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

Assignment 5 - Due date 02/18/25

Daniel Whitehead

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)
```

```
## 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
```

```r
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

## -- 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(cowplot)
```

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

```r
library(readxl)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

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
Table_10_1_Renewable_Energy_Production_and_Consumption_by_Source <- read_excel(
  "~/ENV797/Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xlsx",
  skip = 9)
energy_data <-
  Table_10_1_Renewable_Energy_Production_and_Consumption_by_Source[-1, ]

head(energy_data)
```

```
## # A tibble: 6 x 14
##   Month               `Wood Energy Production` `Biofuels Production`
##   <dttm>              <chr>                    <chr>
## 1 1973-01-01 00:00:00 129.63                   Not Available
```

```
## 2 1973-02-01 00:00:00 117.194                  Not Available
## 3 1973-03-01 00:00:00 129.763                  Not Available
## 4 1973-04-01 00:00:00 125.462                  Not Available
## 5 1973-05-01 00:00:00 129.624                  Not Available
## 6 1973-06-01 00:00:00 125.435                  Not Available
## # i 11 more variables: 'Total Biomass Energy Production' <chr>,
## #   'Total Renewable Energy Production' <chr>,
## #   'Hydroelectric Power Consumption' <chr>,
## #   'Geothermal Energy Consumption' <chr>, 'Solar Energy Consumption' <chr>,
## #   'Wind Energy Consumption' <chr>, 'Wood Energy Consumption' <chr>,
## #   'Waste Energy Consumption' <chr>, 'Biofuels Consumption' <chr>,
## #   'Total Biomass Energy Consumption' <chr>, ...
```

```r
# Get the number of rows (observations) and columns (variables)
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
selected_data <- energy_data %>%
  select(Month, `Solar Energy Consumption`, `Wind Energy Consumption`) %>%
  filter(`Solar Energy Consumption` != "Not Available" & `Wind Energy Consumption` != "Not Available") %
  mutate(
    `Solar Energy Consumption` = as.numeric(`Solar Energy Consumption`),
    `Wind Energy Consumption` = as.numeric(`Wind Energy Consumption`)
  ) %>%
  drop_na()  # Drop rows with any remaining NAs

# View the cleaned data
head(selected_data)
```

```
## # A tibble: 6 x 3
##   Month               'Solar Energy Consumption' 'Wind Energy Consumption'
##   <dttm>                                   <dbl>                     <dbl>
## 1 1984-01-01 00:00:00                      0                         0
## 2 1984-02-01 00:00:00                      0                         0.001
## 3 1984-03-01 00:00:00                      0.001                     0.001
## 4 1984-04-01 00:00:00                      0.001                     0.002
## 5 1984-05-01 00:00:00                      0.002                     0.003
## 6 1984-06-01 00:00:00                      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")")`

```
#I regroup to make the names easier to use in the plot
selected_data_long <- selected_data %>%
  gather(key = "Energy_Type", value = "Consumption", `Solar Energy Consumption`,
         `Wind Energy Consumption`)

ggplot(selected_data_long, aes(x = as.Date(Month, format = "%Y"), y = Consumption)) +
  geom_line(color = "blue") +
  facet_wrap(~ Energy_Type, scales = "fixed") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  ylab("Energy Consumption (Trillion BTU)") +
  xlab("Time")+
  labs(color = "Energy Type", caption = "Figure 1: Wind and Solar Energy
       Consumption Time Series. This plot shows the consumption of solar and
       wind energy from 1984 to present, highlighting the trends over time
       between the two energy types.") +
  theme_minimal() +  # Use a minimal theme
  theme(
    strip.text = element_text(size = 12, face = "bold"),  # Bold facet labels
    axis.title = element_text(size = 14),  # Adjust axis title size
    axis.text = element_text(size = 8)   # Adjust axis text size
  ) +
  ggtitle("Solar and Wind Energy Consumption Over Time")
```

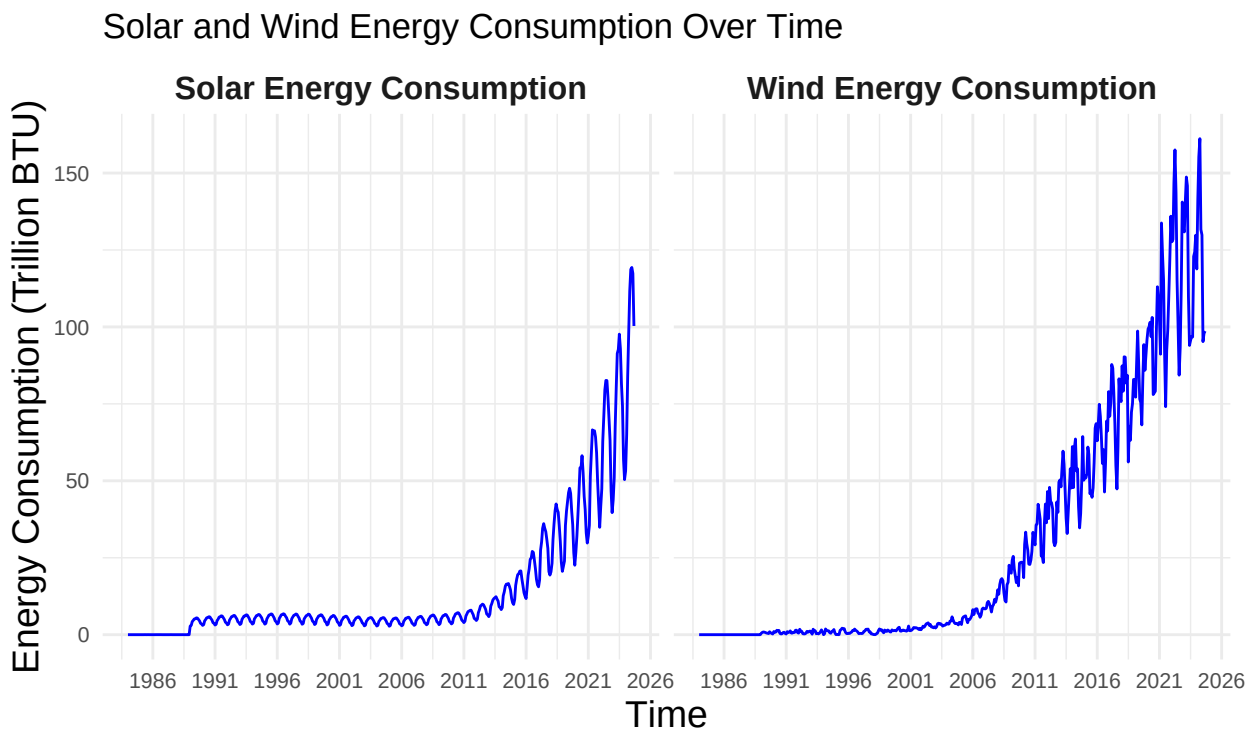


Figure 1: Wind and Solar Energy Consumption Time Series. This plot shows the consumption of solar and wind energy from 1984 to present, highlighting the trends over time between the two energy types.

