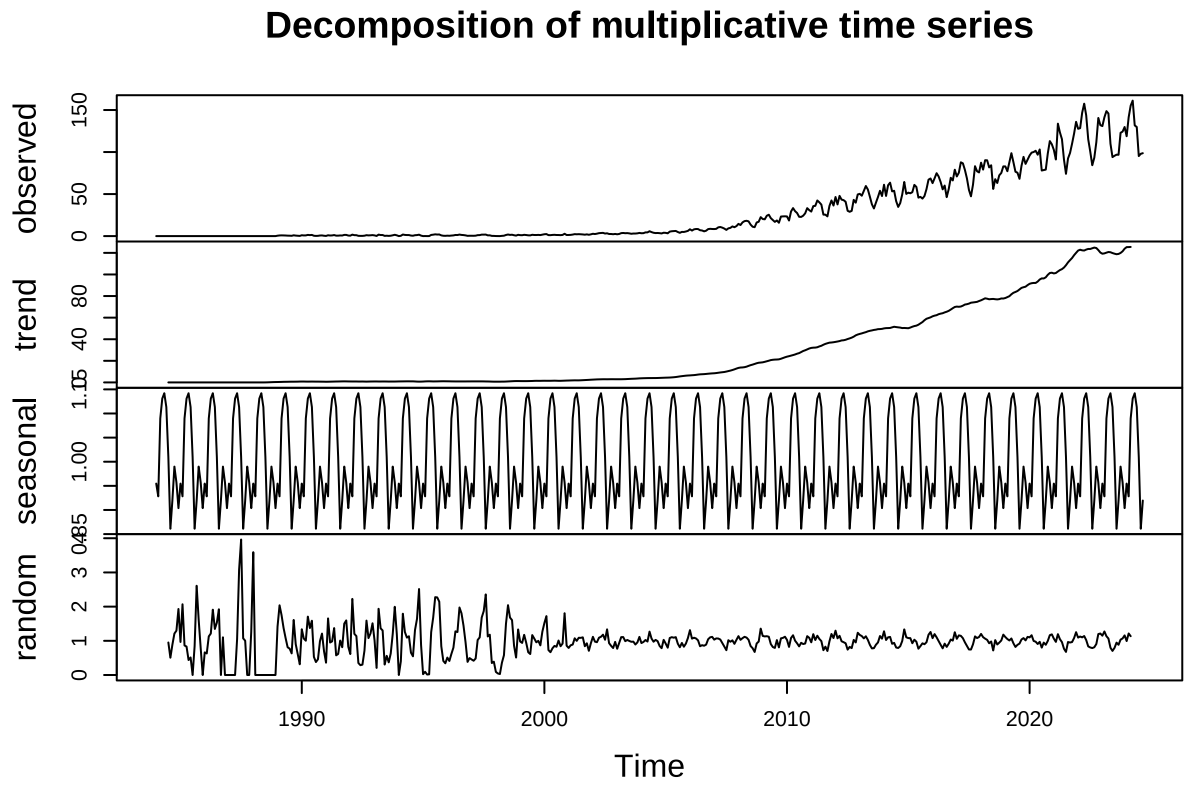
Q5

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?

```r
# Multiplicative Decomposition
solar_decomp_mlt <- decompose(solar_ts, type = "multiplicative")
wind_decomp_mlt <- decompose(wind_ts, type = "multiplicative")

# Plotting decomposition for Solar Energy
plot(solar_decomp_mlt)
```



```r
# Plotting decomposition for Wind Energy
plot(wind_decomp_mlt)
```



The random component is now more stable than before. This is because the multiplicative model scales variations based on the trend. As energy consumption grows, seasonal patterns and noise adjust proportionally. In an additive model, the random component would have larger fluctuations, making it harder to interpret. The multiplicative model provides a better fit since it accounts for the increasing trend in energy consumption.

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 all historical data may be necessary for forecasting the next six months. The data from the 1990s and early 2000s might not be very relevant because energy consumption patterns, technology, and policies have changed significantly over time.

