Data624 - Homework2

Amit Kapoor

02/18/2021

Contents


```
library(fpp2)
library(gridExtra)
```

3.1

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

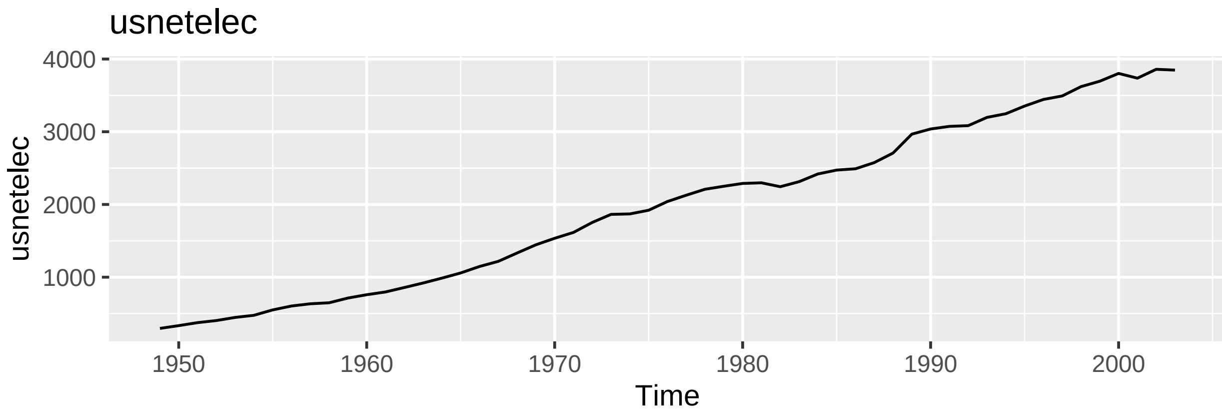
```
# function to draw 2 plots: original and with BoxCox transformation
plot_timeseries <- function(timeseries) {
  lambda <- BoxCox.lambda(timeseries)

  ts_original <- autoplot(timeseries) +
    ggtitle(substitute(timeseries)) +
    xlab("Time") +
    ylab(substitute(timeseries))

  ts_boxcox <- autoplot(BoxCox(timeseries, lambda)) +
    ggtitle(paste('BoxCox transformed lambda=', round(lambda,2))) +
    xlab("Time") +
    ylab(paste(substitute(timeseries), " transformed"))

  grid.arrange(arrangeGrob(ts_original, ts_boxcox, ncol=1, nrow = 2))
}
```

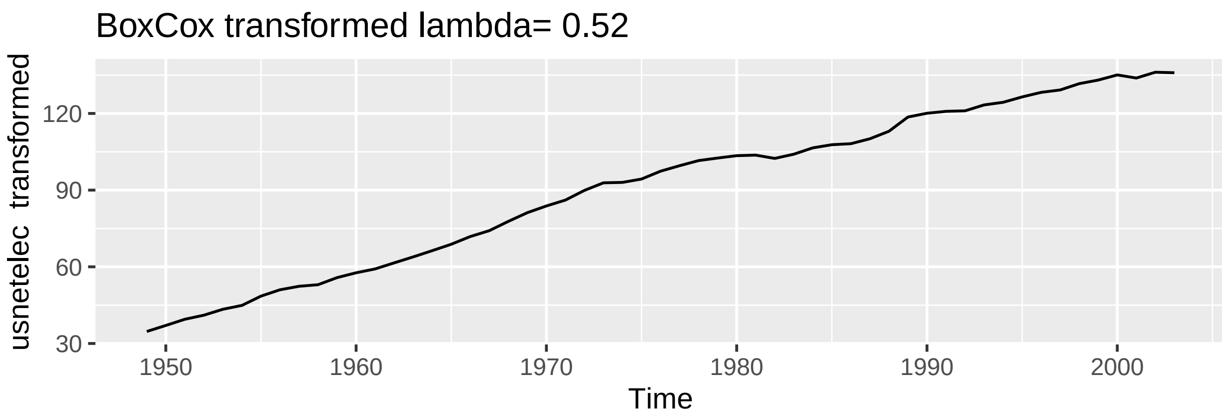
usnetelec

Annual US net electricity generation - Annual US net electricity generation (billion kwh) for 1949-2003

```
?usnetelec
```

```r
plot_timeseries(usnetelec)
```





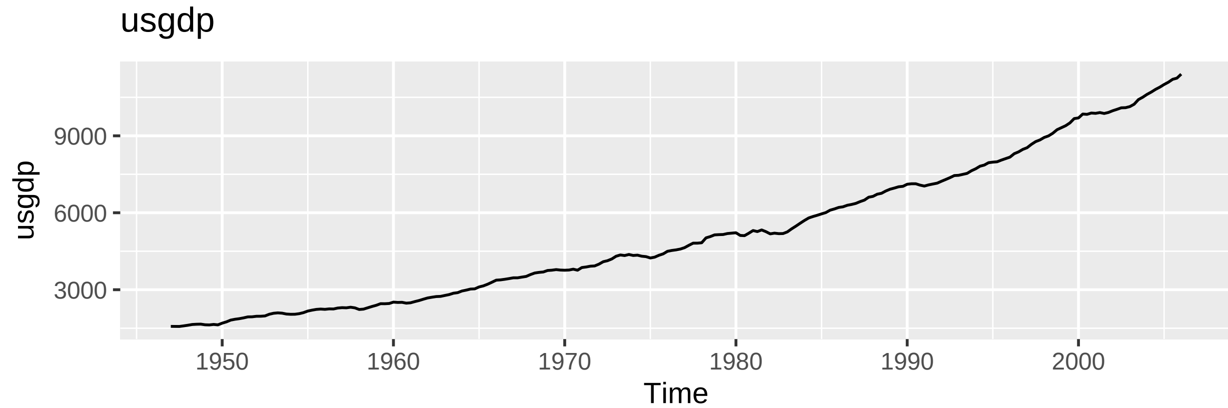
The BoxCox transformation made no apparent difference to reduce the variation in usnetelec data.Therefore no Box-Cox transformation is needed here.

