DATA 624 - Homework 2

OMER OZEREN

Table of Contents

Exercises 3.1, 3.2, 3.3 and 3.8 from the Hyndman online Forecasting book.

```r
#clear the workspace
rm(list = ls())
#load req's packages
library(forecast)
library(readxl)
library(RCurl)
library(fpp2)
library(gridExtra)
library(grid)
library(CombMSC)
```

Question 3.1

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

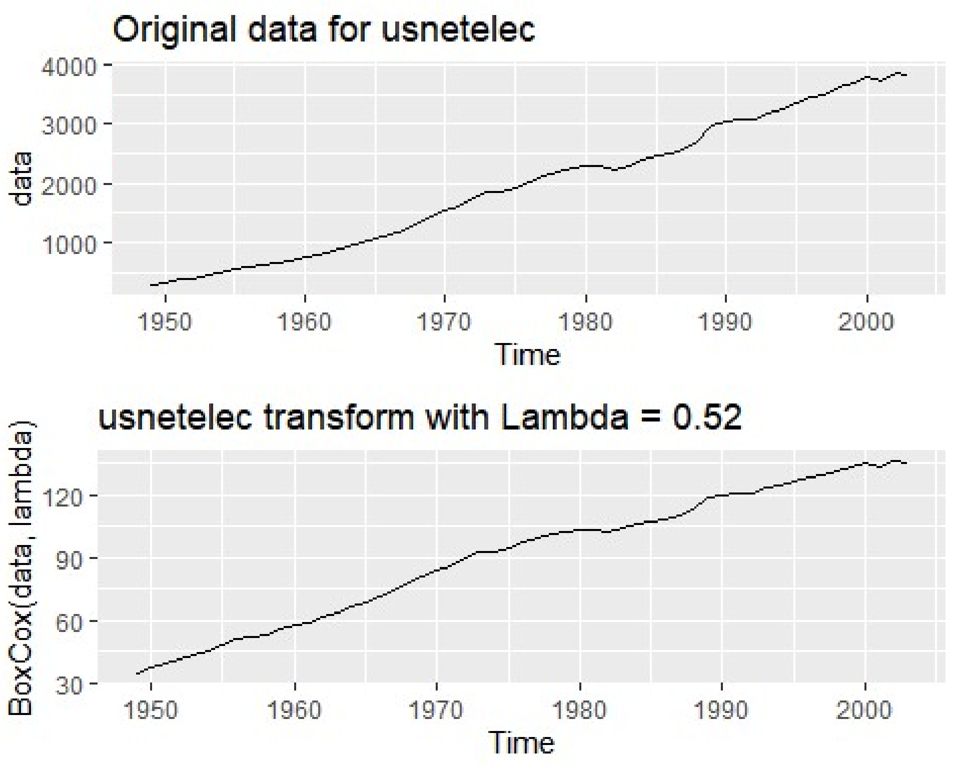
- usnetelec

- usgdp
- mcopper
- enplanements

```
bc.transform <- function(data,series.name) {
  #plot original data
  p1 <- autoplot(data,main=paste0("Original data for ",series.name))
  #find lambda
  (lambda <- BoxCox.lambda(data))
  #plot the transformed data
  p2 <- autoplot(BoxCox(data,lambda),
                 main=paste0(series.name, " transform with Lambda = ",
                             round(lambda,2)))
    print(paste0("An appropriate variance stabilizing Box-Cox transformation
for ", series.name, " is ", round(lambda,2) ))
  grid.arrange(p1, p2)
}
```

usnetelec

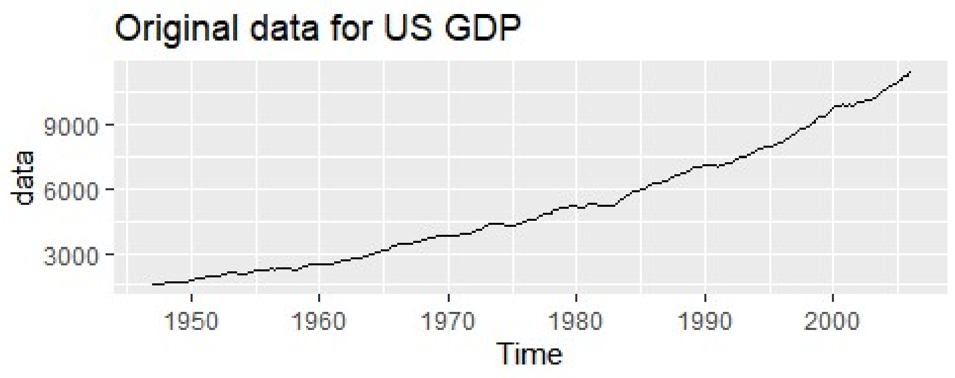
```
bc.transform(usnetelec,"usnetelec")

## [1] "An appropriate variance stabilizing Box-Cox transformation for
usnetelec is 0.52"
```