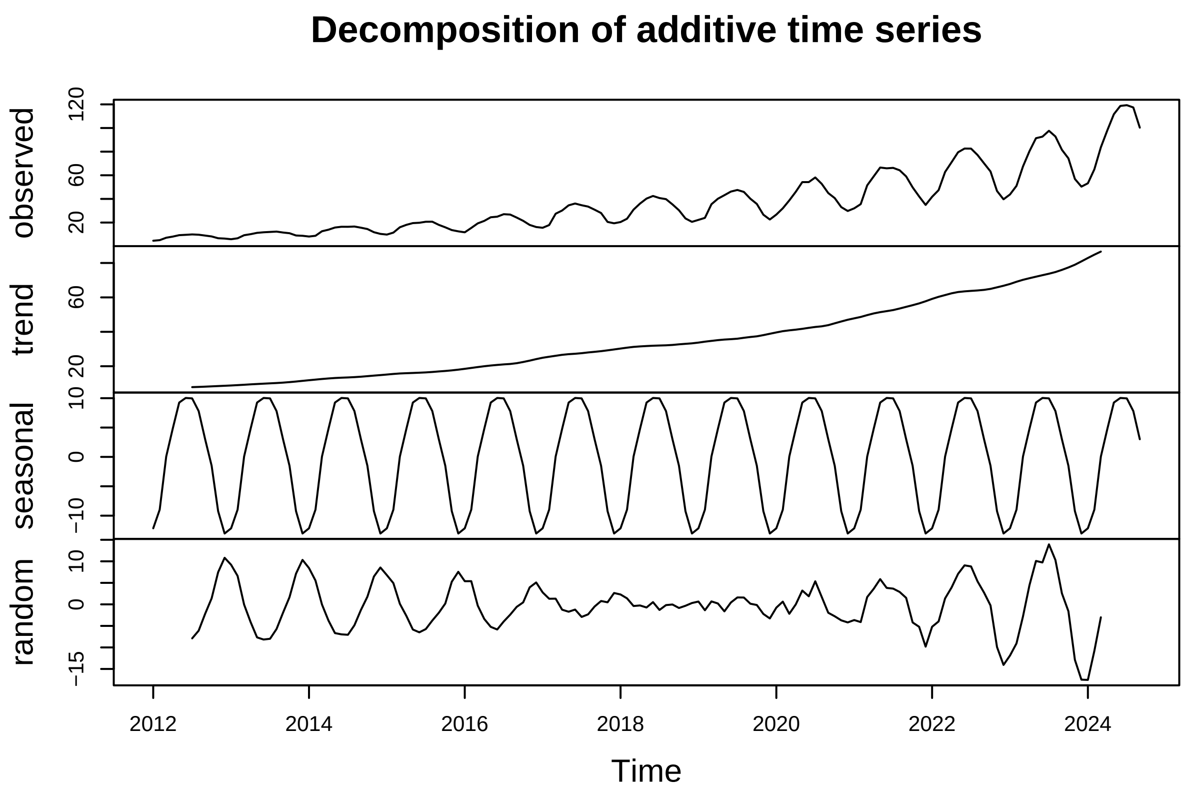
For short-term forecasting, recent data is more important as it better reflects current trends, seasonality, and external factors affecting energy consumption. Older data may introduce outdated patterns that no longer apply. However, if the goal is to understand long-term trends, including historical data can be useful.

In this case, for a six-month forecast, focusing on data from the last 5 to 10 years would likely provide a more accurate model.

