Homework3_201611531

Jeong Hojae

2021 4 4

0. Read the data

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(forecast)
```

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

```
library(seasonal)
```

```
##
## Attaching package: 'seasonal'
```

```
## The following object is masked from 'package:tibble':
##
##     view
```

```
library(aTSA)
```

```
##
## Attaching package: 'aTSA'
```

```
## The following object is masked from 'package:forecast':
##
##     forecast
```

```
## The following object is masked from 'package:graphics':
##
##     identify
```

```
library(timsac)

data <- read_csv("/cloud/project/loadregr.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   MKw = col_double(),
##   Month = col_character(),
##   Time = col_double()
## )
```

```
dim(data)
```

```
## [1] 120   3
```

```
summary(data)
```

```
##       MKw           Month                Time
##  Min.   :6.425   Length:120         Min.   :1970
##  1st Qu.:6.714   Class :character   1st Qu.:1972
##  Median :6.894   Mode  :character   Median :1975
##  Mean   :6.887                      Mean   :1975
##  3rd Qu.:7.032                      3rd Qu.:1977
##  Max.   :7.323                      Max.   :1980
```

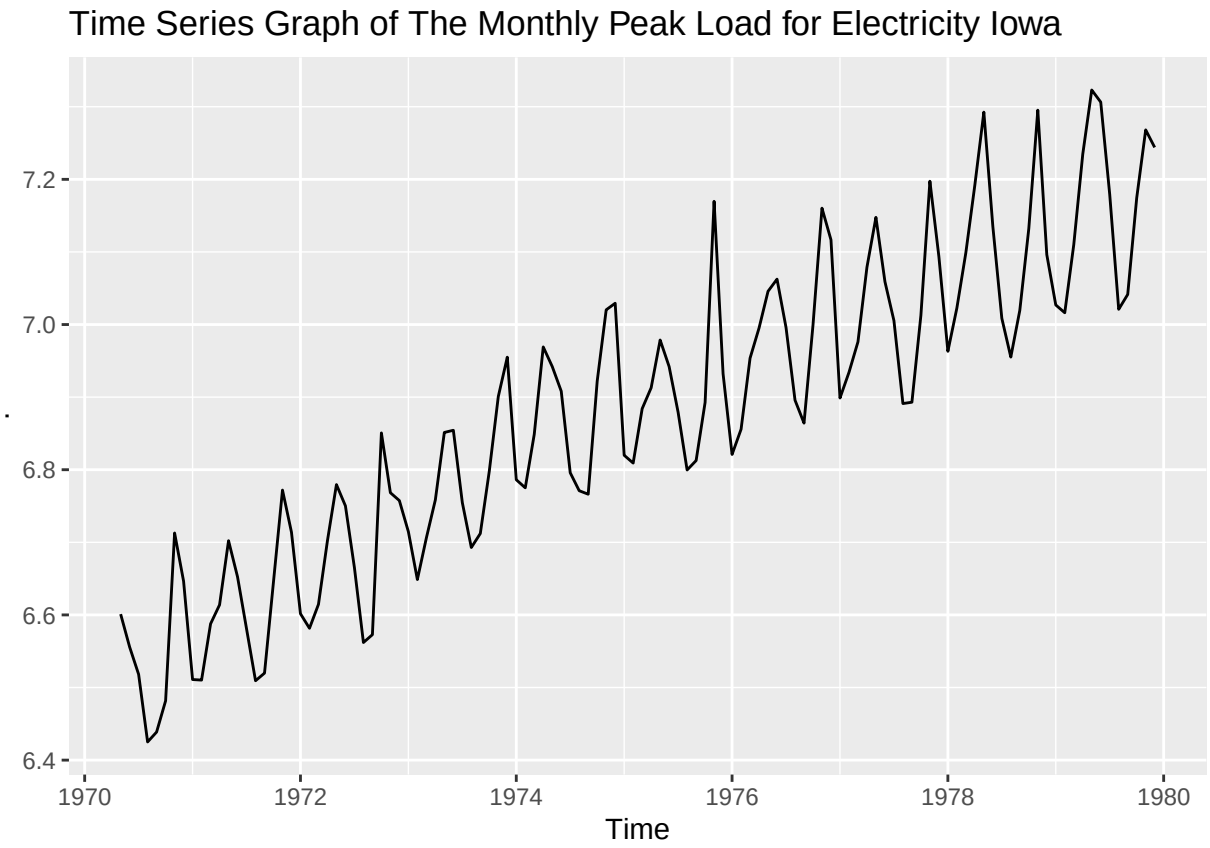
```
head(data)
```

```
## # A tibble: 6 x 3
##     MKw Month  Time
##   <dbl> <chr> <dbl>
## 1  6.60 Jan   1970
## 2  6.56 Feb   1970.
## 3  6.52 Mar   1970.
## 4  6.42 Apr   1970.
## 5  6.44 May   1970.
## 6  6.48 Jun   1970.
```

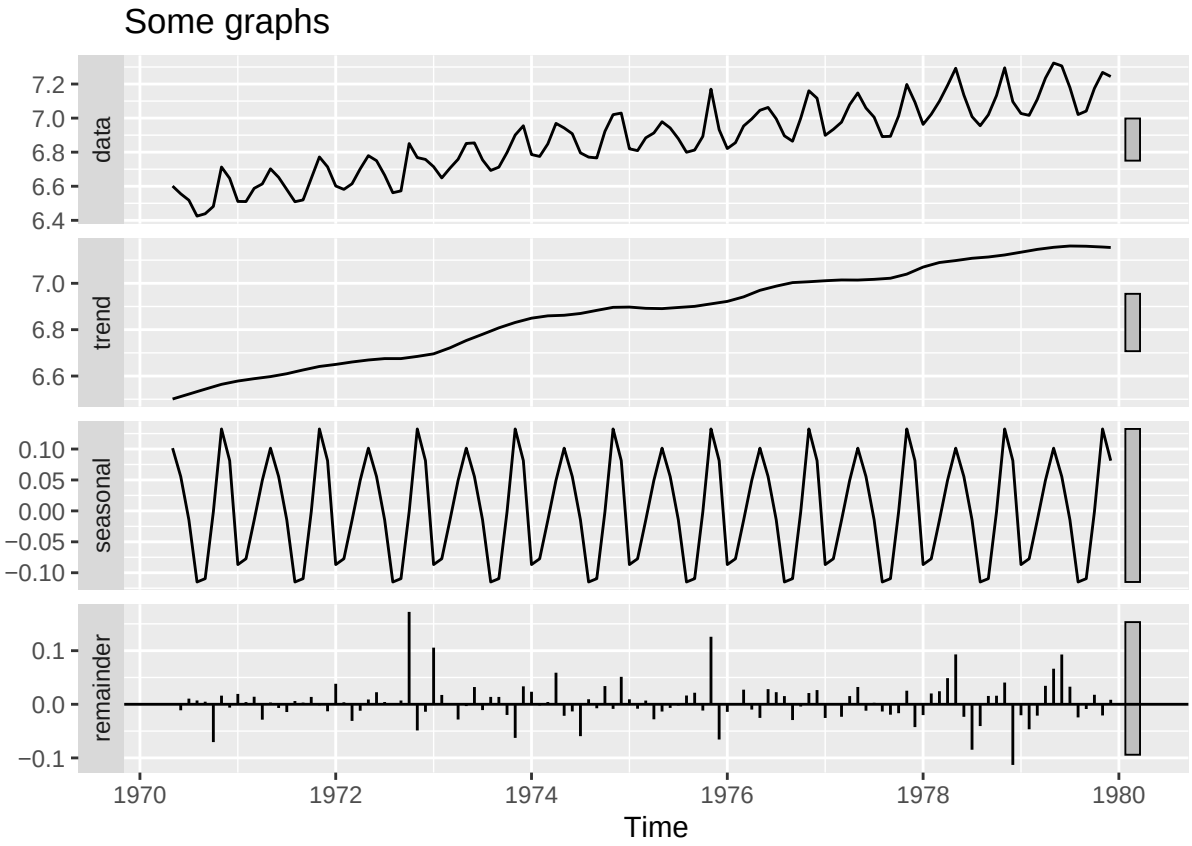
```
data_ts<-ts(data$MKw, start=c(1970, 5), end=c(1979, 12), frequency=12 )
```

1. Draw time series plot

