Data 524 Assignment 2

Warner Alexis

2025-02-16

The Forcaster Toolbox

Excercise 3.1

For the following series, find an appropriate Box-Cox transformation in order to stabilise the variance.

> usnetelec usgdp mcopper enplanements

```
## Registered S3 method overwritten by 'tsibble':
##   method               from
##   as_tibble.grouped_df dplyr

##
## Attaching package: 'tsibble'

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

## -- Attaching packages -------------------------------------------- fpp3 1.0.1 --

## v tibble    3.2.1     v ggplot2   3.5.1
## v dplyr     1.1.4     v feasts    0.4.1
## v tidyr     1.3.1     v fable     0.4.1
## v lubridate 1.9.3

## -- Conflicts ------------------------------------------------- fpp3_conflicts --
## x lubridate::date()     masks base::date()
## x dplyr::filter()       masks stats::filter()
## x tsibble::intersect()  masks base::intersect()
## x lubridate::interval() masks tsibble::interval()
## x dplyr::lag()          masks stats::lag()
## x tsibble::setdiff()    masks base::setdiff()
## x tsibble::union()      masks base::union()

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

## -- Attaching packages ---------------------------------------------- fpp2 2.5 --
## v forecast  8.23.0     v expsmooth 2.3
## v fma       2.5
```

```
##
```

```
##
## Attaching package: 'fpp2'
```

```
## The following object is masked from 'package:fpp3':
##
##     insurance
```

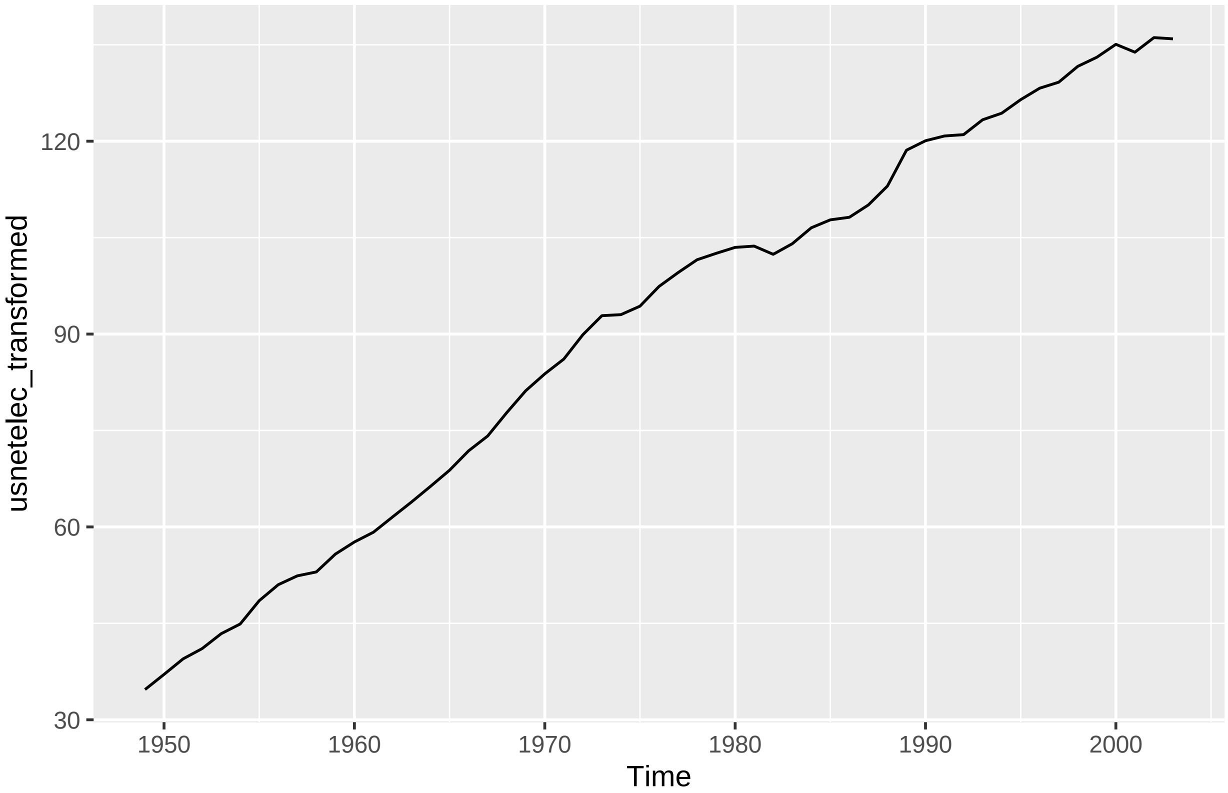
Let create a function that read Box

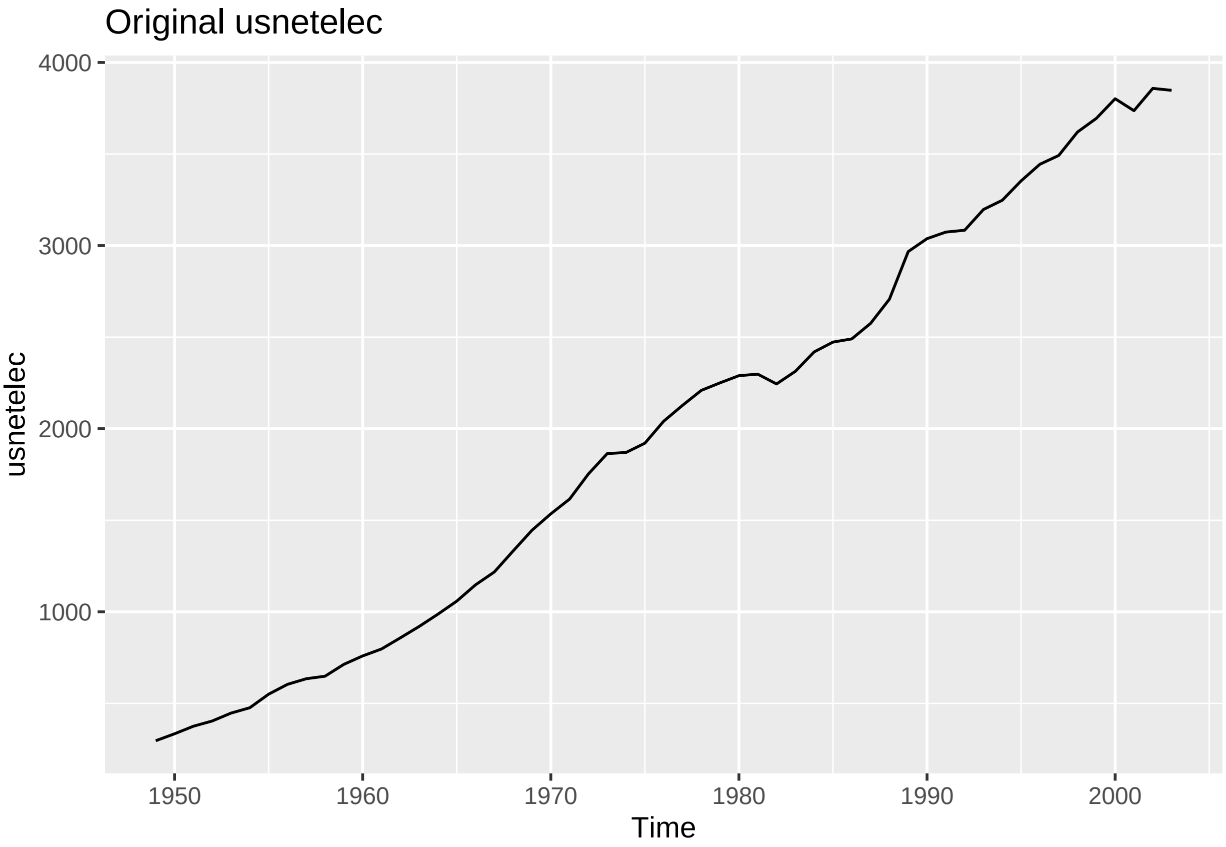
```
## [1] "Optimal Lambda for series: 0.516771443964645"
```

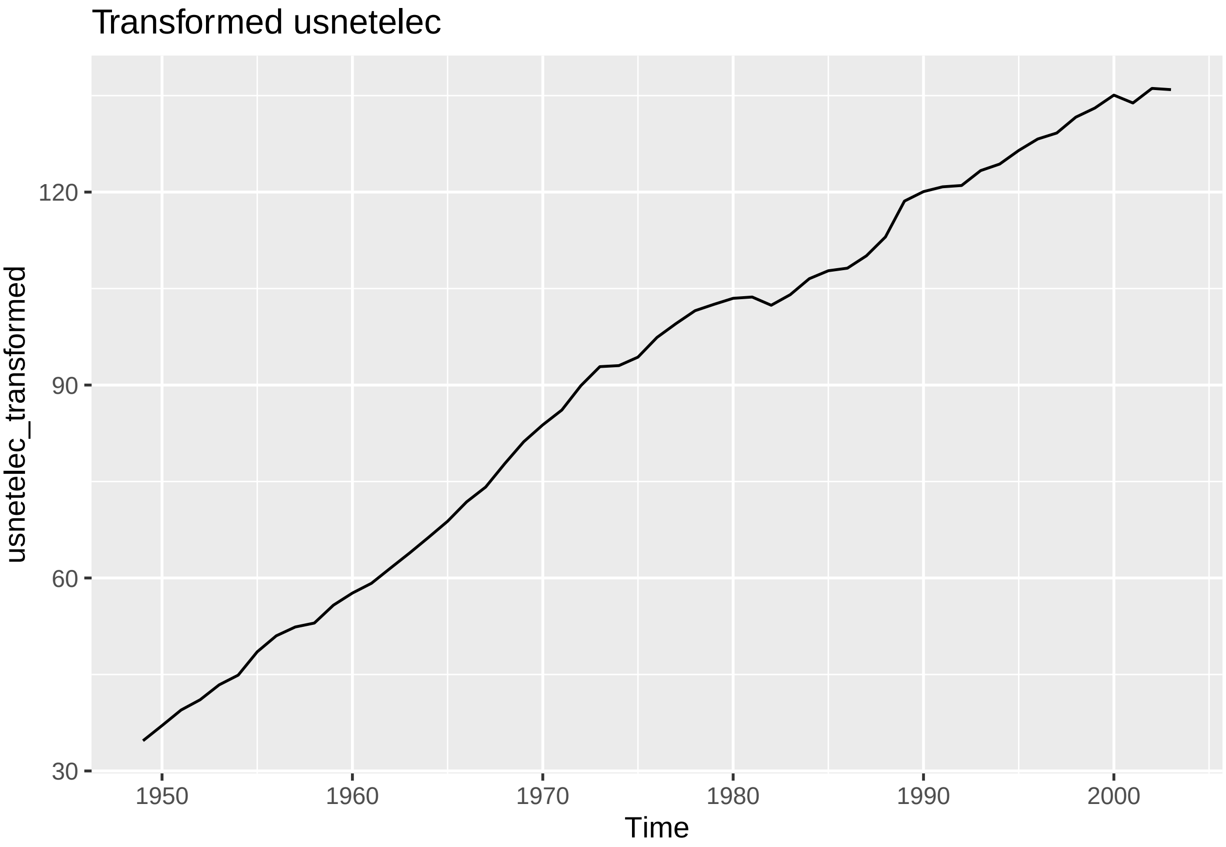
```
## [1] "Optimal Lambda for series: 0.366352049520934"
```