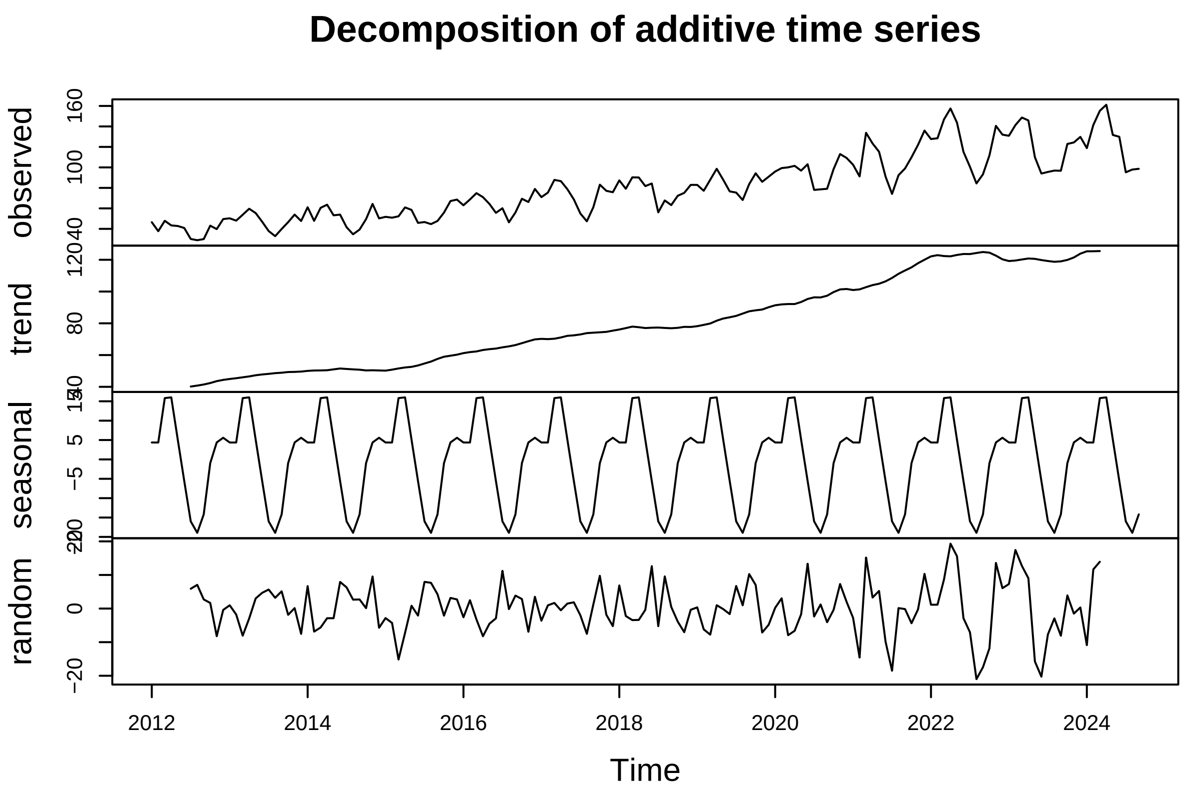
Q7

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
Sol_Wind_2012 <- Sol_Wind %>%
  filter(year(Date) >= 2012)

Sol_Wind_2012_ts <- ts(
  cbind(Solar = Sol_Wind_2012$`Solar Energy Consumption`,
        Wind = Sol_Wind_2012$`Wind Energy Consumption`),
  start = c(2012, 1),
  frequency = 12)

# Additive decomposition
solar_decomp_ad_12 <- decompose(Sol_Wind_2012_ts[,1], type = "additive")
wind_decomp_ad_12 <- decompose(Sol_Wind_2012_ts[,2], type = "additive")

