DATA624-HW2-Forecasting

FPP-Hyndman exercises 3.1, 3.2, 3.3 and 3.8

Michael Y.

2/16/2020

Contents

```r
library(fpp2)
```

```
## Loading required package: ggplot2

## Loading required package: forecast

## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

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

## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.fracdiff    fracdiff
##   residuals.fracdiff fracdiff

## Loading required package: fma

## Loading required package: expsmooth
```

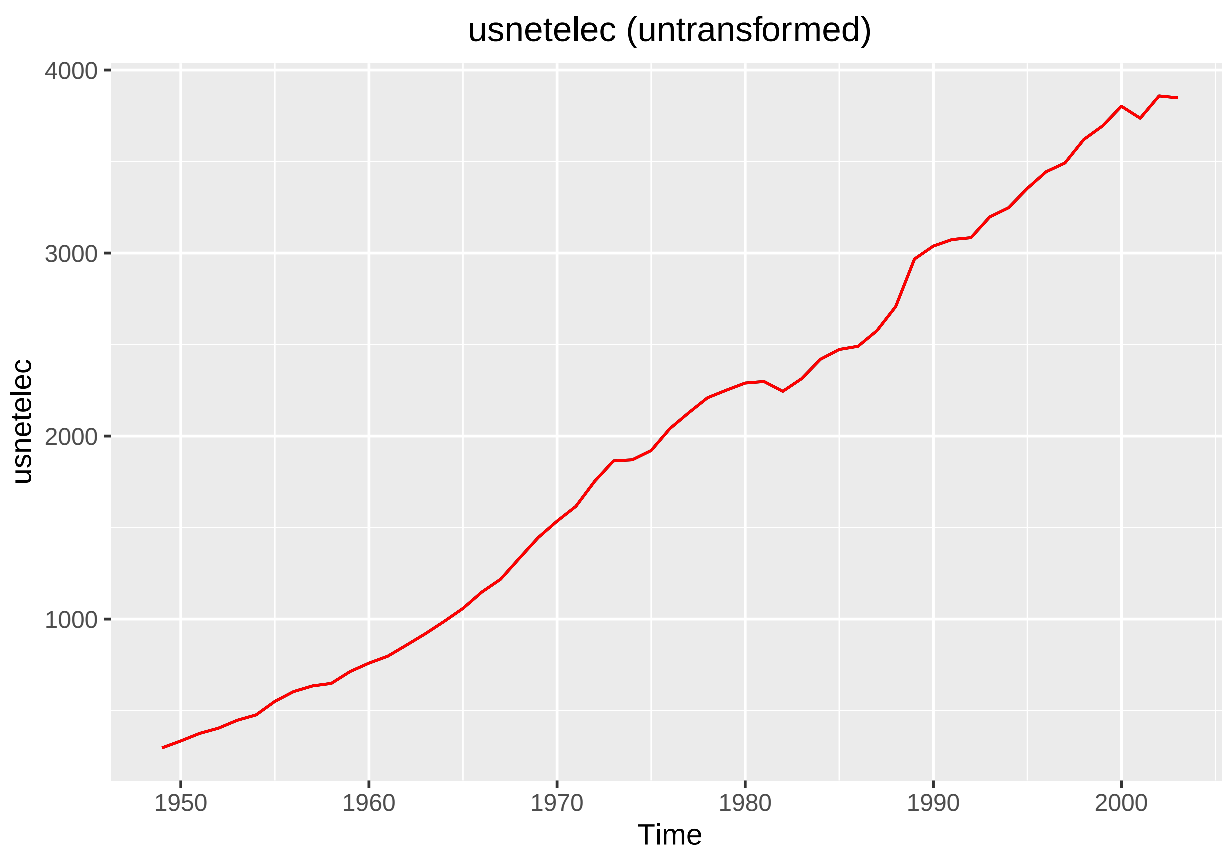
Homework 2 - Forecasting

Do exercises 3.1, 3.2, 3.3 and 3.8 from the online Hyndman book. Please submit both your Rpubs link as well as attach the .rmd file with your code.

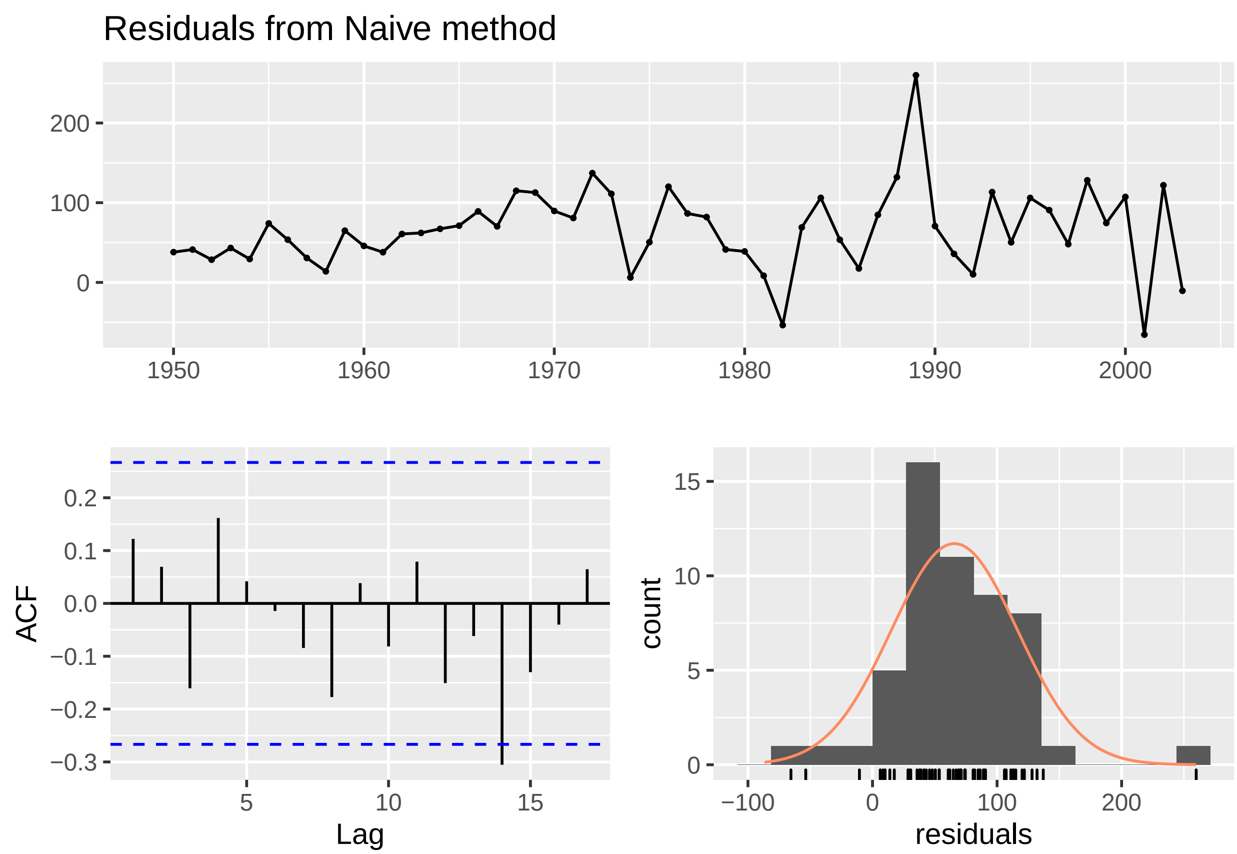
3.1 For the following series, find an appropriate Box-Cox transformation in order to stabilise the variance.

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

```
### plot raw data series
autoplot(usnetelec) +
  ggtitle(paste("usnetelec (untransformed)")) +
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="red")
```




```
usnetelec.ljung <- checkresiduals(naive(usnetelec))
```



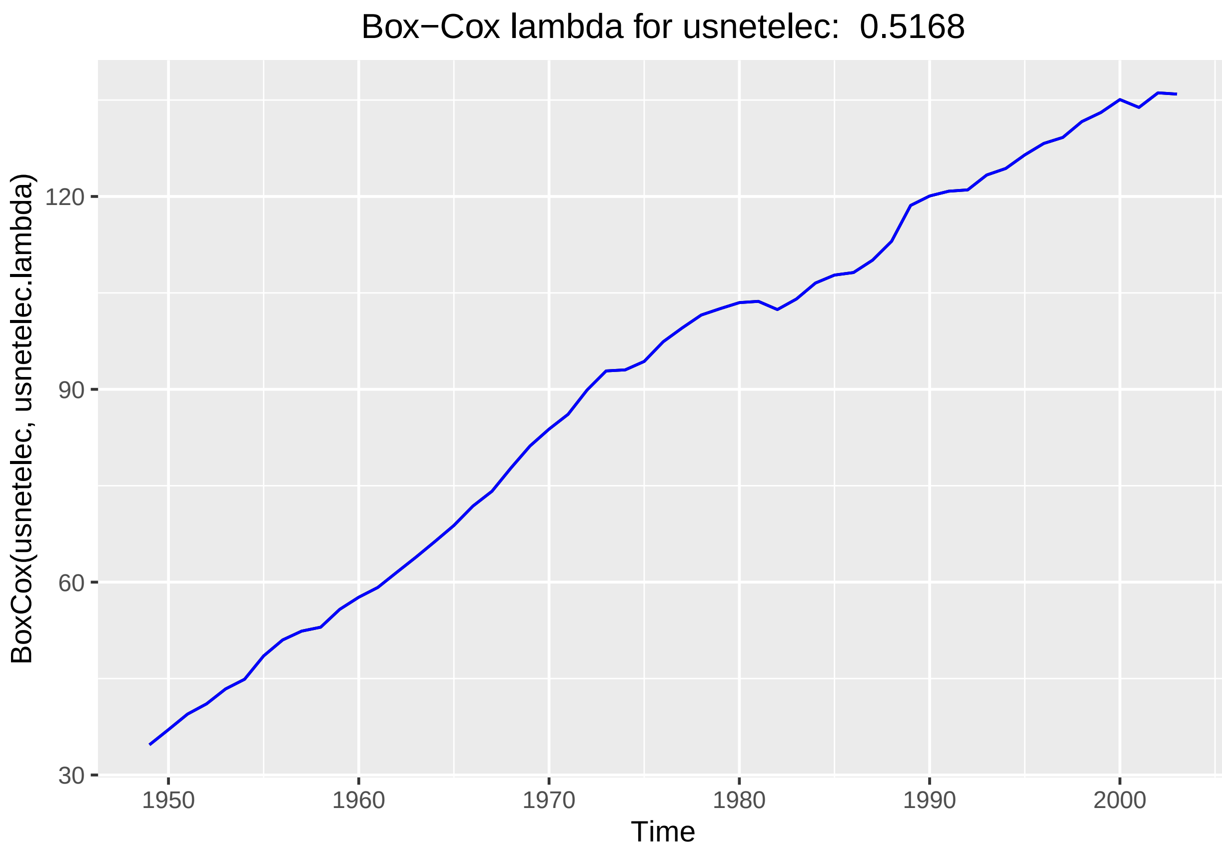

```
##
##  Ljung-Box test
##
## data:  Residuals from Naive method
## Q* = 7.4406, df = 10, p-value = 0.6833
##
## Model df: 0.   Total lags used: 10
```

```r
#usnetelec.ljung

if (usnetelec.ljung$p.value > 0.05) {
  print("Because the p-value on the Ljung-Box test is large,
        the Box-Cox transform is not necessary, but here goes:")
}  else  {
    print("Because the p-value on the Ljung-Box test is small,
          we'll try Box-Cox transform to see if we can achieve constant variance")}
```

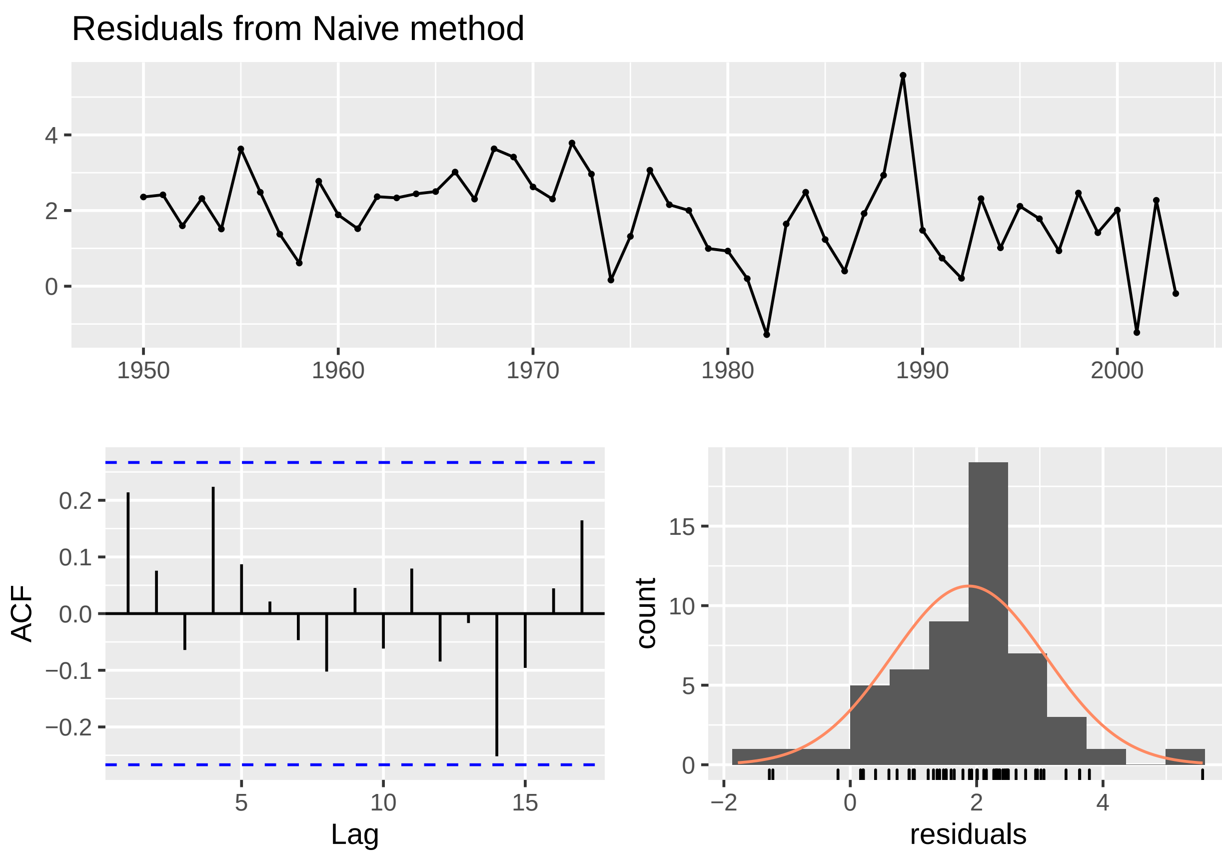
```
## [1] "Because the p-value on the Ljung-Box test is large, \n        the Box-Cox transform is not nece
```

```r
### Box-Cox transform
usnetelec.lambda <- BoxCox.lambda(usnetelec)
### Plot transformed series
#print(paste("Box-Cox lambda for usnetelec: ", round(usnetelec.lambda,3)))
autoplot(BoxCox(usnetelec, usnetelec.lambda)) +
  ggtitle(paste("Box-Cox lambda for usnetelec: ", round(usnetelec.lambda,4))) +
```

```
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="blue")
```




```
usnetelec.xform.ljung <- checkresiduals(naive(BoxCox(usnetelec, usnetelec.lambda)))
```