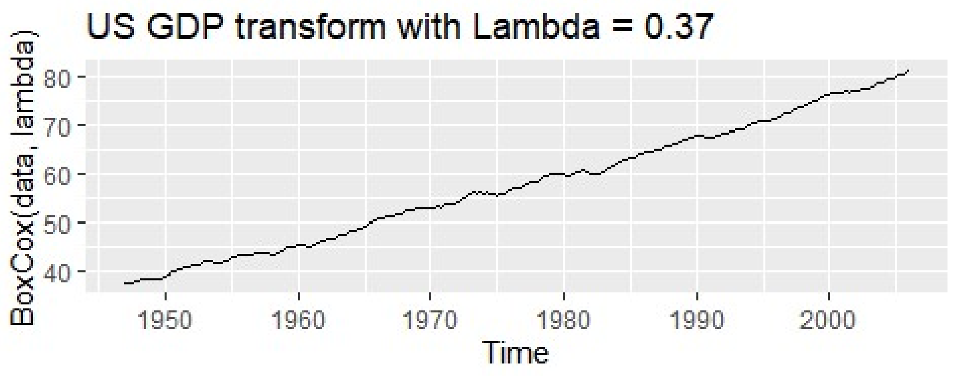


US GDP

```
bc.transform(usgdp,"US GDP")
```

```
## [1] "An appropriate variance stabilizing Box-Cox transformation for US GDP
is 0.37"
```