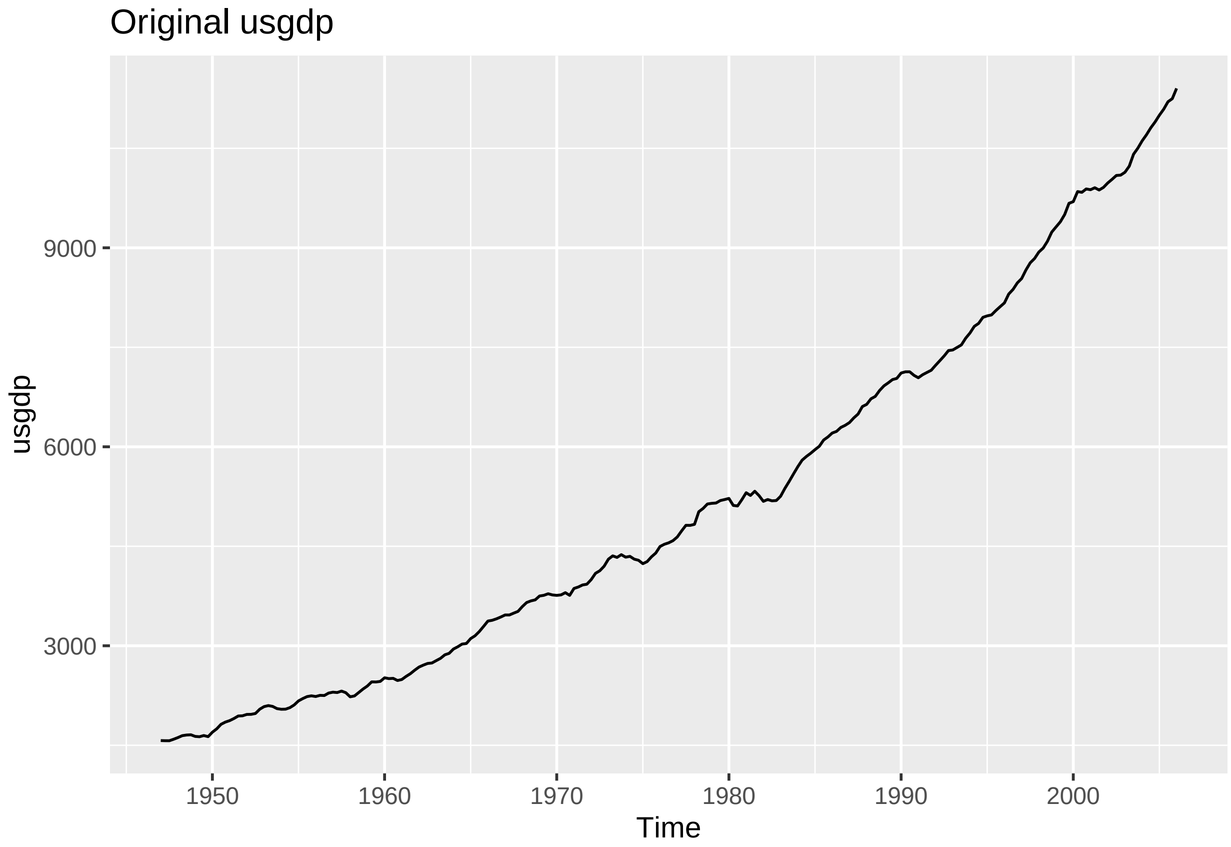
```
## [1] "Optimal Lambda for series: 0.191904709003829"
```

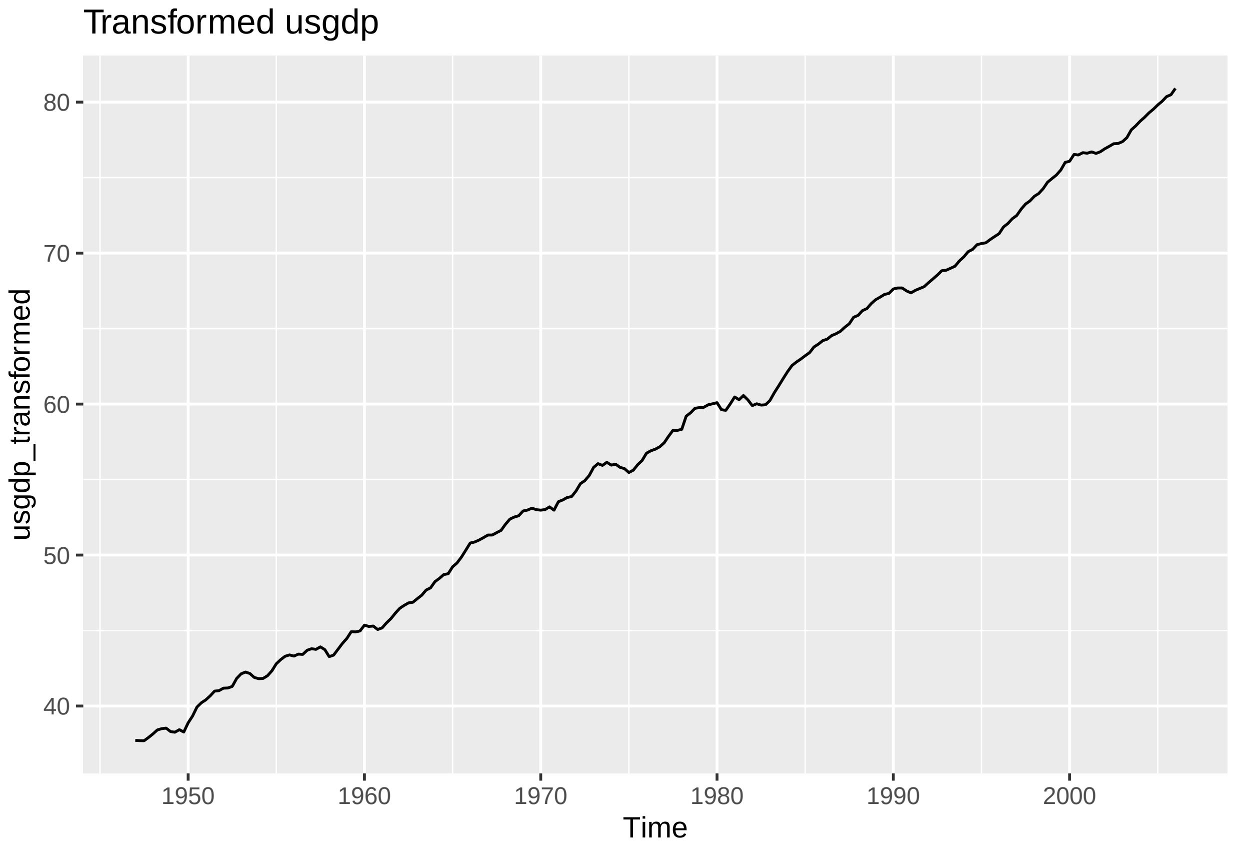
```
## [1] "Optimal Lambda for series: -0.226946111237065"
```

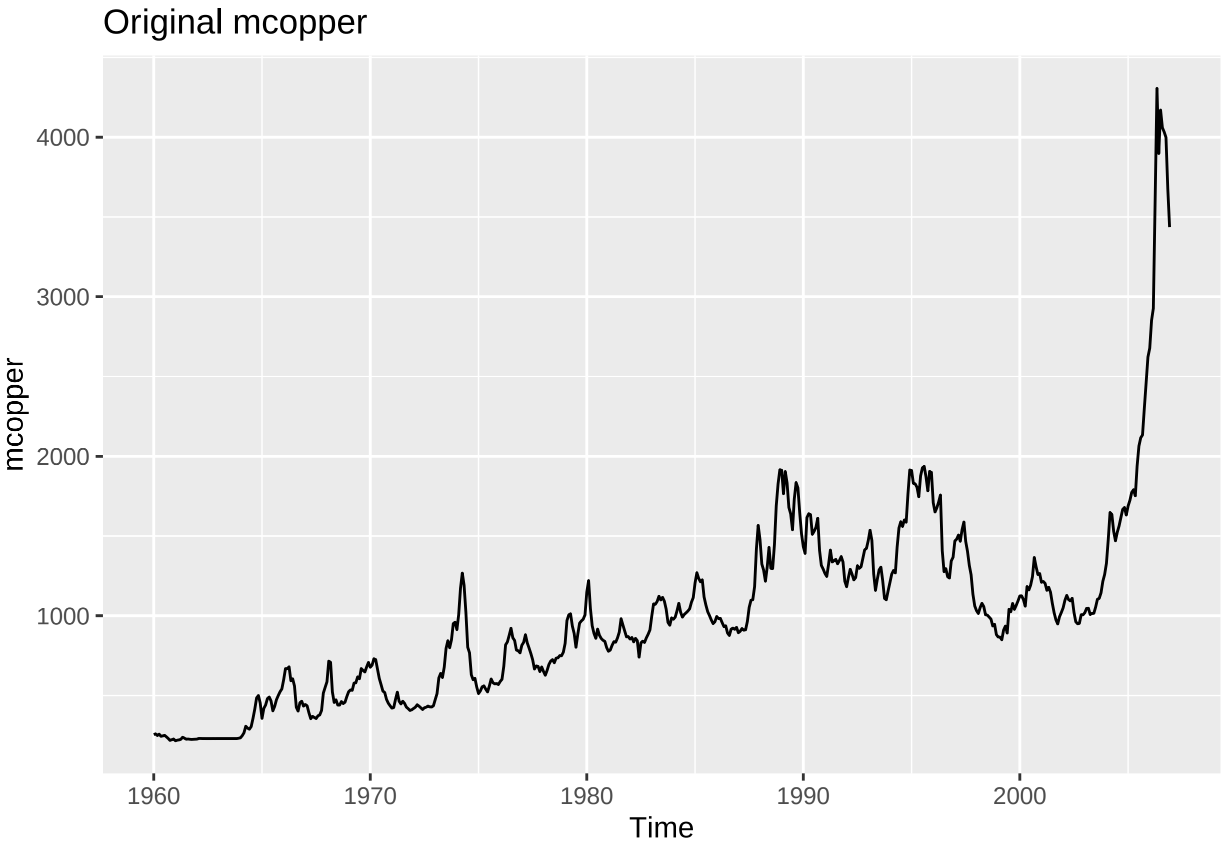


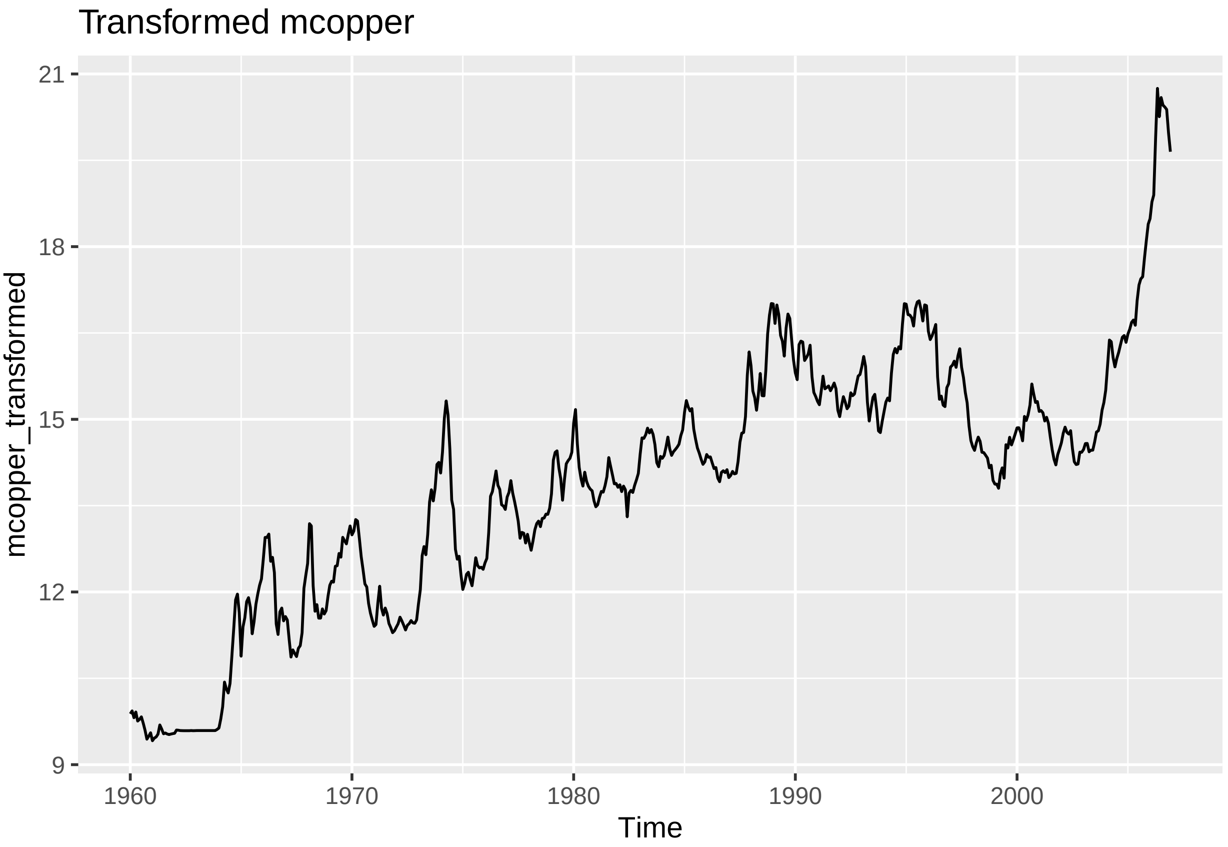

Original usnetelec
4000
3000
2000
1000
usnetelec
1950
1960
1970
1980
1990
2000
Time