```
data_ts %>% autoplot() +
  ggtitle("Time Series Graph of The Monthly Peak Load for Electricity Iowa")
```




```
data_ts %>%
  stl(t.window=13, s.window="periodic", robust=TRUE) %>%
  autoplot()+
  ggtitle("Some graphs")
```



The data has trend variation and seasonal trend. But, it doesn't seem to show about dependent of them.

2. Fit decomposition model and draw a plot with the original data, seasonally adjusted, trend-cycle component all together.

Classical decomposition

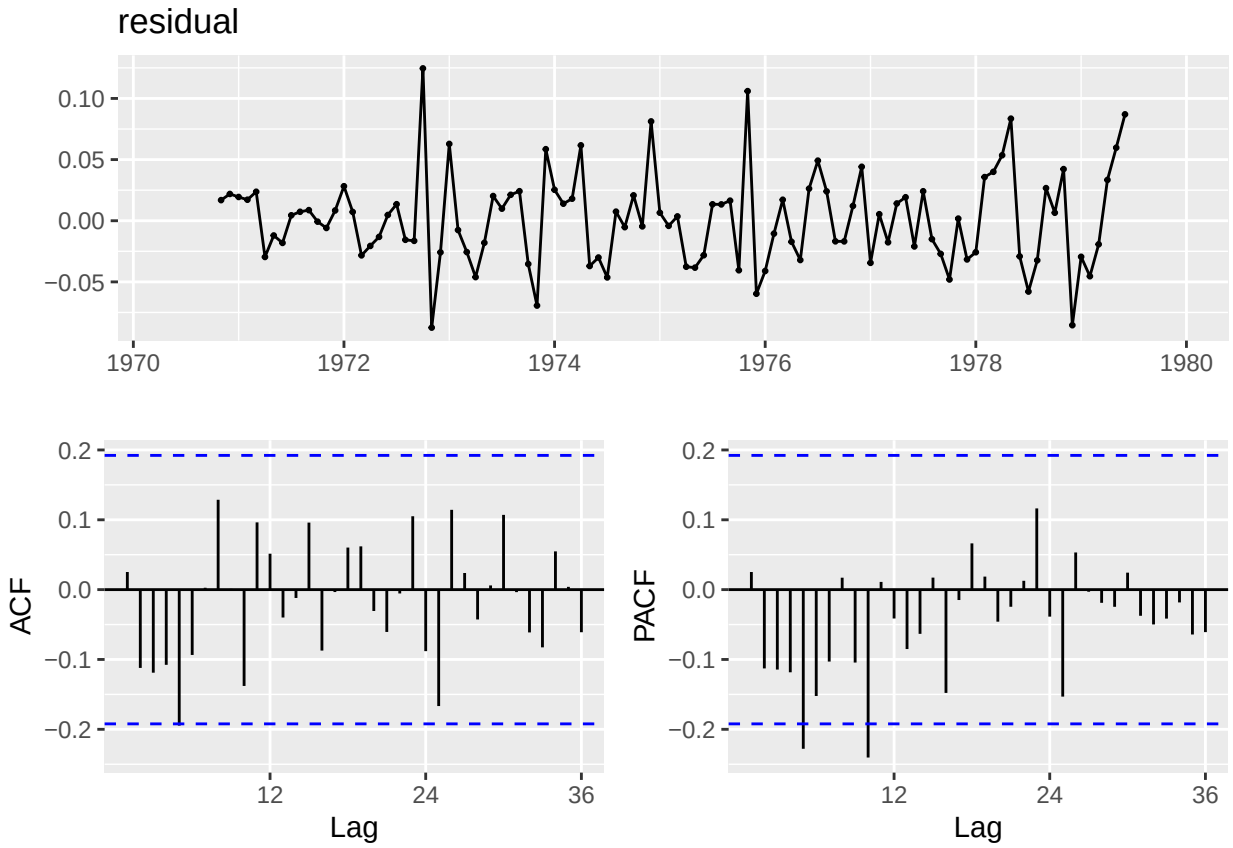
```
# additive
dd1=decompose(data_ts,type="additive")
tseries::kpss.test(dd1$random,null="Level")
```