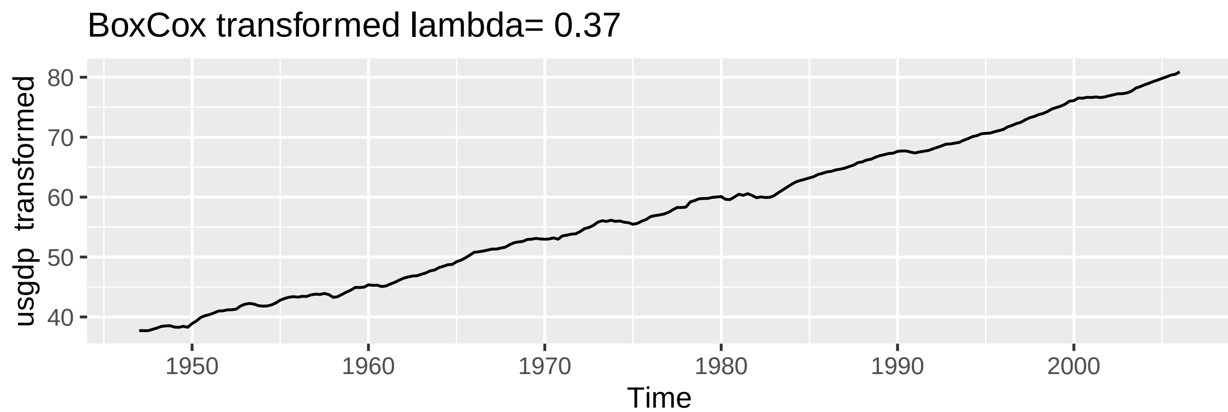
usgdp

Quarterly US GDP - Quarterly US GDP. 1947:1 - 2006.1.

```r
?usgdp
```

```r
plot_timeseries(usgdp)
```





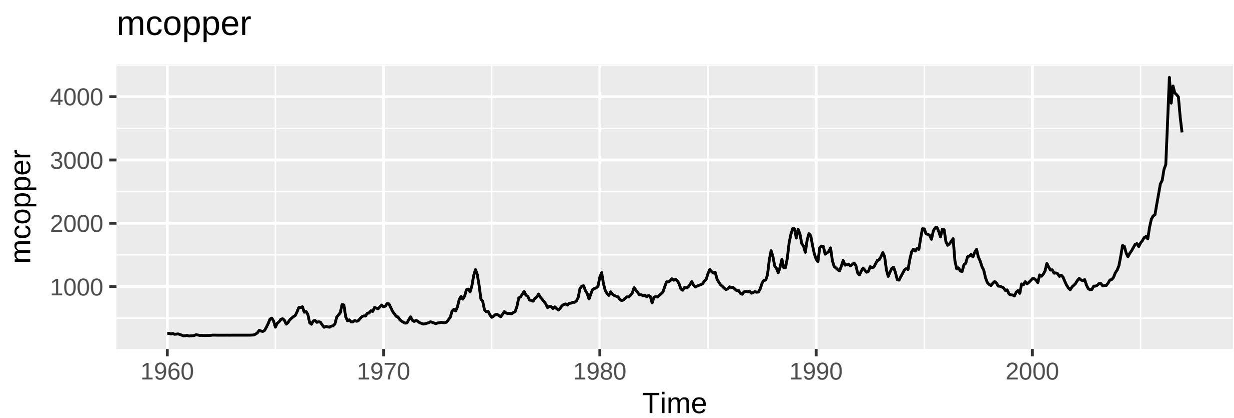
In this case, BoxCox transformation removed the curvature that exists in original data and could make possibility of linear regression model.

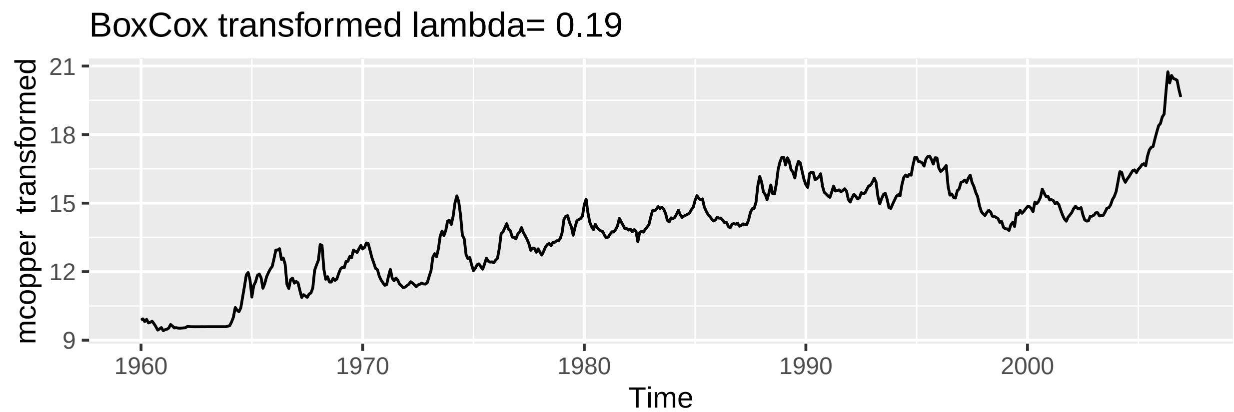
mcopper

Monthly copper prices - Monthly copper prices. Copper, grade A, electrolytic wire bars/cathodes,LME,cash (pounds/ton) Source: UNCTAD (http://stats.unctad.org/Handbook).

```
?mcopper
```

```
plot_timeseries(mcopper)
```





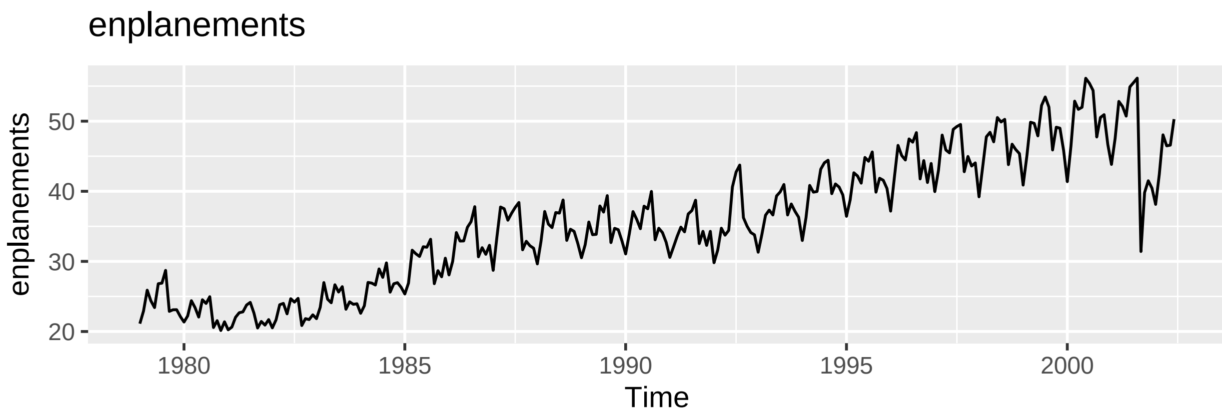
For mcopper data, I dont see any significant change after transformation so dont see a need to apply BoxCox transformation.