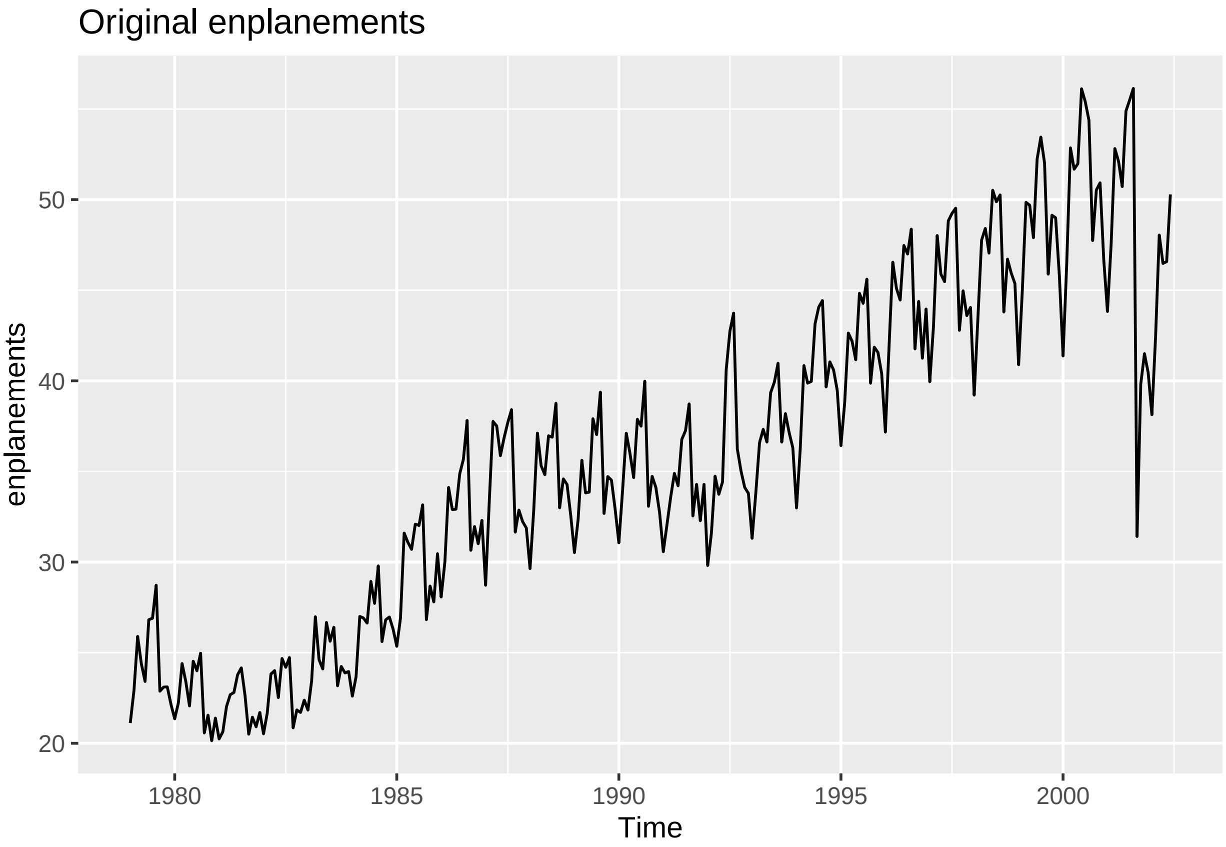

Transformed usnetelec
usnetelec_transformed
120
90
60
30
1950
1960
1970
1980
1990
2000
Time

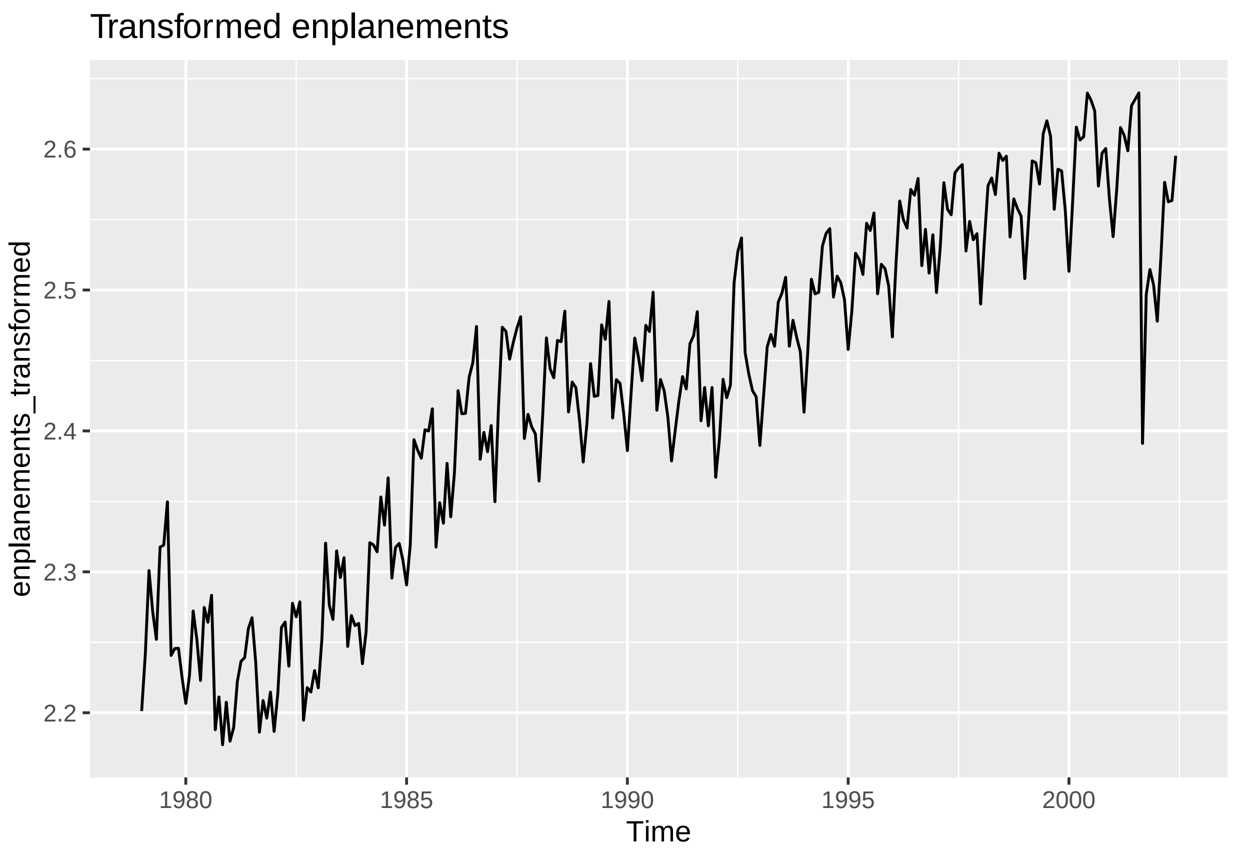

Original usgdp
usgdp
9000
6000
3000
1950
1960
1970
1980
1990
2000
Time


Transformed usgdp
usgdp_transformed
80
70
60
50
40
1950
1960
1970
1980
1990
2000
Time


Original mcopper
mcopper
4000
3000
2000
1000
1960
1970
1980
1990
2000
Time


Transformed mcopper
mcopper_transformed
21
18
15
12
9
1960
1970
1980
1990
2000
Time


Original enplanements
enplanements
50
40
30
20
1980
1985
1990
1995
2000
Time



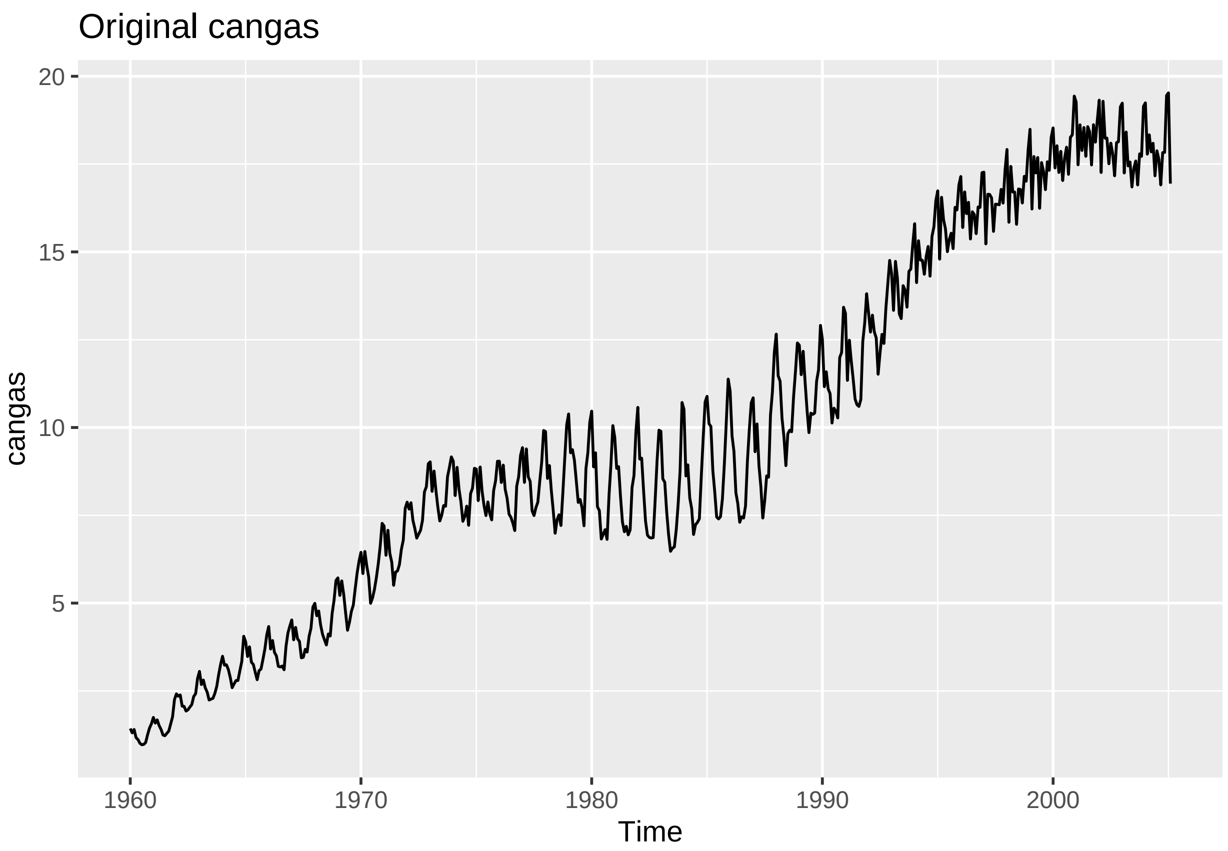
The original time series graphs exhibit non-stationary behavior, with increasing variance and trends over time, especially in **usnetelec**, **usgdp**, and **mcopper**, which show exponential-like growth. The **Box-Cox transformed** graphs demonstrate more stabilized variance and a more linear trend, making them more suitable for time series modeling. In particular, **usnetelec** and **usgdp** show significant variance reduction post-transformation. The **mcopper** and **enplanements** series, which have more irregular fluctuations, also exhibit a more stabilized pattern after transformation, though seasonal patterns remain visible in **enplanements**. Overall, the transformation effectively mitigates heteroscedasticity, improving the series for forecasting purposes.