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(selected_data_long, aes(x = as.Date(Month, format = "%Y"), y = Consumption, color = Energy_Type)
  geom_line() +  # Line graph for both series
  scale_color_manual(values = c("Solar Energy Consumption" = "red",
                                "Wind Energy Consumption" = "blue")) +  # Manually set colors for the l
  scale_x_date(date_breaks = "5 years", date_labels = "%Y") +
  ylab("Energy Consumption (Trillion BTU)") +  # Label for the y-axis
  xlab("Time") +  # Label for the x-axis
  labs(color = "Energy Type", caption = "Figure 2: Solar and Wind Energy
       Consumption Over Time. This plot shows the consumption of solar and
       wind energy from 1984 to present, highlighting the trends over time
       between the two energy types.") +
  theme_minimal() + # Change the legend title to "Energy Type"
  theme_minimal() +  # Use a minimal theme
  theme(
    axis.title = element_text(size = 14),  # Adjust axis title size
    axis.text = element_text(size = 8),    # Adjust axis text size
    legend.title = element_text(size = 12),  # Adjust legend title size
    legend.text = element_text(size = 8),
     plot.caption = element_text(size = 10, hjust = 0) # Adjust legend text size
  ) +
  ggtitle("Solar and Wind Energy Consumption Over Time")  # Plot title
```

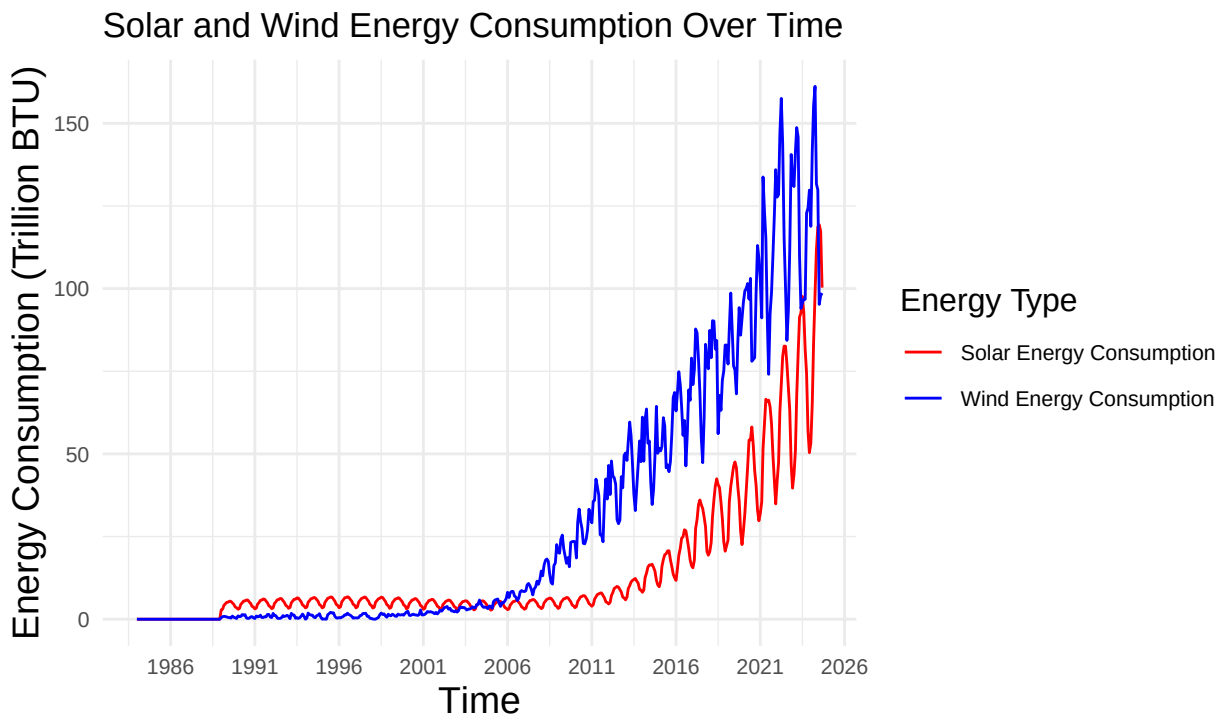


Figure 2: Solar and Wind Energy Consumption Over Time. This plot shows the consumption of solar and wind energy from 1984 to present, highlighting the trends over time between the two energy types.

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(selected_data_long %>%
                 filter(Energy_Type == "Solar Energy Consumption") %>%
                 pull(Consumption),
               start = c(1984, 1), frequency = 12)  # Adjust start and frequency based on your data

# Decompose the Solar Energy Consumption time series
decompose_solar <- decompose(solar_ts, type = "additive")

# Plot the decomposition result for Solar Energy
plot(decompose_solar)
mtext("Decomposition of Solar Energy Consumption", side = 3, line = 0.7, cex = 1.0)
# Create a time series object for Wind Energy Consumption
mtext("Figure 4: Decomposed components of Solar Energy Consumption.", side = 1, line = 4, cex = 0.8)
```