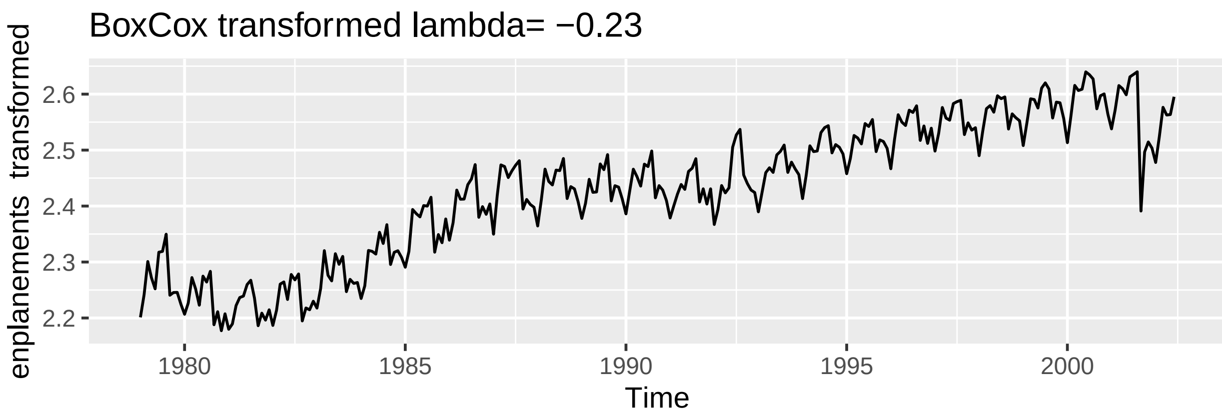
enplanements

Monthly US domestic enplanements - Domestic Revenue Enplanements (millions): 1996-2000. SOURCE: Department of Transportation, Bureau of Transportation Statistics, Air Carrier Traffic Statistic Monthly.

```
?enplanements
```

```
plot_timeseries(enplanements)
```





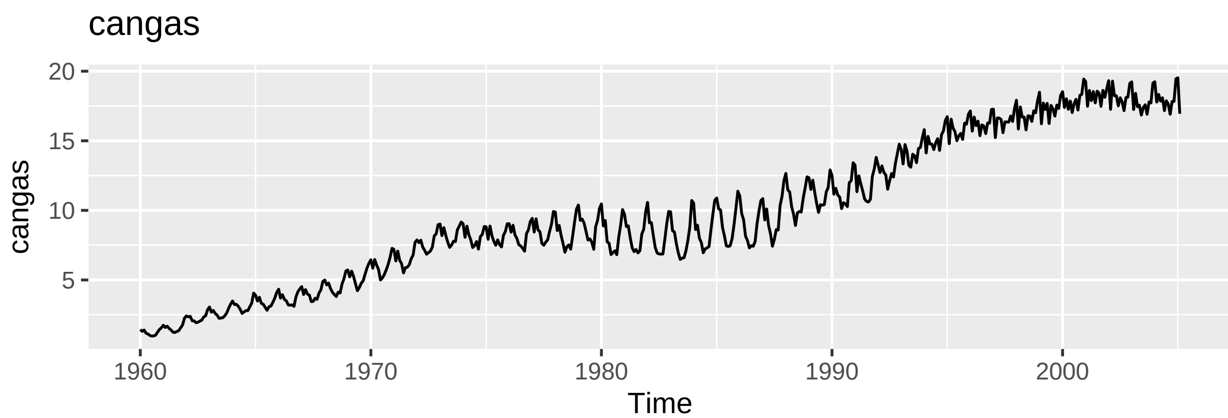
We could see BoxCox transformation did seasonality transformed to show seasonal jump in transformed data.

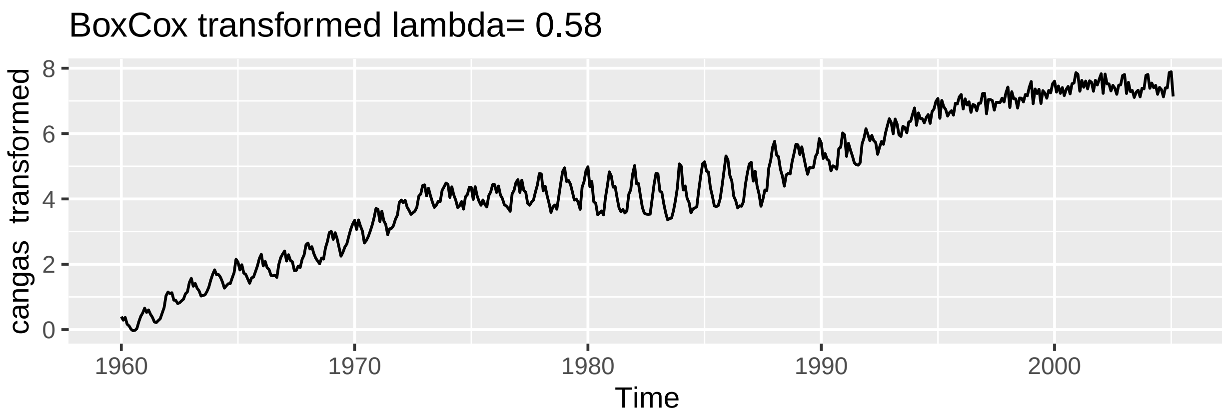
3.2

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

```
?cangas
```

```
plot_timeseries(cangas)
```





For the overall cangas data, the BoxCox transformation doesn't appear to be useful because the middle portion of the data varies much wildly than the lower and upper regions of the data. It could be if the data is separated in 3 regions but with overall data transformation doesn't make any difference.