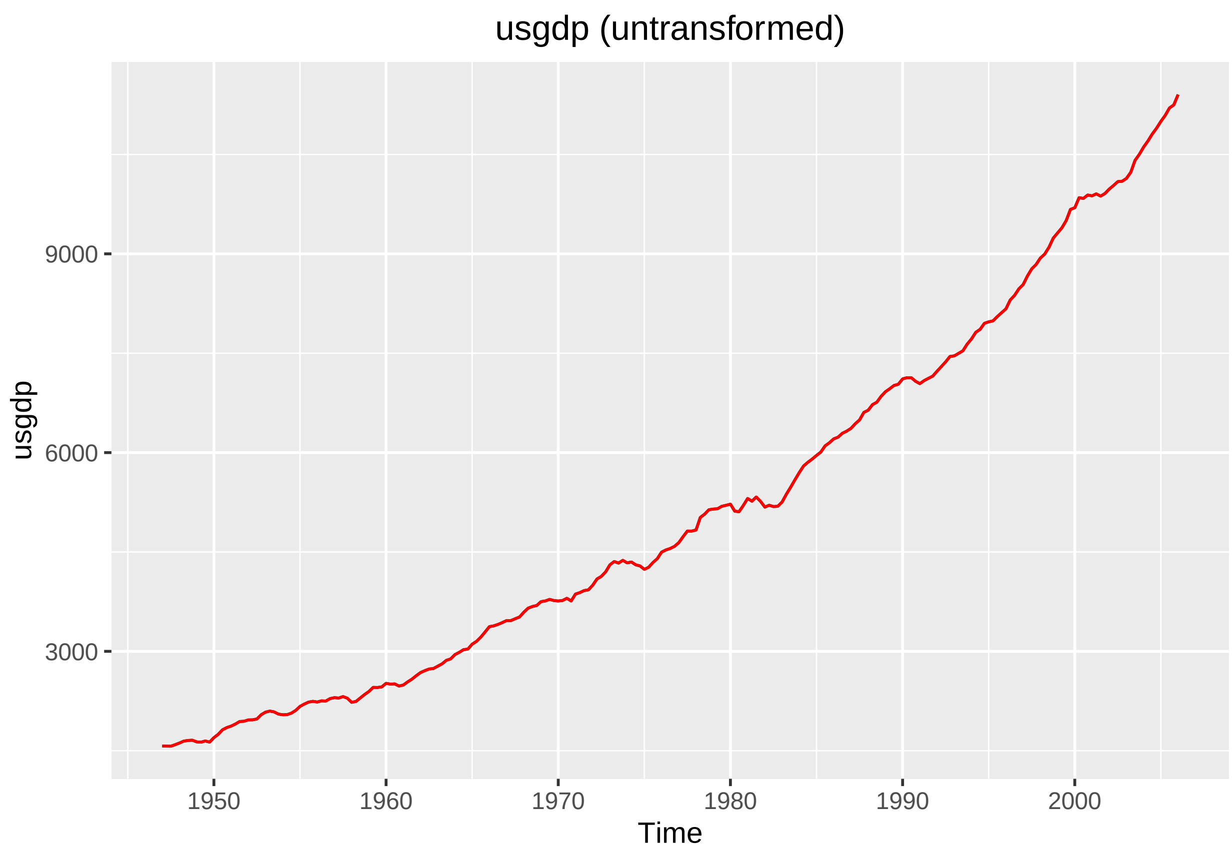



```
##
##  Ljung-Box test
##
## data:  Residuals from Naive method
## Q* = 7.9451, df = 10, p-value = 0.6342
##
## Model df: 0.   Total lags used: 10
```

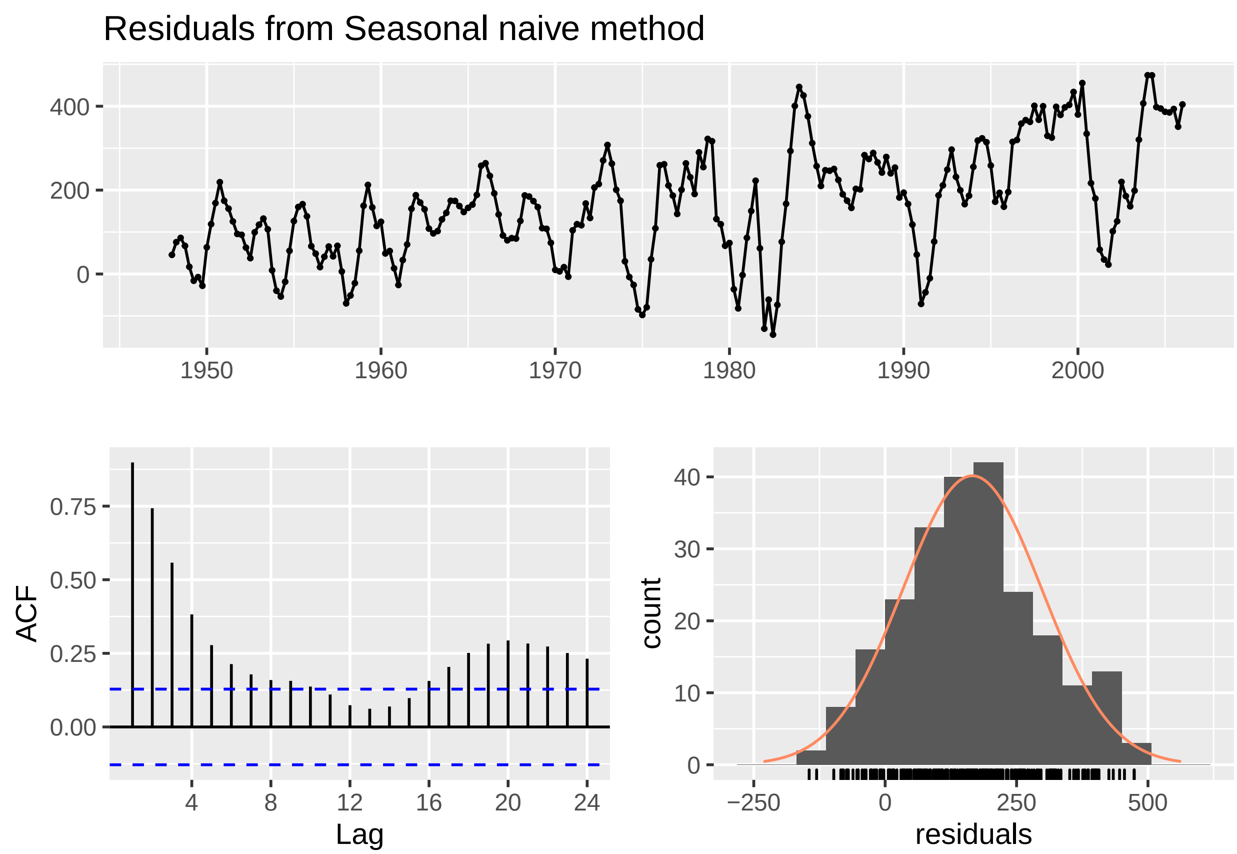
```
#usnetelec.xform.ljung
```

usgdp : Quarterly US GDP. 1947:1 - 2006.1

```r
### plot raw data series
autoplot(usgdp) +
  ggtitle(paste("usgdp (untransformed)")) +
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="red")
```




```r
usgdp.ljung <- checkresiduals(snaive(usgdp))
```



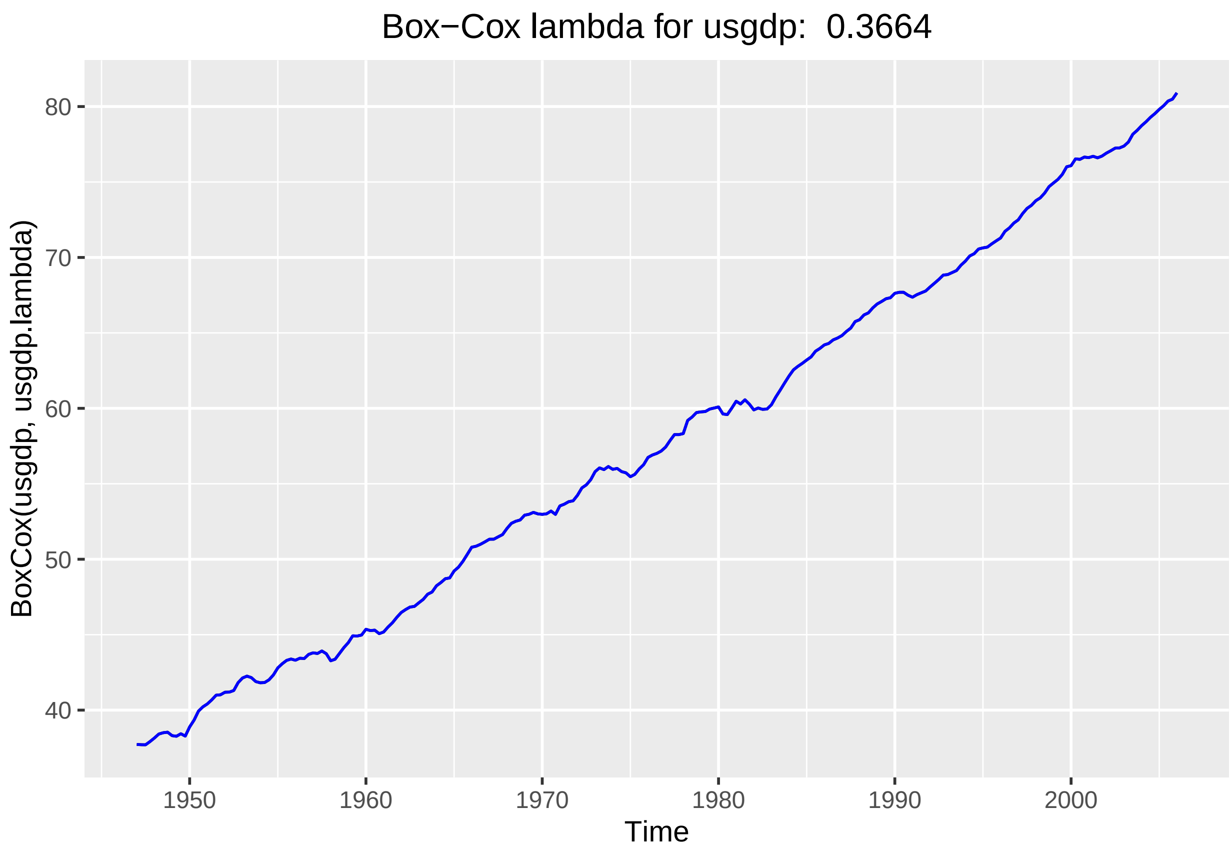

```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 473.66, df = 8, p-value < 0.00000000000000022
##
## Model df: 0.   Total lags used: 8
```

```
#usgdp.ljung

if (usgdp.ljung$p.value > 0.05) {
  print("Because the p-value on the Ljung-Box test is large,
        the Box-Cox transform is not necessary, but here goes:")
}  else  {
    print("Because the p-value on the Ljung-Box test is small,
          we'll try Box-Cox transform to see if we can achieve constant variance")}
```

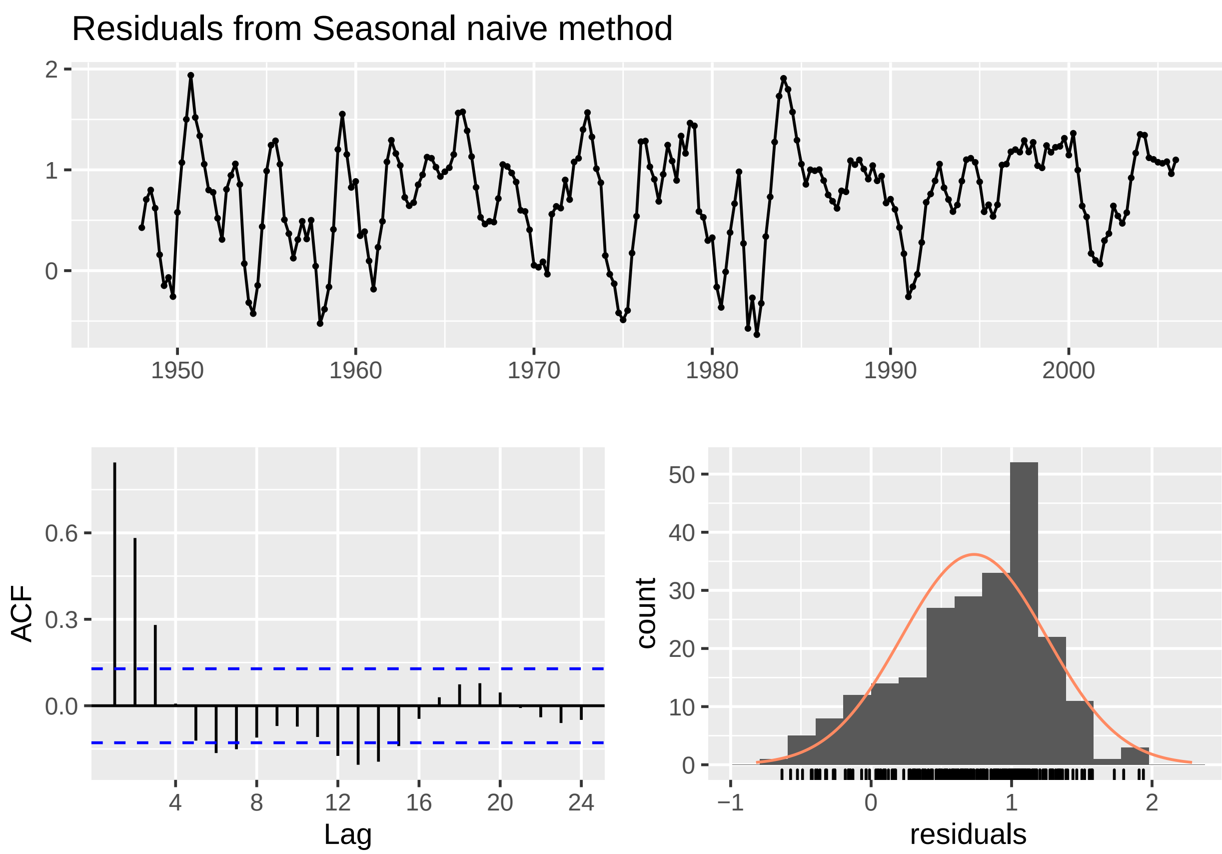
```
## [1] "Because the p-value on the Ljung-Box test is small, \n          we'll try Box-Cox transform to s
```

```
### Box-Cox transform
usgdp.lambda <- BoxCox.lambda(usgdp)
### Plot transformed series
#print(paste("Box-Cox lambda for usgdp: ", round(usgdp.lambda,3)))
autoplot(BoxCox(usgdp, usgdp.lambda)) +
  ggtitle(paste("Box-Cox lambda for usgdp: ", round(usgdp.lambda,4))) +
```

```
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="blue")
```




```
usgdp.xform.ljung <- checkresiduals(snaive(BoxCox(usgdp, usgdp.lambda)))
```