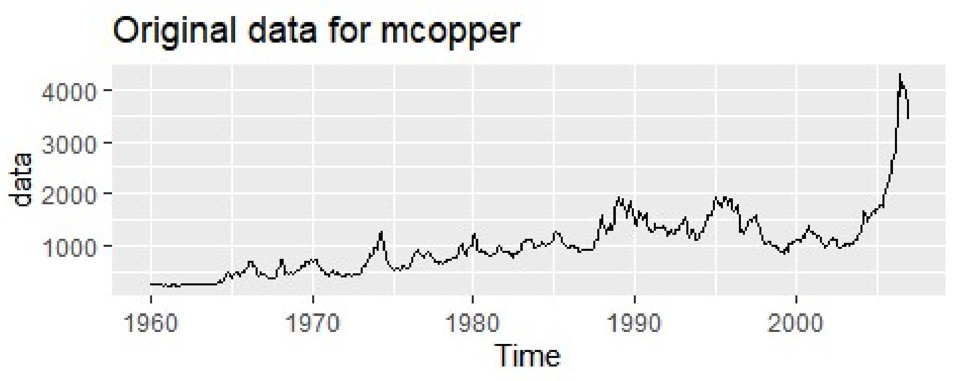


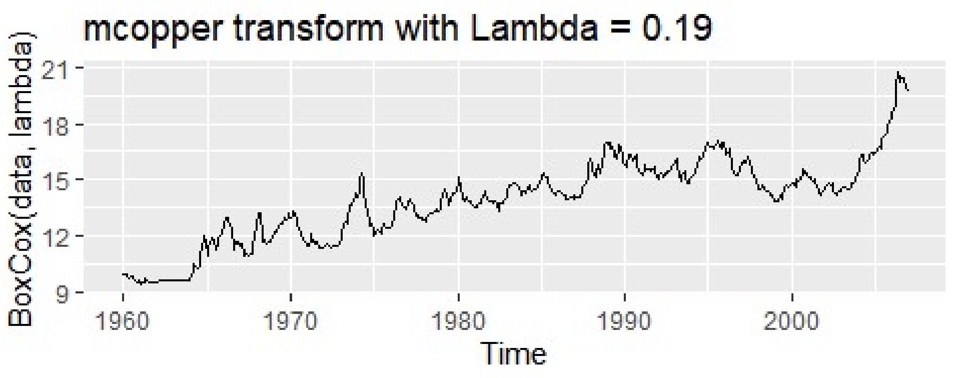


mcopper

```
bc.transform(mcopper,"mcopper")
```

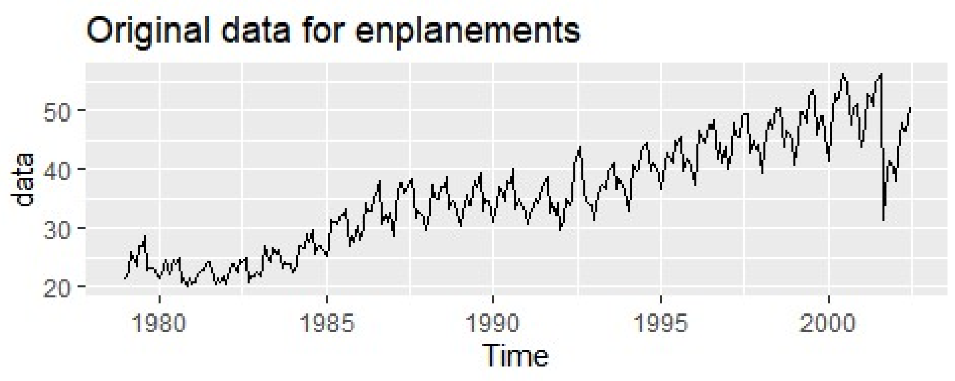
```
## [1] "An appropriate variance stabilizing Box-Cox transformation for
mcopper is 0.19"
```

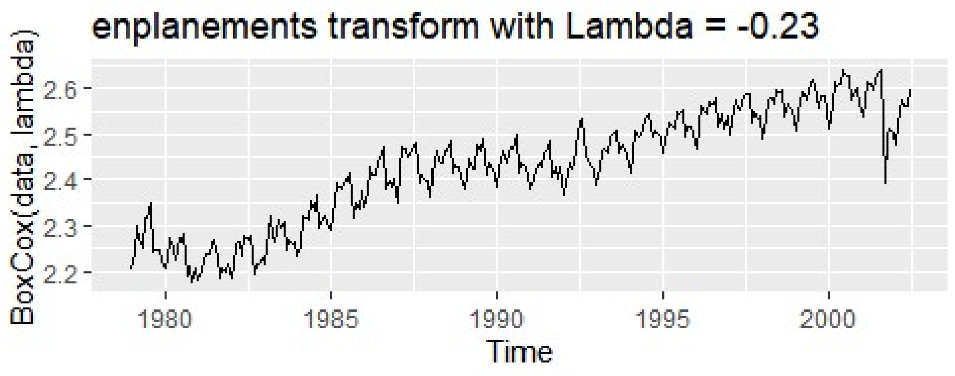




enplanements

```
bc.transform(enplanements,"enplanements")

## [1] "An appropriate variance stabilizing Box-Cox transformation for
enplanements is -0.23"
```