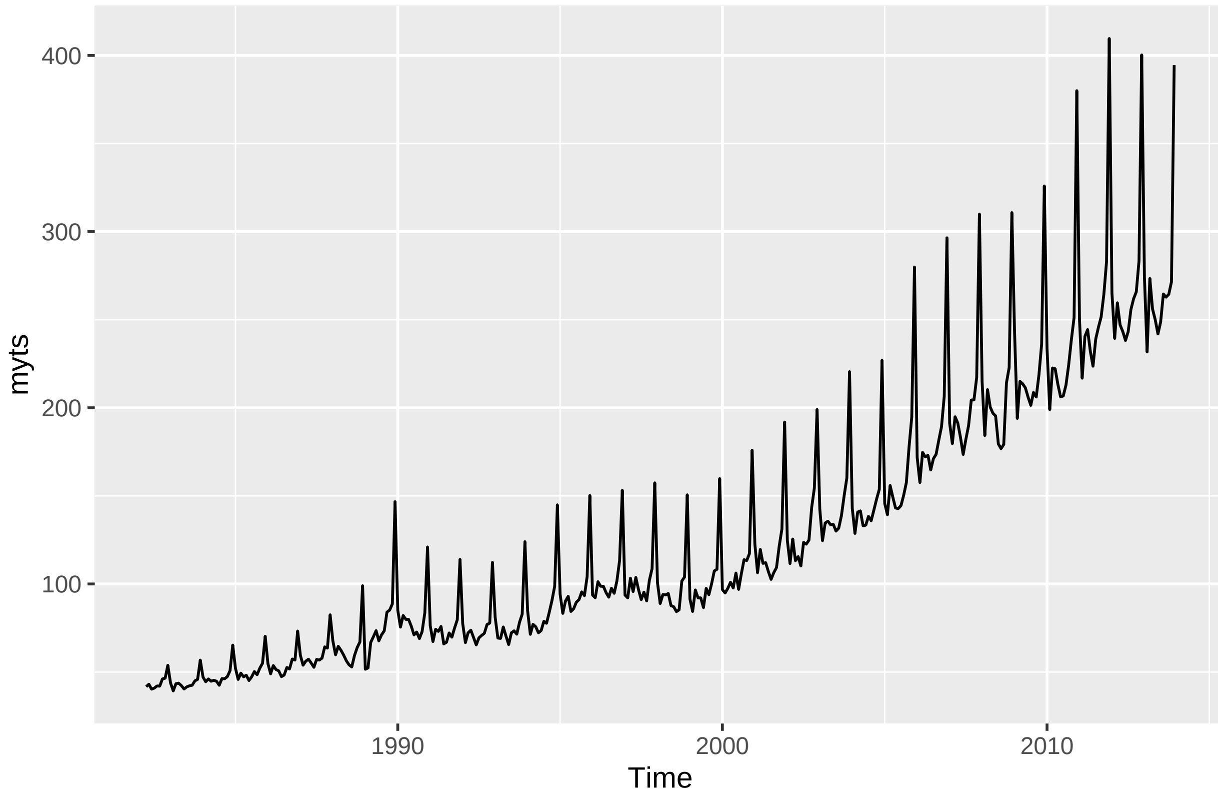
3.3

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

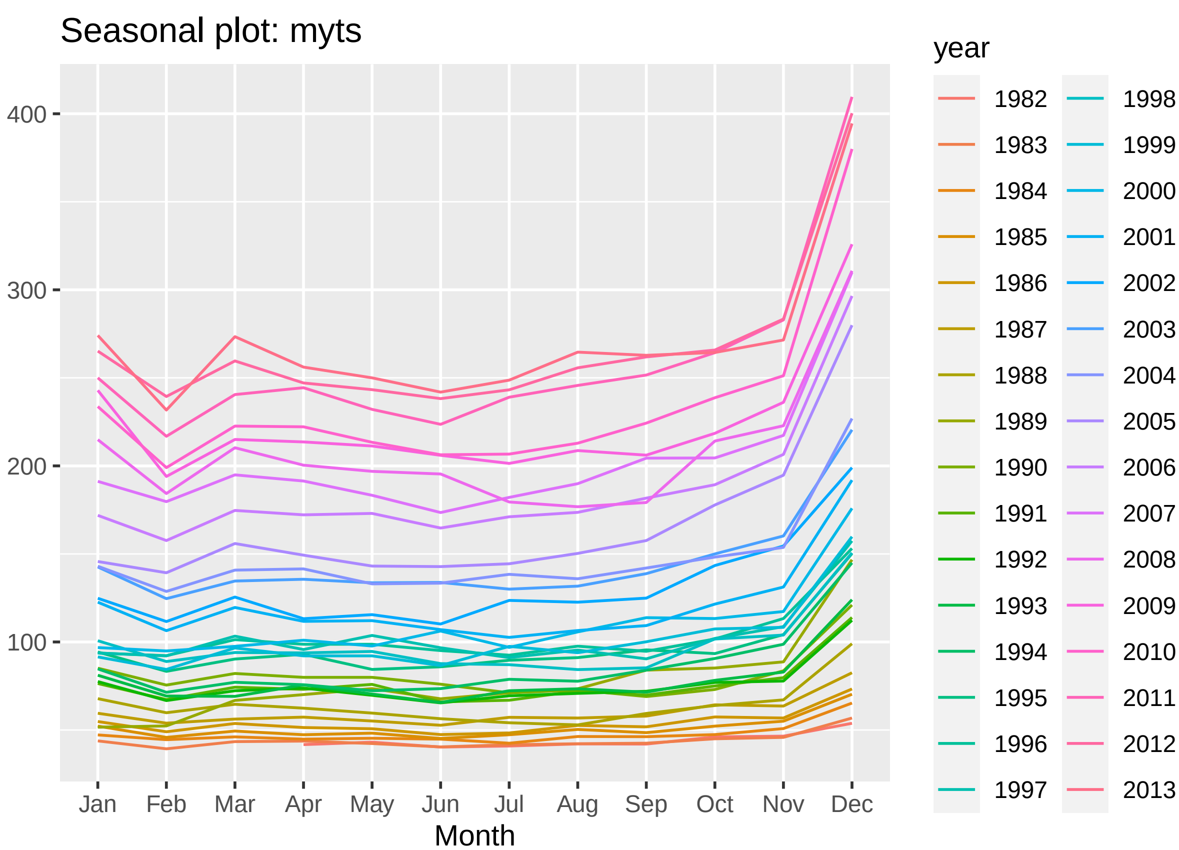
```
retaildata <- readxl::read_excel("retail.xlsx", skip=1)
myts <- ts(retaildata[,"A3349627V"], frequency=12, start=c(1982,4))
head(myts)
```

```
##       Apr  May  Jun  Jul  Aug  Sep
## 1982 41.7 43.1 40.3 40.9 42.1 42.0
```