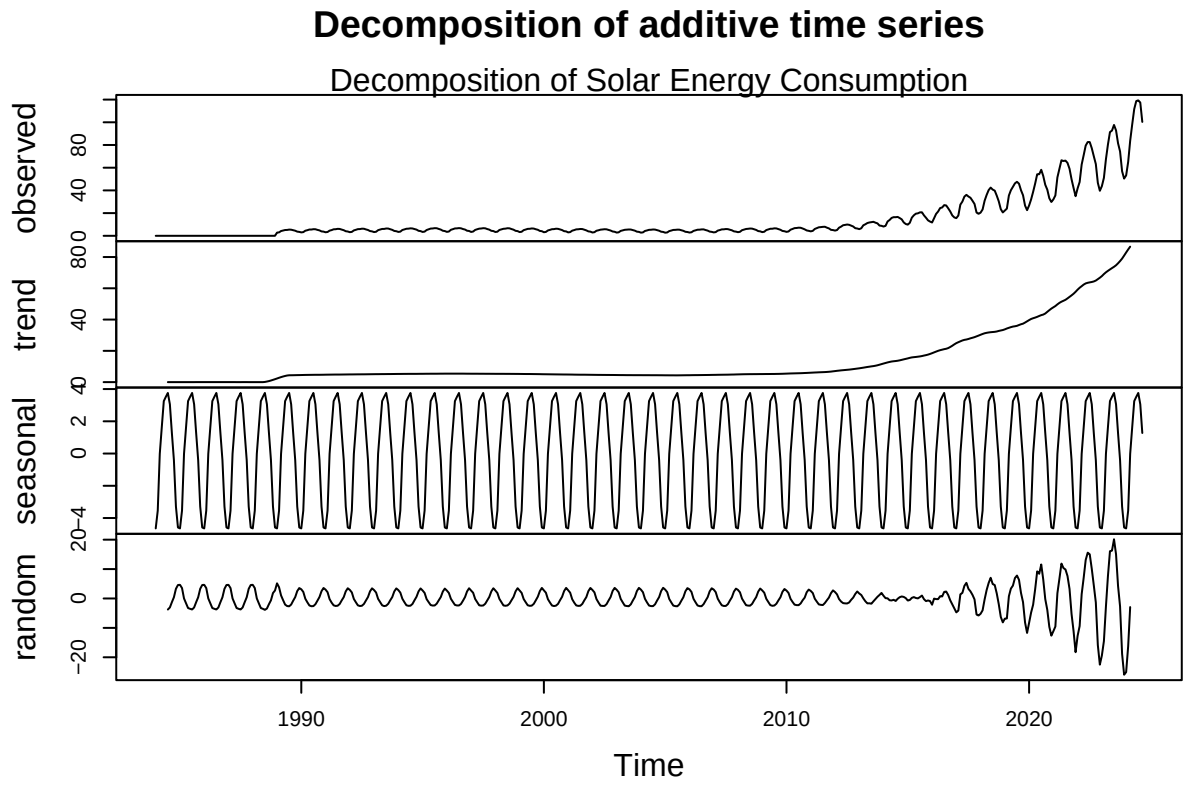


Figure 4: Decomposed components of Solar Energy Consumption.

```r
wind_ts <- ts(selected_data_long %>%
                 filter(Energy_Type == "Wind Energy Consumption") %>%
                 pull(Consumption),
               start = c(1984, 1), frequency = 12)  # Adjust start and frequency based on your data

# Decompose the Wind Energy Consumption time series
decompose_wind <- decompose(wind_ts, type = "additive")

# Plot the decomposition result for Wind Energy
plot(decompose_wind)
mtext("Decomposition of Wind Energy Consumption", side = 3, line = 0.7, cex = 1.0)
```

```
# Add a caption below the plot
mtext("Figure 4: Decomposed components of Wind Energy Consumption.", side = 1, line = 4, cex = 0.8)
```

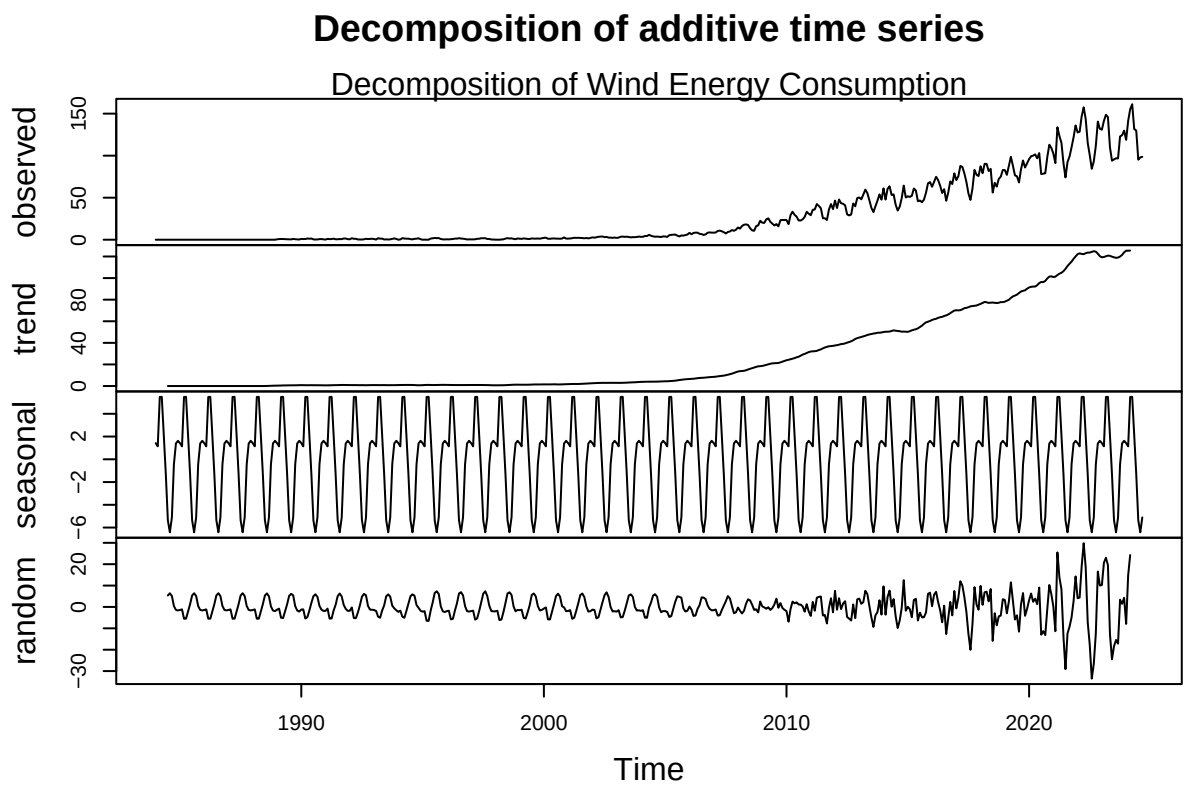


Figure 4: Decomposed components of Wind Energy Consumption.