Excercise 3.2 Why is a Box-Cox transformation unhelpful for the cangas data?

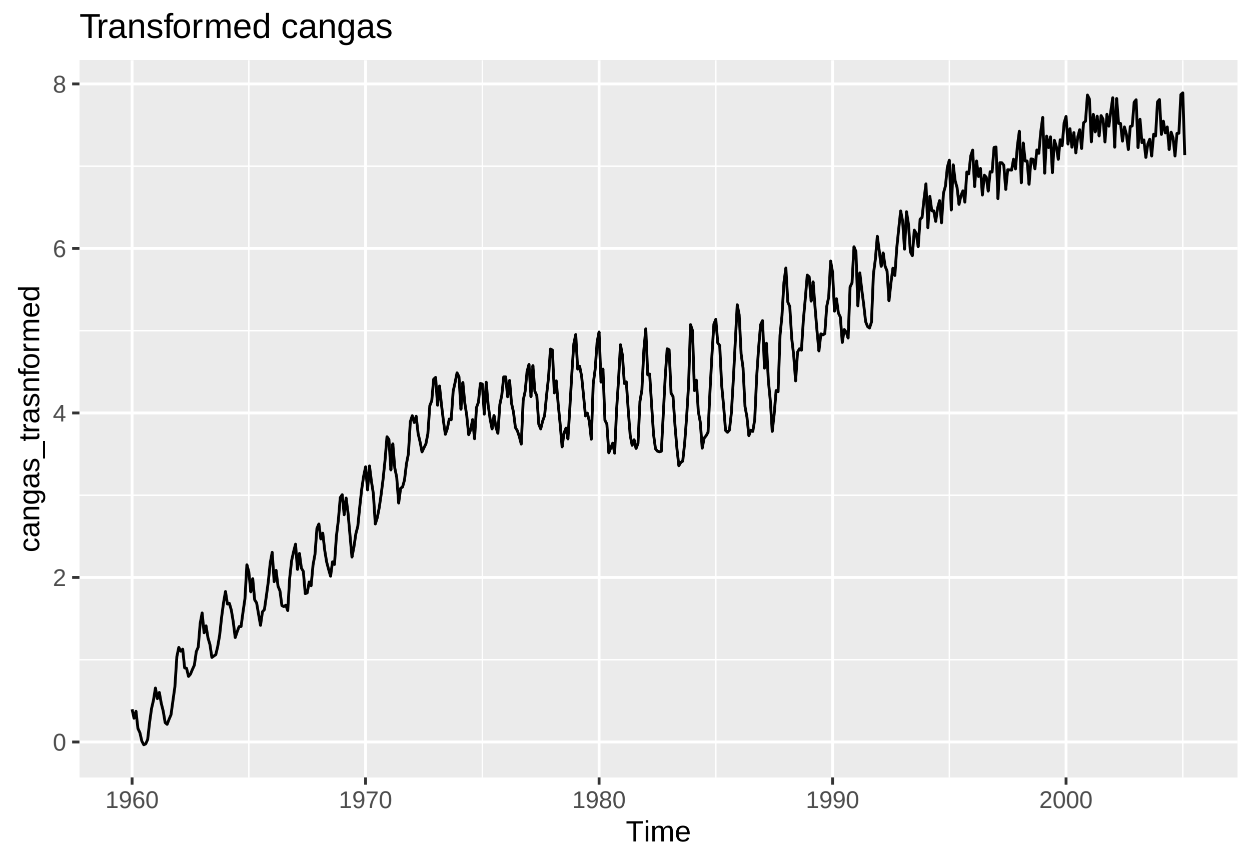
```
# Find lamba for cangas
lambda_cangas <- find_lambda(cangas)
```

```
## [1] "Optimal Lambda for series: 0.576775938228139"
```

```
# Apply Box-Cox transformation to canga series
cangas_trasnformed <- BoxCox(cangas, lambda_cangas)
autoplot(cangas) + ggtitle("Original cangas")
```



```r
autoplot(cangas_trasnformed) + ggtitle("Transformed cangas")
```



The Box-Cox transformation is unhelpful for the **cangas** data because it does not effectively address the key challenges in the series. While the transformation is designed to stabilize variance, the **cangas** data primarily exhibits a strong **upward trend** and pronounced **seasonality**, neither of which are mitigated by the transformation. The seasonal fluctuations remain just as prominent, indicating that **seasonal differencing** would be a more suitable approach. Additionally, while there is some variation in amplitude, the variance instability is not severe enough to warrant a Box-Cox transformation. Instead, methods like **log transformation** or **differencing** (both regular and seasonal) would likely be more effective in making the series stationary and suitable for forecasting.

Exercise 3.3 What Box-Cox transformation would you select for your retail data (from Exercise 3 in Section 2.10)?

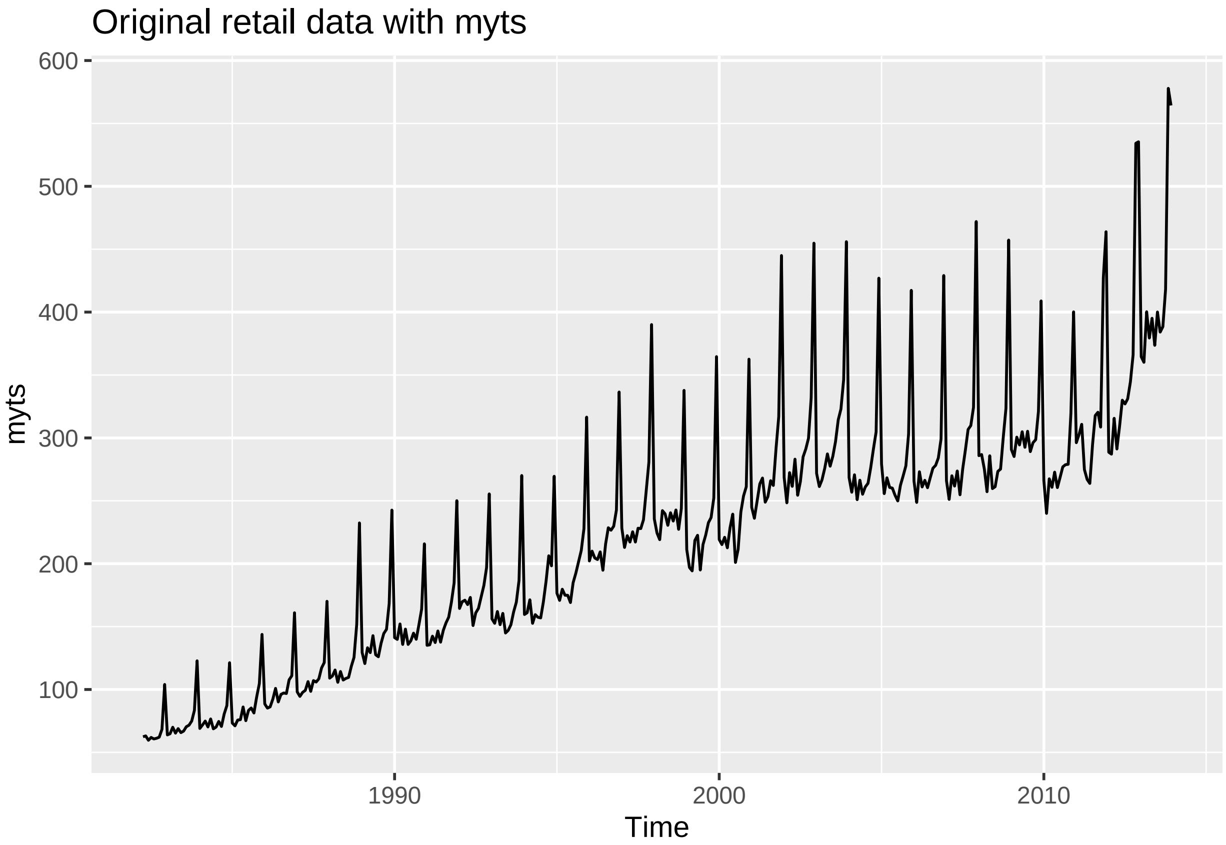
```
library(readxl)
library(httr)
library(openxlsx)
url <- 'https://raw.githubusercontent.com/joewarner89/Data-624-Predictive-Anaytics/main/workspace/retail

temp_file <- tempfile(fileext = ".xlsx")  # Create a temporary file

download.file(url, temp_file, mode = "wb")  # Download
retail <- read_excel(temp_file, skip = 1)  # Read the Excel file

head(retail)
```

```
## # A tibble: 6 x 190
##   ‘Series ID‘         A3349335T A3349627V A3349338X A3349398A A3349468W
##   <dttm>                      <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1 1982-04-01 00:00:00          303.      41.7      63.9      409.      65.8
## 2 1982-05-01 00:00:00          298.      43.1      64        405.      65.8
## 3 1982-06-01 00:00:00          298       40.3      62.7      401       62.3
## 4 1982-07-01 00:00:00          308.      40.9      65.6      414.      68.2
## 5 1982-08-01 00:00:00          299.      42.1      62.6      404.      66
## 6 1982-09-01 00:00:00          305.      42        64.4      412.      62.3
## # i 184 more variables: A3349336V <dbl>, A3349337W <dbl>, A3349397X <dbl>,
## #   A3349399C <dbl>, A3349874C <dbl>, A3349871W <dbl>, A3349790V <dbl>,
## #   A3349556W <dbl>, A3349791W <dbl>, A3349401C <dbl>, A3349873A <dbl>,
## #   A3349872X <dbl>, A3349709X <dbl>, A3349792X <dbl>, A3349789K <dbl>,
## #   A3349555V <dbl>, A3349565X <dbl>, A3349414R <dbl>, A3349799R <dbl>,
## #   A3349642T <dbl>, A3349413L <dbl>, A3349564W <dbl>, A3349416V <dbl>,
## #   A3349643V <dbl>, A3349483V <dbl>, A3349722T <dbl>, A3349727C <dbl>, ...
```