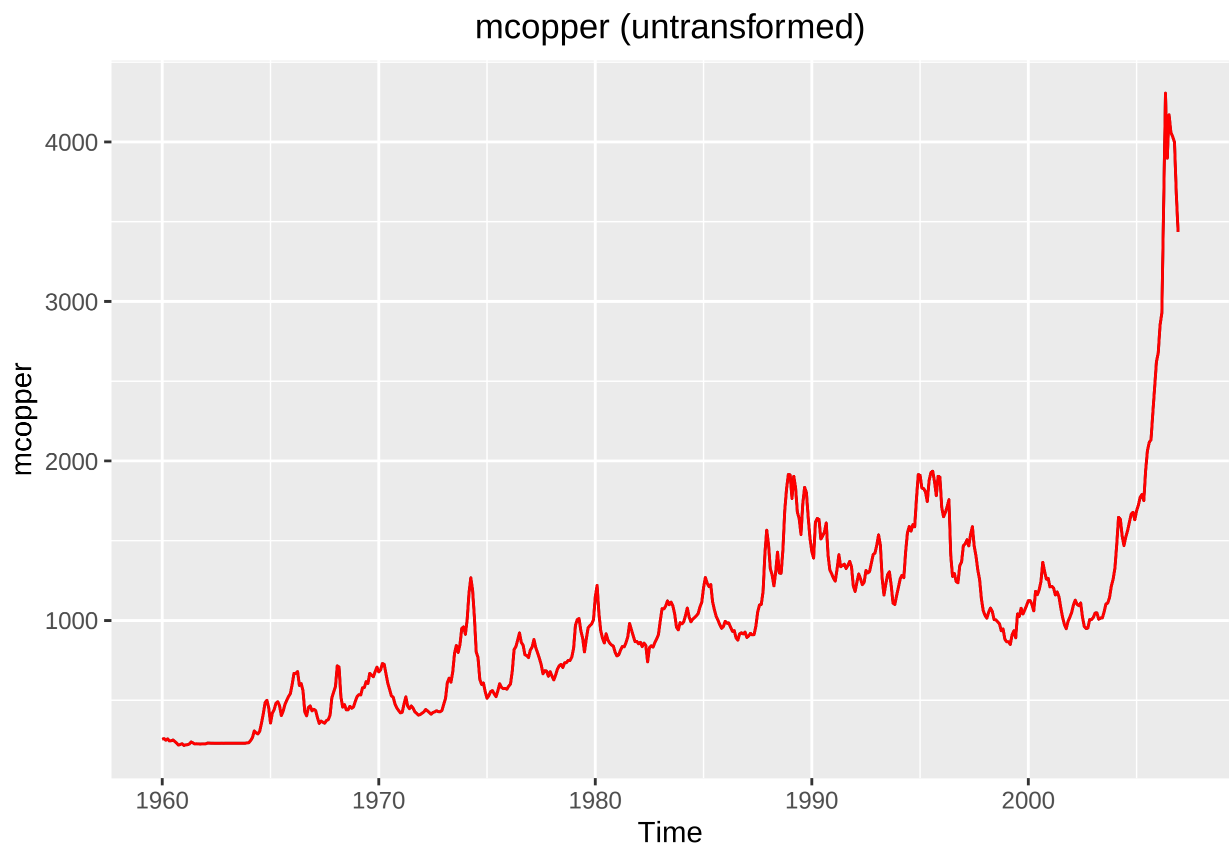



```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 285.76, df = 8, p-value < 0.00000000000000022
##
## Model df: 0.   Total lags used: 8
```

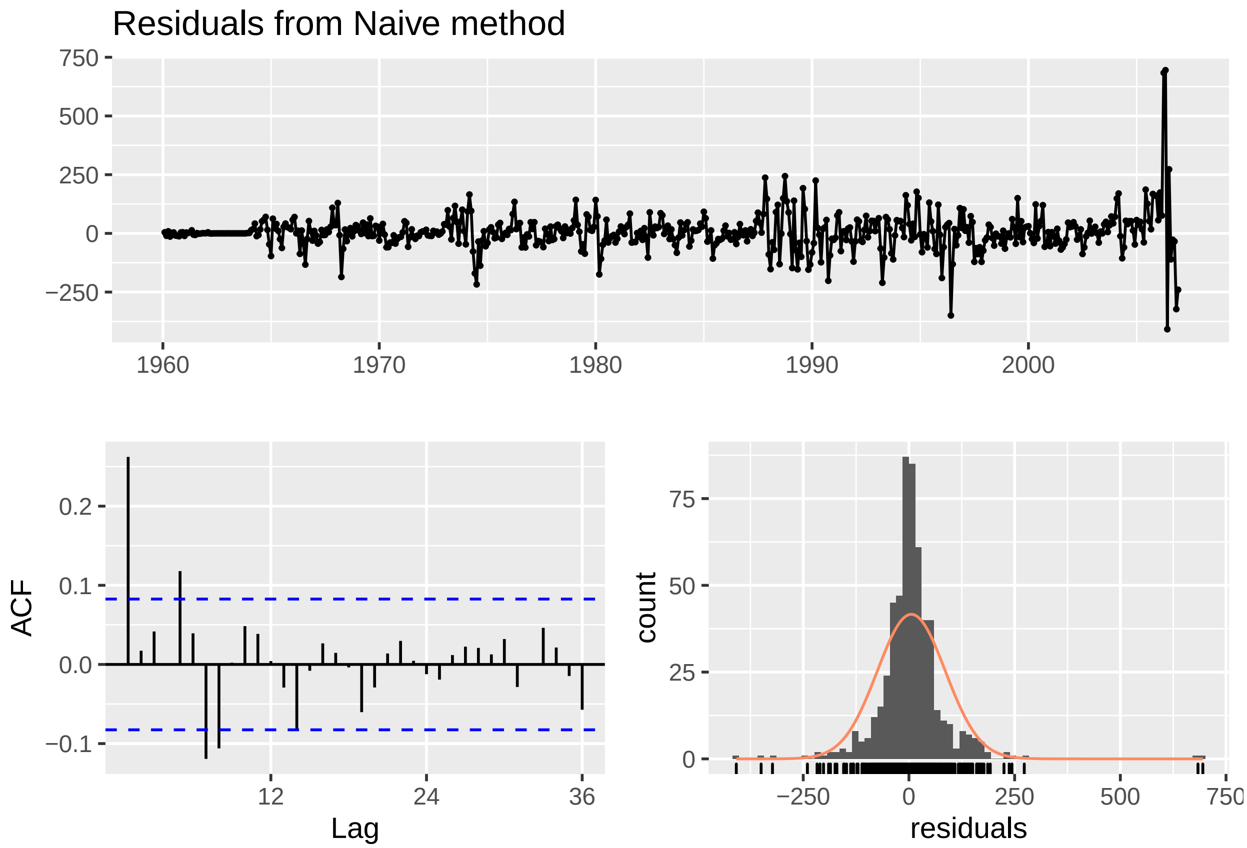
```
#usgdp.xform.ljung
```

mcopper : Monthly copper prices. Copper, grade A, electrolytic wire bars/cathodes,LME,cash (pounds/ton)

```
### plot raw data series
autoplot(mcopper) +
  ggtitle(paste("mcopper (untransformed)")) +
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="red")
```




```
mcopper.ljung <- checkresiduals(naive(mcopper))
```



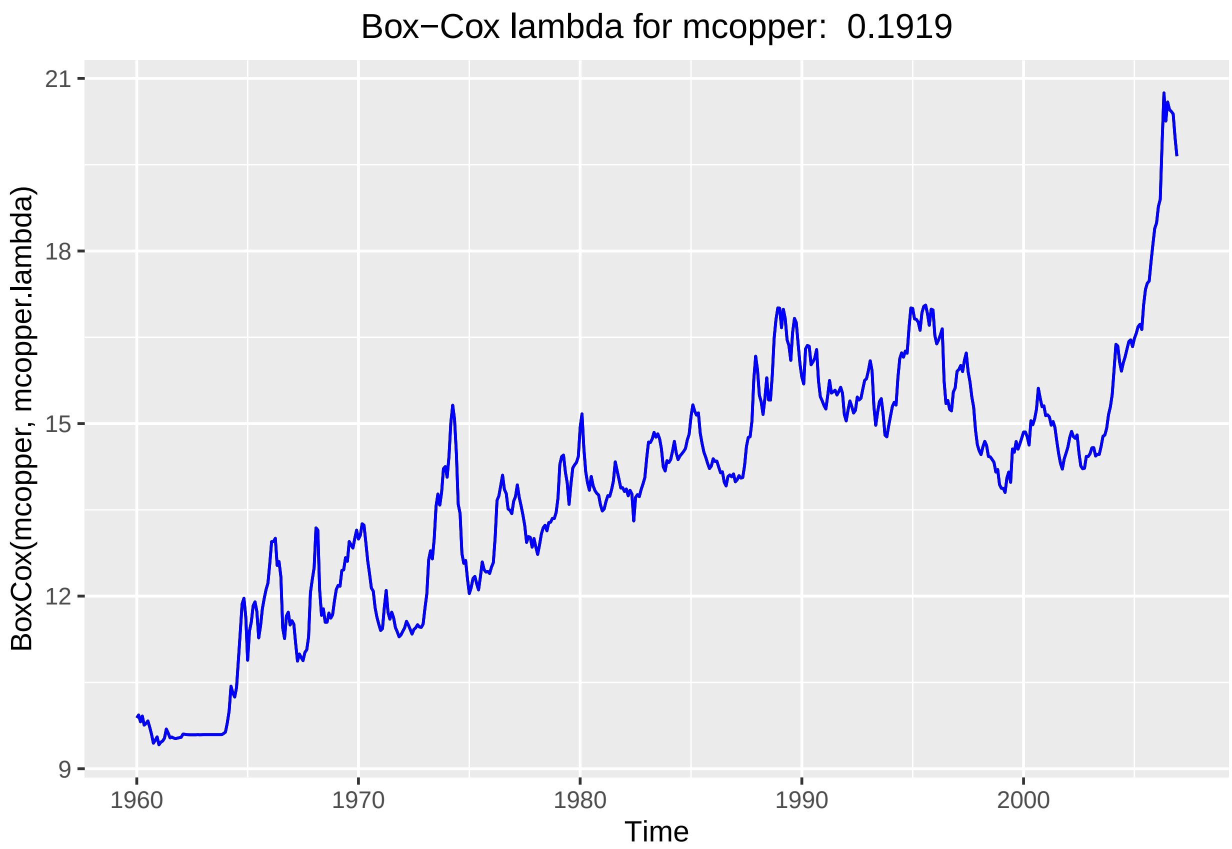

```
##
##  Ljung-Box test
##
## data:  Residuals from Naive method
## Q* = 74.063, df = 24, p-value = 0.0000005216
##
## Model df: 0.   Total lags used: 24
```

```
#mcopper.ljung

if (mcopper.ljung$p.value > 0.05) {
  print("Because the p-value on the Ljung-Box test is large,
        the Box-Cox transform is not necessary, but here goes:")
}  else  {
    print("Because the p-value on the Ljung-Box test is small,
          we'll try Box-Cox transform to see if we can achieve constant variance")}
```

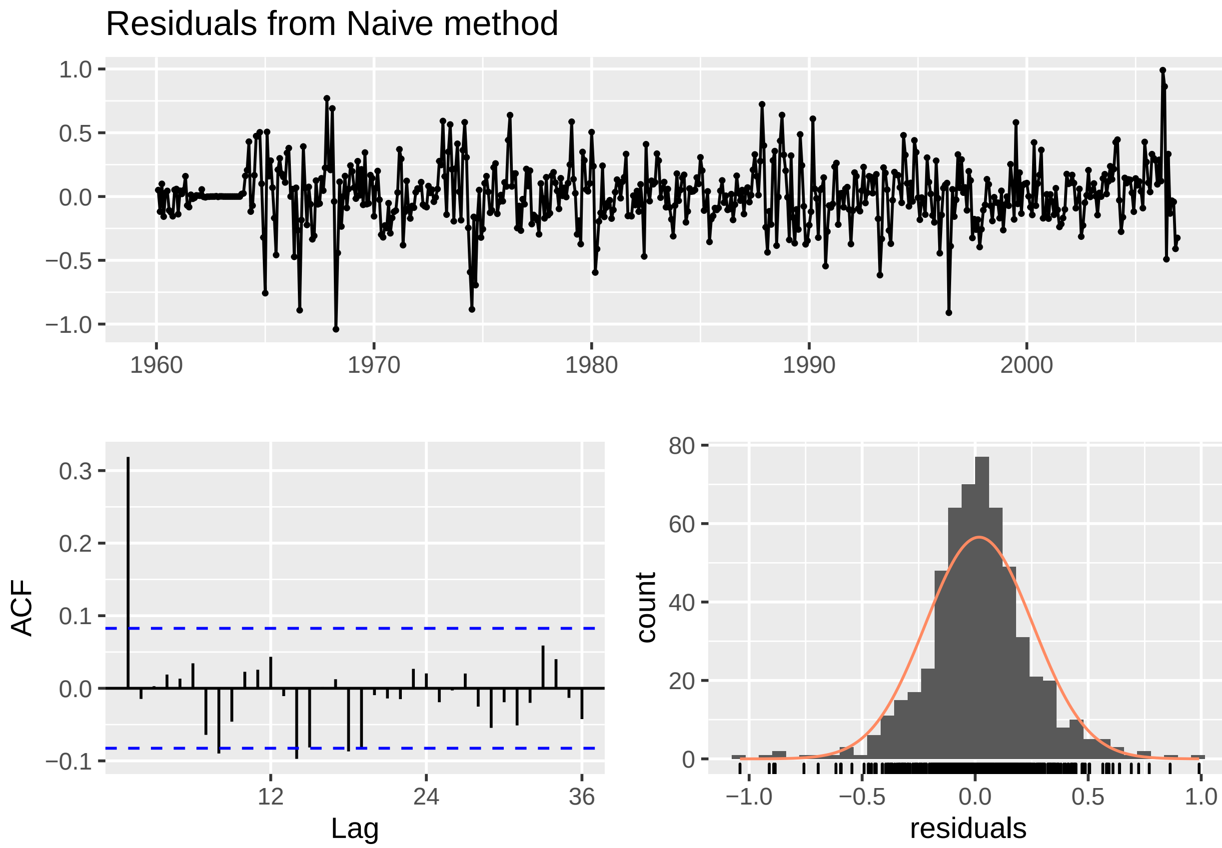
```
## [1] "Because the p-value on the Ljung-Box test is small, \n          we'll try Box-Cox transform to s
```

```
### Box-Cox transform
mcopper.lambda <- BoxCox.lambda(mcopper)
### Plot transformed series
#print(paste("Box-Cox lambda for mcopper: ", round(mcopper.lambda,3)))
autoplot(BoxCox(mcopper, mcopper.lambda)) +
  ggtitle(paste("Box-Cox lambda for mcopper: ", round(mcopper.lambda,4))) +
```

```
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="blue")
```




```
mcopper.xform.ljung <- checkresiduals(naive(BoxCox(mcopper, mcopper.lambda)))
```