```
## Warning in tseries::kpss.test(dd1$random, null = "Level"): p-value greater than
## printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  dd1$random
## KPSS Level = 0, Truncation lag parameter = 4, p-value = 0.1
```

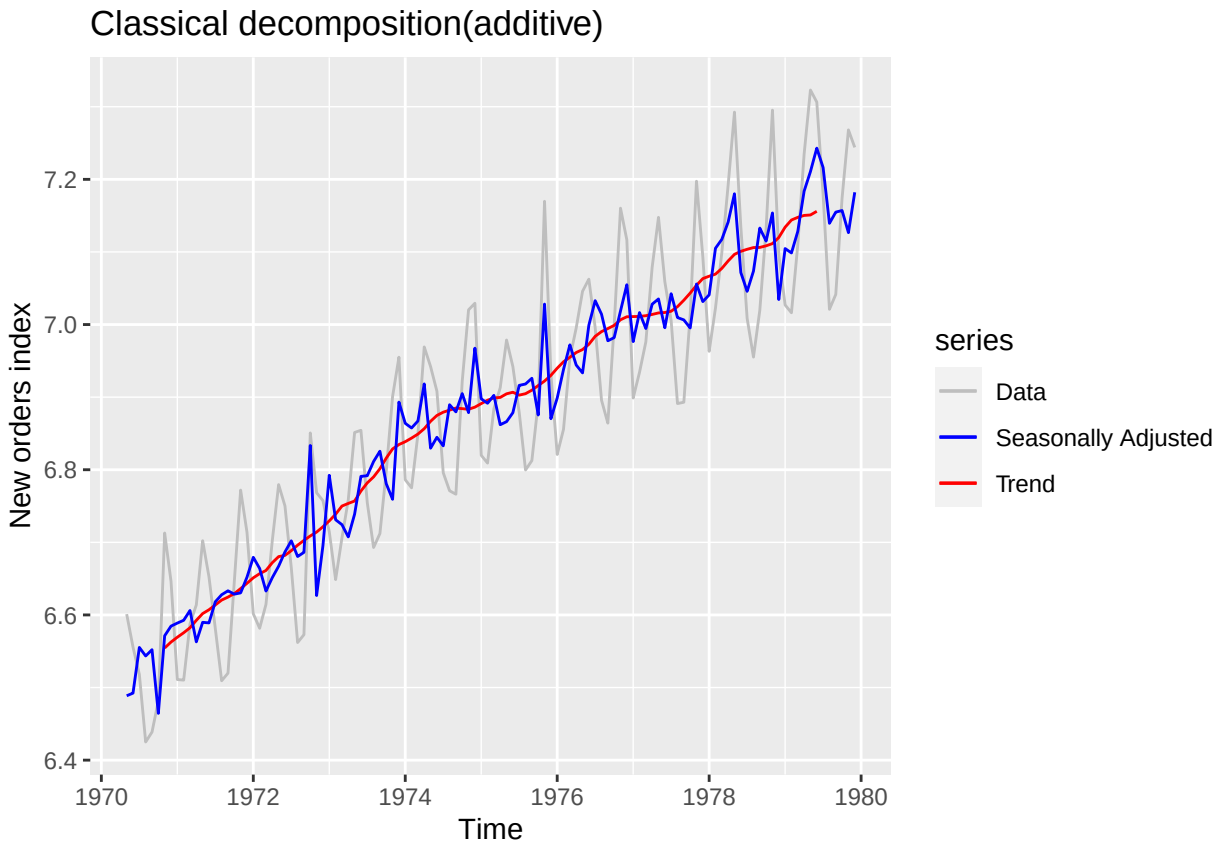
```
ggtsdisplay(dd1$random, main="residual")
```

```
## Warning: Removed 12 rows containing missing values (geom_point).
```




```
autoplot(data_ts, series="Data")+
  autolayer(trendcycle(dd1),series="Trend") +
  autolayer(seasadj(dd1),series="Seasonally Adjusted") +
  xlab("Time") + ylab("New orders index")+
  ggtitle("Classical decomposition(additive)") +
  scale_color_manual(values=c("gray","blue","red"),breaks=c("Data","Seasonally Adjusted","Trend"))
```

```
## Warning: Removed 12 row(s) containing missing values (geom_path).
```



Additive model is used when the seasons are constant regardless of the trend

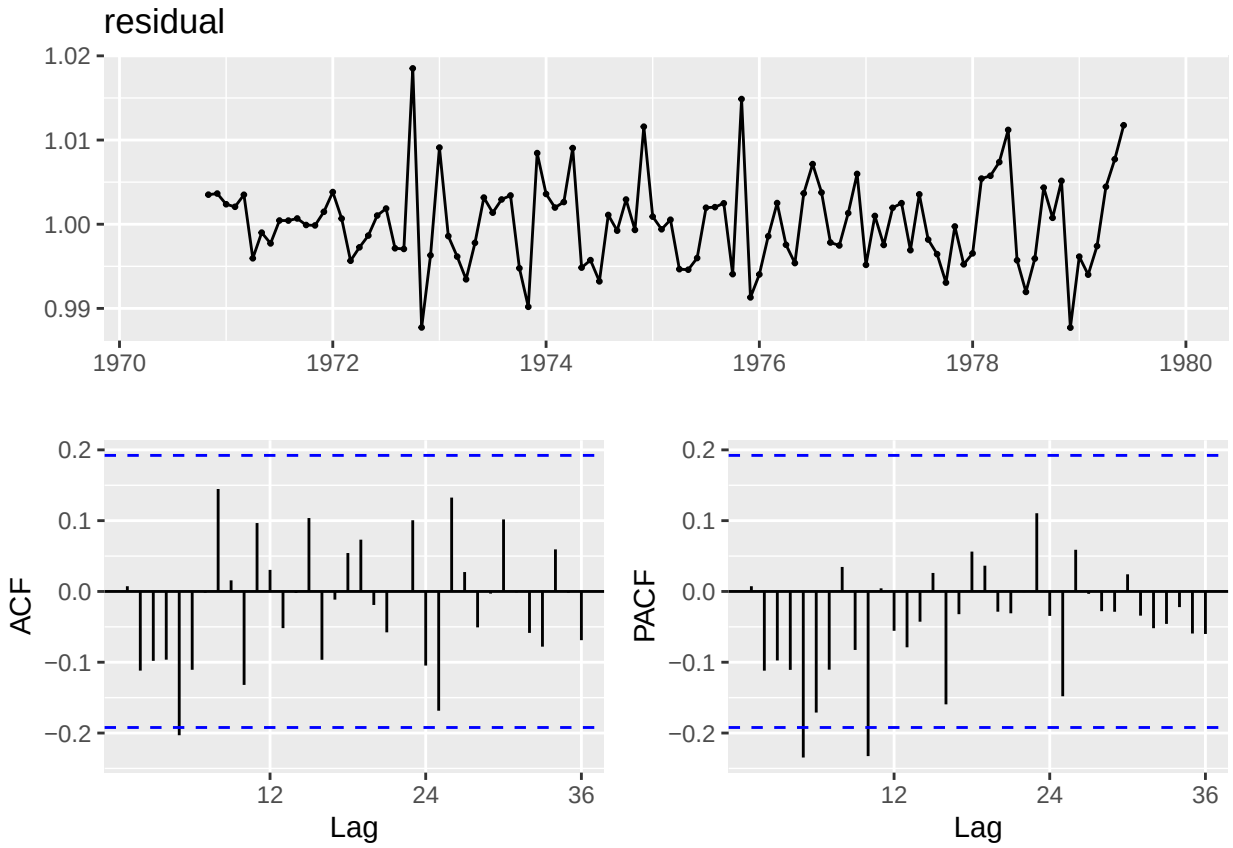
```
# multiplicative
dd2=decompose(data_ts,type="multiplicative")
tseries::kpss.test(dd2$random,null="Level")
```