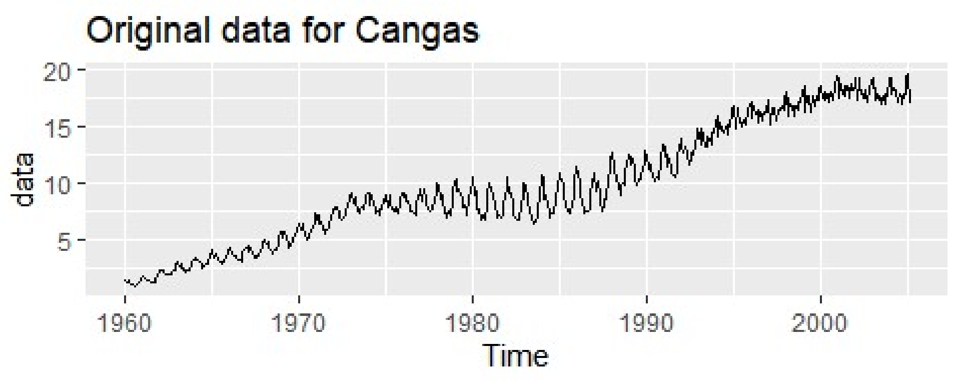


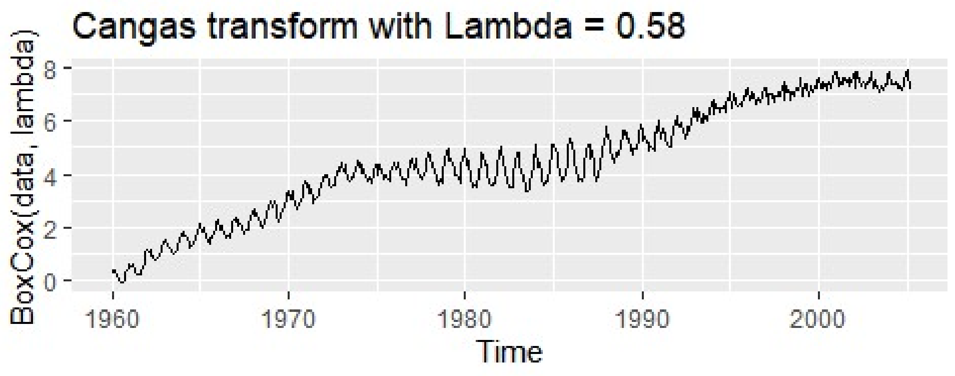


Question 3.2

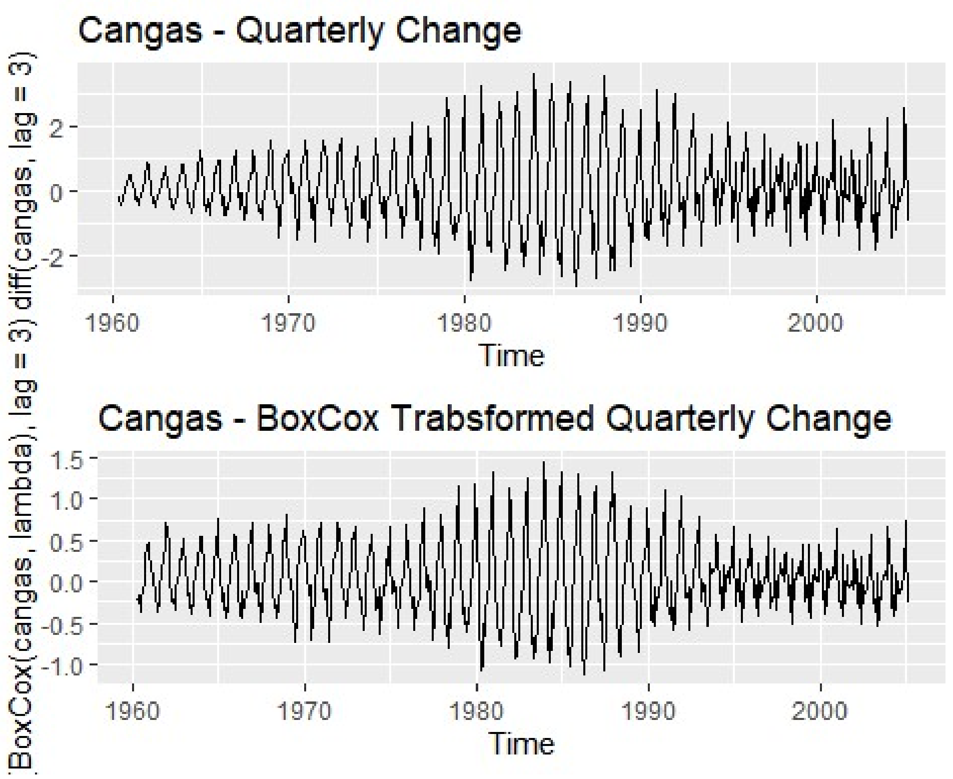
Why is a Box-Cox transformation unhelpful for the cangas data?

```
bc.transform(cangas,"Cangas")

## [1] "An appropriate variance stabilizing Box-Cox transformation for Cangas
is 0.58"
```





```
(lambda <- BoxCox.lambda(cangas))

## [1] 0.5767759

p1 <- autoplot(diff(cangas,lag=3),main = "Cangas - Quarterly Change")
p2 <- autoplot(diff(BoxCox(cangas,lambda),lag=3),main = "Cangas - BoxCox
Trabsformed Quarterly Change")
grid.arrange(p1, p2)
```



The box-cox transformaiton is unhelpful because it does not seem to provide any effect in terms of variance standardization / stabilization. I suspect that this is because of the nature of the data. We appear to have 3 regimes, as is evident in the quarterly change plots.

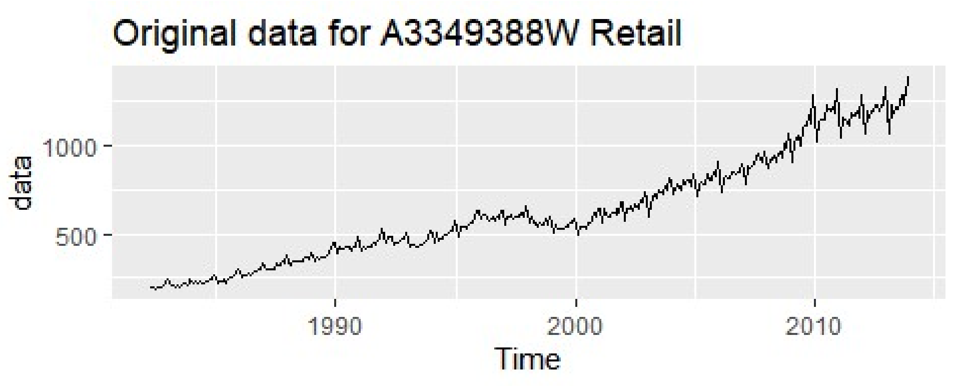
- moderate variance regime for approx the first 1/3rd of the data
- high variance for the mid-regime
- low variance for the last 3rd