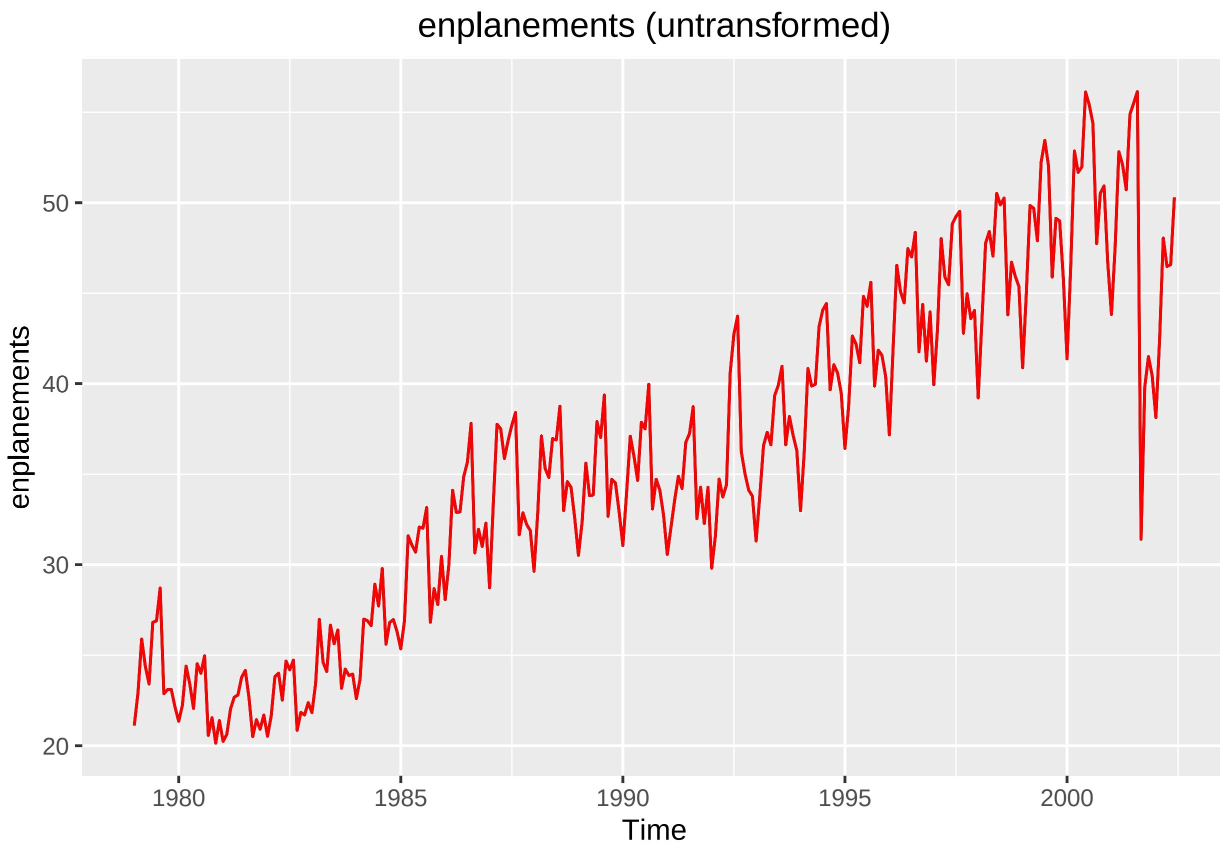



```
##
##  Ljung-Box test
##
## data:  Residuals from Naive method
## Q* = 87.597, df = 24, p-value = 0.000000003588
##
## Model df: 0.   Total lags used: 24
```

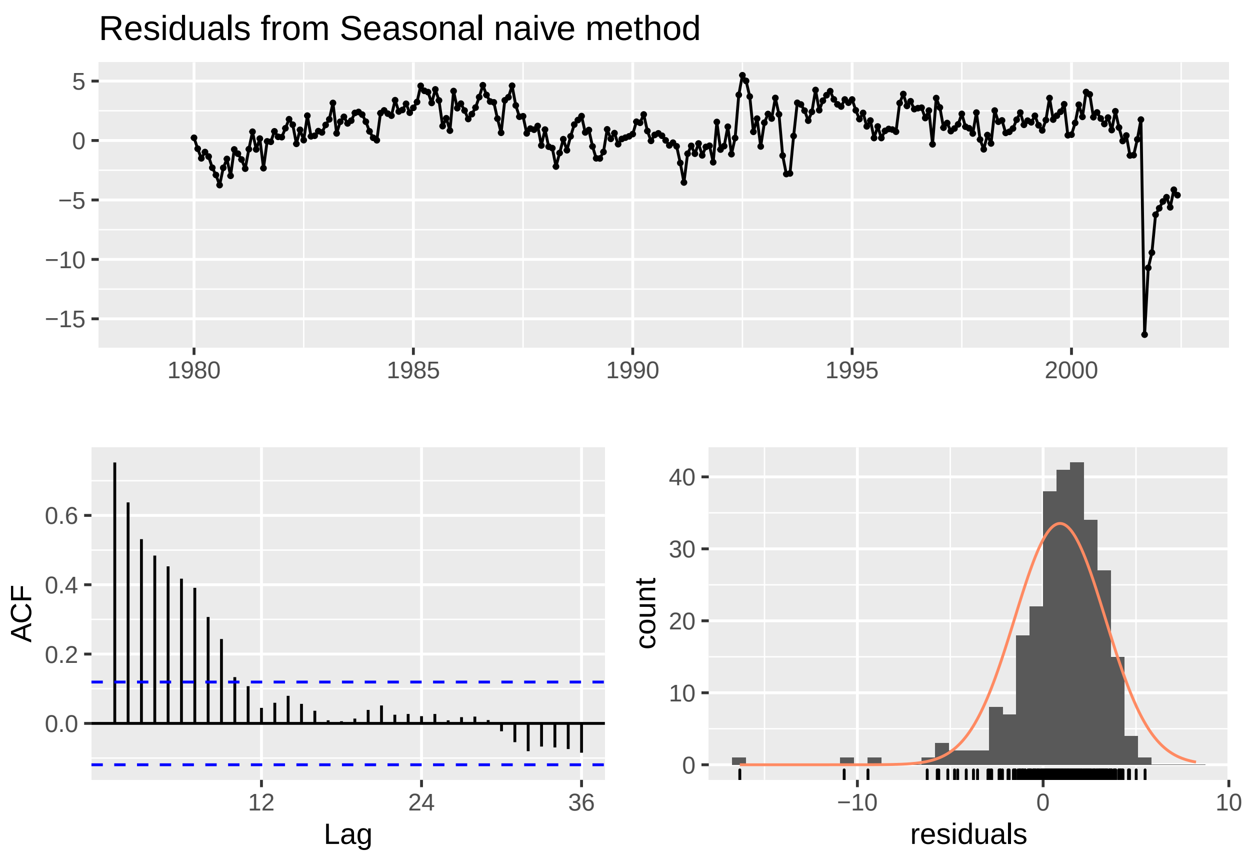
```
#mcopper.xform.ljung
```

`enplanements` : Monthly US domestic enplanements (millions): 1996-2000

```
### plot raw data series
autoplot(enplanements) +
  ggtitle(paste("enplanements (untransformed)")) +
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="red")
```




```
enplanements.ljung <- checkresiduals(snaive(enplanements))
```



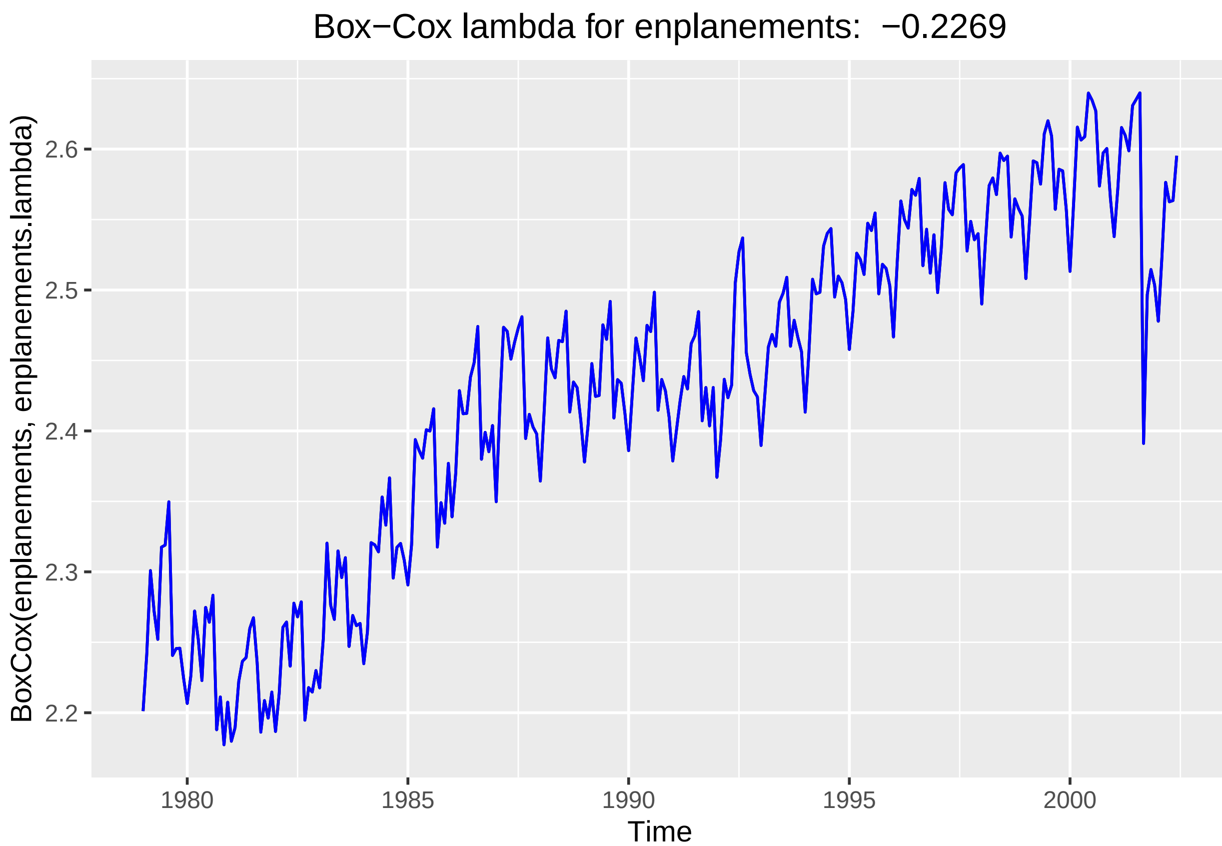

```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 614.48, df = 24, p-value < 0.00000000000000022
##
## Model df: 0.   Total lags used: 24
```

```
#enplanements.ljung

if (enplanements.ljung$p.value > 0.05) {
  print("Because the p-value on the Ljung-Box test is large,
        the Box-Cox transform is not necessary, but here goes:")
}  else  {
    print("Because the p-value on the Ljung-Box test is small,
          we'll try Box-Cox transform to see if we can achieve constant variance")}
```

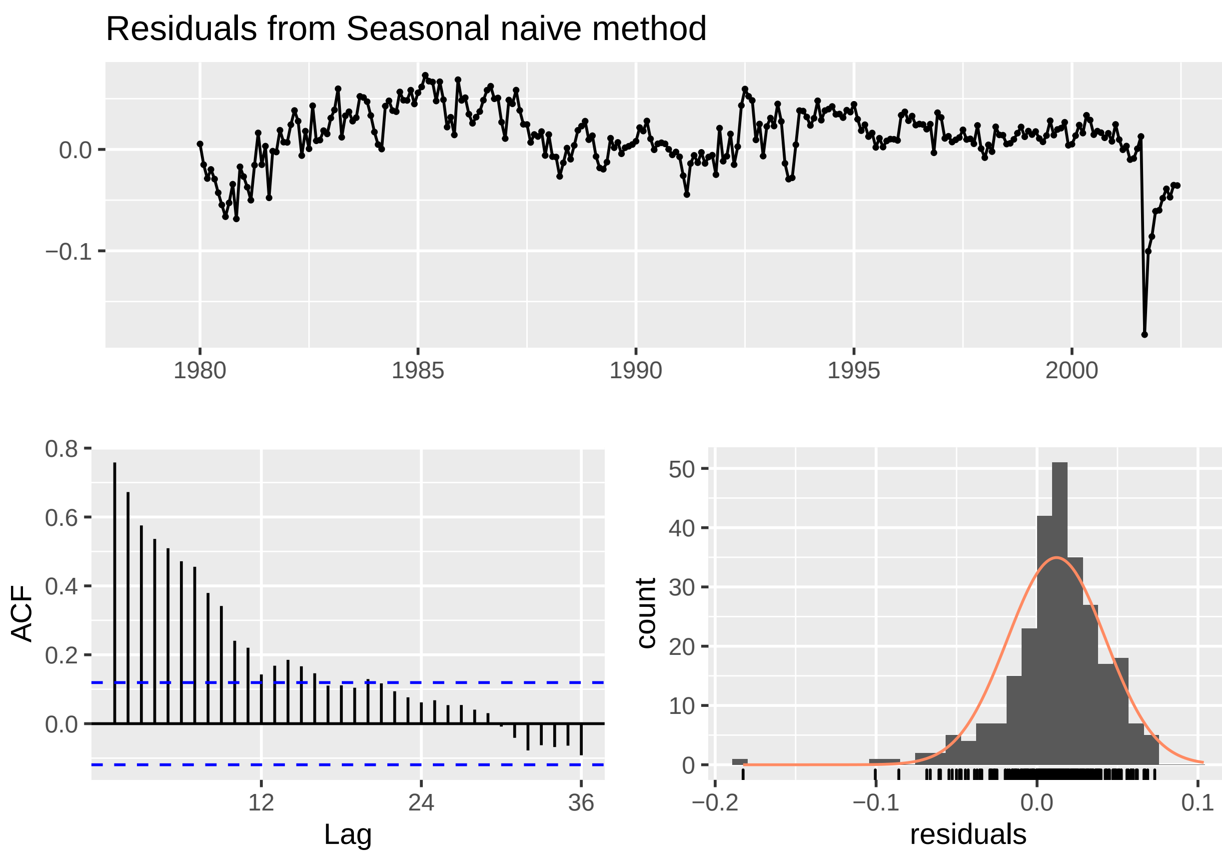
```
## [1] "Because the p-value on the Ljung-Box test is small, \n          we'll try Box-Cox transform to
```

```
### Box-Cox transform
enplanements.lambda <- BoxCox.lambda(enplanements)
### Plot transformed series
#print(paste("Box-Cox lambda for enplanements: ", round(enplanements.lambda,3)))
autoplot(BoxCox(enplanements, enplanements.lambda)) +
  ggtitle(paste("Box-Cox lambda for enplanements: ", round(enplanements.lambda,4))) +
```

```r
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="blue")
```




```r
enplanements.xform.ljung <- checkresiduals(snaive(BoxCox(enplanements, enplanements.lambda)))
```



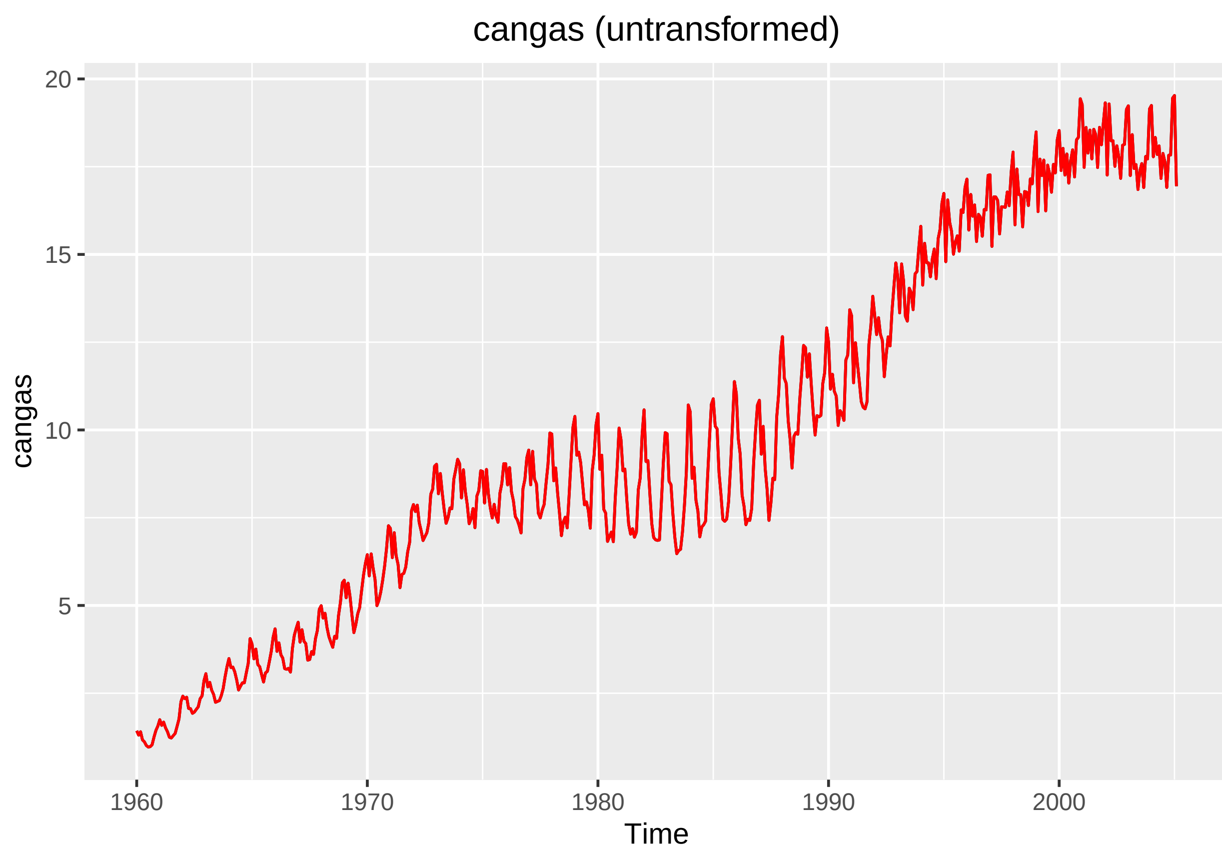

```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 809.1, df = 24, p-value < 0.00000000000000022
##
## Model df: 0.   Total lags used: 24
```

```
#enplanements.xform.ljung
```