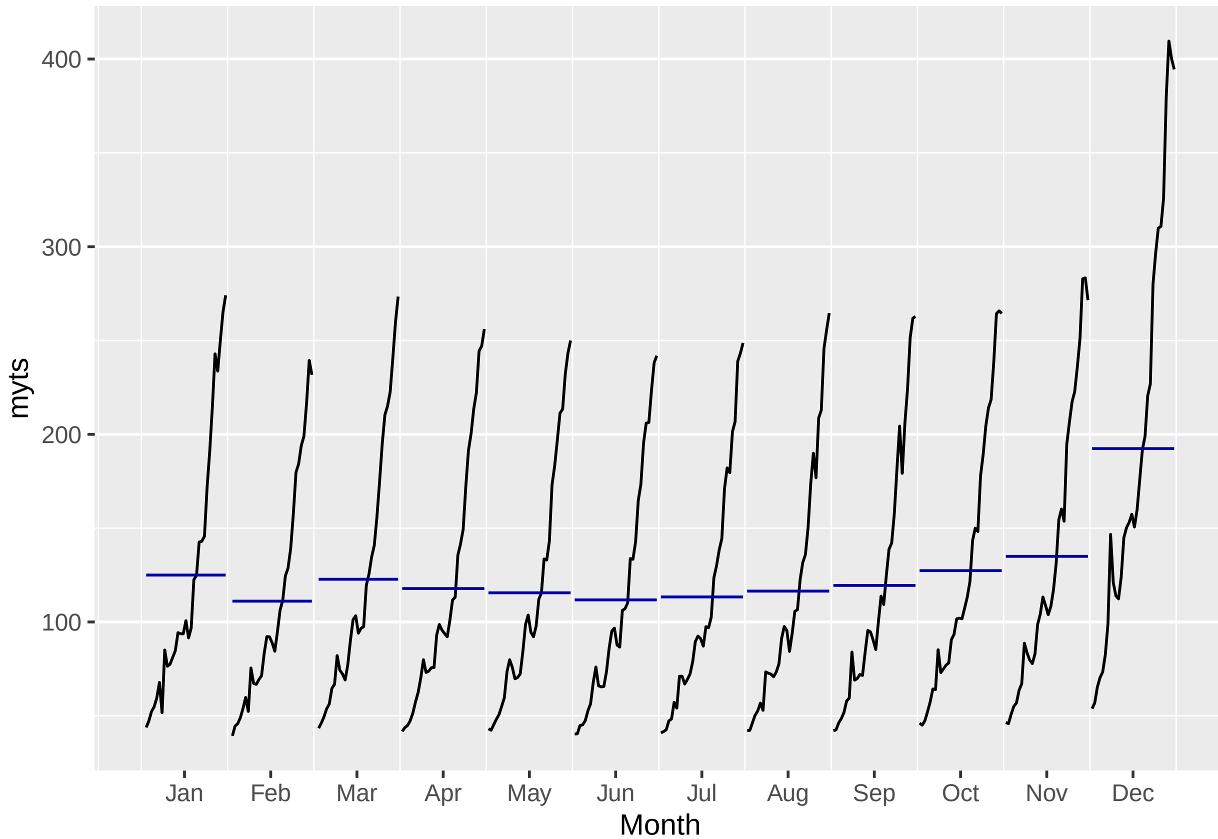
```
autoplot(myts)
```



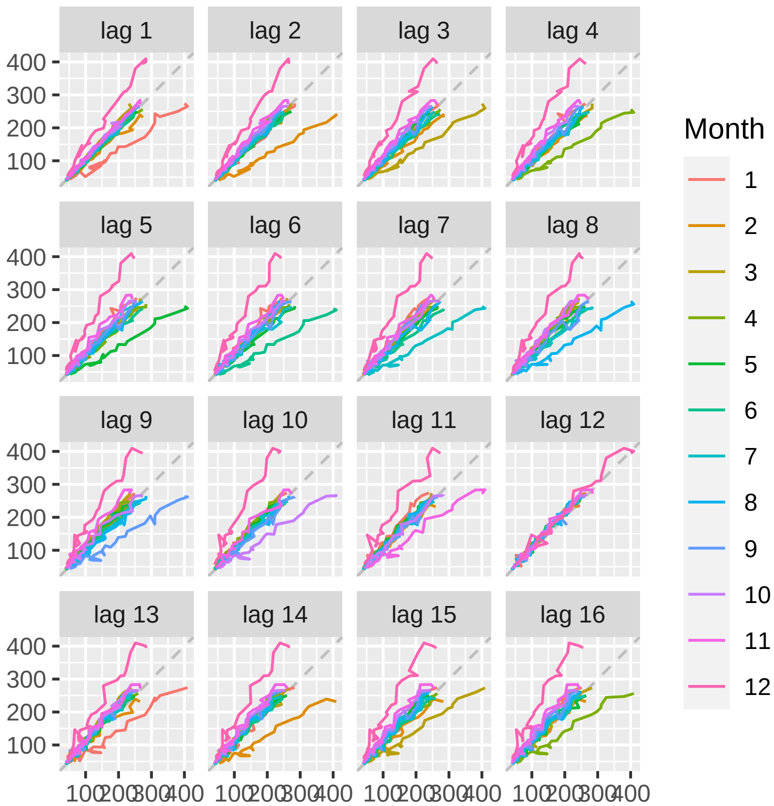

```r
ggseasonplot(myts)
```



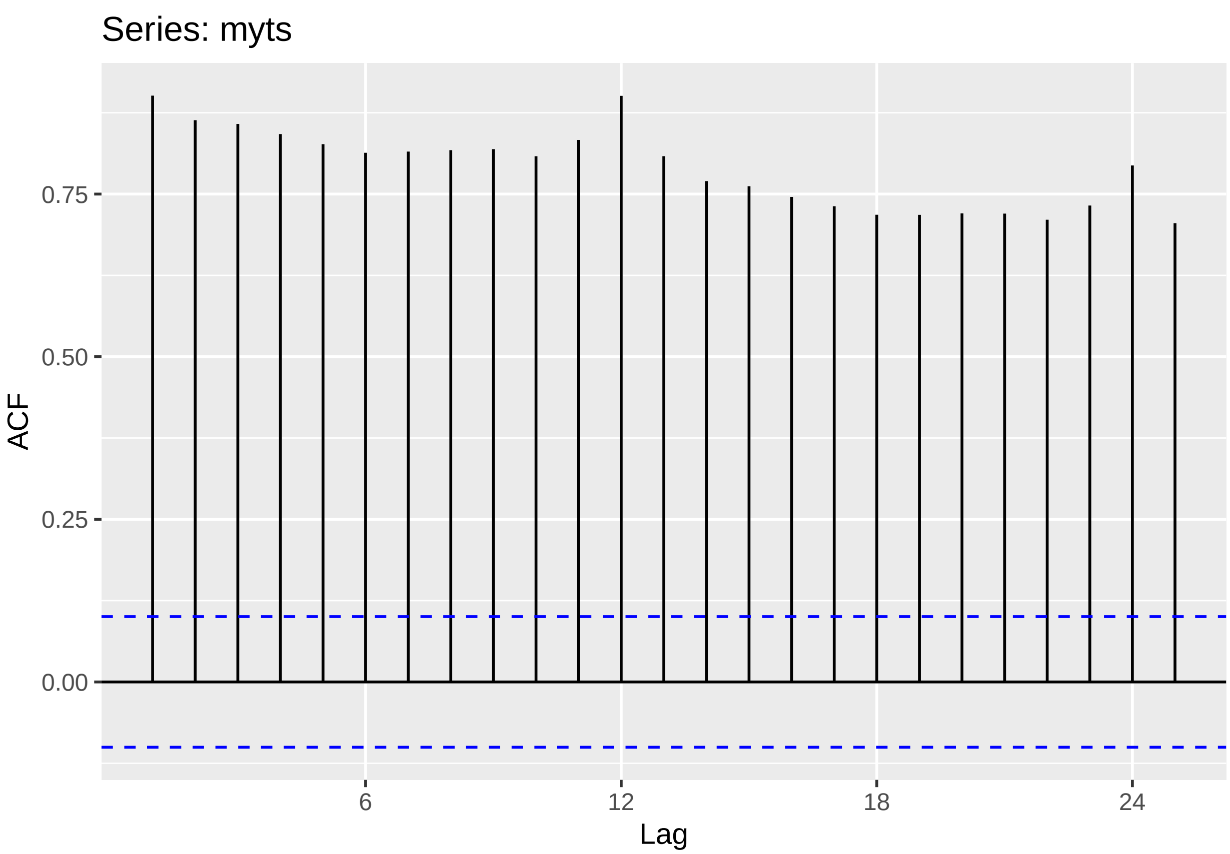
```r
ggsubseriesplot(myts)
```



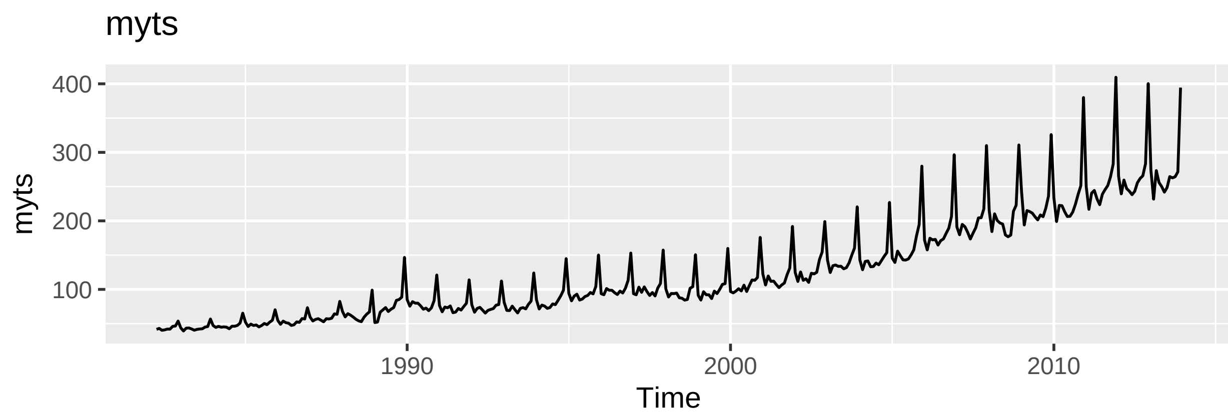