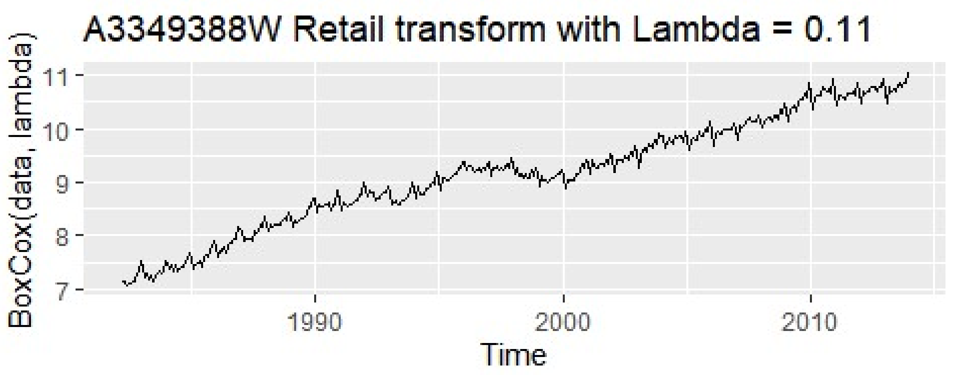
Question 3.3

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

```r
#load the A3349388W retail data
temp_file <- tempfile(fileext = ".xlsx")
download.file(url =
"https://github.com/omerozeren/DATA624/raw/master/HMW1/retail.xlsx",
              destfile = temp_file,
              mode = "wb",
              quiet = TRUE)
retaildata <- readxl::read_excel(temp_file,skip=1)
A3349388W.retail <- ts(retaildata[,"A3349388W"],
  frequency=12, start=c(1982,4))
#run my function
bc.transform(A3349388W.retail,"A3349388W Retail")
```

```
## [1] "An appropriate variance stabilizing Box-Cox transformation for
A3349388W Retail is 0.11"
```





An appropriate value for lambda to be used for the A3349388W retail data series that was used for homework#1 would be 0.11. As can be seen from the plots above, we see a reasonable smoothing of the data and the appearance of exponential growth present in the original data is significantly diminished.

Question 3.8

For your retail time series (from Exercise 3 in Section 2.10):

A - Split the Data

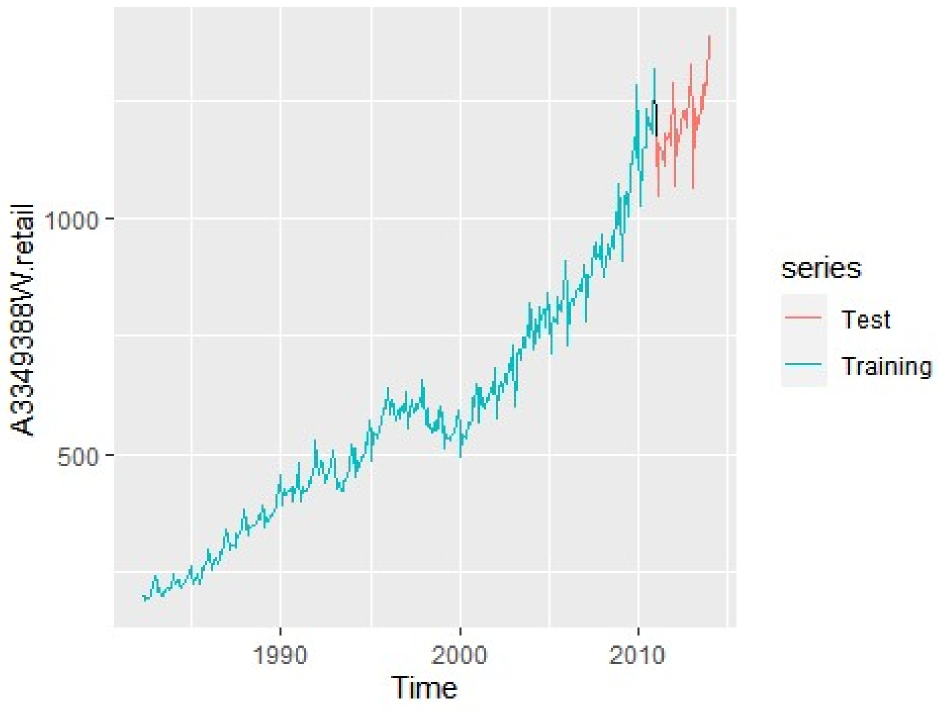
Split the data into two parts using:

```
myts.train <- window(A3349388W.retail, end=c(2010,12))
myts.test <- window(A3349388W.retail, start=2011)
```

B - Check the split

Check that your data have been split appropriately by producing the following plot.

```
autoplot(A3349388W.retail) +
  autolayer(myts.train, series="Training") +
  autolayer(myts.test, series="Test")
```



C - Calculate Forecasts

Calculate forecasts using snaive applied to *myts.train*.