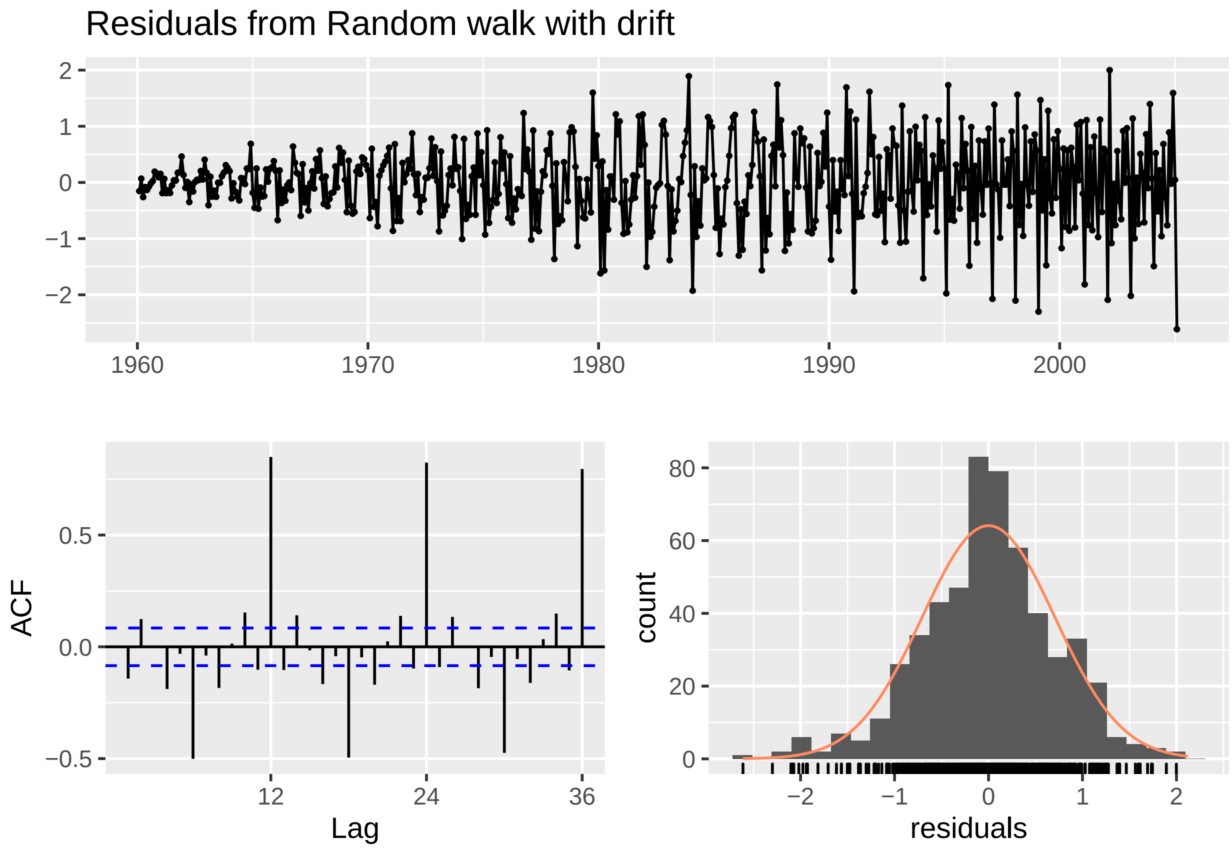
3.2 Why is a Box-Cox transformation unhelpful for the cangas data?

cangas : Monthly Canadian gas production, billions of cubic metres, January 1960 - February 2005

```
### plot raw data series
autoplot(cangas) +
  ggtitle(paste("cangas (untransformed)")) +
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="red")
```




```
cangas.ljung <- checkresiduals(rwf(cangas,h=2*frequency(cangas),drift=T))
```



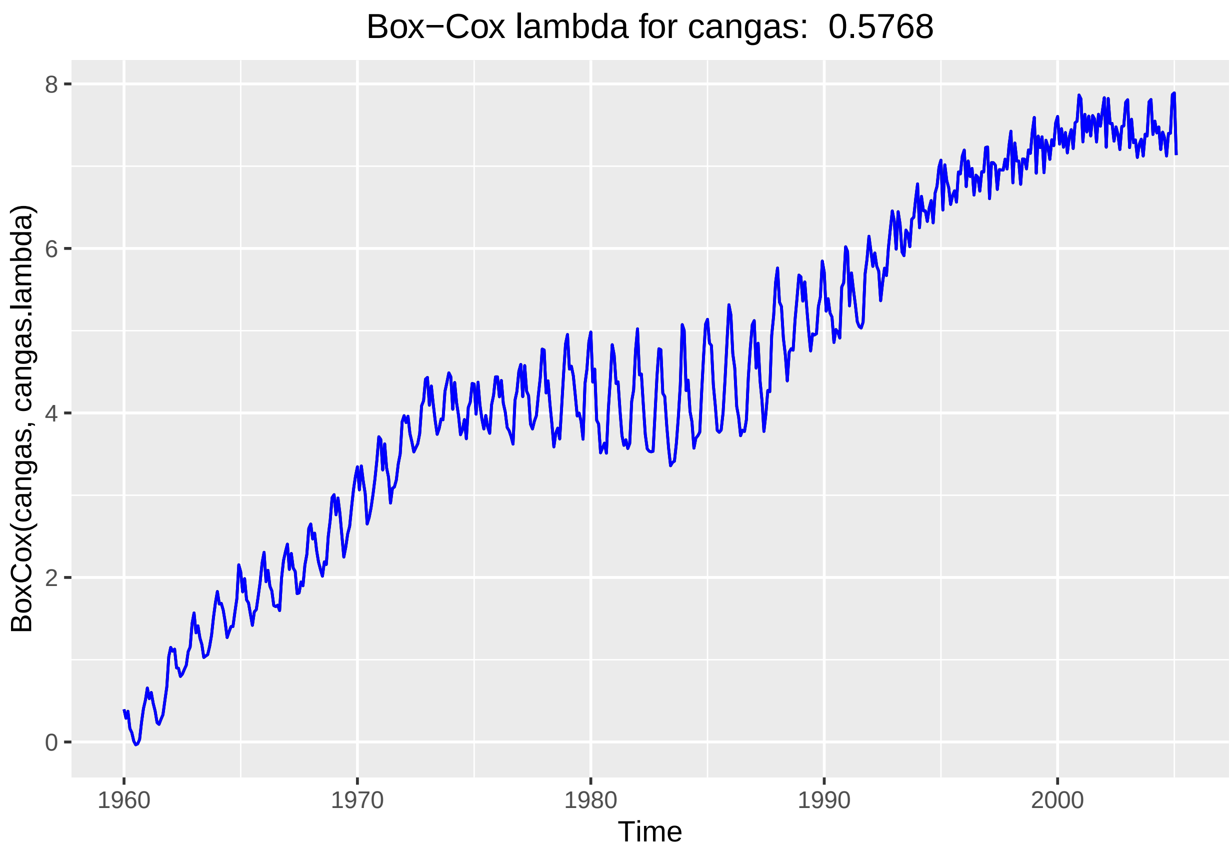

```
##
##  Ljung-Box test
##
## data:  Residuals from Random walk with drift
## Q* = 1207.3, df = 23, p-value < 0.00000000000000022
##
## Model df: 1.   Total lags used: 24
```

```r
#cangas.ljung

if (cangas.ljung$p.value > 0.05) {
  print("Because the p-value on the Ljung-Box test is large,
        the Box-Cox transform is not necessary, but here goes:")
}  else  {
    print("Because the p-value on the Ljung-Box test is small,
          we'll try Box-Cox transform to see if we can achieve constant variance")}
```

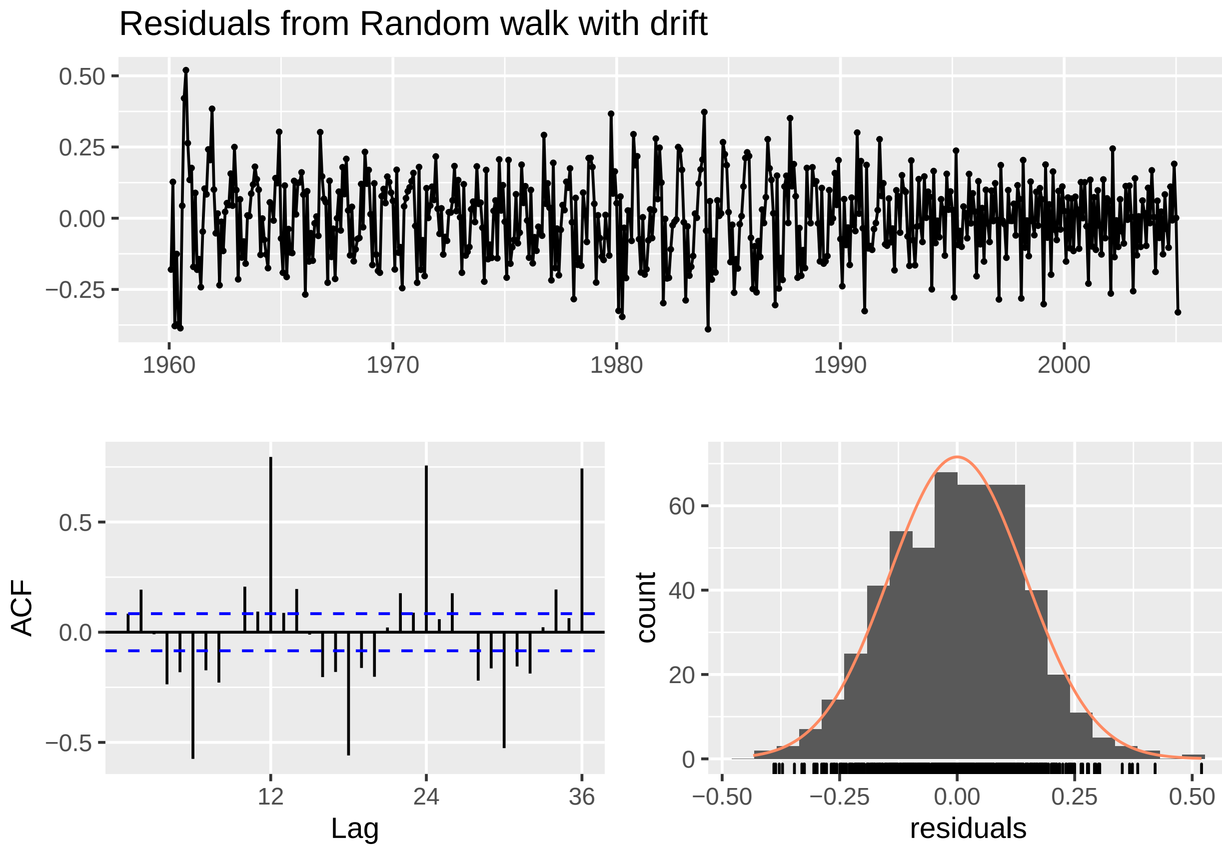
```
## [1] "Because the p-value on the Ljung-Box test is small, \n          we'll try Box-Cox transform to s
```

```r
### Box-Cox transform
cangas.lambda <- BoxCox.lambda(cangas)
### Plot transformed series
#print(paste("Box-Cox lambda for cangas: ", round(cangas.lambda,3)))
autoplot(BoxCox(cangas, cangas.lambda)) +
  ggtitle(paste("Box-Cox lambda for cangas: ", round(cangas.lambda,4))) +
```

```r
theme(plot.title = element_text(hjust = 0.5))+
geom_line(color="blue")
```




```r
cangas.xform.ljung <- checkresiduals(rwf(BoxCox(cangas, cangas.lambda),h=2*frequency(cangas),drift=T,lam
```



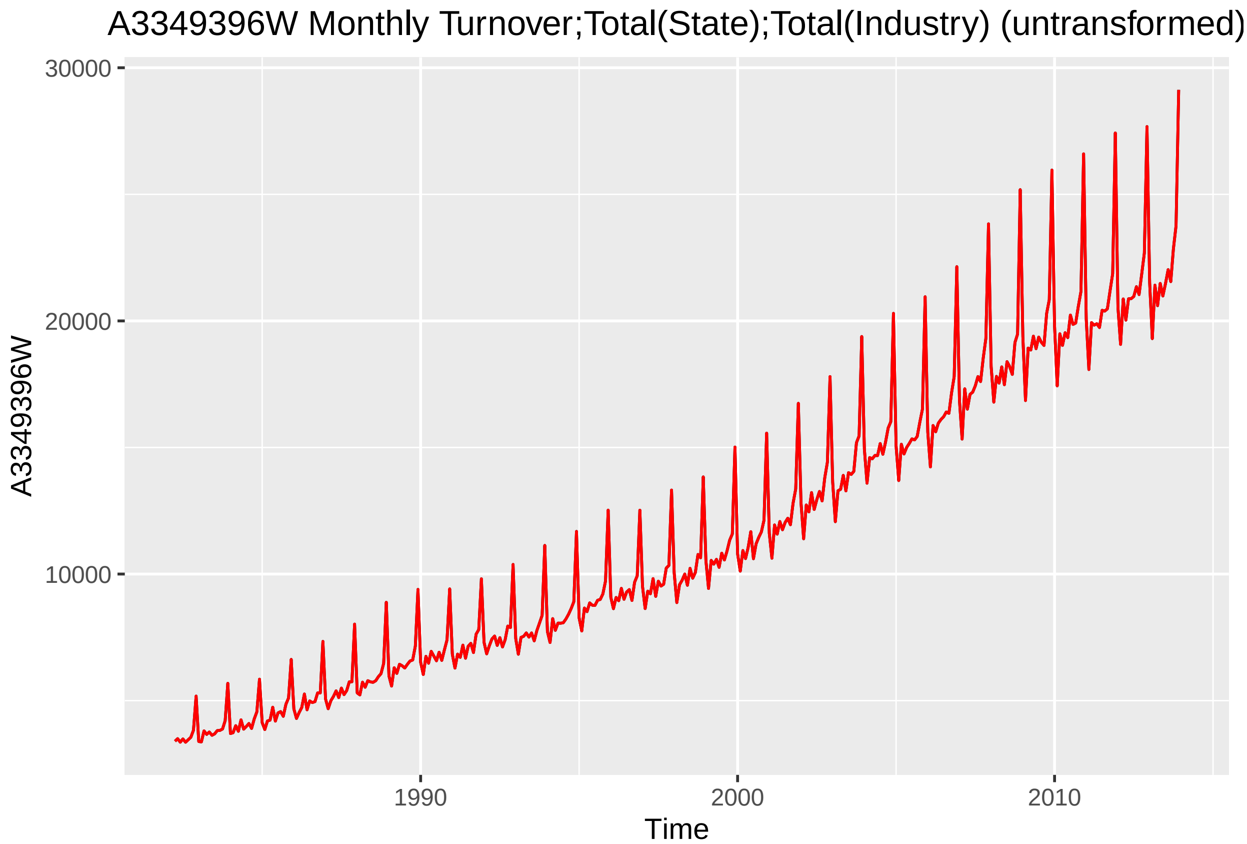

```
##
##  Ljung-Box test
##
## data:  Residuals from Random walk with drift
## Q* = 1307.8, df = 23, p-value < 0.00000000000000022
##
## Model df: 1.   Total lags used: 24
```

```
#cangas.xform.ljung
```