```r
gglagplot(myts)
```

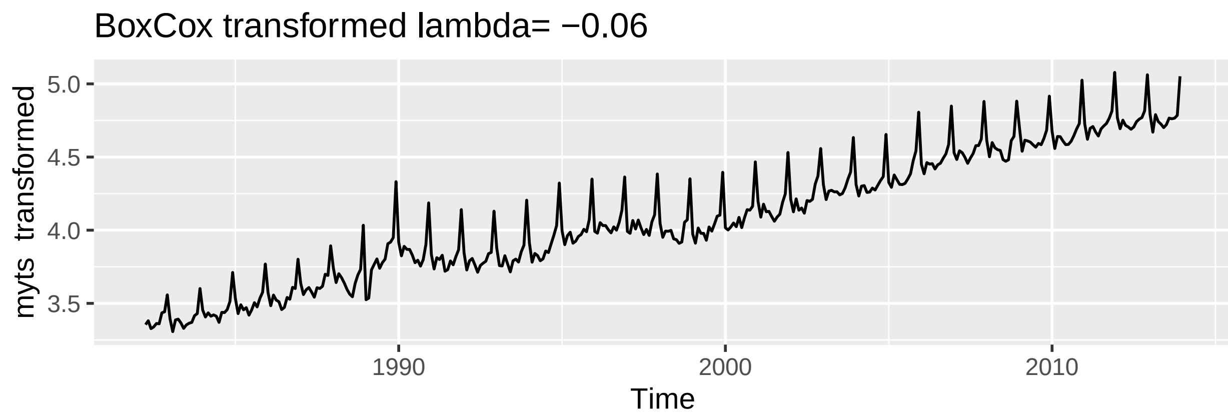

lag 1
lag 2
lag 3
lag 4
lag 5
lag 6
lag 7
lag 8
lag 9
lag 10
lag 11
lag 12
lag 13
lag 14
lag 15
lag 16
400
300
200
100
100 200 300 400
Month
1
2
3
4
5
6
7
8
9
10
11
12

```
ggAcf(myts)
```