< 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? < Answer: The trend shows the rise that becomes a quicker, steeper rise starting in about 2010, as there is a rise in energy consumption from both solar and wind. The random component is really only varying to a wide degree past about 2012-2015. This means that the random component stayed around 0 up until this point in history. There appears to maybe be a little bit of seasonality, as there are still highs and lows of the data, and the data starts to be more volatile past the mid 2010's.
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?

```
solar_ts <- ts(selected_data_long %>%
                 filter(Energy_Type == "Solar Energy Consumption") %>%
                 pull(Consumption),
               start = c(1984, 1), frequency = 12)  # Adjust start and frequency based on your data

# Decompose the Solar Energy Consumption time series
decompose_solar <- decompose(solar_ts, type = "multiplicative")

# Plot the decomposition result for Solar Energy
plot(decompose_solar)
mtext("Decomposition of Solar Energy Consumption", side = 3, line = 0.7, cex = 1.0)
```

```
mtext("Figure 5: Decomposed components of Solar Energy Consumption (multi).",
      side = 1, line = 4, cex = 0.8)
```

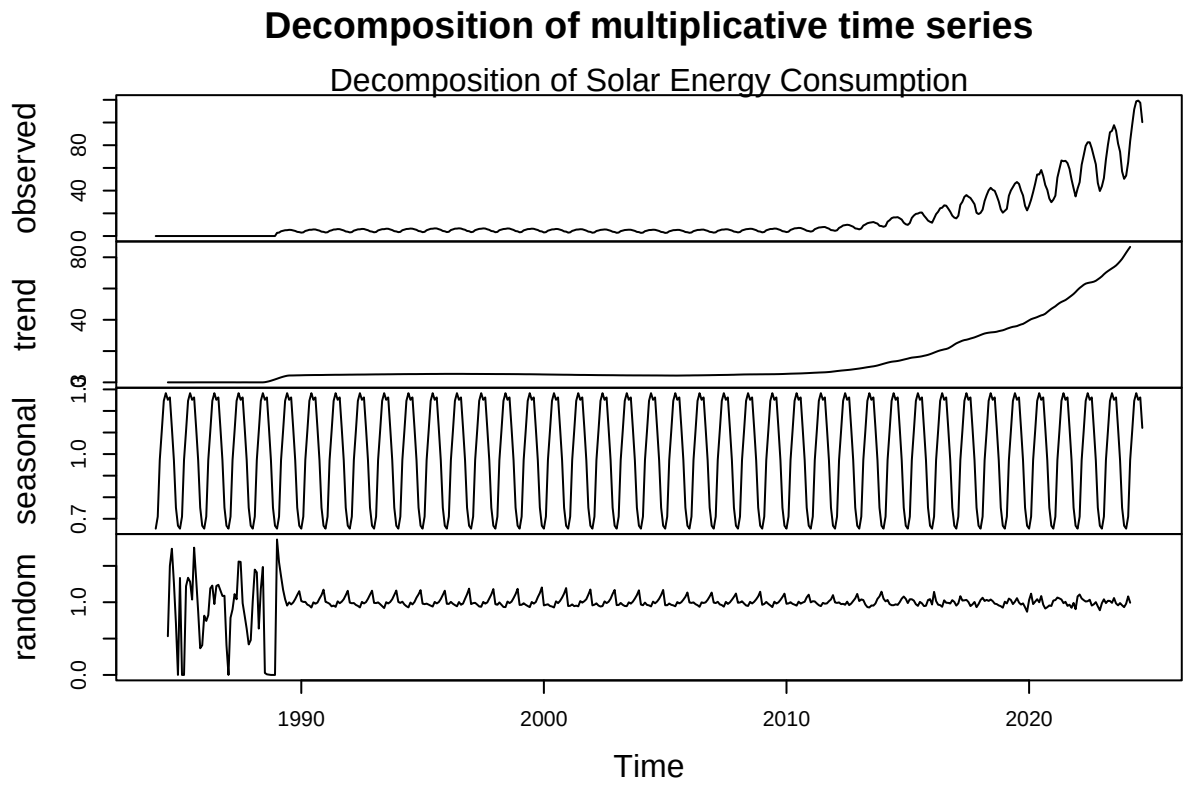


Figure 5: Decomposed components of Solar Energy Consumption (multi).

```
# Create a time series object for Wind Energy Consumption
wind_ts <- ts(selected_data_long %>%
                filter(Energy_Type == "Wind Energy Consumption") %>%
                pull(Consumption),
              start = c(1984, 1), frequency = 12)  # Adjust start and frequency based on your data

# Decompose the Wind Energy Consumption time series
decompose_wind <- decompose(wind_ts, type = "multiplicative")

# Plot the decomposition result for Wind Energy
plot(decompose_wind)
mtext("Decomposition of Wind Energy Consumption", side = 3, line = 0.7,
      cex = 1.0)
mtext("Figure 6: Decomposed components of Wind Energy Consumption (multi).",
      side = 1, line = 4, cex = 0.8)
```