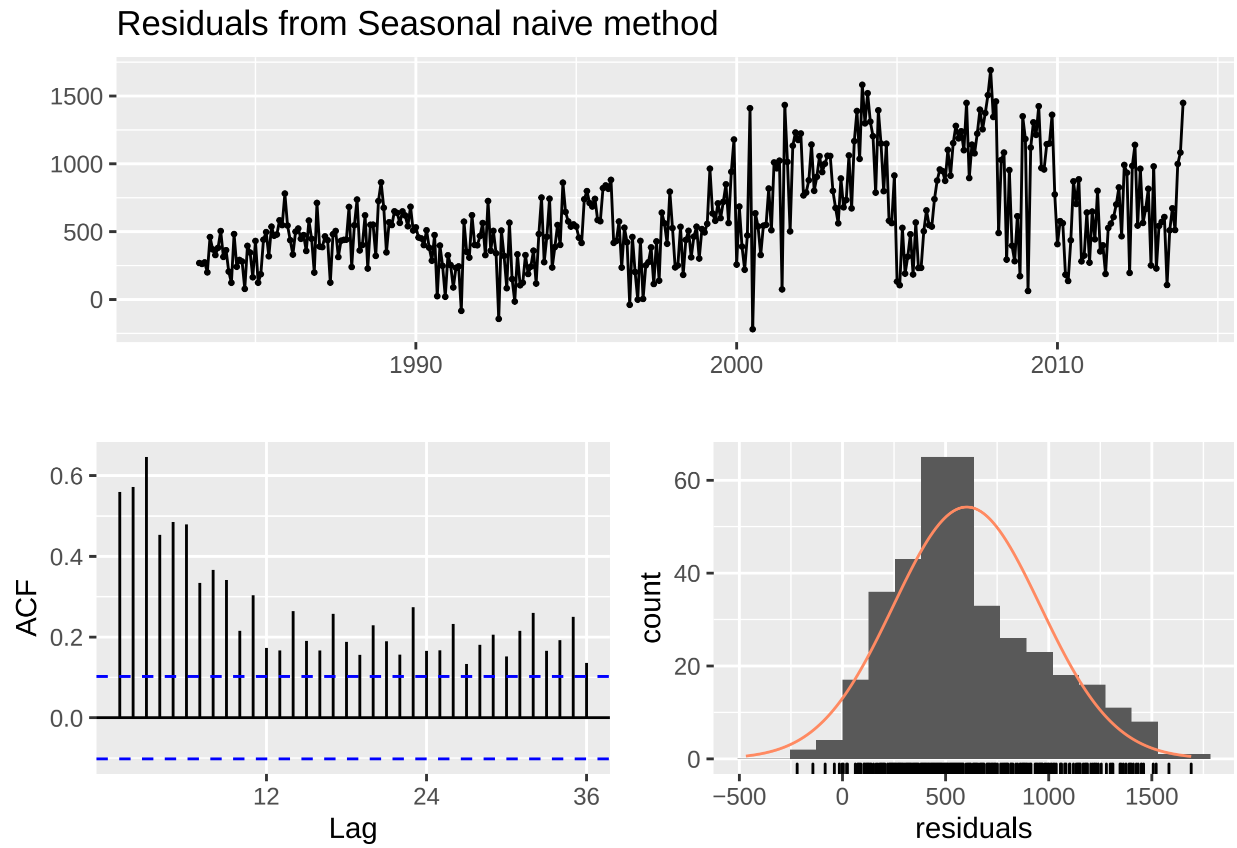
The raw data series for `cangas` exhibits much higher variance during the middle years (late 1970s through early 1990s) and lower variance in the early and later years. The Box-Cox transformation is unable to stabilize this pattern.

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

```
#### You can read the data into R with the following script:
#### readxl does not read straight from URL without local download
####retaildata <- readxl::read_excel("https://otexts.com/fpp2/extrafiles/retail.xlsx", skip=1)
retaildata <- readxl::read_excel("retail.xlsx", skip=1)
#### The second argument (`skip=1`) is required because the Excel sheet has two header rows.

##myts <- ts(retaildata[,"A3349873A"],
##  frequency=12, start=c(1982,4))
#### Select one of the time series as follows
#### (but replace the column name with your own chosen column):

mycode <- "A3349396W"
mytitle <-  "Monthly Turnover;Total(State);Total(Industry)"
mymain <- paste(mycode,mytitle)
myts <- ts(retaildata[,"A3349396W"],
  frequency=12, start=c(1982,4))


### plot raw data series
autoplot(myts) +
  ggtitle(paste(mymain, "(untransformed)")) +
  ylab(mycode)+
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="red")
```




```r
myts.ljung <- checkresiduals(snaive(myts))
```



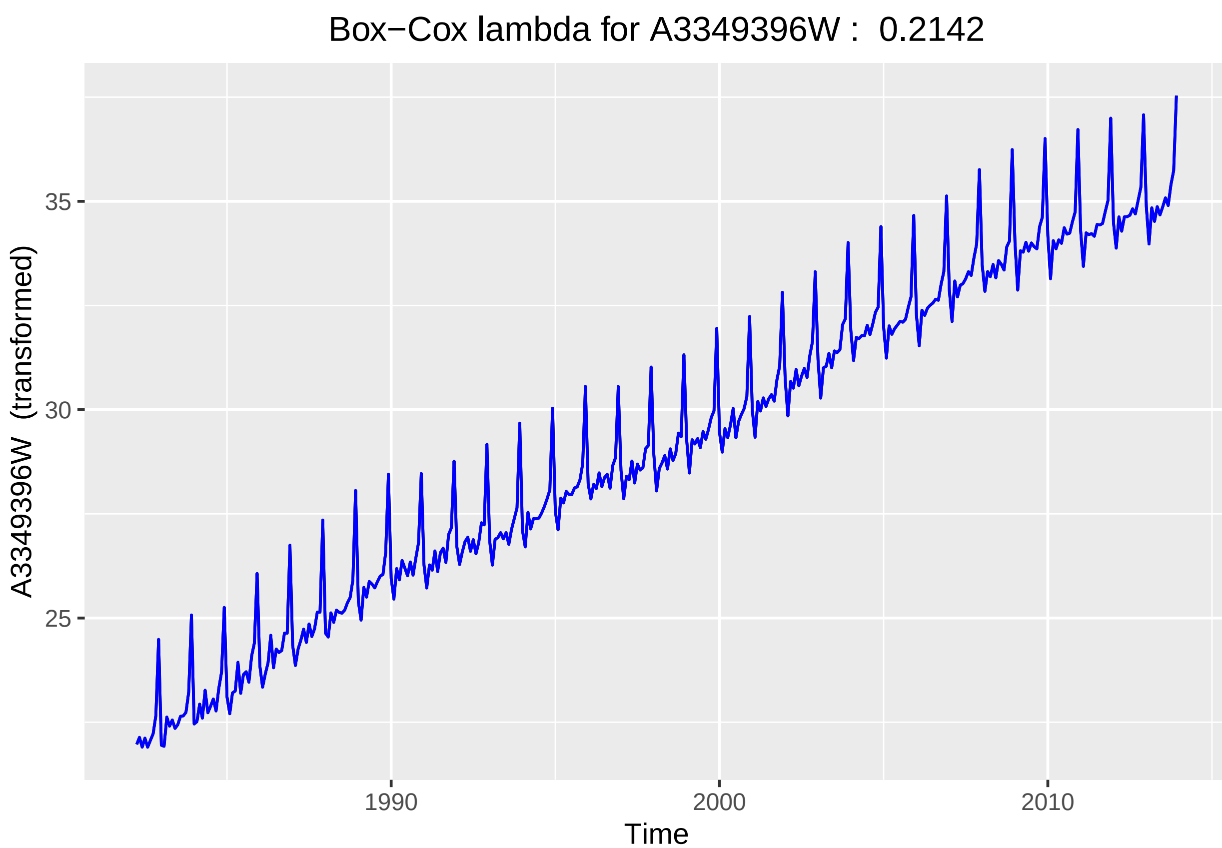

```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 1045.2, df = 24, p-value < 0.0000000000000022
##
## Model df: 0.   Total lags used: 24
```

```
#myts.ljung

if (myts.ljung$p.value > 0.05) {
  print("Because the p-value on the Ljung-Box test is large,
        the Box-Cox transform is not necessary, but here goes:")
}  else  {
    print("Because the p-value on the Ljung-Box test is small,
          we'll try Box-Cox transform to see if we can achieve constant variance")}
```

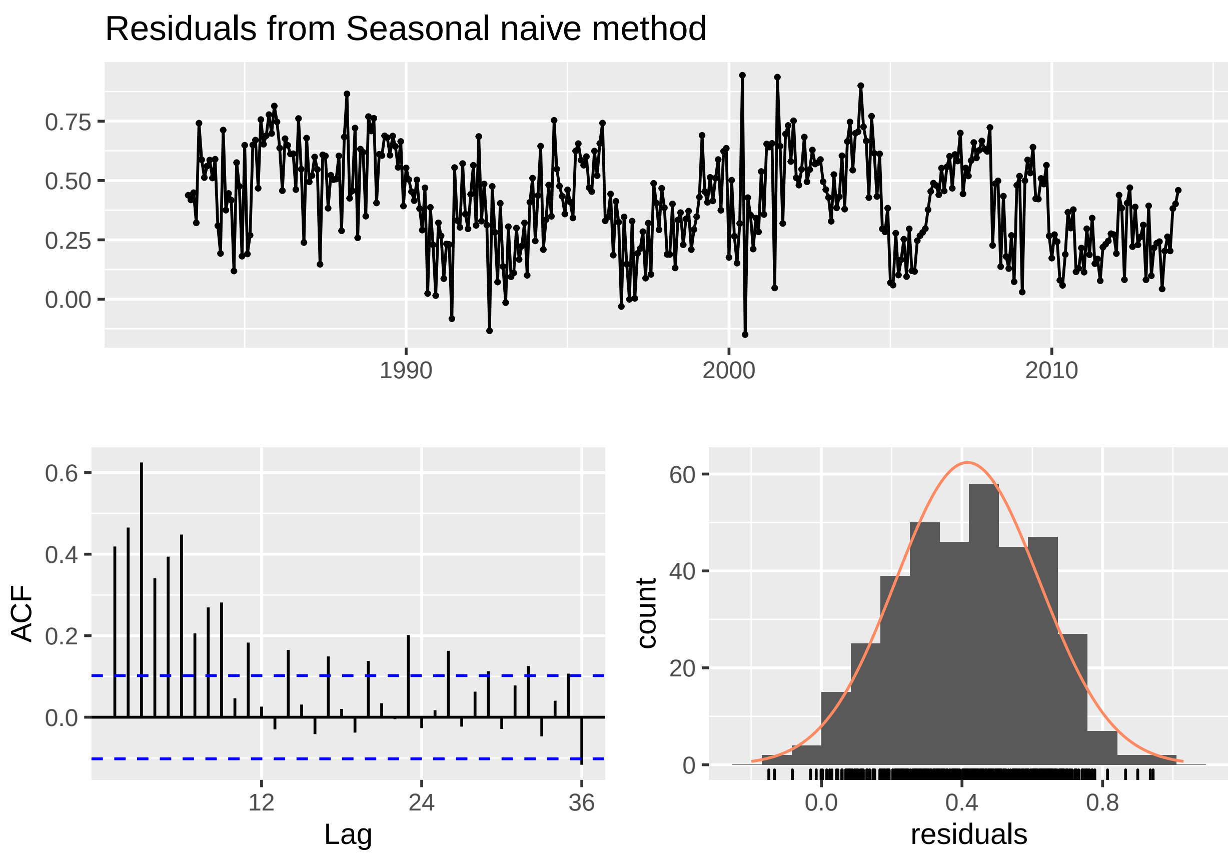
```
## [1] "Because the p-value on the Ljung-Box test is small, \n          we'll try Box-Cox transform to
```

```
### Box-Cox transform
myts.lambda <- BoxCox.lambda(myts)
### Plot transformed series
#print(paste("Box-Cox lambda for myts: ", round(myts.lambda,3)))
autoplot(BoxCox(myts, myts.lambda)) +
  ggtitle(paste("Box-Cox lambda for", mycode, ": ", round(myts.lambda,4))) +
```

```
  ylab(paste(mycode," (transformed)"))+
  theme(plot.title = element_text(hjust = 0.5))+
  geom_line(color="blue")
```




```
myts.xform.ljung <- checkresiduals(snaive(BoxCox(myts, myts.lambda)))
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 602.81, df = 24, p-value < 0.00000000000000022
##
## Model df: 0.   Total lags used: 24
```

```
#myts.xform.ljung
```

The Box-Cox transformation selected for this data series is $\lambda = 0.2142$.