# Plotting decomposition for Solar Energy
plot(solar_decomp_ad_12)
```



```r
# Plotting decomposition for Wind Energy
plot(wind_decomp_ad_12)
```



```r
#Use Multiplicative model also
```

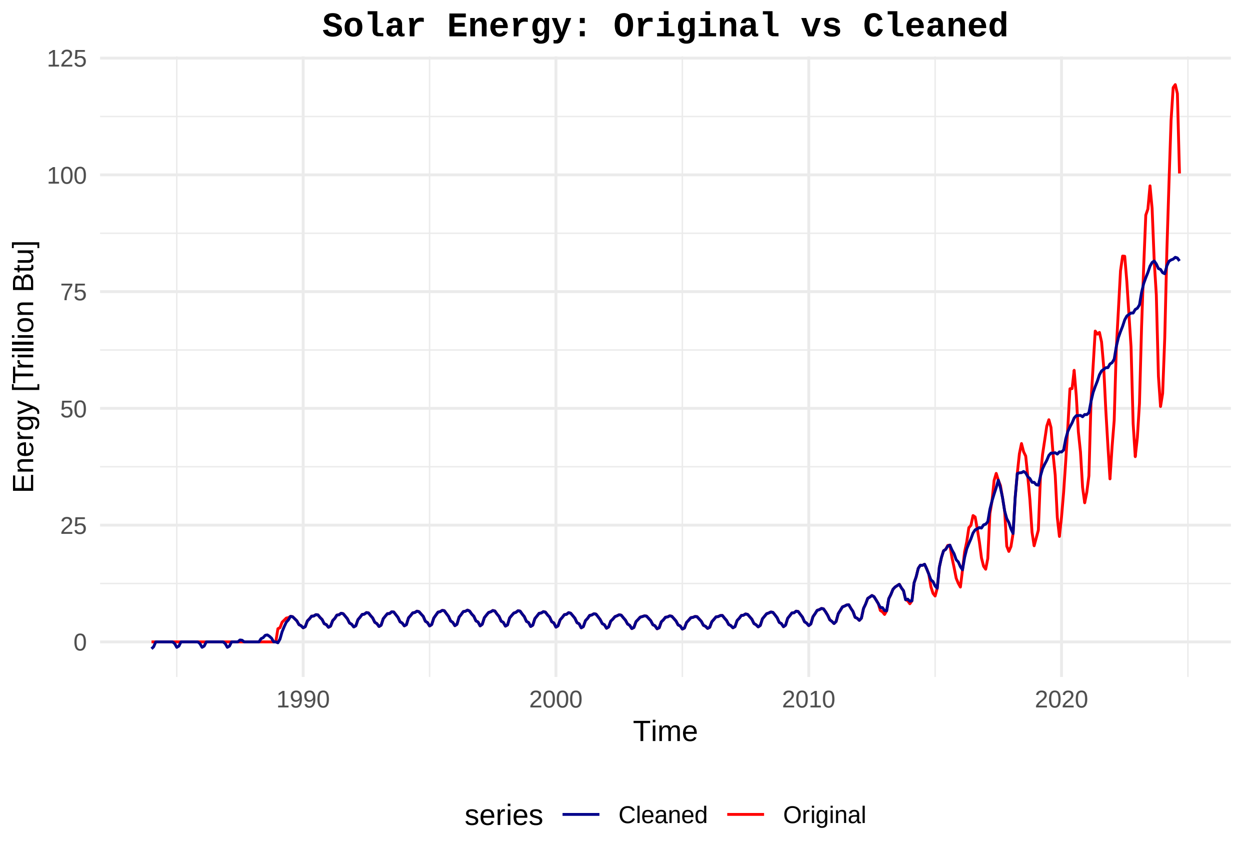
Answer: After filtering the data from January 2012, the trend shows a steady increase. This reflects the long-term growth in energy consumption. The seasonal component stays the same, repeating each year. But now the random component does not look fully random. It still shows patterns, which means some seasonal effects or external factors are not fully removed.

Identify and Remove outliers

Q8

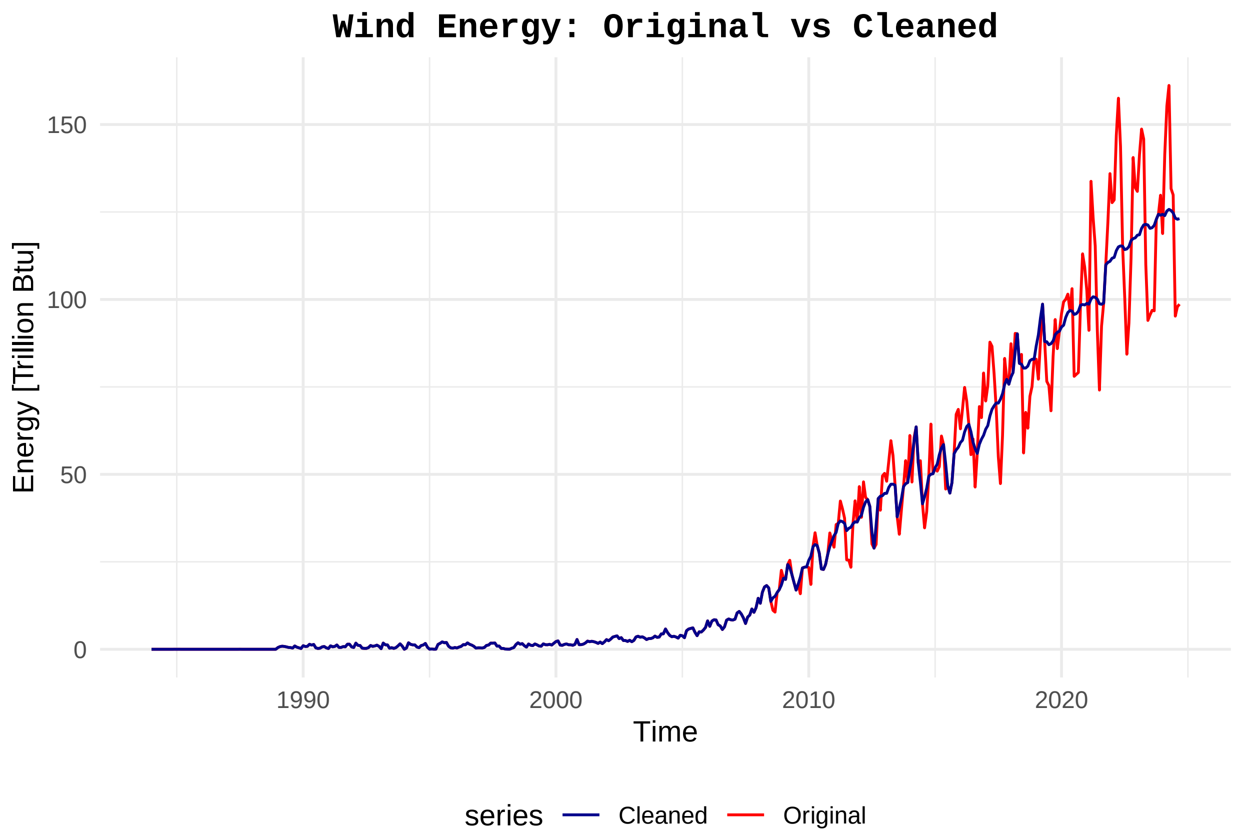
Apply the `tsclean()` to both series from Q7. Did the function removed any outliers from the series? Hint: Use `autoplot()` to check if there is difference between cleaned series and original series.