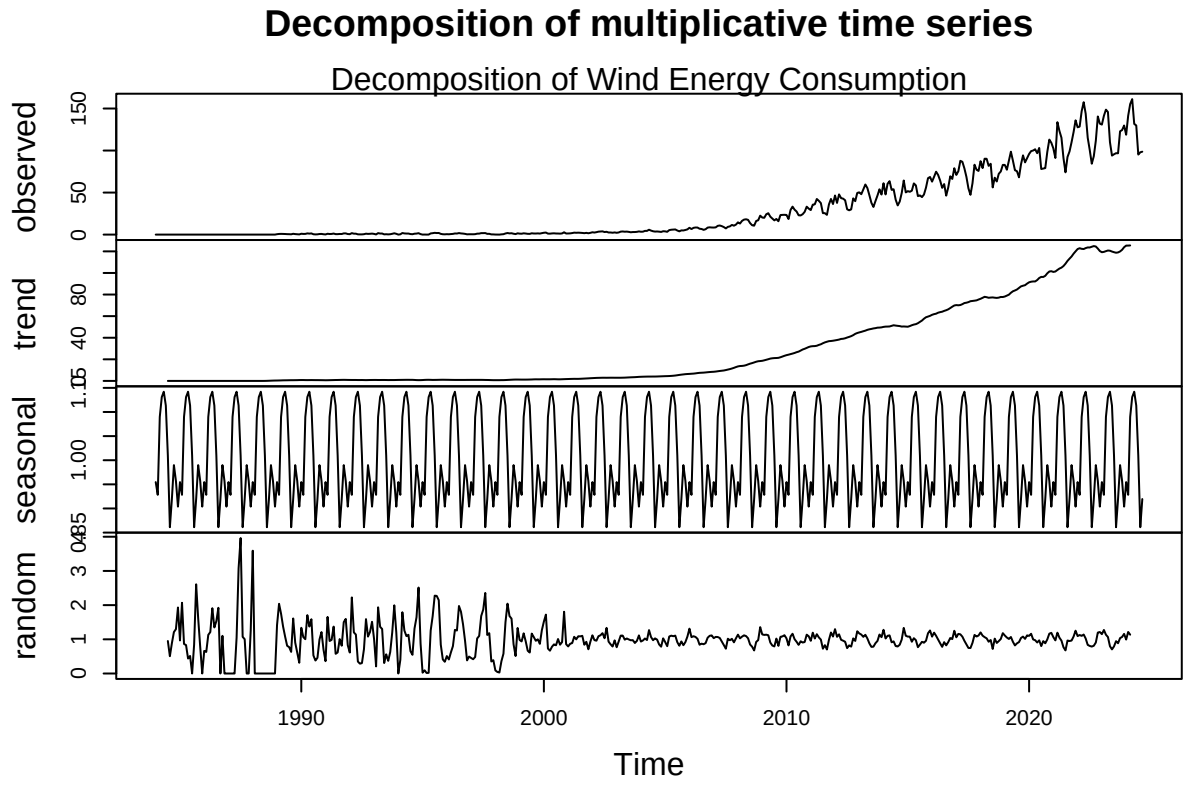


Figure 6: Decomposed components of Wind Energy Consumption (multi).

< This time, the random component is a lot more volatile before about 2000 for the wind series, and aout 1990 for the solar series. This means that the data before this time may be more irrlelevant in terms of our total data we are looking at. ### 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 would say that the data before about 2010 is somewhat irrelevant to our forecast, as this data seems very out-dated. Additionally, the data before about 2010 has very small values, and our data does not really start trending upwards in consumption (Trillion BTU) until just before 2010. Solar and Wind have scaled up so much in consumption since 2010, so it would make sense to only use this recent data for our forecasts.

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.

```
selected_data_filtered <- selected_data_long %>%
  filter(year(as.Date(Month, format = "%Y-%m")) >= 2012)  # Filter data from January 2012 onward

# Create a time series object for Solar Energy Consumption
```

```r
solar_ts_new <- ts(selected_data_filtered %>%
                     filter(Energy_Type == "Solar Energy Consumption") %>%
                     pull(Consumption),
                   start = c(2012, 1), frequency = 12)  # Adjust start and frequency based on the filte

# Decompose the Solar Energy Consumption time series using additive decomposition
decompose_solar_new <- decompose(solar_ts_new, type = "additive")

# Plot the decomposition result for Solar Energy (after filtering from 2012)
plot(decompose_solar_new)
mtext("Decomposition of Solar Energy Consumption", side = 3, line = 0.7, cex = 1.0)
mtext("Figure 7: Time Series of Decomposed components of Solar Energy Consumption (additive).", side =
```

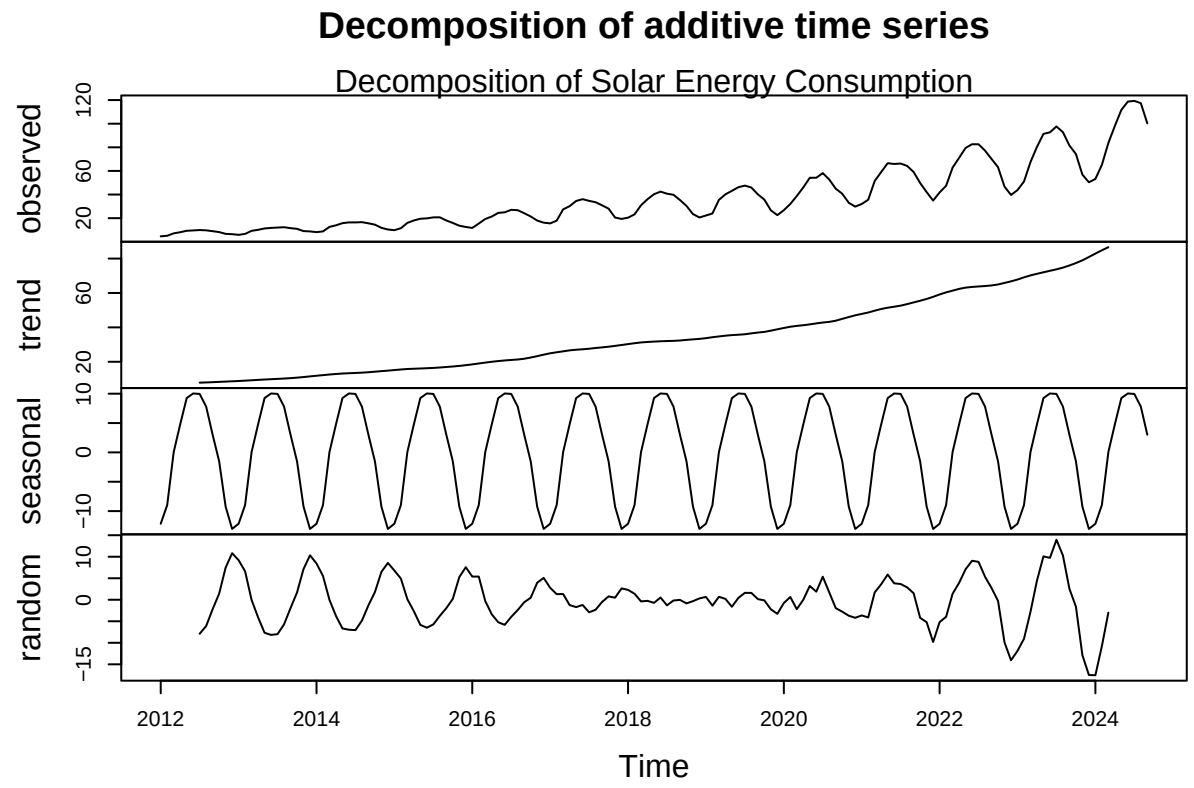


Figure 7: Time Series of Decomposed components of Solar Energy Consumption (additive).

```r
# Create a time series object for Wind Energy Consumption
wind_ts_new <- ts(selected_data_filtered %>%
                     filter(Energy_Type == "Wind Energy Consumption") %>%
                     pull(Consumption),
                   start = c(2012, 1), frequency = 12)  # Adjust start and frequency based on the filte

# Decompose the Wind Energy Consumption time series using additive decomposition
decompose_wind_new <- decompose(wind_ts_new, type = "additive")

# Plot the decomposition result for Wind Energy (after filtering from 2012)
plot(decompose_wind_new)
mtext("Decomposition of Wind Energy Consumption", side = 3, line = 0.7, cex = 1.0)
```

```
mtext("Figure 8: Time Series of Decomposed components of Wind Energy Consumption (additive).", side = 1
```