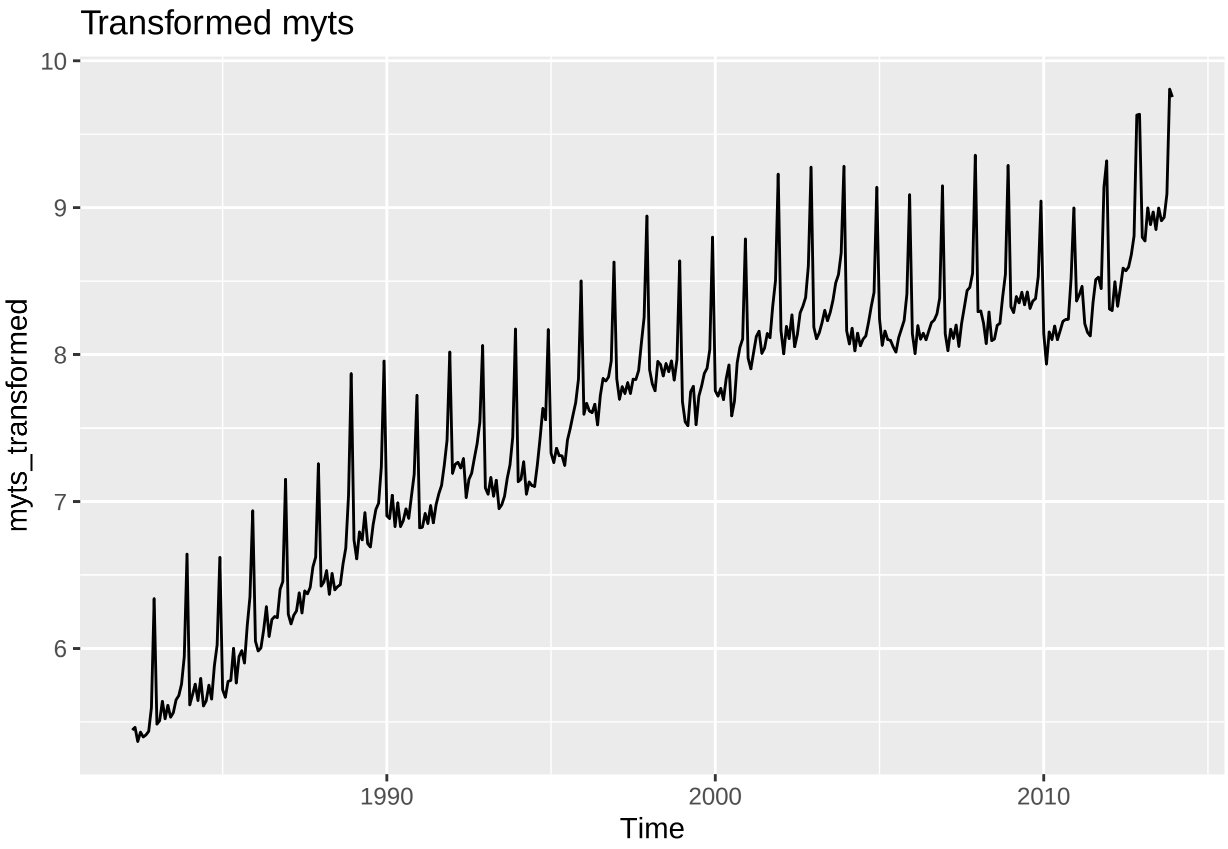
```
myts <- ts(retail[,"A3349873A"],
           frequency=12, start=c(1982,4))

# Find lambda for each series
lambda_myts <- find_lambda(myts)
```

```
## [1] "Optimal Lambda for series: 0.127636859661548"
```

```
myts_transformed <- BoxCox(myts,lambda_myts)

autoplot(myts) + ggtitle("Original retail data with myts")
```

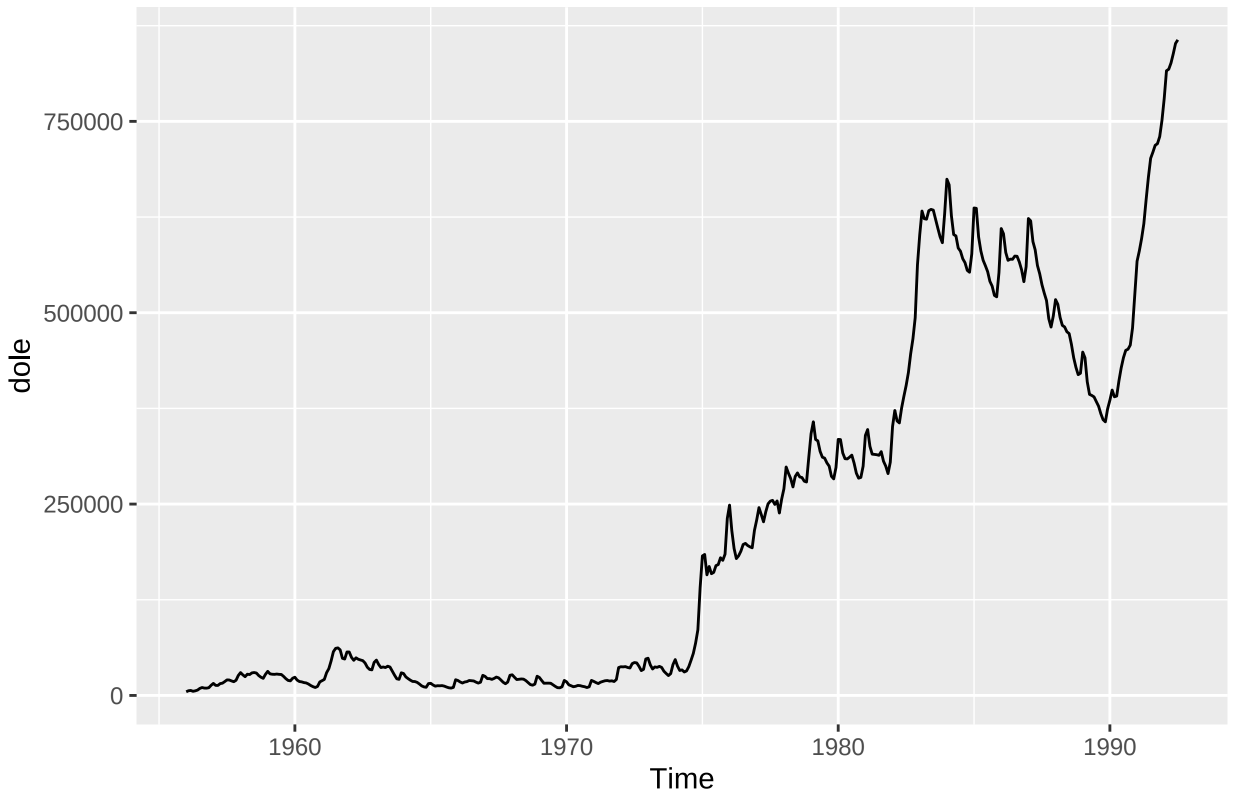


```
autoplot(myts_transformed) + ggtitle("Transformed myts")
```

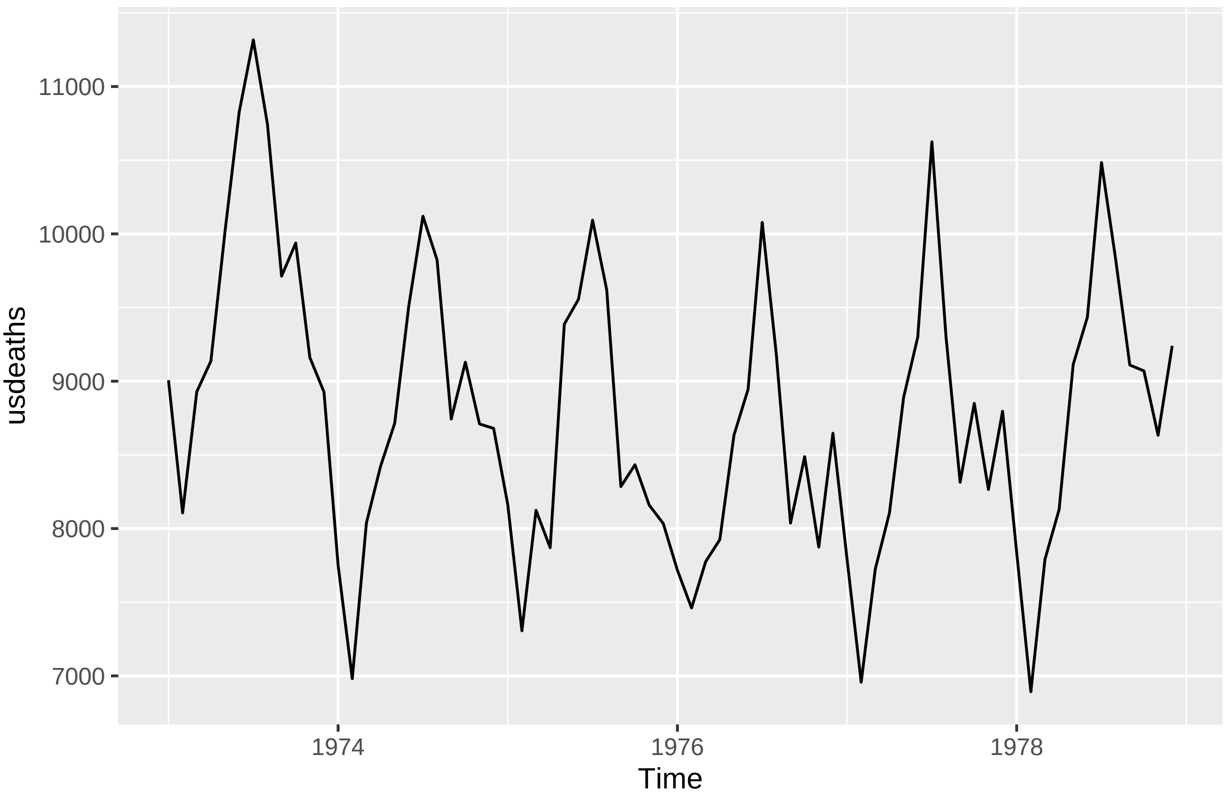


Exercise 3.4 For each of the following series, make a graph of the data. If transforming seems appropriate, do so and describe the effect. dole, usdeaths, bricksq.

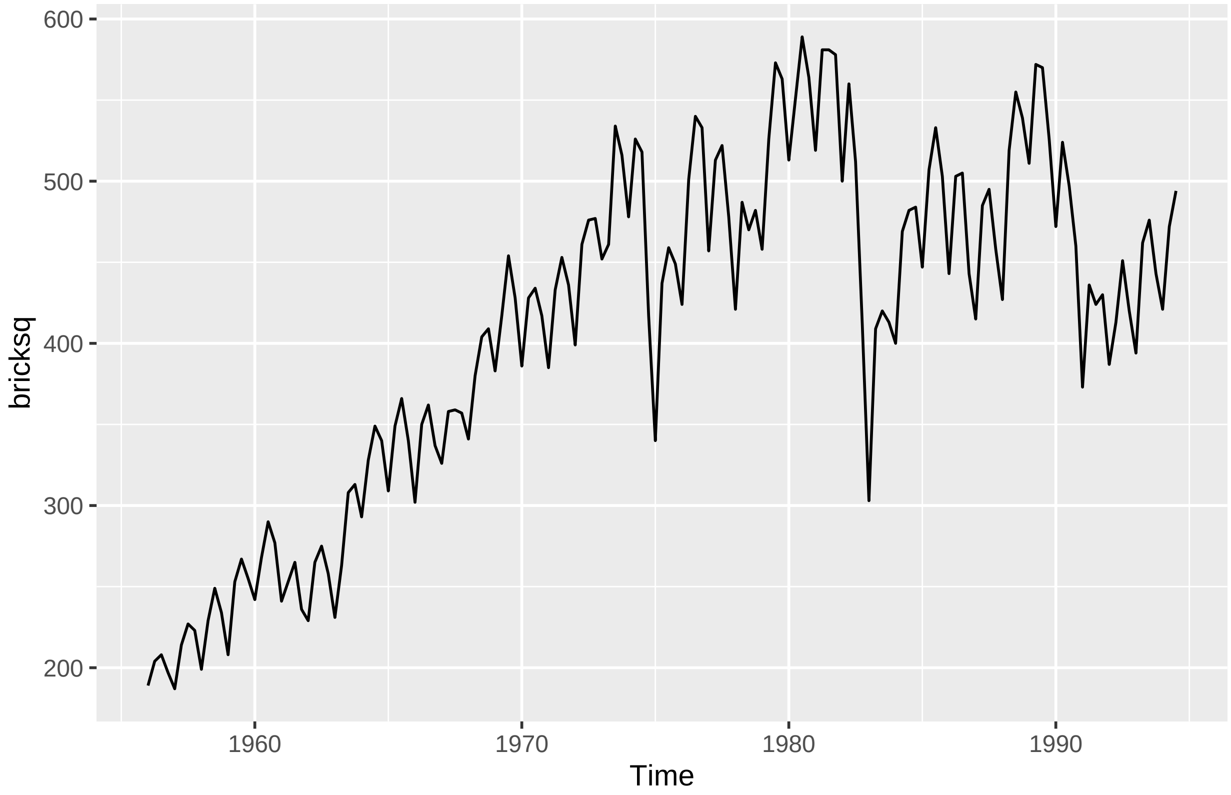
```
autoplot(dole)
```



```r
autoplot(usdeaths)
```