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

This method works remarkably well for many economic and financial time series. However, we will be using the seasonal naive method. "In this case, we set each forecast to be equal to the last observed value from the same season of the year (e.g., the same month of the previous year).

D - Compare Accuracy

Compare the accuracy of your forecasts against the actual values stored in *myts.test*.

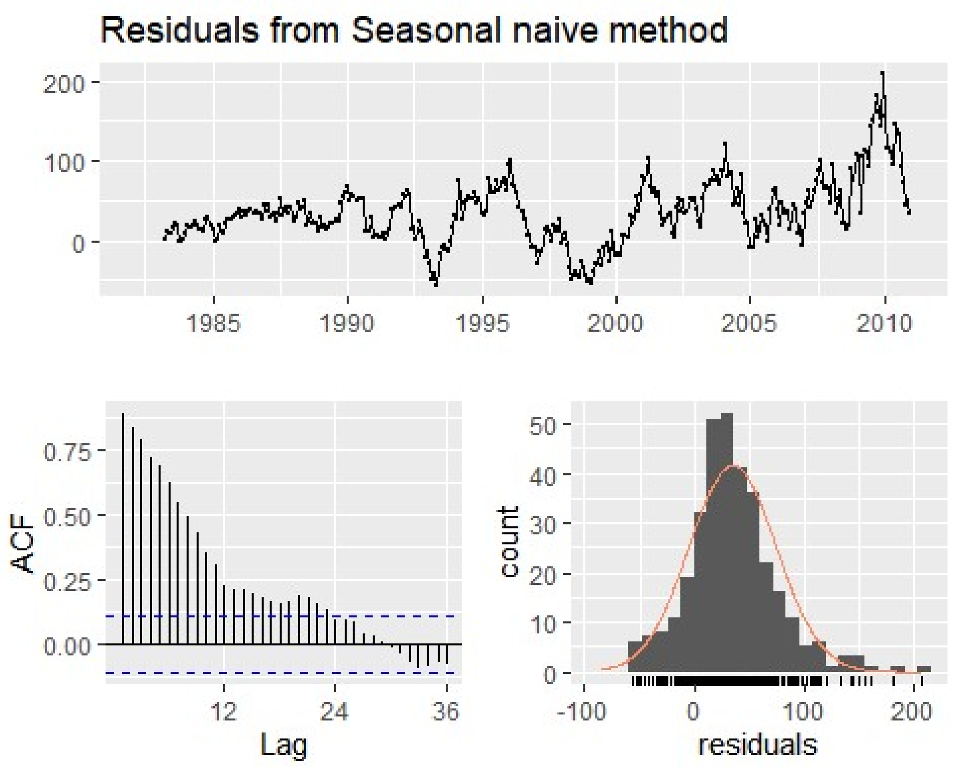
```
acc <- accuracy(fc,myts.test)
```

According to the textbook, chapter 3.4, "When choosing models, it is common practice to separate the available data into two portions, training and test data, where the training data is used to estimate any parameters of a forecasting method and the test data is used to evaluate its accuracy. Because the test data is not used in determining the forecasts, it should provide a reliable indication of how well the model is likely to forecast on new data."