```
## Warning in tseries::kpss.test(dd2$random, null = "Level"): p-value greater than
## printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  dd2$random
## KPSS Level = 0.025886, Truncation lag parameter = 4, p-value = 0.1
```

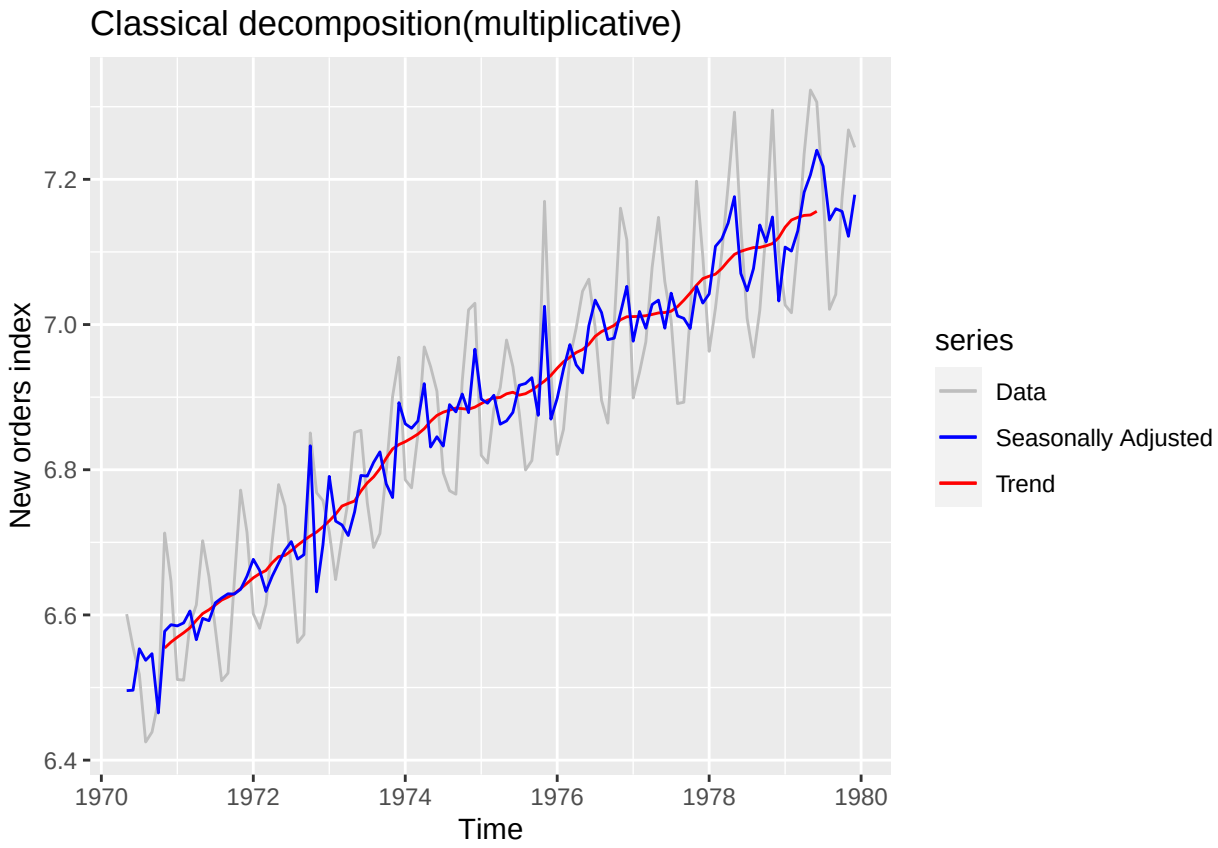
```
ggtsdisplay(dd2$random, main="residual")
```

```
## Warning: Removed 12 rows containing missing values (geom_point).
```




```
autoplot(data_ts, series="Data")+
  autolayer(trendcycle(dd2),series="Trend") +
  autolayer(seasadj(dd2),series="Seasonally Adjusted") +
  xlab("Time") + ylab("New orders index")+
  ggtitle("Classical decomposition(multiplicative)") +
  scale_color_manual(values=c("gray","blue","red"),breaks=c("Data","Seasonally Adjusted","Trend"))
```

```
## Warning: Removed 12 row(s) containing missing values (geom_path).
```



Multiplicative model is used when the seasons change with the trend.

In classical decomposition, the first and last few observations of the trend-cycle factor is removed.

So, Let's apply another method. The following three methods don't make data loss.

Plus, We cannot reject H0 in kpss.test. And the value is usually with in the significance level in the ACF and PACF graph. So, Data is stationary.

SEATS decomposition

```
dd3<-seas(data_ts)
names(dd3)
```

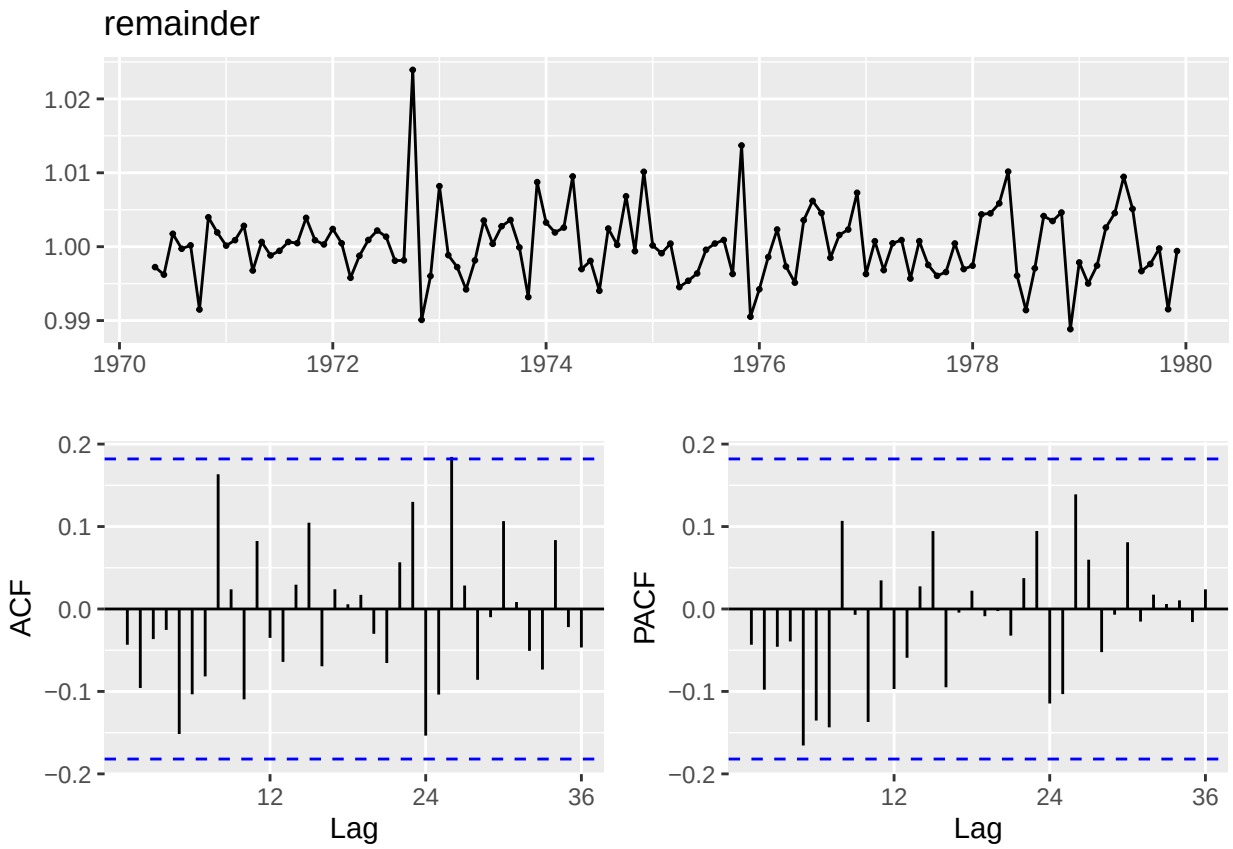
```
##  [1] "series"      "udg"         "data"        "err"         "est"
##  [6] "model"       "fivebestmdl" "wdir"        "iofile"      "call"
## [11] "list"        "x"           "spc"
```