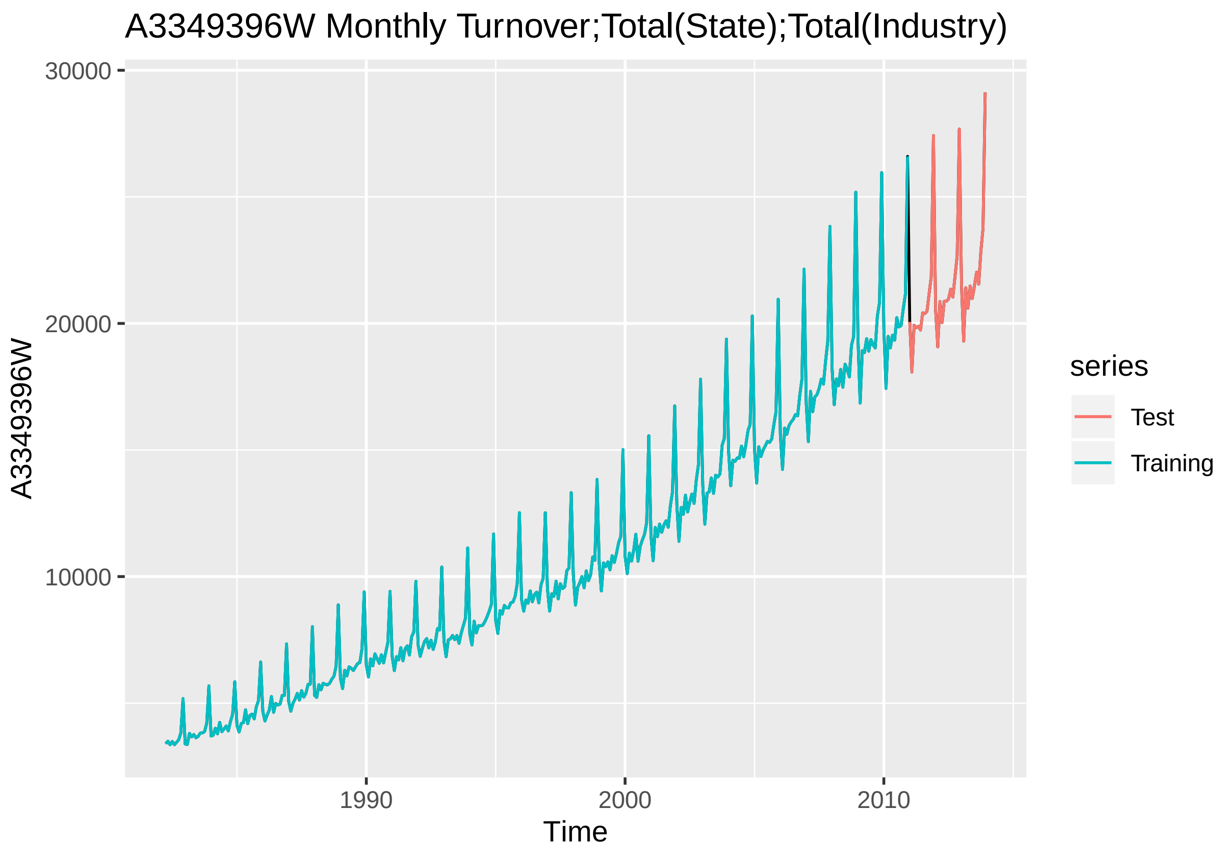
3.8 For your retail time series (from Exercise 3 in Section 2.10):

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")  +
  ggtitle(mymain)+
  ylab(mycode)
```



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

Note: To get 36 months of forecast (3 years) we have to specify h=36

```
fc <- snaive(myts.train,h=length(myts.test))
fc
```

If we don't specify a value for h then we will get the default, which is only 2 years (24 months)

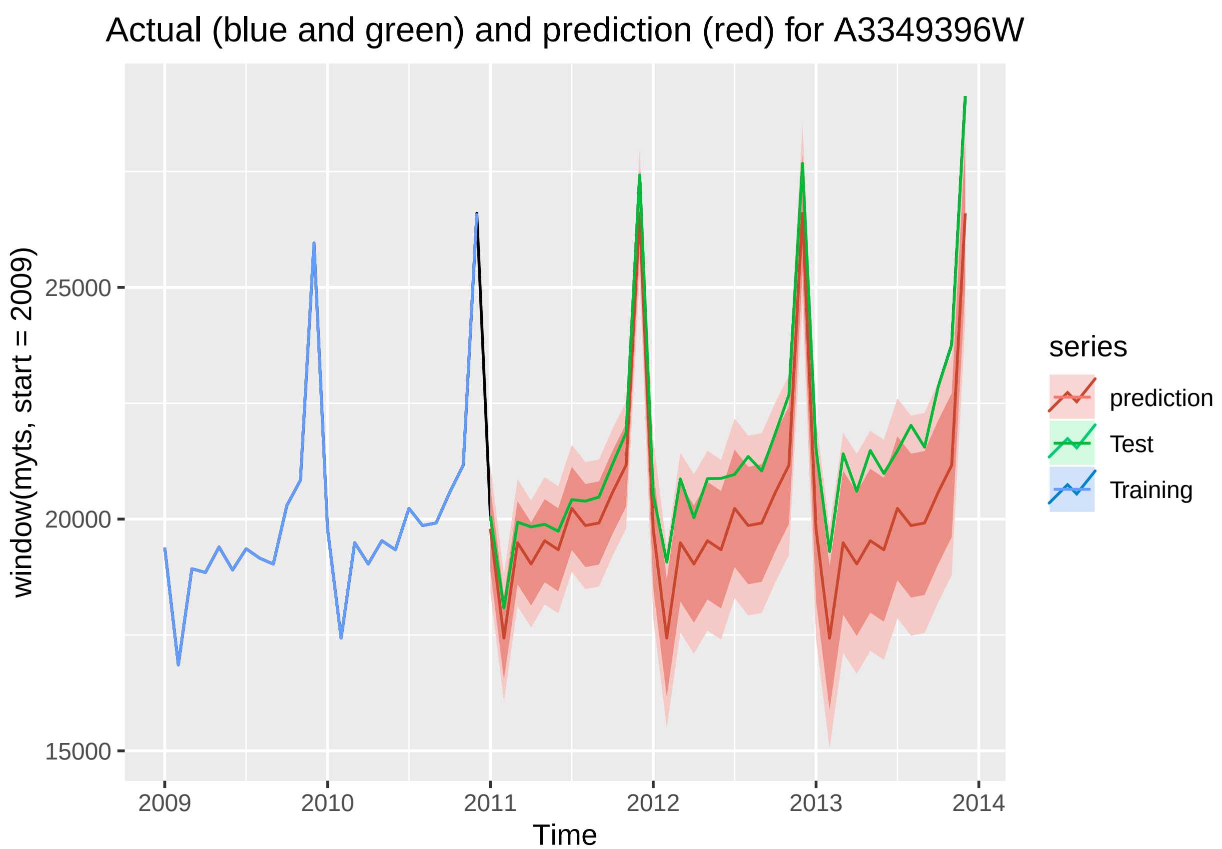
```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 2011        19792.0 18895.89 20688.11 18421.52 21162.48
## Feb 2011        17431.5 16535.39 18327.61 16061.02 18801.98
## Mar 2011        19490.3 18594.19 20386.41 18119.82 20860.78
## Apr 2011        19032.0 18135.89 19928.11 17661.52 20402.48
## May 2011        19533.6 18637.49 20429.71 18163.12 20904.08
## Jun 2011        19339.1 18442.99 20235.21 17968.62 20709.58
## Jul 2011        20231.6 19335.49 21127.71 18861.12 21602.08
## Aug 2011        19860.8 18964.69 20756.91 18490.32 21231.28
## Sep 2011        19916.3 19020.19 20812.41 18545.82 21286.78
## Oct 2011        20575.4 19679.29 21471.51 19204.92 21945.88
## Nov 2011        21163.7 20267.59 22059.81 19793.22 22534.18
## Dec 2011        26599.2 25703.09 27495.31 25228.72 27969.68
## Jan 2012        19792.0 18524.71 21059.29 17853.85 21730.15
## Feb 2012        17431.5 16164.21 18698.79 15493.35 19369.65
## Mar 2012        19490.3 18223.01 20757.59 17552.15 21428.45
## Apr 2012        19032.0 17764.71 20299.29 17093.85 20970.15
## May 2012        19533.6 18266.31 20800.89 17595.45 21471.75
## Jun 2012        19339.1 18071.81 20606.39 17400.95 21277.25
## Jul 2012        20231.6 18964.31 21498.89 18293.45 22169.75
## Aug 2012        19860.8 18593.51 21128.09 17922.65 21798.95
## Sep 2012        19916.3 18649.01 21183.59 17978.15 21854.45
## Oct 2012        20575.4 19308.11 21842.69 18637.25 22513.55
## Nov 2012        21163.7 19896.41 22430.99 19225.55 23101.85
## Dec 2012        26599.2 25331.91 27866.49 24661.05 28537.35
## Jan 2013        19792.0 18239.90 21344.10 17418.26 22165.74
## Feb 2013        17431.5 15879.40 18983.60 15057.76 19805.24
## Mar 2013        19490.3 17938.20 21042.40 17116.56 21864.04
## Apr 2013        19032.0 17479.90 20584.10 16658.26 21405.74
## May 2013        19533.6 17981.50 21085.70 17159.86 21907.34
## Jun 2013        19339.1 17787.00 20891.20 16965.36 21712.84
## Jul 2013        20231.6 18679.50 21783.70 17857.86 22605.34
## Aug 2013        19860.8 18308.70 21412.90 17487.06 22234.54
## Sep 2013        19916.3 18364.20 21468.40 17542.56 22290.04
## Oct 2013        20575.4 19023.30 22127.50 18201.66 22949.14
## Nov 2013        21163.7 19611.60 22715.80 18789.96 23537.44
## Dec 2013        26599.2 25047.10 28151.30 24225.46 28972.94
```

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

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

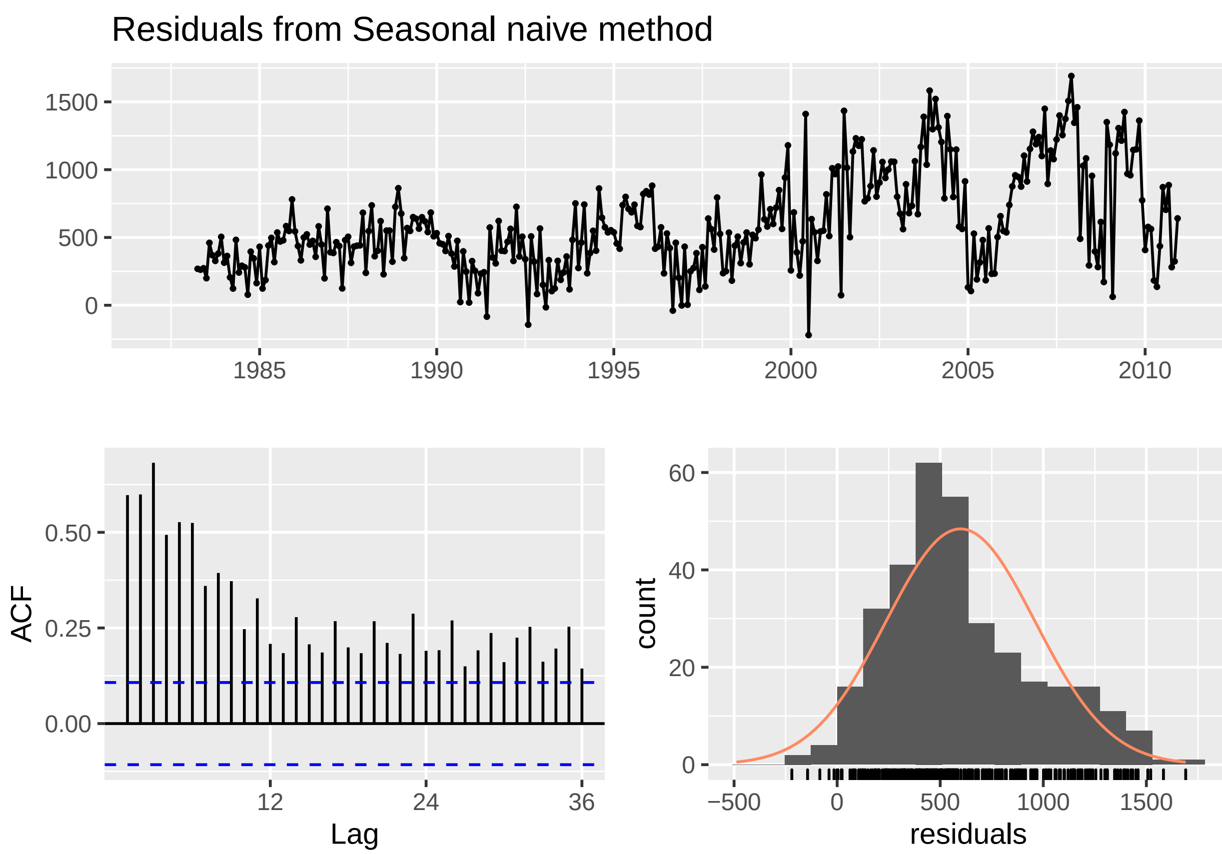
```
##                      ME     RMSE      MAE      MPE     MAPE     MASE      ACF1 Theil's U
## Training set   598.4838  699.236  601.506 5.961516 5.997664 1.000000 0.5972618        NA
## Test set      1230.1556 1389.337 1230.156 5.668187 5.668187 2.045126 0.7093345 0.6485265
```

```r
autoplot(window(myts,start=2009)) +
  ggtitle(paste("Actual (blue and green) and prediction (red) for", mycode))+
  theme(plot.title = element_text(hjust = 0.5))+
  autolayer(window(myts.train,start=2009), series="Training") +
  autolayer(fc, series="prediction")+
  autolayer(myts.test, series="Test", )
```



e) Check the residuals.

```r
checkresiduals(fc)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 1101, df = 24, p-value < 0.00000000000000022
##
## Model df: 0.   Total lags used: 24
```

Do the residuals appear to be uncorrelated and normally distributed? No, the residuals exhibit strong autocorrelation across all lags. Additionally the histogram shows much more density to the left of the mode, with a right tail. The residuals are clearly biased as they do not account for the year-over-year upward trend observed in the actual data; a seasonal trend model would be more appropriate.

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

Below I take the dataset and rerun the naive model a dozen times, each time moving one additional year's worth of data between training and test. We start with the year 2000 being the cut-point between training and test, and advance one year at a time until reaching 2012.

The summary of the sensitivity is as follows:

As more data is moved into the TRAINING set, and less data is in the TEST set:

- The accuracy of **all TEST metrics improves:**
  - i.e., the **TEST** MAE, RMSE, ME, MAPE, MASE, and MPE **all** become **SMALLER**.
- **The change in accuracy of the** TRAINING** metrics is **not uniform:**
  - The accuracy of the **TRAINING** MAE, RMSE, and ME actually **WORSEN** (i.e., become **LARGER**), while
  - the **TRAINING** MAPE and MPE initially **WORSEN**, but then eventually **IMPROVE**.

The results are displayed below.

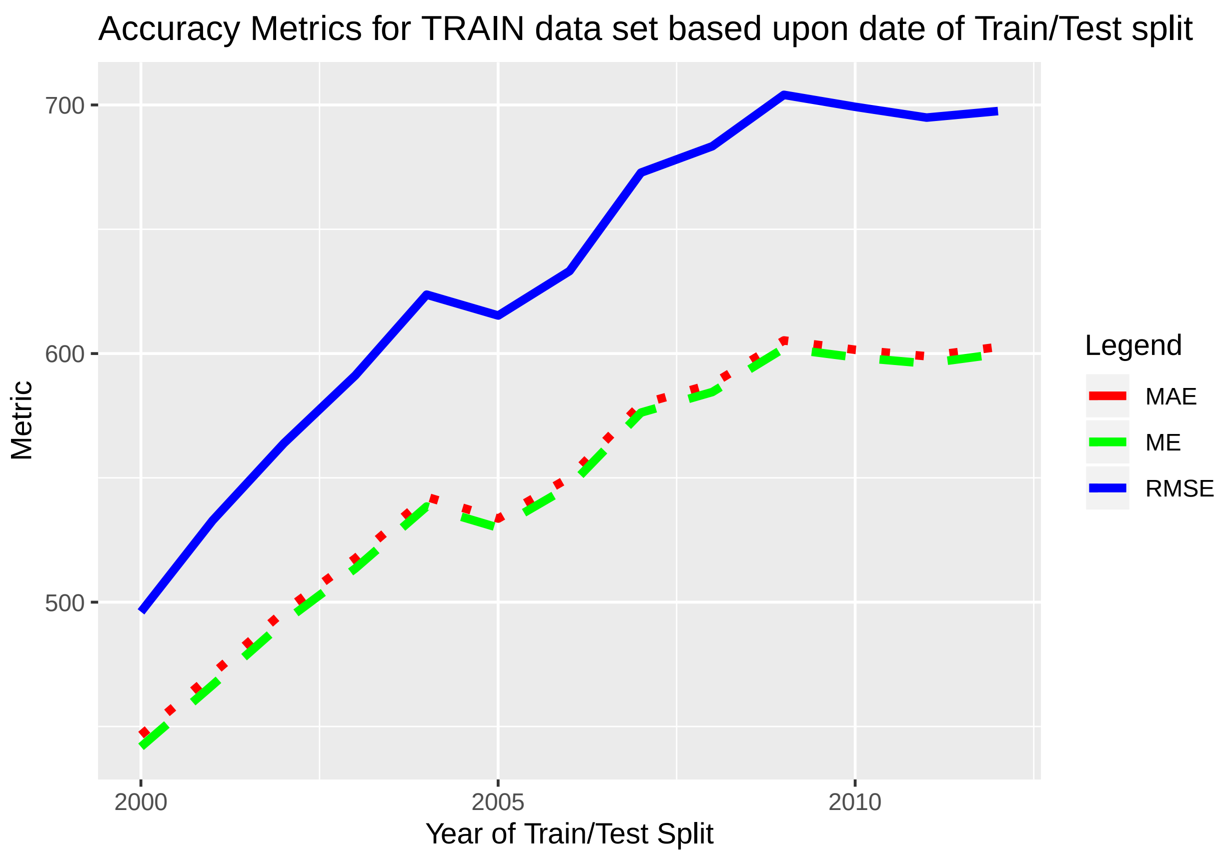
```
firstyear = TRUE
## Loop through 13 years
for (year in 2000:2012) {
  myts.train <- window(myts, end=c(year,12))
  myts.test <- window(myts, start=year+1)
  fc <- snaive(myts.train,h=length(myts.test))
  #print(length(fc$mean))
  #print(paste("YEAR: ", year, "TRAINING SIZE: ", length(myts.train), "TEST SIZE: ",length(myts.test) ))
  ac=accuracy(fc,myts.test)
  if(firstyear == TRUE) {
    # split the "train" and "test" metrics out into two separate matrices for train and test accuracy:
    trainac  = c(YEAR=year,TRAINSIZE=length(myts.train),ac[1,])
    testac   = c(YEAR=year,TESTSIZE=length(myts.test),  ac[2,])
    firstyear = FALSE
  }
  else {
    # append the results from this year onto the existing matrices for train accuracy and test accuracy:
    trainac = rbind(trainac, c(YEAR=year,TRAINSIZE=length(myts.train),ac[1,]))
    testac  = rbind(testac,  c(YEAR=year,TESTSIZE=length(myts.test),  ac[2,]))
  }