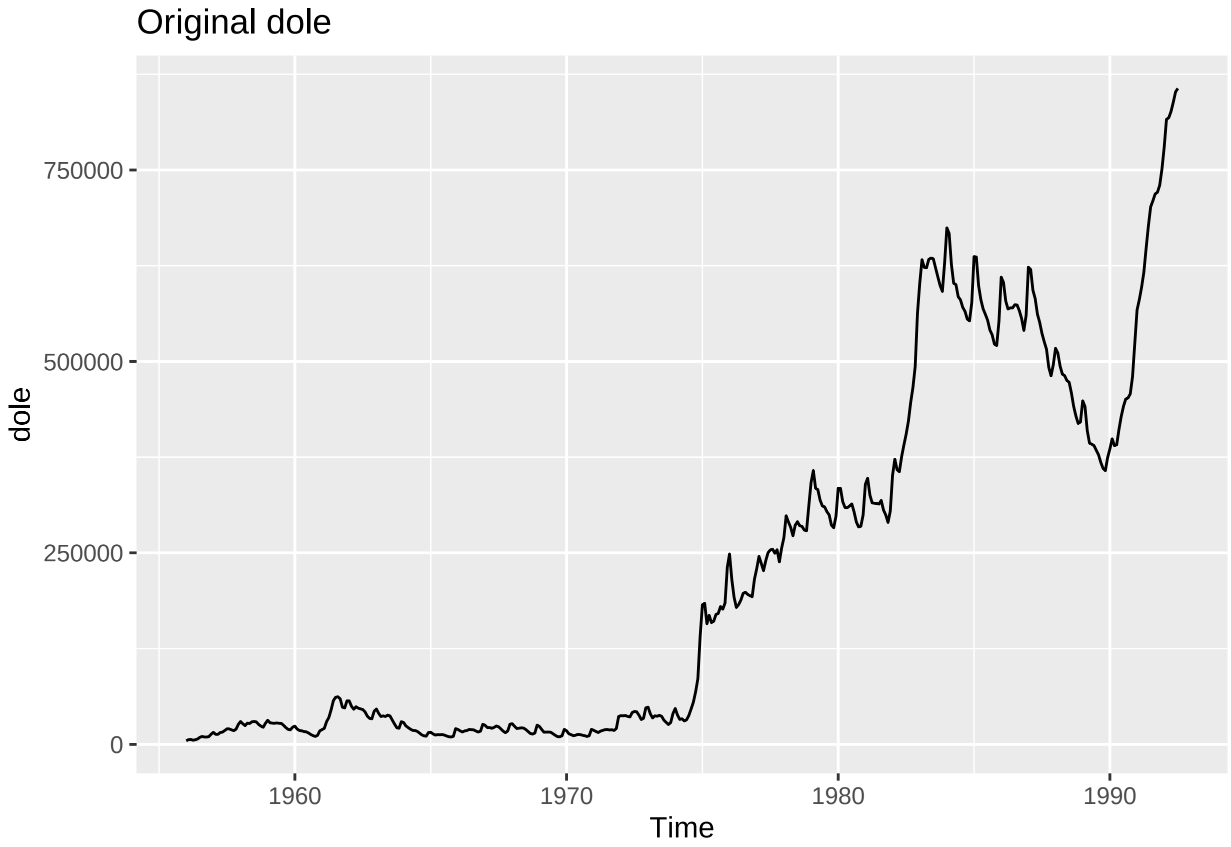
```
autoplot(bricksq)
```



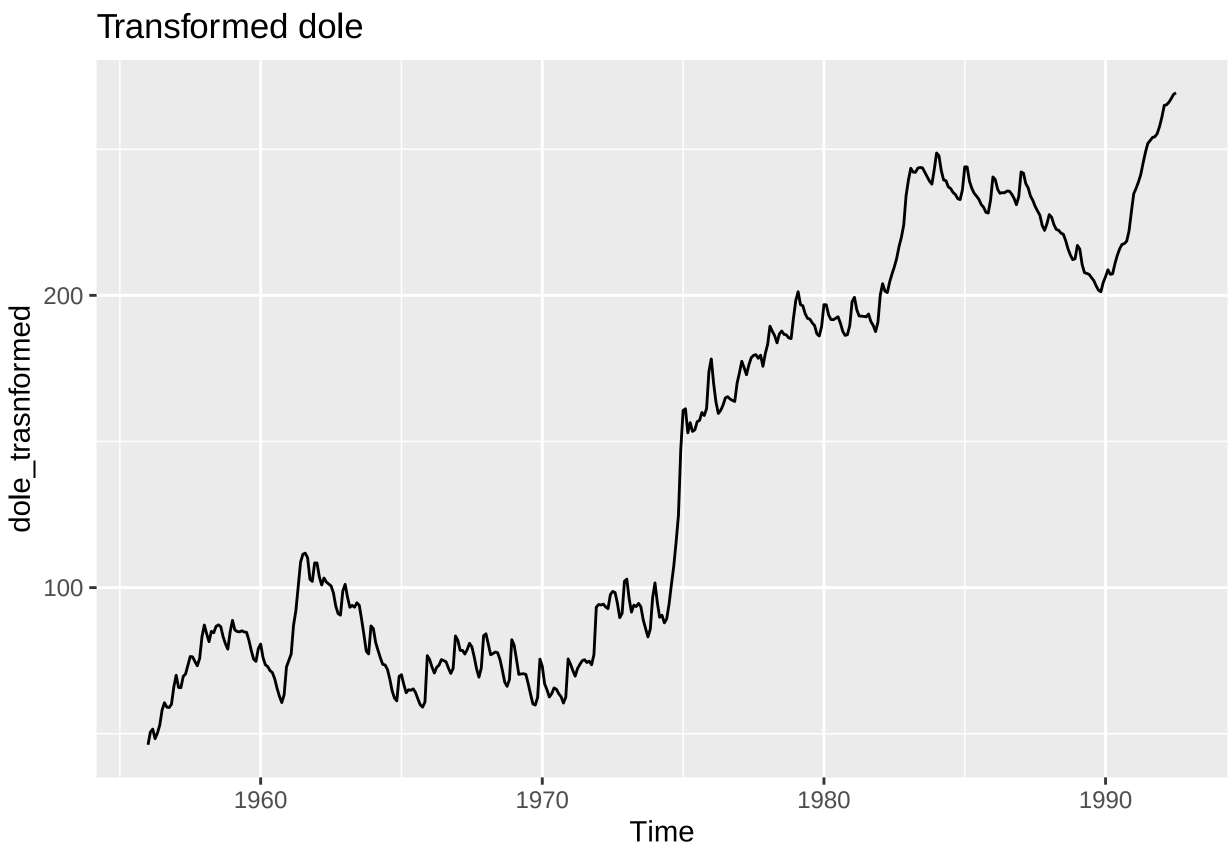
```
# Find lamba for dole
lambda_dole <- find_lambda(dole)
```

```
## [1] "Optimal Lambda for series: 0.329092227570859"
```

```
# Apply Box-Cox transformation to  series
dole_trasnformed <- BoxCox(dole, lambda_dole)
autoplot(dole) + ggtitle("Original dole")
```



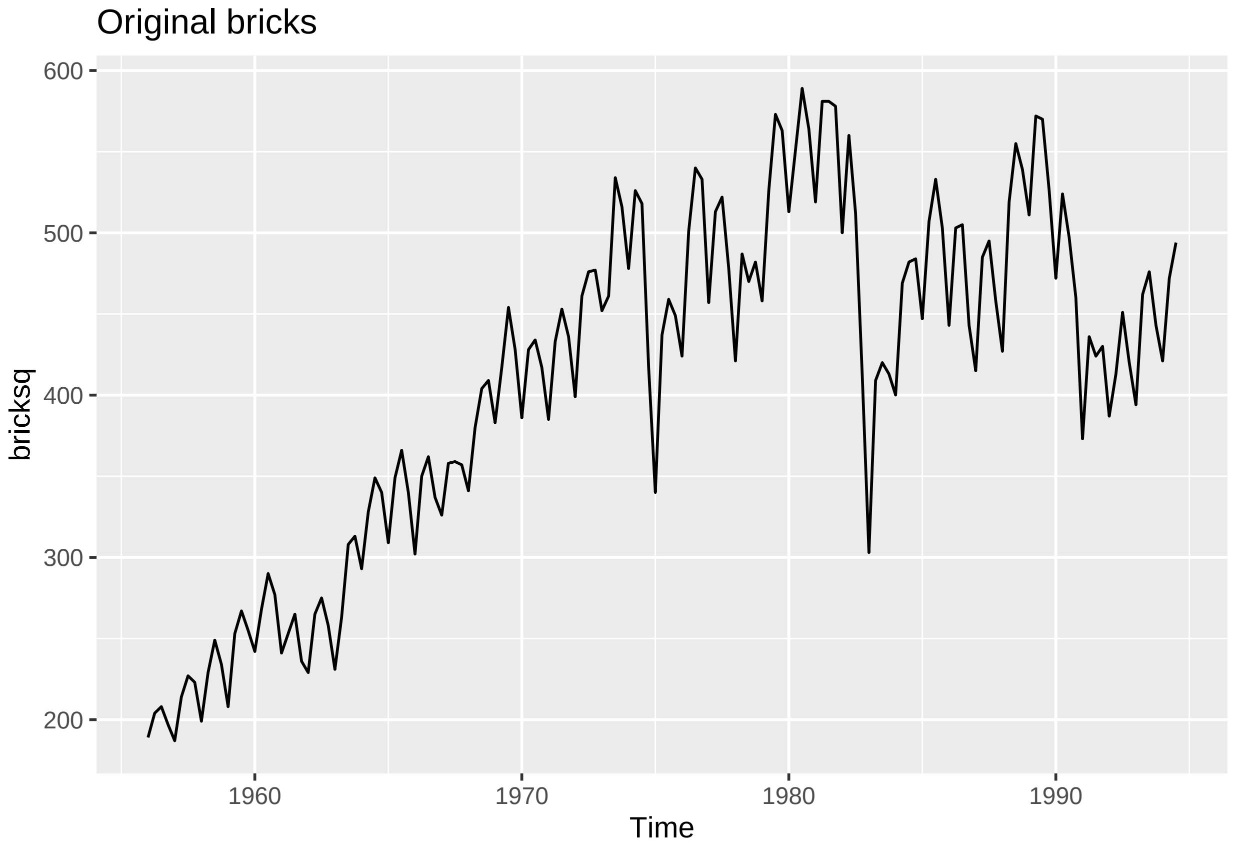
```
autoplot(dole_trasnformed) + ggtitle("Transformed dole")
```