Series: myts
ACF
0.75
0.50
0.25
0.00
6
12
18
24
Lag

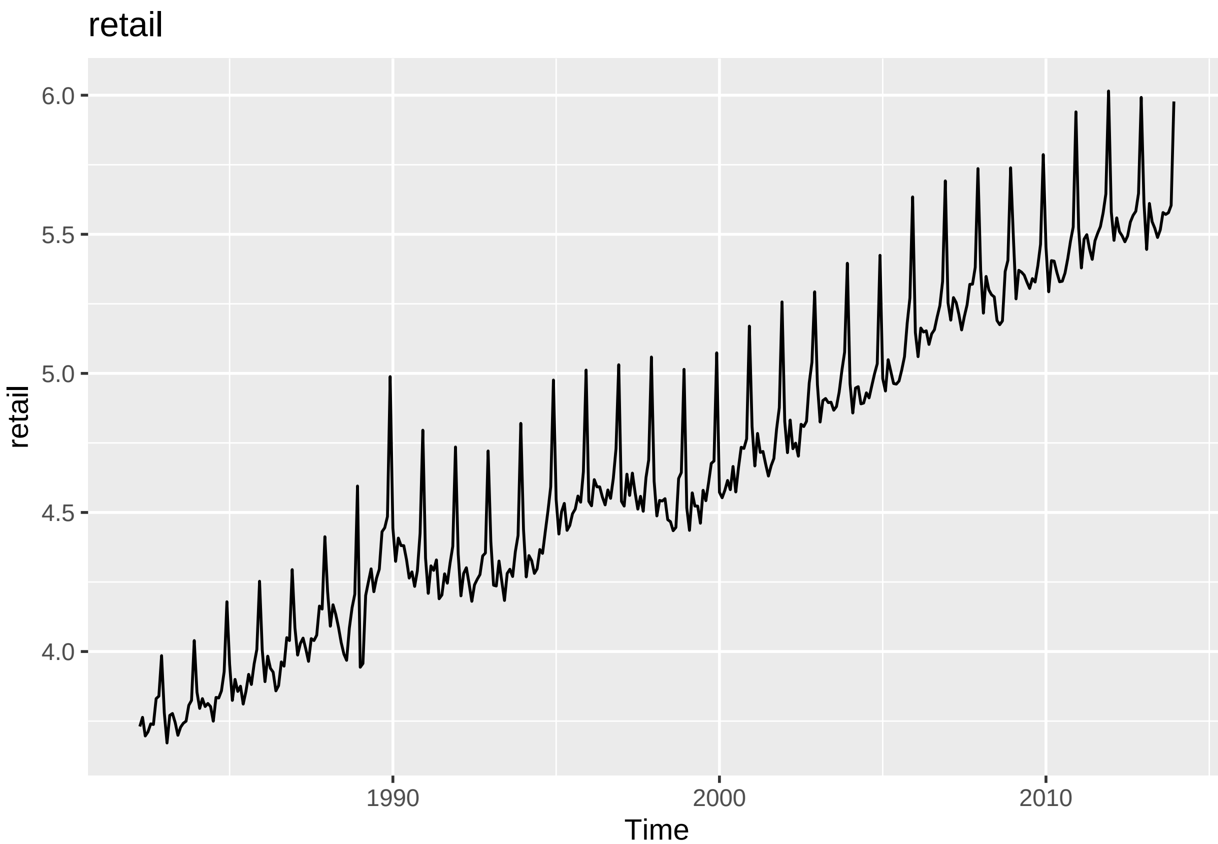
```r
plot_timeseries(myts)
```





Now the best lambda chosen is ~0 so BoxCox transformation would be log transformation.

```r
autoplot(log(myts)) +
    ggtitle("retail") +
    xlab("Time") +
    ylab("retail")
```



3.8

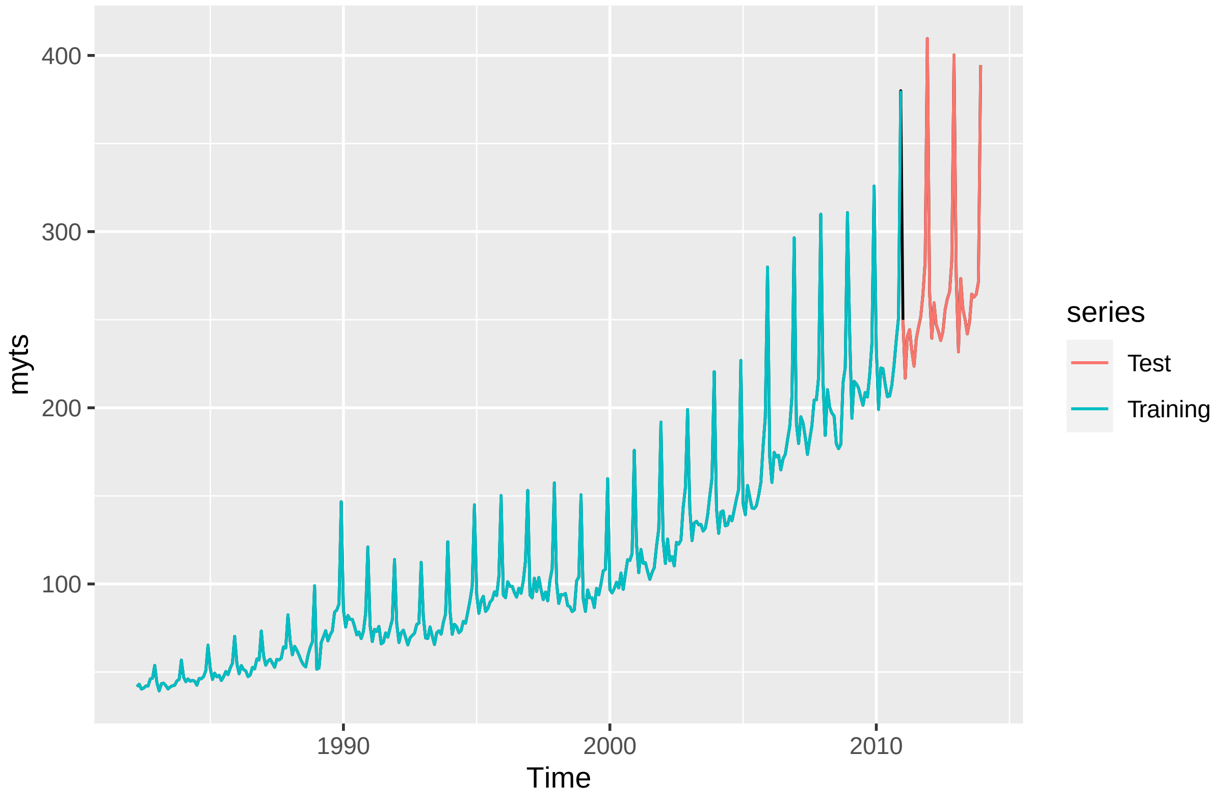
For your retail time series (from Exercise 3 in Section 2.10):

a. Split the data into two parts using.

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

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

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



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

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

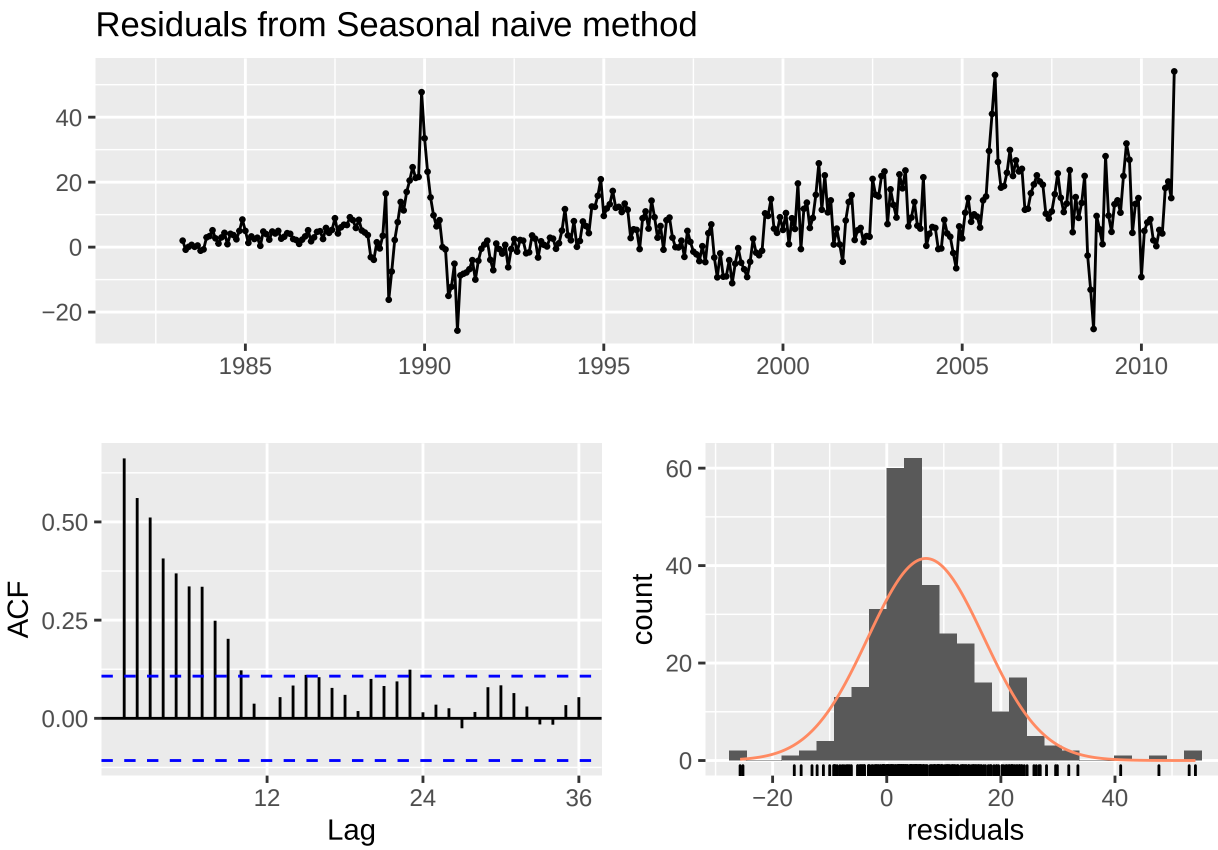
d. Compare the accuracy of your forecasts against the actual values stored in myts.test.

```
accuracy(fc,myts.test)
```

```
##                      ME     RMSE       MAE       MPE      MAPE    MASE      ACF1
## Training set  6.870871 12.27525  8.893093  5.476112  7.780981 1.00000 0.6617306
## Test set     28.400000 29.39091 28.400000 11.015822 11.015822 3.19349 0.5697915
##              Theil's U
## Training set        NA
## Test set     0.7493485
```