```
dd3_r<-remainder(dd3)
tseries::kpss.test(dd3_r,null="Level")
```

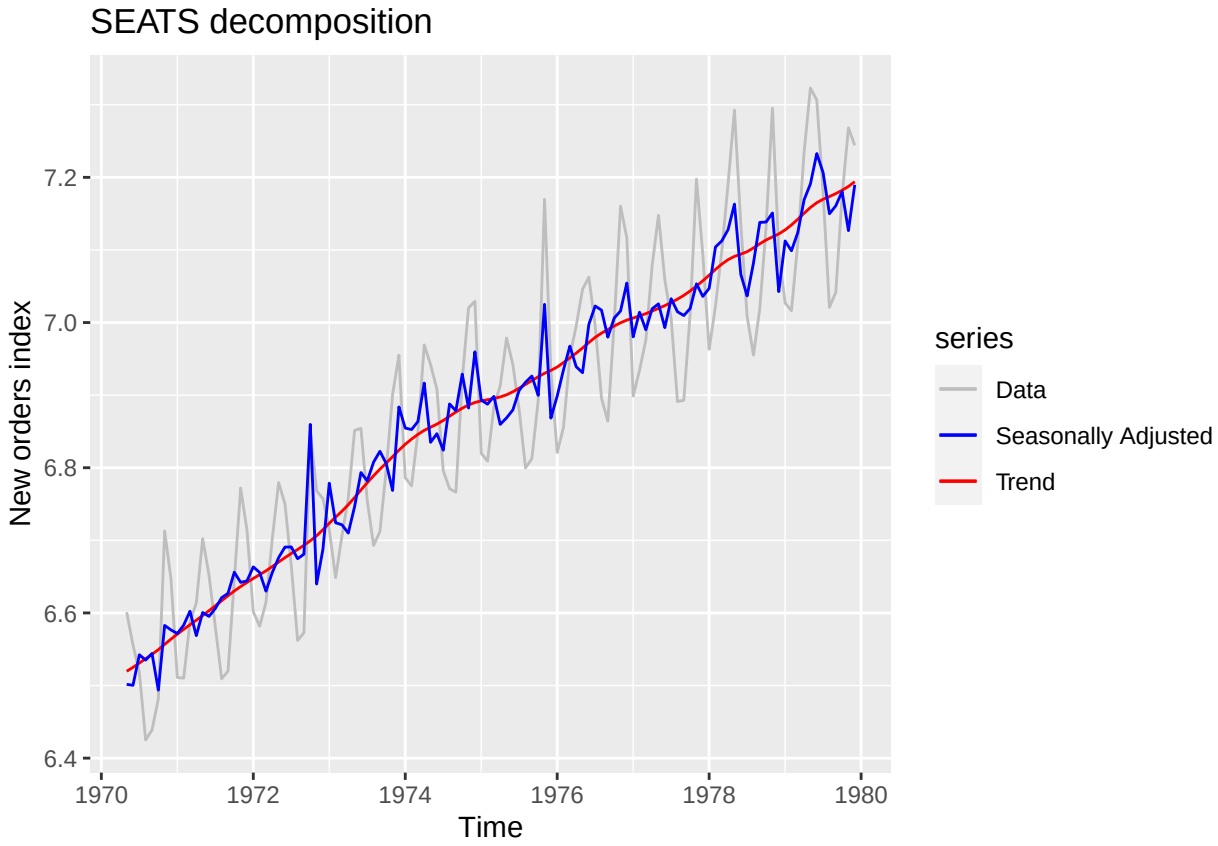
```
## Warning in tseries::kpss.test(dd3_r, null = "Level"): p-value greater than
## printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  dd3_r
## KPSS Level = 0.066507, Truncation lag parameter = 4, p-value = 0.1
```

```
ggtsdisplay(dd3_r, main="remainder")
```




```
autoplot(data_ts, series="Data")+
  autolayer(trendcycle(dd3),series="Trend") +
  autolayer(seasadj(dd3),series="Seasonally Adjusted") +
  xlab("Time") + ylab("New orders index")+
  ggtitle("SEATS decomposition") +
  scale_color_manual(values=c("gray","blue","red"),breaks=c("Data","Seasonally Adjusted","Trend"))
```



The procedure works only with quarterly and monthly data.

There is no problem because the data is monthly.

Plus, We cannot reject H0 in kpss.test. And the value is usually with in the significance level in the ACF and PACF graph. So, Data is stationary.

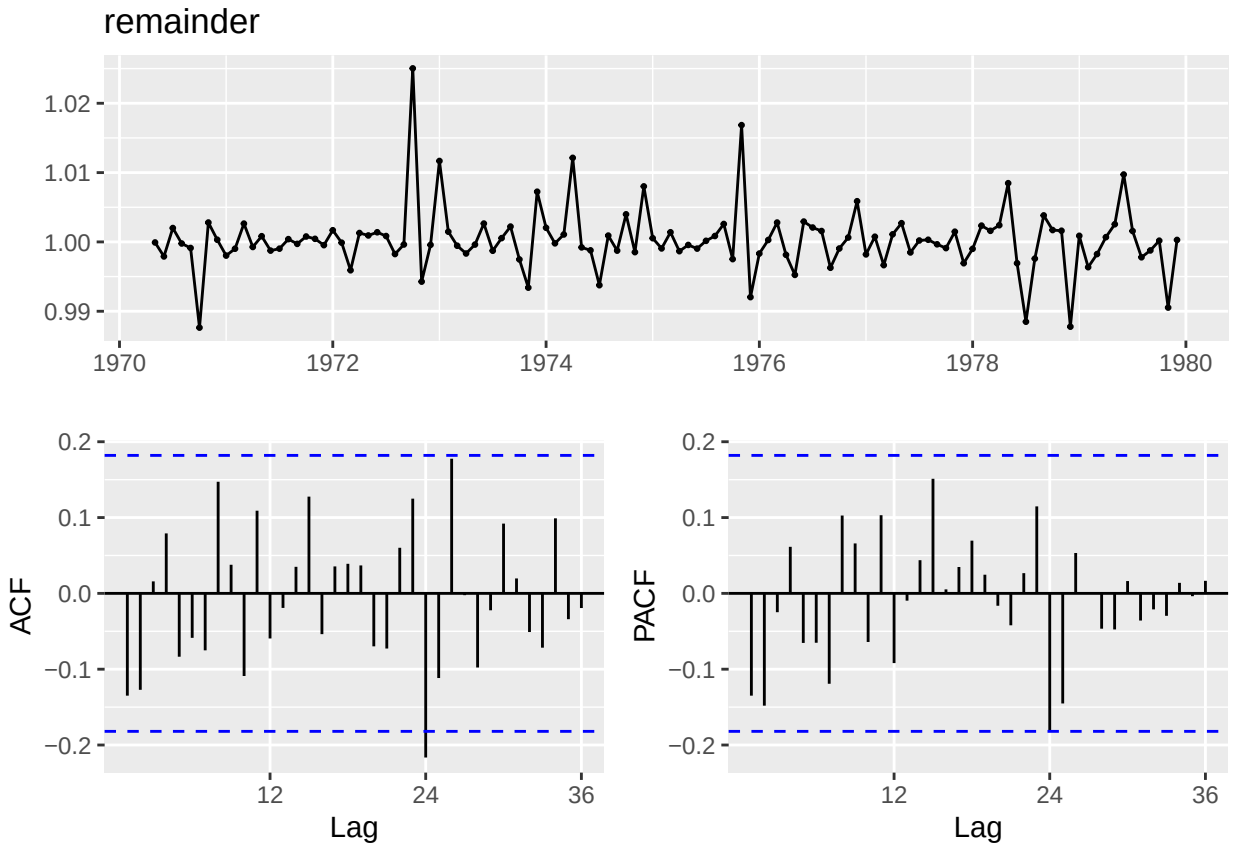
X11 decomposition