E - Check Residuals

Check the residuals.

```r
checkresiduals(fc)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 1631.7, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

According to chapter 3.3 “Residual Diagnostics”:

The “residuals” in a time series model are what is left over after fitting a model. For many (but not all) time series models, the residuals are equal to the difference between the observations and the corresponding fitted values. Residuals are useful in checking whether a model has adequately captured the information in the data. A good forecasting method will yield residuals with the following properties:

There appears to be a significant amount of correlation in the residuals in this series. Residual variance does not look constant. The distribution of the residuals appears to be skewed to the right. I suspect that this model is missing something (inflation adjustment? Normalization by some measure of econ performance?)If the residuals have a mean other than zero, then the forecasts are biased.

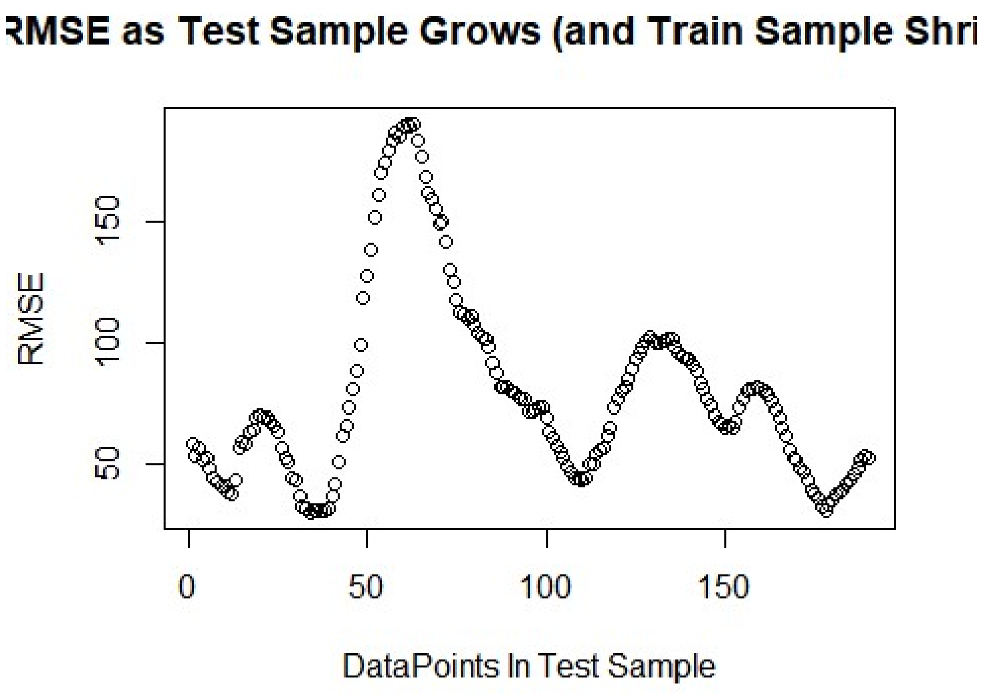
Any forecasting method that does not satisfy these properties can be improved. However, that does not mean that forecasting methods that satisfy these properties cannot be improved. It is possible to have several different forecasting methods for the same data set, all of which satisfy these properties. Checking these properties is important in order to see whether a method is using all of the available information, but it is not a good way to select a forecasting method.

These two properties make the calculation of prediction intervals easier (see Section 3.5 for an example). However, a forecasting method that does not satisfy these properties cannot necessarily be improved. Sometimes applying a Box-Cox transformation may assist with these properties, but otherwise there is usually little that you can do to ensure that your residuals have constant variance and a normal distribution. Instead, an alternative approach to obtaining prediction intervals is necessary.

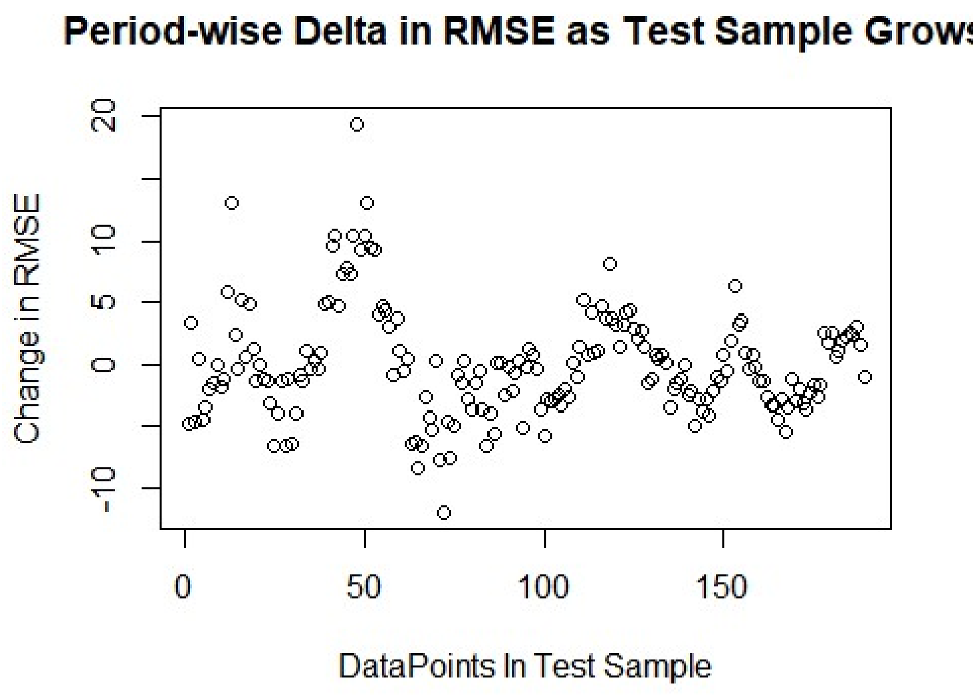
F - Sensitivity Analysis

How sensitive are the accuracy measures to the training/test split?

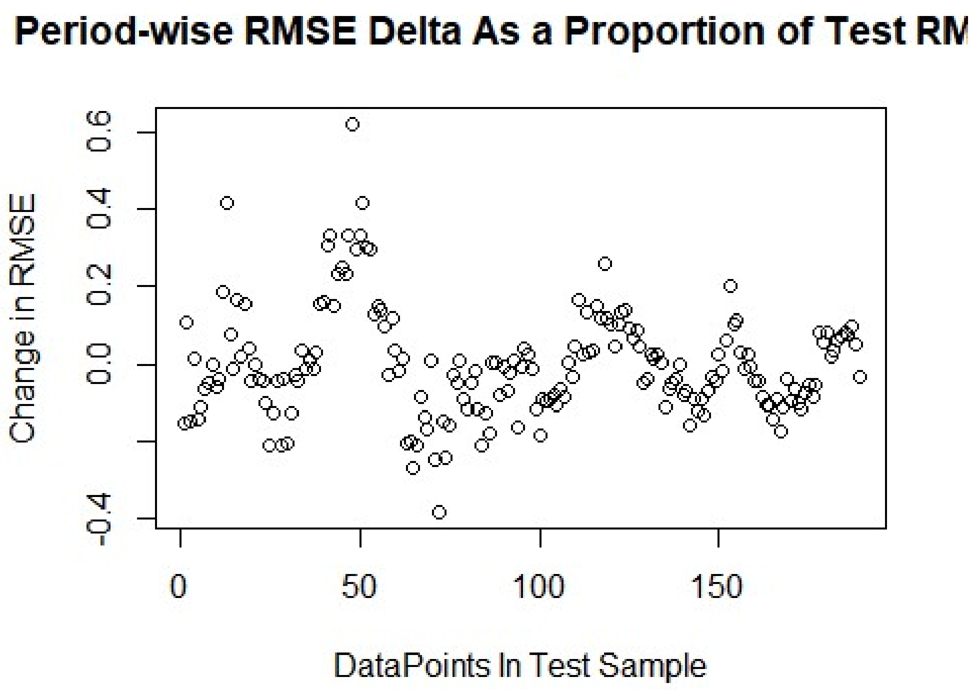
```
set.seed(108)
len <- 190
results <- matrix(0,nrow=len,ncol=1)
for (i in 1:len){
  tts <- splitTrainTest(A3349388W.retail, numTrain = length(A3349388W.retail)
- i)
  myts.train <- tts$train
  myts.test <- tts$test