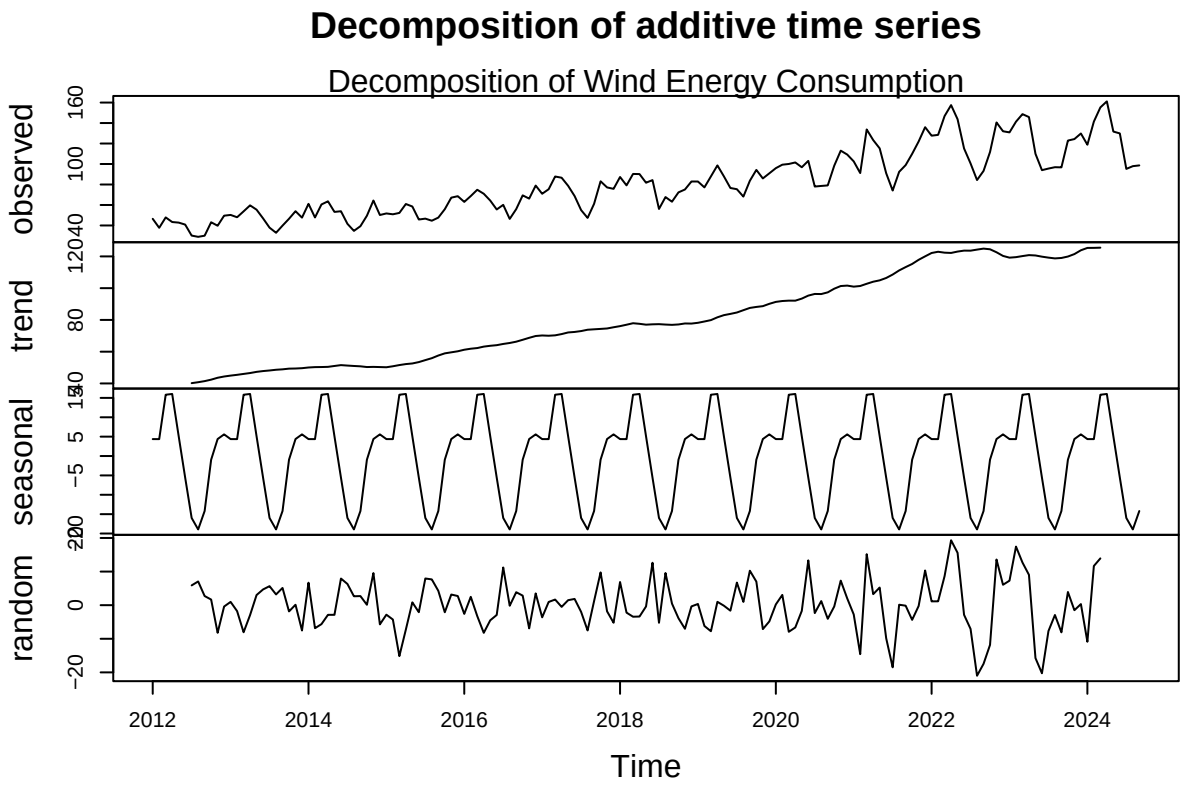


Figure 8: Time Series of Decomposed components of Wind Energy Consumption (additive).

Answer: For the solar series, the randomness is pretty random, though there is less volatility between about 2018-2020. As for the wind data, the data is continuously rising and falling, dipping in and out of the negatives. However, there may be an aspect of AR or there being a slight historical consideration by the system, as data can kind of rise and fall with other points.