e. Check the residuals.

```
checkresiduals(fc)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 591.71, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

Do the residuals appear to be uncorrelated and normally distributed?

Based on the plots shown above, the residuals seems to be normally distributed with slightly right skewed. The ACF plot shows significant correlations between time lags of residuals. The mean of the residuals is not centered around 0 thats shows bias in forecast.

f. How sensitive are the accuracy measures to the training/test split

Accuracy measures are very sensitive to split. It is shows below for different years to split the data.

```
# function to get accuracy based on year
cal_acc <- function(split_yr){
  train <- window(myts, end=c(split_yr, 12))
  test <- window(myts, start=split_yr+1)
  acc <- accuracy(snaive(train), test)
  return(acc)
}

# splits
splits <- c(2000:2011)
```

```r
# loop
for (year in splits){
  acc <- cal_acc(year)
  print(acc)
}
```

```
##                    ME      RMSE       MAE       MPE     MAPE     MASE      ACF1
## Training set  3.72770  8.859789  6.289202  4.556147  7.81053 1.000000 0.7112533
## Test set     15.71667 18.766504 16.091667 12.160534 12.50363 2.558618 0.3657062
##              Theil's U
## Training set        NA
## Test set      0.928694
##                     ME      RMSE       MAE       MPE      MAPE     MASE
## Training set  4.086222  9.164725  6.551111  4.763703  7.881116 1.000000
## Test set     17.500000 19.939931 17.500000 12.495470 12.495470 2.671303
##                   ACF1 Theil's U
## Training set 0.7044425        NA
## Test set     0.6914277  1.100219
##                     ME      RMSE       MAE       MPE      MAPE     MASE
## Training set  4.412658  9.414384  6.752743  4.922626  7.882196 1.000000
## Test set     15.162500 16.837223 15.229167 10.496018 10.539392 2.255256
##                    ACF1 Theil's U
## Training set 0.70455284        NA
## Test set     0.03199491 0.7603186
##                     ME     RMSE       MAE      MPE     MAPE     MASE      ACF1
## Training set  4.871486  9.77985  7.098795 5.155021 7.971960 1.000000 0.6938236
## Test set     11.429167 17.92597 12.204167 6.557991 7.074205 1.719188 0.5793994
##              Theil's U
## Training set        NA
## Test set     0.6546334
##                     ME      RMSE       MAE       MPE      MAPE     MASE
## Training set  4.760536  9.607498  6.956705  4.994935  7.729828 1.000000
## Test set     28.441667 33.028674 28.441667 15.259542 15.259542 4.088382
##                   ACF1 Theil's U
## Training set 0.6834907        NA
## Test set     0.6836064  1.140126
##                     ME     RMSE      MAE       MPE      MAPE     MASE      ACF1
## Training set  5.339927 10.57731  7.43956  5.208671  7.823349 1.000000 0.6817184
## Test set     28.887500 30.62823 28.88750 15.277977 15.277977 3.882958 0.6049068
##              Theil's U
## Training set        NA
## Test set      1.037071
##                     ME     RMSE       MAE      MPE     MAPE     MASE      ACF1
## Training set  5.999298 11.27013  8.010526 5.484696 7.989282 1.000000 0.7405585
## Test set     18.412500 20.97696 18.620833 9.078205 9.194463 2.324546 0.3839928
##              Theil's U
## Training set        NA
## Test set     0.6591721
##                     ME    RMSE       MAE      MPE     MAPE     MASE      ACF1
## Training set  6.394276 11.5251  8.324242 5.586108 7.989499 1.000000 0.7499074
## Test set     13.362500 20.0530 16.770833 5.995657 7.905696 2.014698 0.2962003
##              Theil's U
## Training set        NA
## Test set      0.603503
```

```
##                     ME     RMSE      MAE      MPE     MAPE     MASE      ACF1
## Training set  6.350809 11.65192  8.47055 5.459058 7.917465 1.000000 0.7208639
## Test set     21.654167 25.74213 21.65417 9.257211 9.257211 2.556406 0.3224365
##              Theil's U
## Training set        NA
## Test set     0.7033985
##                     ME     RMSE       MAE      MPE     MAPE     MASE      ACF1
## Training set  6.718069 11.96976  8.758567 5.530198 7.896703 1.000000 0.7058969
## Test set     23.020833 31.12061 23.787500 8.538663 8.866719 2.715912 0.4686515
##              Theil's U
## Training set        NA
## Test set     0.7299163
##                     ME     RMSE       MAE       MPE      MAPE    MASE      ACF1
## Training set  6.870871 12.27525  8.893093  5.476112  7.780981 1.00000 0.6617306
## Test set     28.400000 29.39091 28.400000 11.015822 11.015822 3.19349 0.5697915
##              Theil's U
## Training set        NA
## Test set     0.7493485
##                    ME     RMSE       MAE      MPE     MAPE     MASE      ACF1
## Training set 7.471014 12.92370  9.422899 5.612836 7.837536 1.000000 0.6958216
## Test set     9.825000 14.96852 12.808333 4.008724 4.871206 1.359277 0.3652316
##              Theil's U
## Training set        NA
## Test set     0.3676318
```