```
data_ts %>% seas(x11="") ->dd4
dd4_r=remainder(dd4)
tseries::kpss.test(dd4_r,null="Level")
```

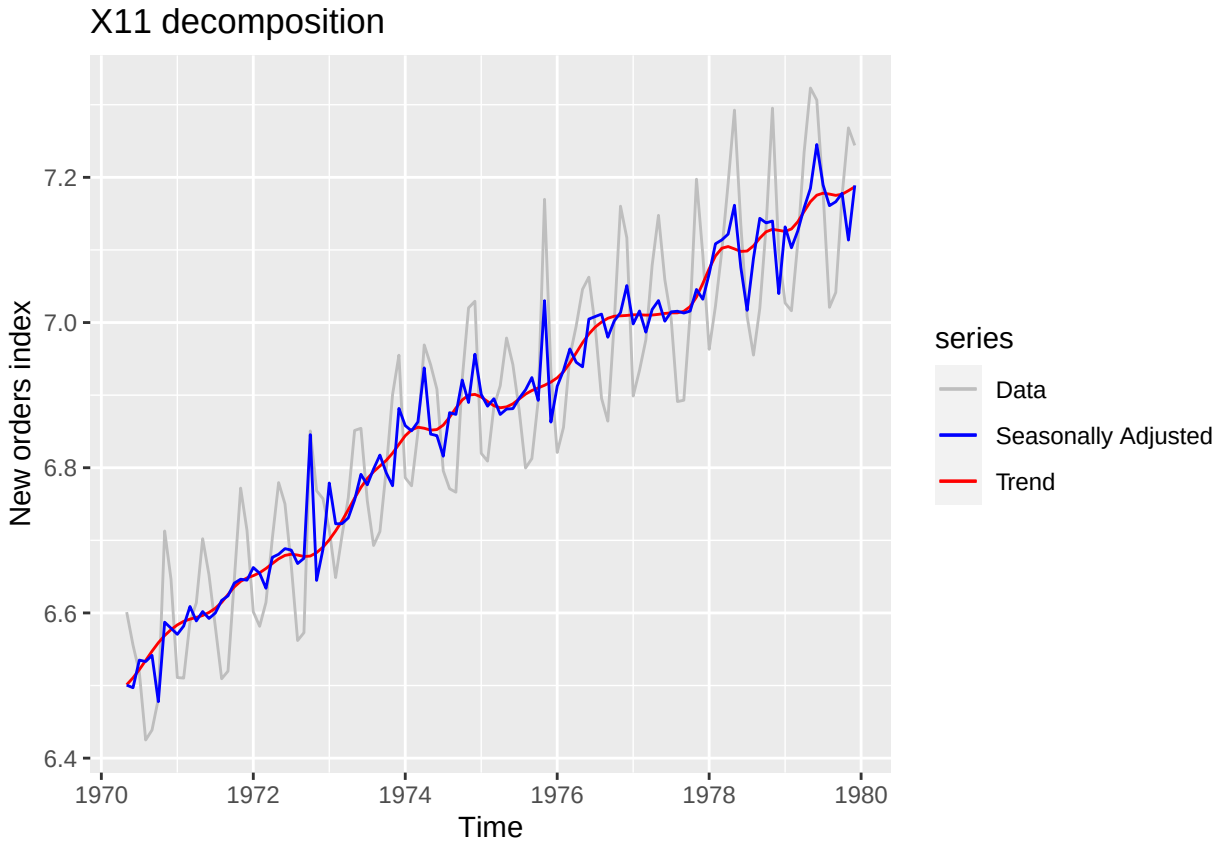
```
## Warning in tseries::kpss.test(dd4_r, null = "Level"): p-value greater than
## printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  dd4_r
## KPSS Level = 0.16255, Truncation lag parameter = 4, p-value = 0.1
```