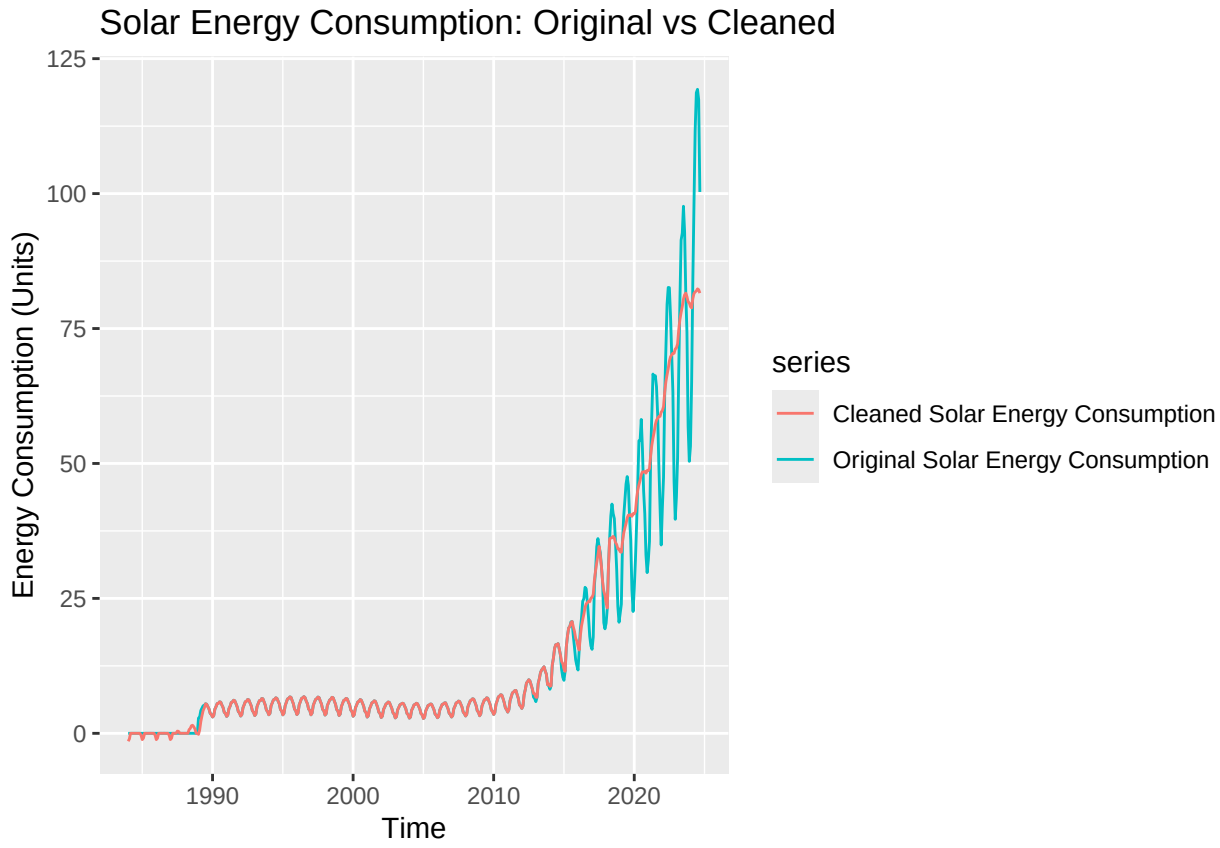
Identify and Remove outliers

Q8

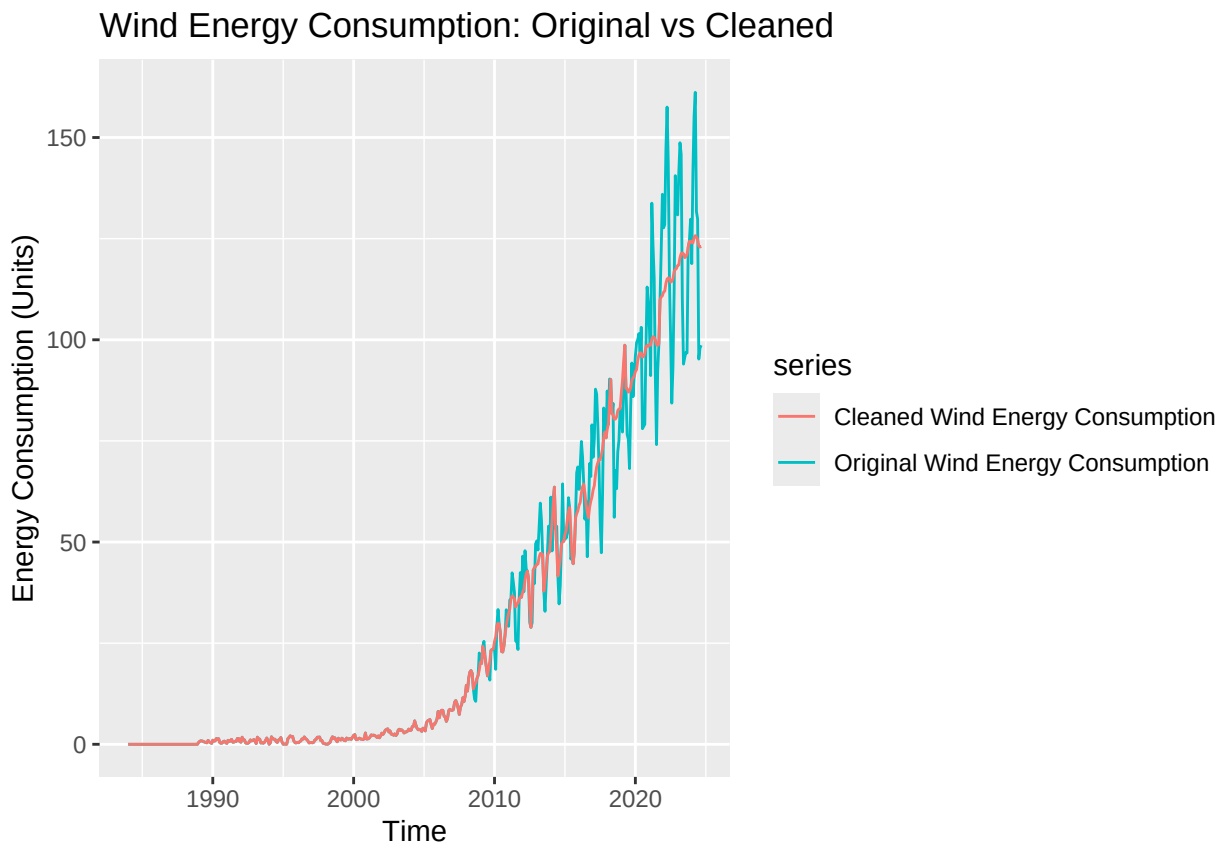
Apply the `tsclean()` to both series from Q4. Did the function removed any outliers from the series? Hint: Use `autoplot()` to check if there is difference between cleaned series and original series.

```
# Apply tsclean() to the original Solar Energy Consumption time series
solar_ts_clean_original <- tsclean(solar_ts)

# Apply tsclean() to the original Wind Energy Consumption time series
wind_ts_clean_original <- tsclean(wind_ts)

# Plot the original and cleaned Solar Energy Consumption time series
autoplot(solar_ts, series = "Original Solar Energy Consumption") +
  autolayer(solar_ts_clean_original, series = "Cleaned Solar Energy Consumption") +
  ggtitle("Solar Energy Consumption: Original vs Cleaned") +
```

```
  ylab("Energy Consumption (Units)") +
  xlab("Time")
```



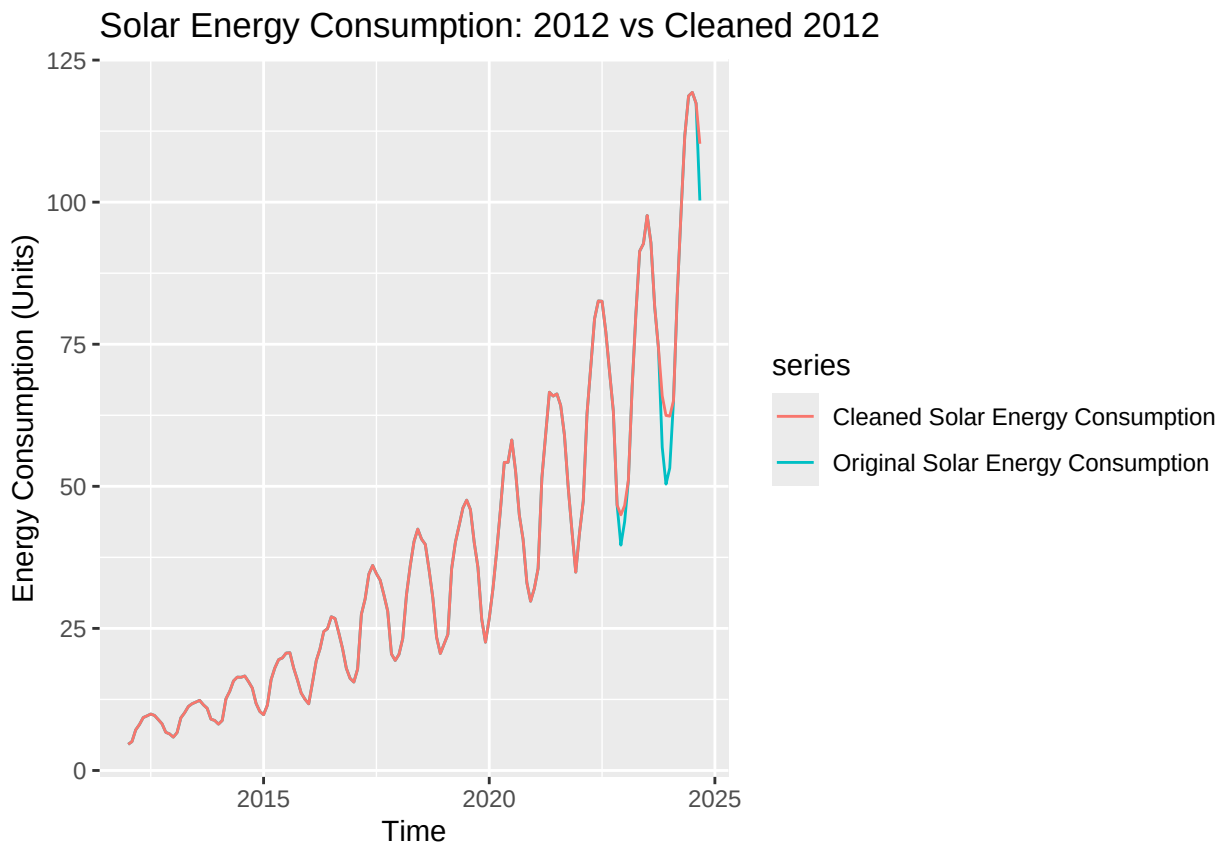

```
# Plot the original and cleaned Wind Energy Consumption time series
autoplot(wind_ts, series = "Original Wind Energy Consumption") +
  autolayer(wind_ts_clean_original, series = "Cleaned Wind Energy Consumption") +
  ggtitle("Wind Energy Consumption: Original vs Cleaned") +
  ylab("Energy Consumption (Units)") +
  xlab("Time")
```