  fc <- snaive(myts.train)
  ac <- accuracy(fc,myts.test)

  results[i] <- ac[2,2]

}
plot(results,
     main="RMSE as Test Sample Grows (and Train Sample Shrinks)",
     xlab="DataPoints In Test Sample",
     ylab="RMSE")
```



```r
plot(diff(results,lag=1),
     main="Period-wise Delta in RMSE as Test Sample Grows",
     xlab="DataPoints In Test Sample",
     ylab="Change in RMSE")
```



```
plot(diff(results,lag=1)/acc[2,2] ,
     main="Period-wise RMSE Delta As a Proportion of Test RMSE",
     xlab="DataPoints In Test Sample",
     ylab="Change in RMSE")
```



The accuracy measure are always sensitive to the training/test split. There are better ways to check the robustness of the methods in terms of accuracy such as using a time series cross-validation tsCV(). According to chapter 3.4:

"A more sophisticated version of training/test sets is time series cross-validation. In this procedure, there are a series of test sets, each consisting of a single observation. The corresponding training set consists only of observations that occurred prior to the observation that forms the test set. Thus, no future observations can be used in constructing the forecast. Since it is not possible to obtain a reliable forecast based on a small training set, the earliest observations are not considered as test sets.

The forecast accuracy is computed by averaging over the test sets. This procedure is sometimes known as "evaluation on a rolling forecasting origin" because the "origin" at which the forecast is based rolls forward in time.

With time series forecasting, one-step forecasts may not be as relevant as multi-step forecasts. In this case, the cross-validation procedure based on a rolling forecasting origin can be modified to allow multi-step errors to be used.

Time series cross-validation is implemented with the tsCV() function."

Here we look at how RMSE (just to pick one of the measures of model accuracy) varies as we modify the train/test split. Here we look at an out-sample size of 1 to 190 (where 1 == a single point & 190 == half of the data). We can see that the RMSE varies dramatically as the relative split changes and as such, we can conclude that yes, the accuracy measures are quite sensitive in this case.

The third plot shows the difference in RMSE as the test sample grows as a proportion of the original test RMSE. As we can see, the differences are as much as 40%.