```
ggtsdisplay(dd4_r, main="remainder")
```




```
autoplot(data_ts, series="Data")+
  autolayer(trendcycle(dd4),series="Trend") +
  autolayer(seasadj(dd4),series="Seasonally Adjusted") +
  xlab("Time") + ylab("New orders index")+
  ggtitle("X11 decomposition") +
  scale_color_manual(values=c("gray","blue","red"),breaks=c("Data","Seasonally Adjusted","Trend"))
```



Trend-cycle estimates are also available for all observations including the end points.

And the seasonal component is allowed to vary slowly over time.

So, that line isn't flatter than other procedure.

Plus, We cannot reject H0 in kpss.test. And the value is usually with in the significance level in the ACF and PACF graph. So, Data is stationary.

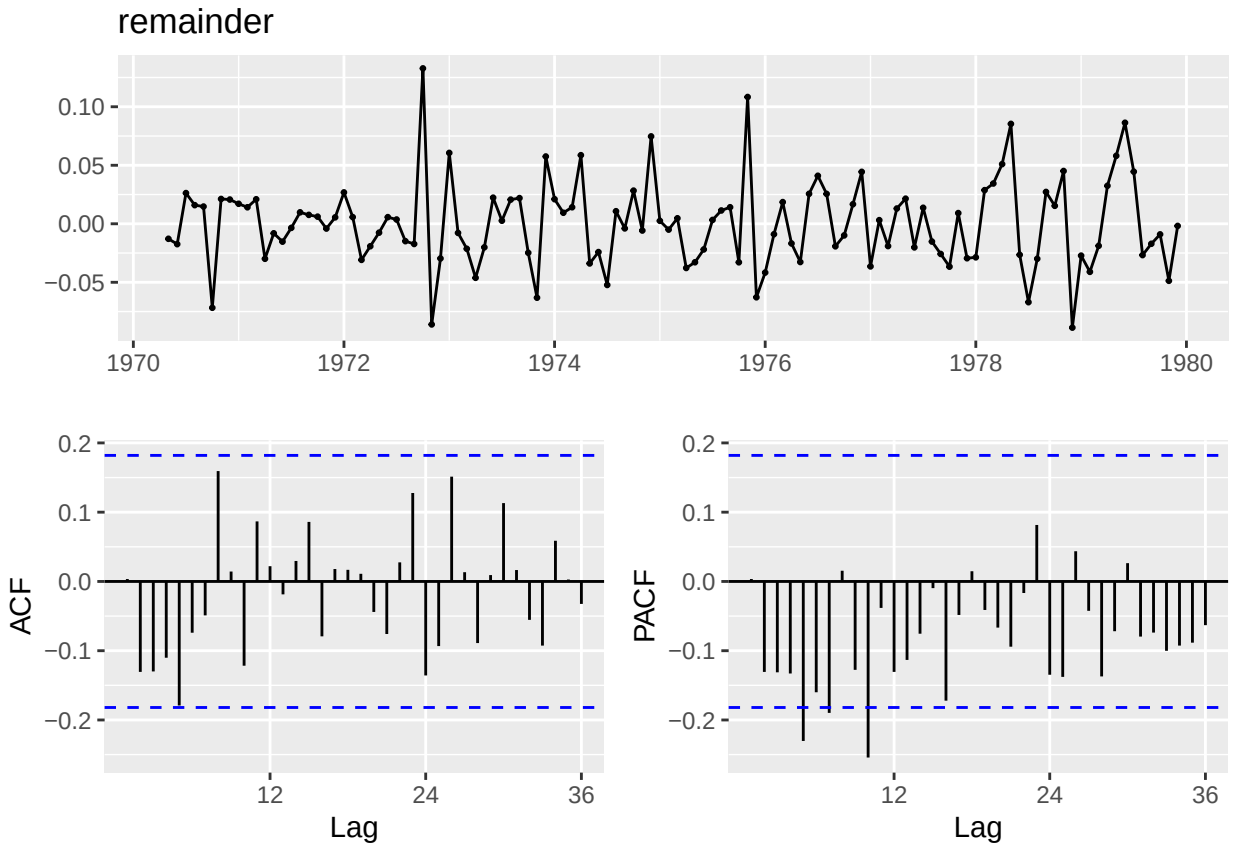
STL decomposition

```
dd5<-stl(data_ts,'periodic')
dd5_r=remainder(dd5)
tseries::kpss.test(dd5_r,null="Level")
```

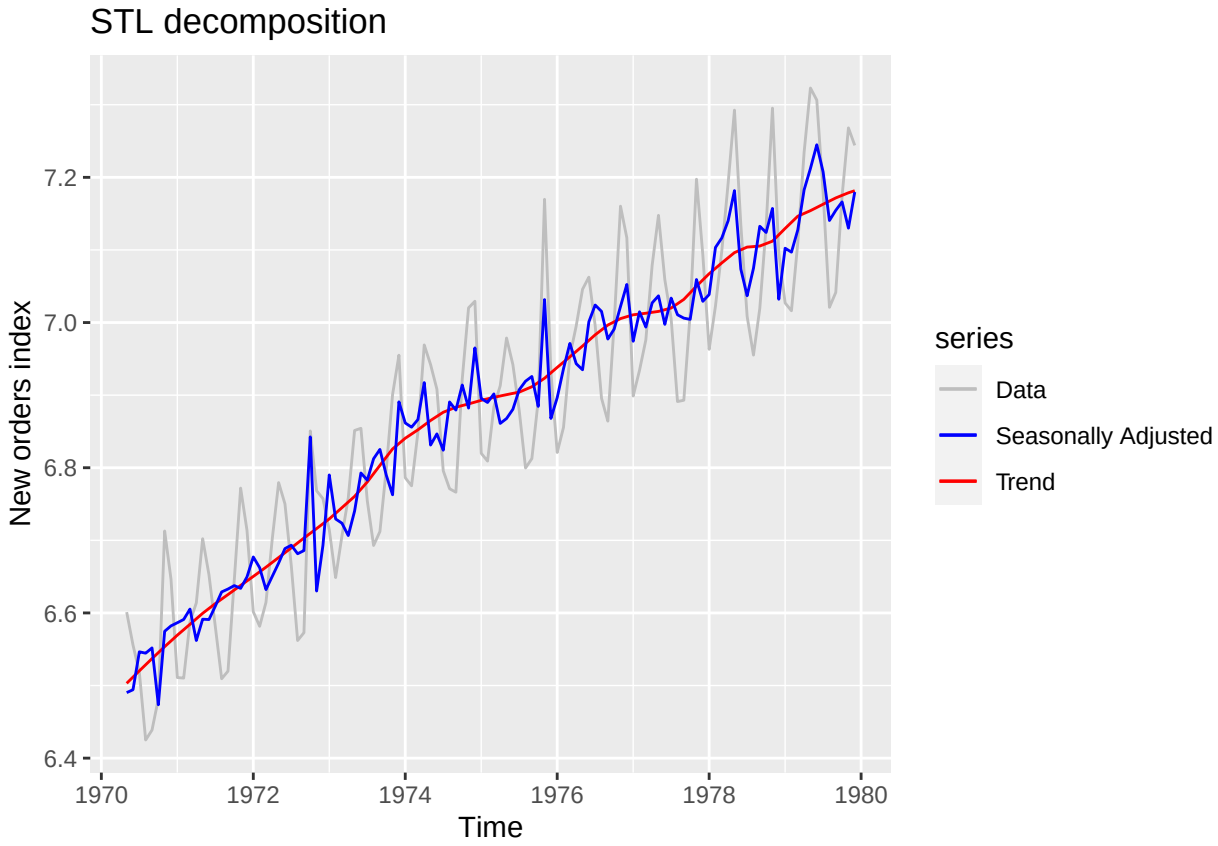
```
## Warning in tseries::kpss.test(dd5_r, null = "Level"): p-value greater than
## printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  dd5_r
## KPSS Level = 0.016203, Truncation lag parameter = 4, p-value = 0.1
```