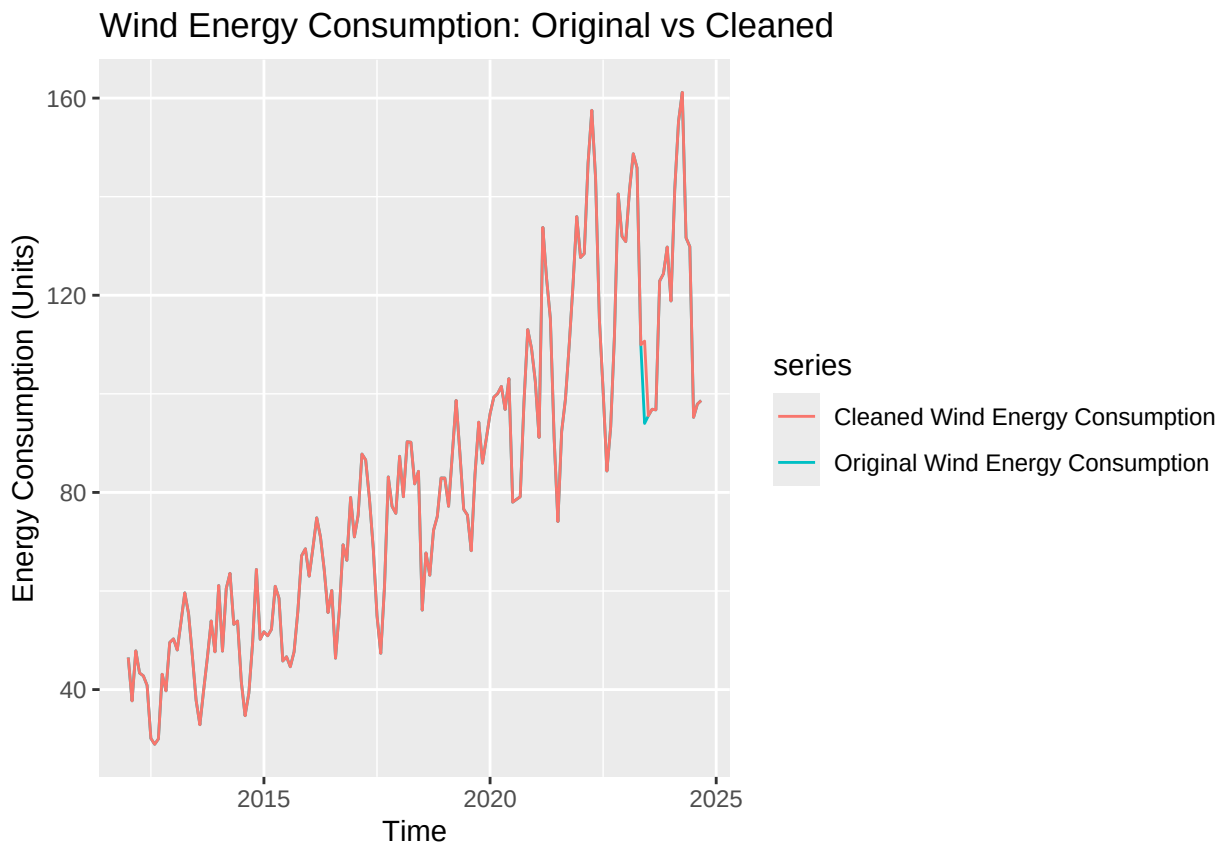
Q9

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?

```
solar_ts_clean <- tsclean(solar_ts_new)
wind_ts_clean <- tsclean(wind_ts_new)


# Plot the original and cleaned Solar Energy Consumption time series
autoplot(solar_ts_new, series = "Original Solar Energy Consumption") +
  autolayer(solar_ts_clean, series = "Cleaned Solar Energy Consumption") +
  ggtitle("Solar Energy Consumption: 2012 vs Cleaned 2012") +
  ylab("Energy Consumption (Units)") +
  xlab("Time")
```




```
# Plot the original and cleaned Wind Energy Consumption time series
autoplot(wind_ts_new, series = "Original Wind Energy Consumption") +
  autolayer(wind_ts_clean, series = "Cleaned Wind Energy Consumption") +
  ggtitle("Wind Energy Consumption: Original vs Cleaned") +
  ylab("Energy Consumption (Units)") +
  xlab("Time")
```



> Answer: There is a large difference in the outliers filtered. When we clean the whole timeline of data, it shows numerous outliers in the 2012-present range, deeming a lot of this data to be outside of the buffer (within ~ standard deviations) for the trend. However, when we focus into the 2012-present data and clean that data, it shows only a few outliers in the data. This means that the 2012-present data has a different trend and standard deviation compared to the overall data. The result of these plots support my answer for Q6 because it shows that the data before 2012 is somewhat irrelevant to our forecasts.