```r
sol_clean <- tsclean(solar_ts) #applying tsclean to original time series
wind_clean <- tsclean(wind_ts)

autoplot(solar_ts,
            series = "Original") +
  autolayer(sol_clean,
            series = "Cleaned") +
  scale_color_manual(
    values = c("Original" = "red",
               "Cleaned" = "darkblue"
               )) +
  ggtitle("Solar Energy: Original vs Cleaned") +
  xlab("Time") +
  ylab("Energy [Trillion Btu]") +
  theme_minimal()+
  theme(legend.position = "bottom")+
  theme(plot.title = element_text(hjust = 0.5,
                                  face = "bold",
                                  family = "Courier"))
```



```
#This code will generate the same plot

#autoplot(cbind(Original = Sol_Wind_2012_ts[,1], Cleaned = sol_ts_clean)) +
  #ggtitle("Solar Energy Consumption: Original vs Cleaned") +
  #xlab("Year") +
  #ylab("Trillion Btu") +
  #theme_minimal()

#Wind Energy: Original vs Cleaned

autoplot(wind_ts,
```

```r
          series = "Original") +
  autolayer(wind_clean,
            series = "Cleaned") +
  scale_color_manual(
    values = c("Original" = "red",
               "Cleaned" = "darkblue"
               )) +
  ggtitle("Wind Energy: Original vs Cleaned") +
  xlab("Time") +
  ylab("Energy [Trillion Btu]")+
  theme_minimal() +
    theme(legend.position = "bottom") +
    theme(plot.title = element_text(hjust = 0.5,
                                    face = "bold",
                                    family = "Courier"))
```

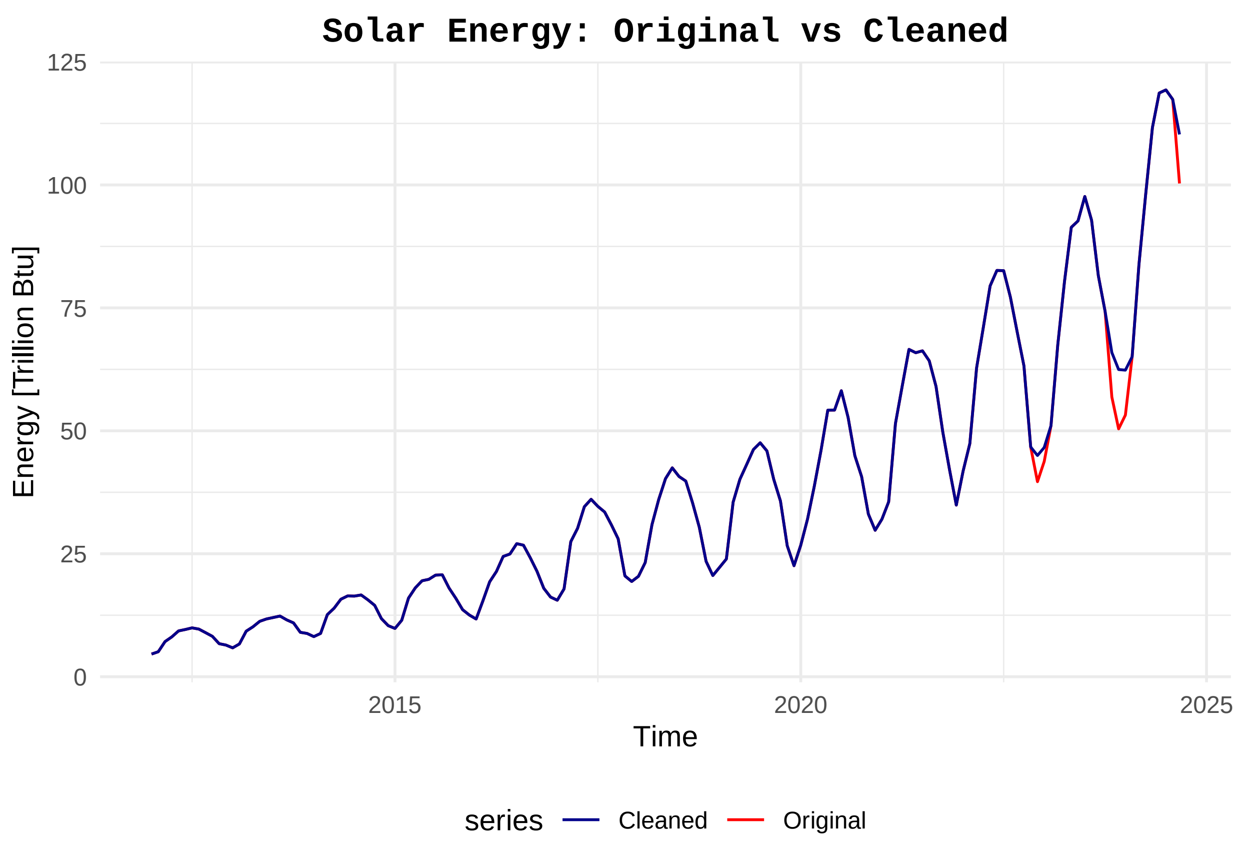


Yes, the tsclean function removed outliers from the series. In the plot, the original series (red) shows sharp spikes or large fluctuations, while the cleaned series (blue) follows a smoother path. The differences between the two lines highlight where outliers were detected and adjusted.

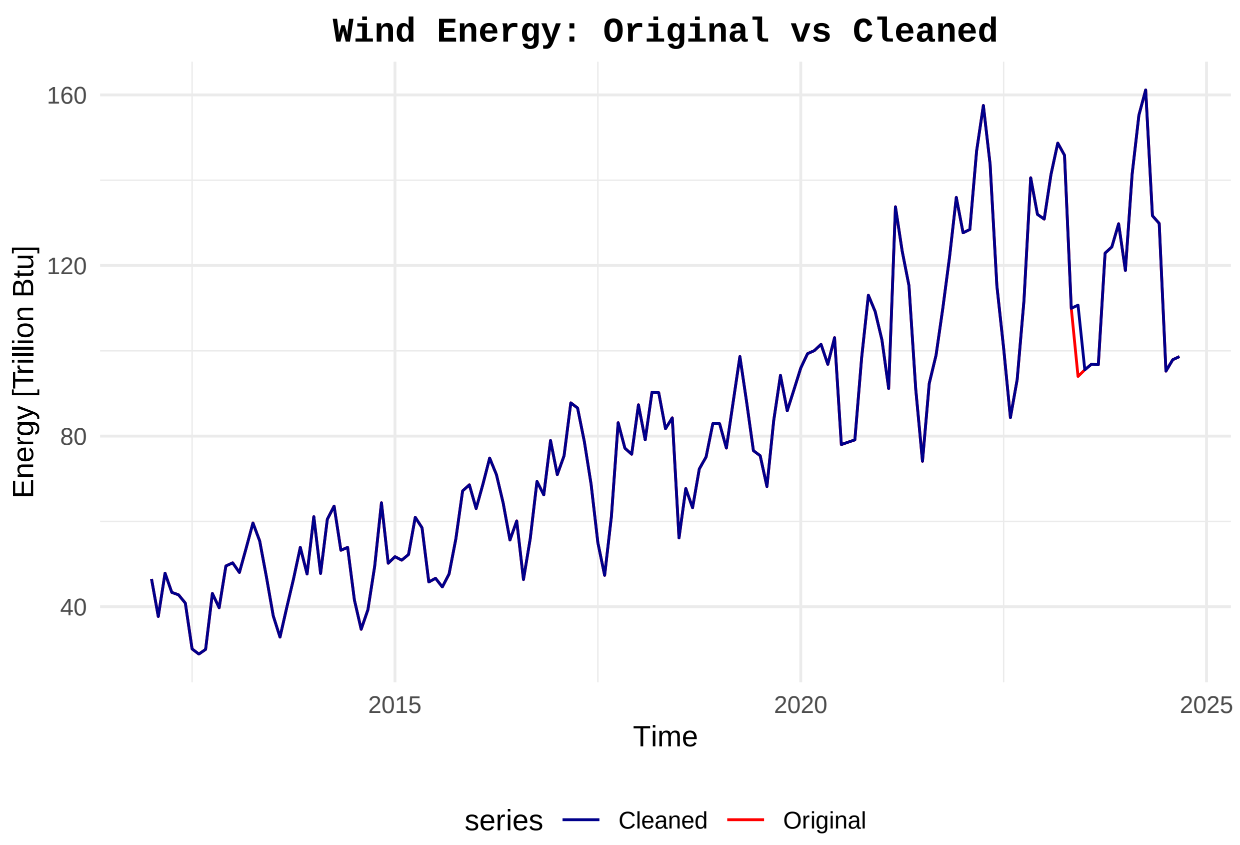
Q9

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?