```
ggtsdisplay(dd5_r, main="remainder")
```




```
autoplot(data_ts, series="Data")+
  autolayer(trendcycle(dd5),series="Trend") +
  autolayer(seasadj(dd5),series="Seasonally Adjusted") +
  xlab("Time") + ylab("New orders index")+
  ggtitle("STL decomposition") +
  scale_color_manual(values=c("gray","blue","red"),breaks=c("Data","Seasonally Adjusted","Trend"))
```



STL will handle any type of seasonality, not only monthly and quarterly data.

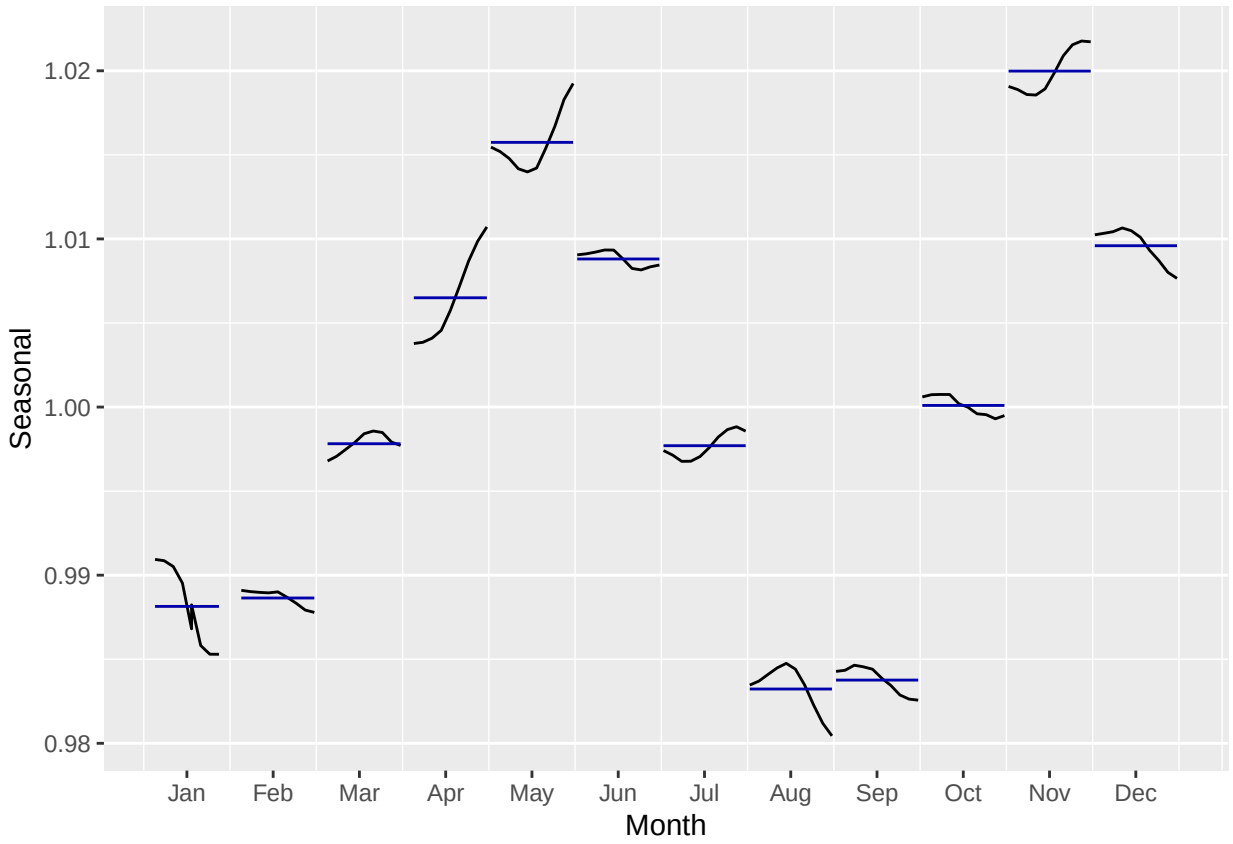
But, it does not handle trading day or calendar variation automatically, and it only provides facilities for additive decompositions.

Plus, We cannot reject H0 in kpss.test. And the value is usually with in the significance level in the ACF and PACF graph. So, Data is stationary.

Therefore, X11 seems most appropriate method considering the shape and stationary of the data.

3. Draw sub-series seasonal effect graph in each month.

```
dd4 %>% seasonal() %>% ggsubseriesplot()+ylab("Seasonal")
```

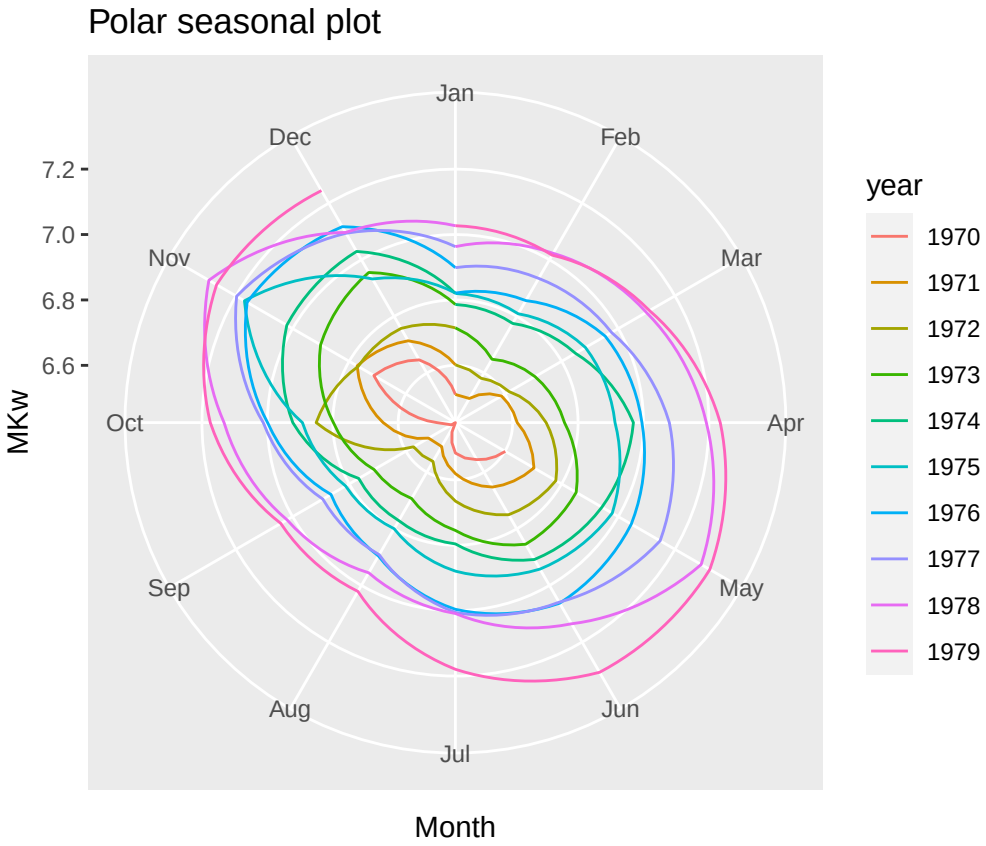


This graph shows shape of seasonal effect of each month (Blue line is constant seasonal effect of each month)

The monthly peak load for electricity Iowa is high around May and November. And it is low around the end of January and the end of January

4. Draw a polar seasonal plot.

```
ggseasonplot(data_ts, polar=TRUE) +
  ylab("MKw") +
  ggtitle("Polar seasonal plot")
```



The graph show not only just the same results at the graph of number three but also an increase over the years.