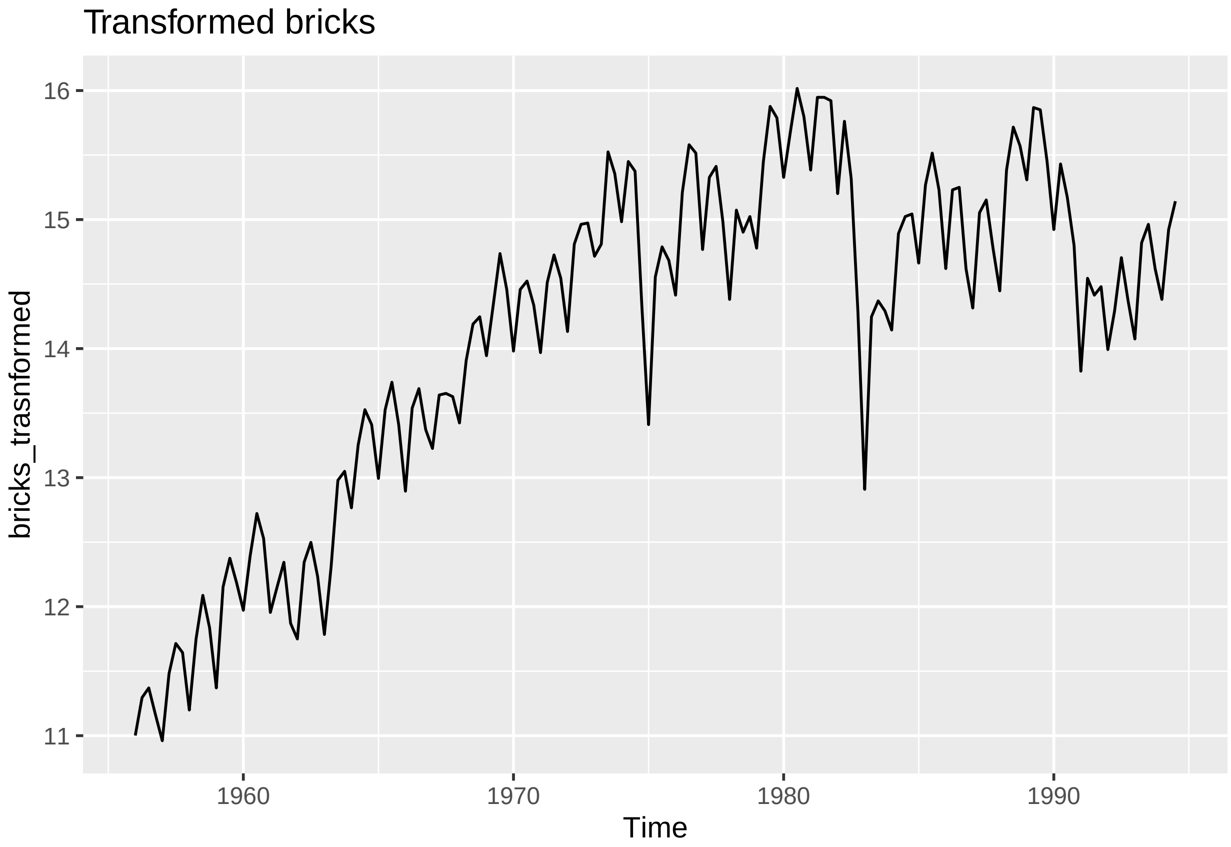
```
# Find lamba for bricks
lambda_bricks <- find_lambda(bricksq)
```

```
## [1] "Optimal Lambda for series: 0.254892859591799"
```

```
# Apply Box-Cox transformation to bricks series
bricks_trasnformed <- BoxCox(bricksq, lambda_bricks)
autoplot(bricksq) + ggtitle("Original bricks")
```



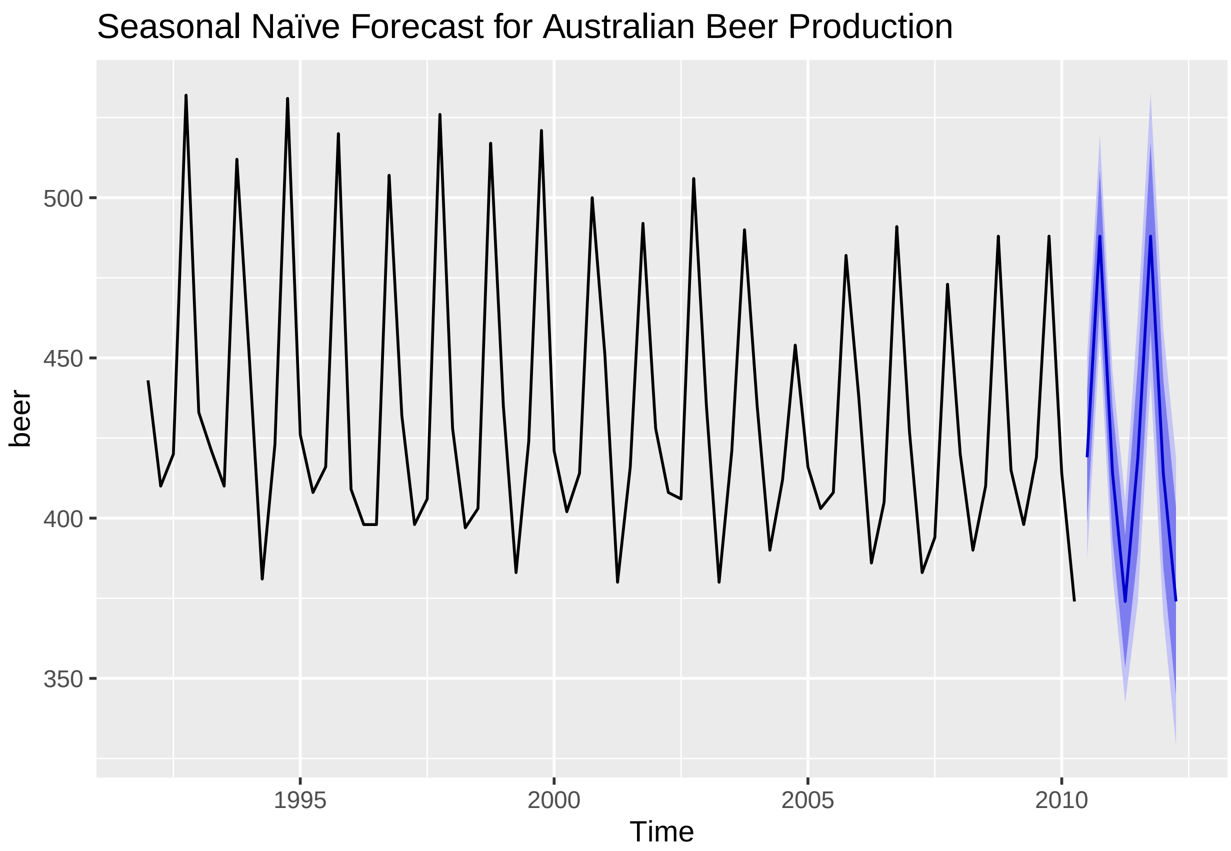
```r
autoplot(bricks_trasnformed) + ggtitle("Transformed bricks")
```



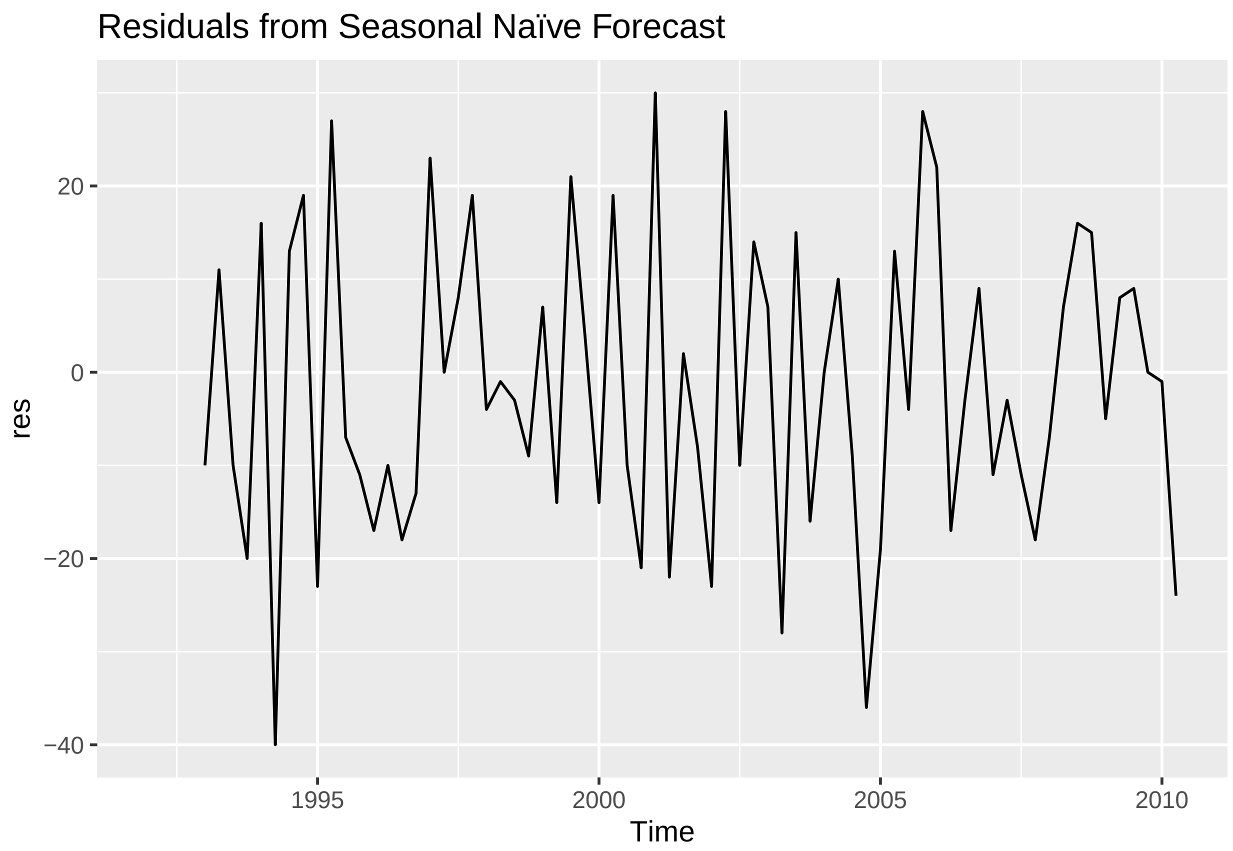
Dole can transformed by either log or Box-Cox due to large fluctuations in variance and bricks by Box-cox because it can help reduce variance and stabilize the series.

Exercise 3.5

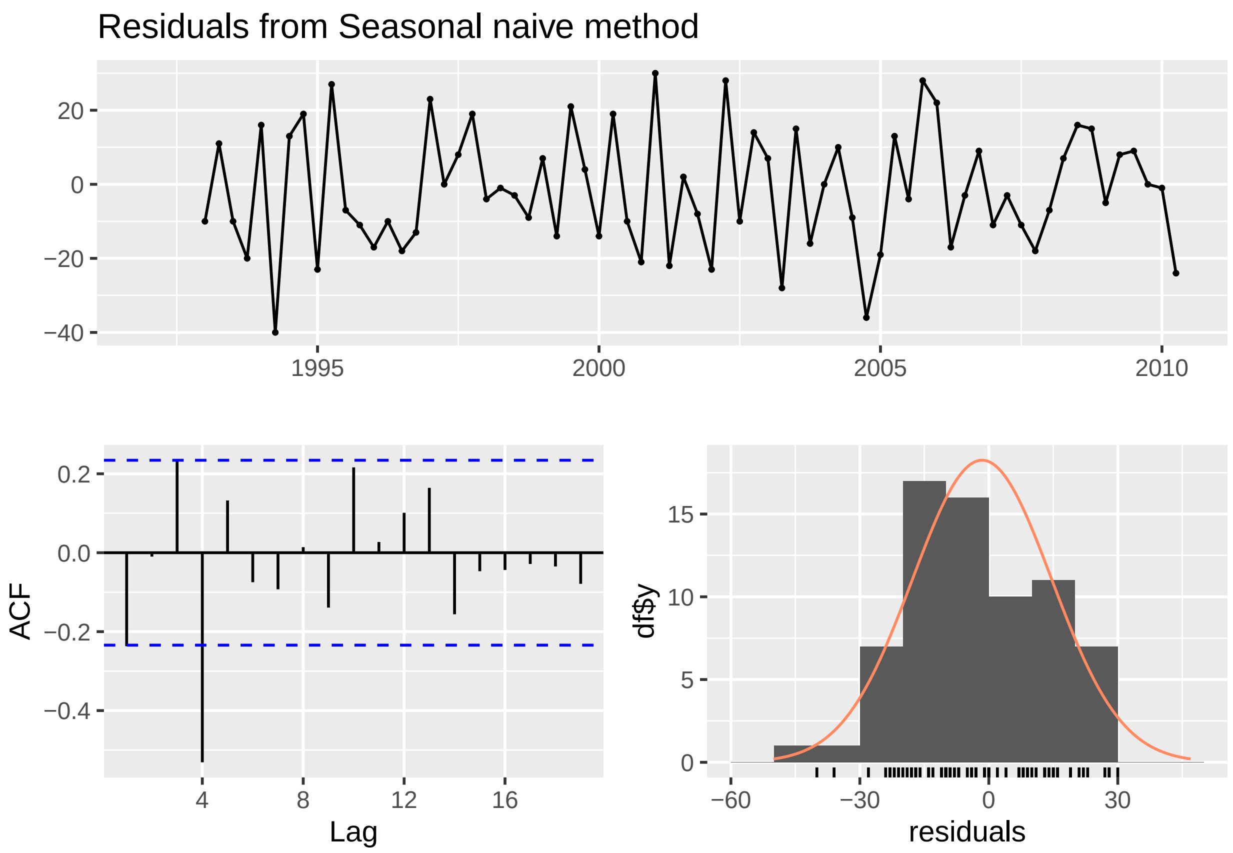
```r
# Load necessary libraries



# Extract quarterly Australian beer production data from 1992 onward
beer <- window(ausbeer, start=1992)

# Apply a seasonal naïve forecast
fc <- snaive(beer)

# Plot the forecast
autoplot(fc) + ggtitle("Seasonal Naïve Forecast for Australian Beer Production")
```



```
# Compute and plot residuals
res <- residuals(fc)
autoplot(res) + ggtitle("Residuals from Seasonal Naïve Forecast")
```



```
# Check if residuals are white noise and normally distributed
checkresiduals(fc)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 32.269, df = 8, p-value = 8.336e-05
##
## Model df: 0.   Total lags used: 8
```

Exercise 3.7 Are the following statements true or false? Explain your answer.