```r
#Solar Energy: Original vs Cleaned

sol_ts_12 <- ts(Sol_Wind_2012$`Solar Energy Consumption`,
                start = c(2012, 1),
                frequency = 12)
wind_ts_12 <- ts(Sol_Wind_2012$`Wind Energy Consumption`,
                start = c(2012, 1),
                frequency = 12)

sol_clean_12 <- tsclean(sol_ts_12)
wind_clean_12 <- tsclean(wind_ts_12)


autoplot(sol_ts_12,
          series = "Original") +
  autolayer(sol_clean_12,
            series = "Cleaned") +
  scale_color_manual(
    values = c("Original" = "red",
               "Cleaned" = "darkblue"
               )) +
  ggtitle("Solar Energy: Original vs Cleaned") +
  xlab("Time") +
  ylab("Energy [Trillion Btu]") +
  theme_minimal()+
  theme(legend.position = "bottom") +
  theme(plot.title = element_text(hjust = 0.5,
                                  face = "bold",
                                  family = "Courier"))
```



```r
#Wind Energy: Original vs Cleaned
autoplot(wind_ts_12,
            series = "Original") +
  autolayer(wind_clean_12,
            series = "Cleaned") +
  scale_color_manual(
    values = c("Original" = "red",
               "Cleaned" = "darkblue"
               )) +
  ggtitle("Wind Energy: Original vs Cleaned") +
  xlab("Time") +
  ylab("Energy [Trillion Btu]")+
  theme_minimal()+
  theme(legend.position = "bottom") +
  theme(plot.title = element_text(hjust = 0.5,
                                  face = "bold",
                                  family = "Courier"))
```



Answer: Yes, it looks like the function removed outliers from the solar energy data. However, I would argue whether these are truly outliers. The spikes were reduced only slightly, which, in my opinion, requires further investigation.

In the case of wind energy, the function appears to have smoothed only one data point. This raises the question of whether these values were actual outliers. Additionally, removing them may not have had a significant impact on the overall analysis.