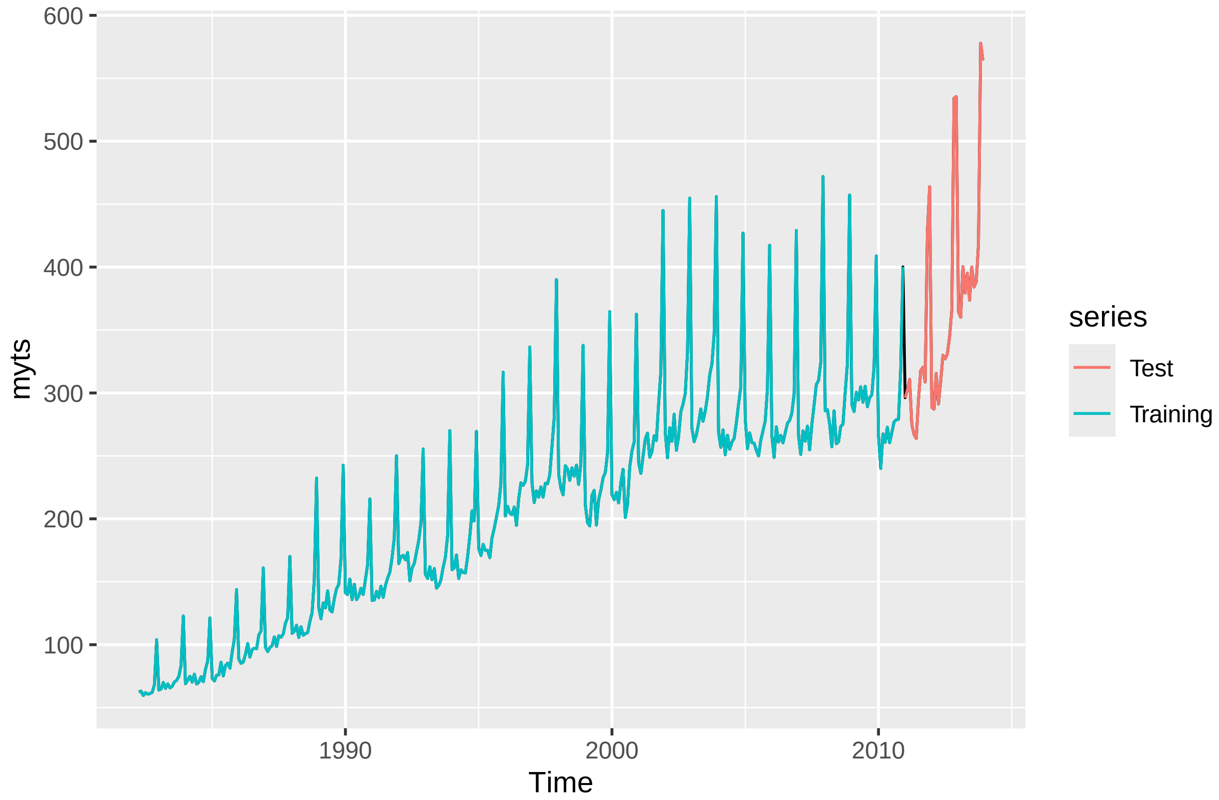
Good forecast methods should have normally distributed residuals. A model with small residuals will give good forecasts. The best measure of forecast accuracy is MAPE. If your model doesn't forecast well, you should make it more complicated. Always choose the model with the best forecast accuracy as measured on the test set.

While accuracy is crucial, it is not the only factor when selecting a model.

> The best model should also be interpretable, robust, and computationally efficient.
>
> Overfitting can lead to a model that performs exceptionally well on the test set but poorly in real-world applications.
>
> Practical considerations, such as data availability and ease of use, should also influence model selection.

Exercise 3.8

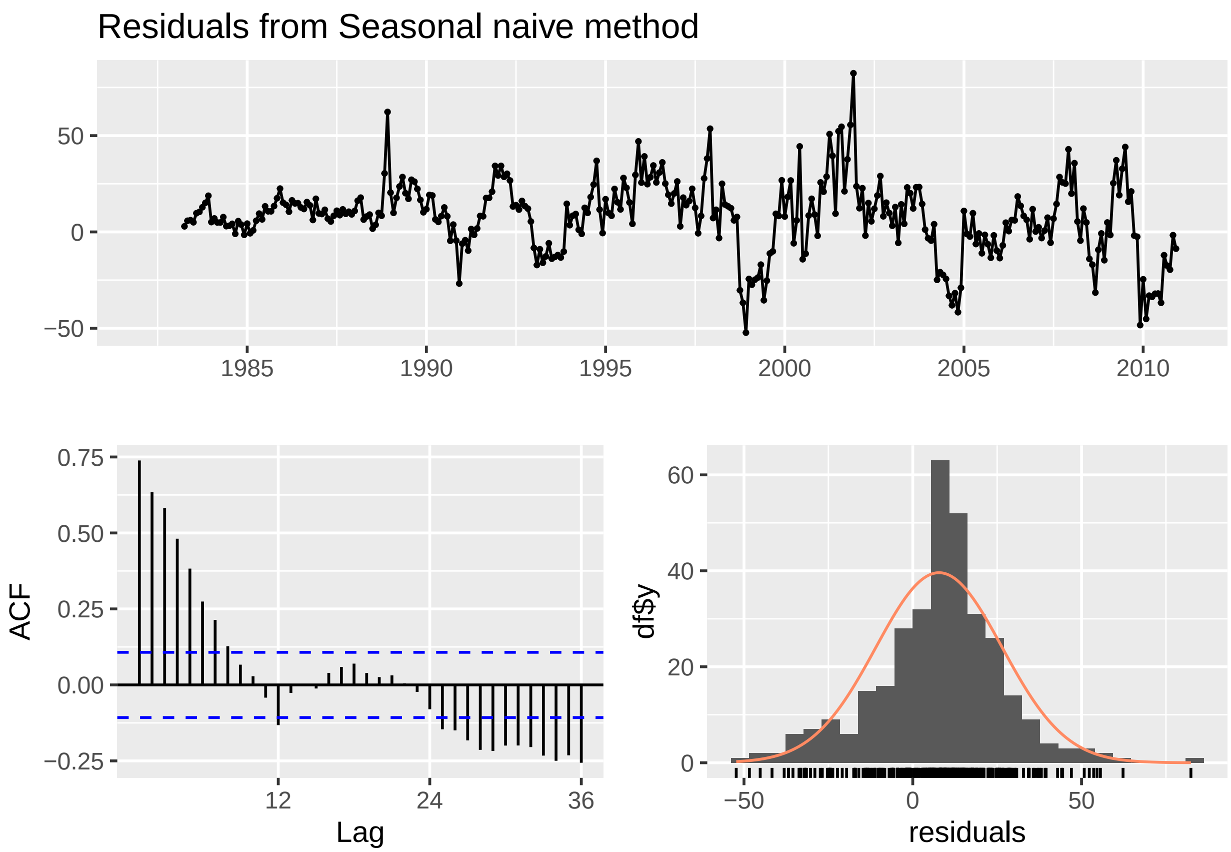
```
myts.train <- window(myts, end=c(2010,12))
myts.test <- window(myts, start=2011)

autoplot(myts) +
  autolayer(myts.train, series="Training") +
  autolayer(myts.test, series="Test")
```



```
fc <- snaive(myts.train)

accuracy(fc,myts.test)
```

```
##                     ME     RMSE      MAE       MPE      MAPE     MASE      ACF1
## Training set  7.772973 20.24576 15.95676  4.702754  8.109777 1.000000 0.7385090
## Test set     55.300000 71.44309 55.78333 14.900996 15.082019 3.495907 0.5315239
##              Theil's U
## Training set        NA
## Test set      1.297866
```

```
checkresiduals(fc)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 624.45, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

No. The residuals show autocorrelation and deviation from normality, suggesting that the seasonal naïve model is not fully appropriate. The accuracy is Highly sensitive. The test set follows a stronger trend, and the naïve model struggles to adapt, showing that forecast accuracy depends on the chosen split.

```
# Create training sets using window function
vsight1 <- window(visnights[, "QLDMetro"], end = c(2015, 4))
vsight2 <- window(visnights[, "QLDMetro"], end = c(2014, 4))
vsight3 <- window(visnights[, "QLDMetro"], end = c(2013, 4))
#Generate forecasts using snaive method
fc1 <- forecast(snaive(vsight1), h = 4)
fc2 <- forecast(snaive(vsight3), h = 4)
fc3 <- forecast(snaive(vsight3), h = 4)


# Calculate accuracy measures for each forecast
accuracy_fc1 <- accuracy(fc1)
accuracy_fc2 <- accuracy(fc2)
```

```r
accuracy_fc3 <- accuracy(fc3)

# Extract MAPE values
mape_fc1 <- accuracy_fc1[,"MAPE"]
mape_fc2 <- accuracy_fc2[,"MAPE"]
mape_fc3 <- accuracy_fc3[,"MAPE"]

# Print MAPE values
print(mape_fc1)
```

```
## [1] 7.97676
```

```r
print(mape_fc2)
```

```
## [1] 8.271365
```

```r
print(mape_fc3)
```

```
## [1] 8.271365
```

Yes, the accuracy measures **are sensitive** to the training/test split. The **Mean Absolute Percentage Error (MAPE)** values for three different forecasts (`fc1`, `fc2`, and `fc3`) show slight variations:

- **MAPE(fc1) = 7.98%**
- **MAPE(fc2) = 8.27%**
- **MAPE(fc3) = 8.27%**

This variation suggests that the accuracy metrics change depending on the **chosen training and test sets**. Even small adjustments in the split point can **impact forecast accuracy**, particularly when the time series has **trends, seasonality, or structural changes**.

1. **Different Trend Phases**: If the test set includes a period of **higher growth or fluctuations**, the forecast accuracy may decline.
2. **Seasonality Effects**: Some quarters may have more unpredictable patterns (e.g., tourism data in **visnights** may have seasonal peaks).
3. **Model Overfitting**: If a model is optimized for a specific training period, its performance may drop when tested on a different data segment.

The results confirm that **MAPE is sensitive to how the training and test data are divided**. To get a more reliable measure of accuracy, a **rolling forecast approach (cross-validation)** should be considered, ensuring that the model's performance is evaluated across multiple test periods rather than a single split.