}
```

```r
# display the results of the Train Accuracy matrix
print(trainac)
```

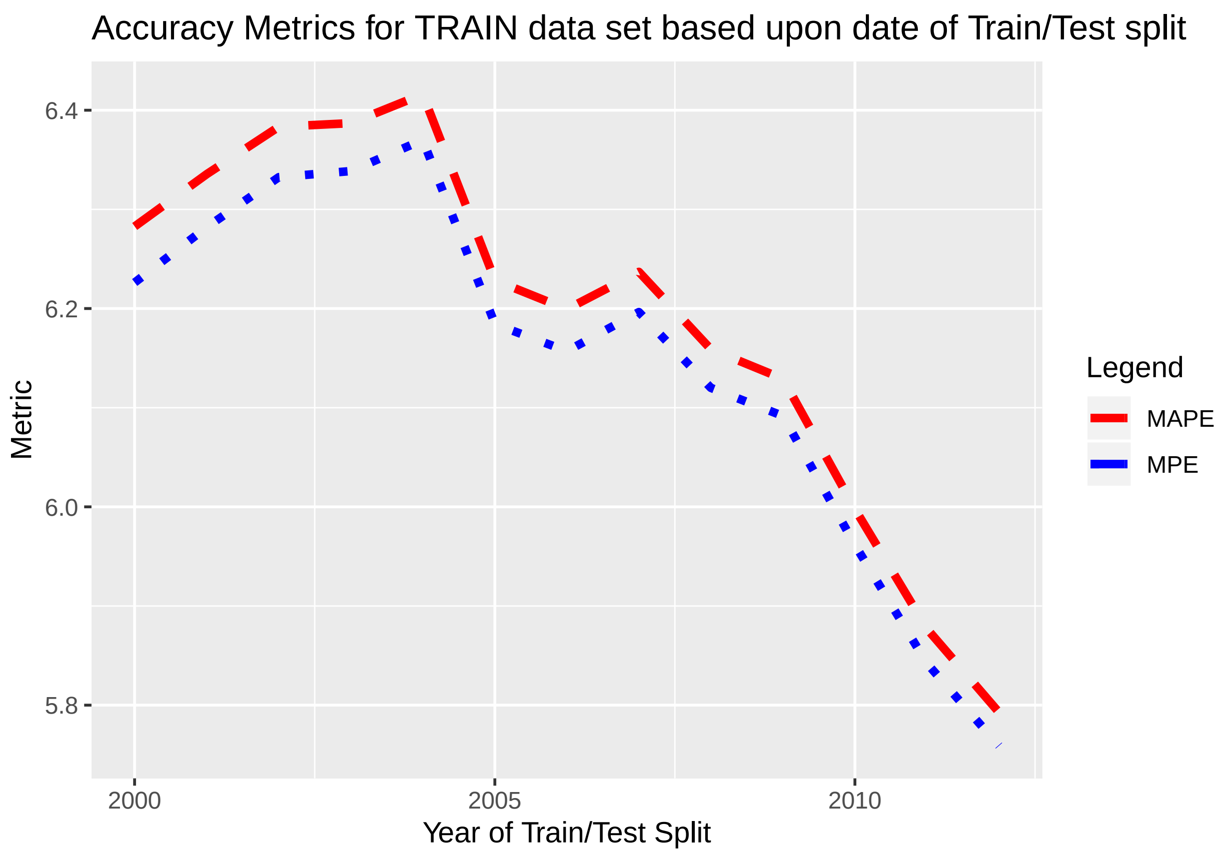
Accuracy metrics for TRAIN data set

```
##         YEAR TRAINSIZE       ME     RMSE      MAE      MPE     MAPE MASE      ACF1 Theil's U
## trainac 2000       225 441.8272 496.1132 446.5521 6.226271 6.282784    1 0.2244231        NA
##         2001       237 466.6796 532.8469 471.1524 6.282127 6.335626    1 0.2828268        NA
##         2002       249 492.0793 563.9284 496.3257 6.332516 6.383306    1 0.3985305        NA
##         2003       261 513.5542 591.2021 517.5960 6.338633 6.386976    1 0.4452951        NA
##         2004       273 538.4889 623.7055 542.3448 6.370047 6.416167    1 0.5321618        NA
##         2005       285 529.9436 615.2940 533.6300 6.184376 6.228469    1 0.5231794        NA
##         2006       297 546.6414 633.1954 550.1726 6.157042 6.199278    1 0.5515157        NA
##         2007       309 576.2492 672.7710 579.6377 6.196449 6.236978    1 0.6144665        NA
##         2008       321 584.5184 683.3828 587.7754 6.120005 6.158961    1 0.5946147        NA
##         2009       333 602.1380 704.0654 605.2732 6.091628 6.129127    1 0.6024399        NA
##         2010       345 598.4838 699.2360 601.5060 5.961516 5.997664    1 0.5972618        NA
##         2011       357 596.0075 694.9102 598.9246 5.842082 5.876972    1 0.5918599        NA
##         2012       369 599.8387 697.5348 602.6577 5.759032 5.792750    1 0.5727110        NA
```

```r
# make it into a data frame, for easier plotting
traindf <- as.data.frame(trainac)
traindf$YEAR <- as.integer(traindf$YEAR)
traindf$DATE <- ISOdate(traindf$YEAR,01,01)

colors <- c("RMSE" = "blue", "MAE" = "red", "ME" = "green")

ggplot(traindf, aes(x = DATE)) +
    geom_line(aes(y = MAE, color = "MAE"), linetype="dotted", size = 1.5) +
    geom_line(aes(y = RMSE, color = "RMSE"), size = 1.5) +
    geom_line(aes(y = ME, color = "ME"), linetype="dashed", size = 1.5) +
    labs(x = "Year of Train/Test Split",
         y = "Metric",
         color = "Legend") +
    scale_color_manual(values = colors)+
  xlim(ISOdate(2000,01,01),ISOdate(2012,01,01))+
  ggtitle("Accuracy Metrics for TRAIN data set based upon date of Train/Test split")
```




```
colors <- c("MPE" = "blue", "MAPE" = "red", "MASE" = "green")

ggplot(traindf, aes(x = DATE)) +
    geom_line(aes(y = MPE, color = "MPE"), linetype="dotted", size = 1.5) +
    geom_line(aes(y = MAPE, color = "MAPE"), linetype="dashed", size = 1.5) +
    labs(x = "Year of Train/Test Split",
         y = "Metric",
         color = "Legend") +
    scale_color_manual(values = colors)+
  xlim(ISOdate(2000,01,01),ISOdate(2012,01,01))+
  ggtitle("Accuracy Metrics for TRAIN data set based upon date of Train/Test split")
```

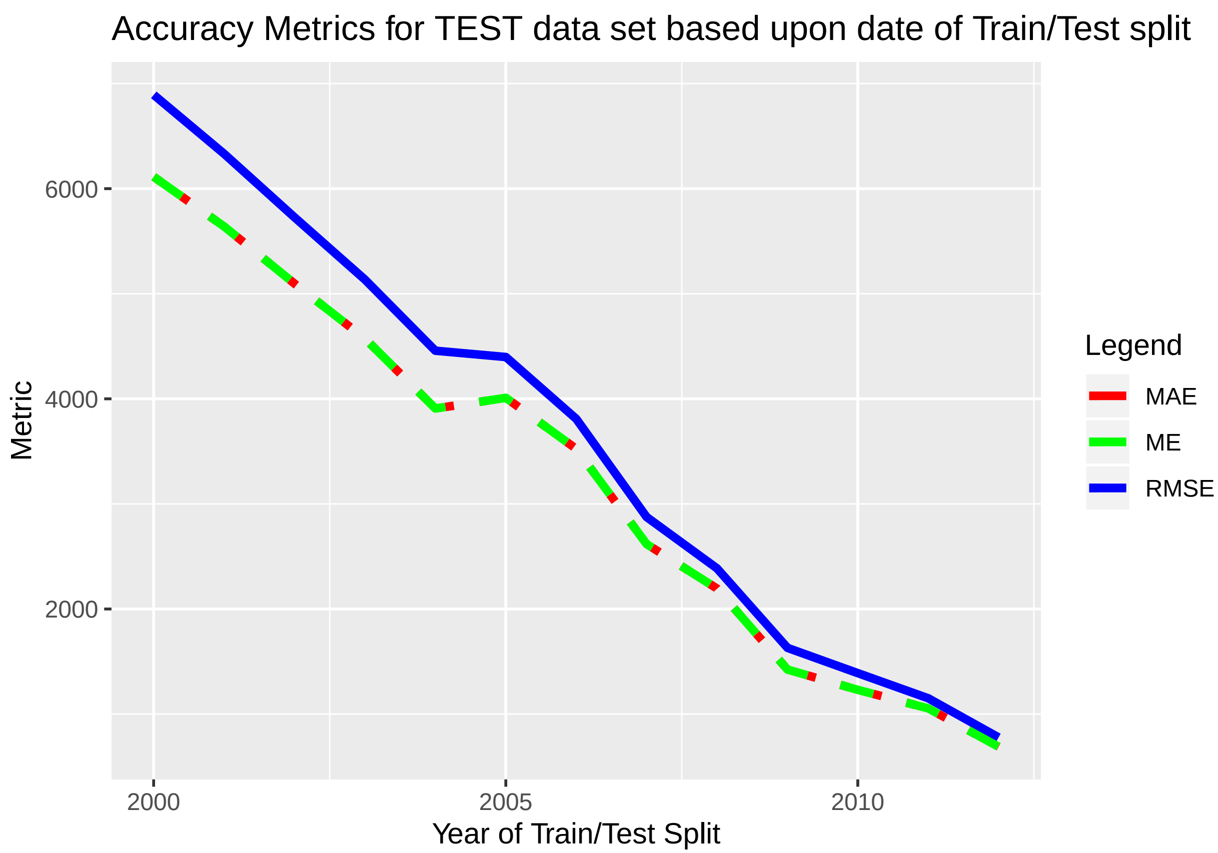

Accuracy Metrics for TRAIN data set based upon date of Train/Test split
Metric
6.4
6.2
6.0
5.8
2000
2005
2010
Year of Train/Test Split
Legend
MAPE
MPE

```r
print(testac)
```

Accuracy metrics for TEST data set

```
##        YEAR TESTSIZE       ME      RMSE      MAE       MPE      MAPE      MASE      ACF1 Theil's U
## testac 2000      156 6112.876 6892.8262 6112.876 32.553804 32.553804 13.689053 0.9337717 3.2207222
##        2001      144 5638.823 6330.5128 5638.823 29.528101 29.528101 11.968150 0.9486687 2.9191664
##        2002      132 5095.089 5725.8973 5095.089 26.174141 26.174141 10.265614 0.9364892 2.6141260
##        2003      120 4573.146 5134.2088 4573.146 23.164552 23.164552  8.835358 0.9438856 2.3266503
##        2004      108 3908.069 4457.8267 3908.069 19.277010 19.277010  7.205876 0.9222842 2.0008593
##        2005       96 4009.484 4397.8005 4009.484 19.694820 19.694820  7.513603 0.9174346 1.9795269
##        2006       84 3523.392 3809.2684 3523.392 17.121308 17.121308  6.404157 0.8926470 1.6967257
##        2007       72 2617.951 2875.0868 2617.951 12.543746 12.543746  4.516530 0.8693843 1.2654264
##        2008       60 2194.522 2387.0439 2194.522 10.388966 10.388966  3.733606 0.8016731 1.0665422
##        2009       48 1423.350 1629.8337 1423.350  6.629138  6.629138  2.351583 0.8135086 0.7346592
##        2010       36 1230.156 1389.3374 1230.156  5.668187  5.668187  2.045126 0.7093345 0.6485265
##        2011       24 1054.296 1150.4211 1054.296  4.830782  4.830782  1.760315 0.2867600 0.5760630
##        2012       12  688.625  777.9451  688.625  3.008222  3.008222  1.142647 0.3078987 0.3891341
```

```r
testdf <- as.data.frame(testac)
testdf$YEAR <- as.integer(testdf$YEAR)
testdf$DATE <- ISOdate(testdf$YEAR,01,01)


colors <- c("RMSE" = "blue", "MAE" = "red", "ME" = "green")

ggplot(testdf, aes(x = DATE)) +
    geom_line(aes(y = MAE, color = "MAE"), linetype="dotted", size = 1.5) +
    geom_line(aes(y = RMSE, color = "RMSE"), size = 1.5) +
    geom_line(aes(y = ME, color = "ME"), linetype="dashed", size = 1.5) +
    labs(x = "Year of Train/Test Split",
         y = "Metric",
         color = "Legend") +
    scale_color_manual(values = colors)+
  xlim(ISOdate(2000,01,01),ISOdate(2012,01,01))+
  ggtitle("Accuracy Metrics for TEST data set based upon date of Train/Test split")
```




```r
colors <- c("MPE" = "blue", "MAPE" = "red", "MASE" = "green")

ggplot(testdf, aes(x = DATE)) +
    geom_line(aes(y = MPE, color = "MPE"), linetype="dotted", size = 1.5) +
    geom_line(aes(y = MASE, color = "MASE"), size = 1.5) +
    geom_line(aes(y = MAPE, color = "MAPE"), linetype="dashed", size = 1.5) +
    labs(x = "Year of Train/Test Split",
         y = "Metric",
         color = "Legend") +
    scale_color_manual(values = colors)+
  xlim(ISOdate(2000,01,01),ISOdate(2012,01,01))+
```

```r
ggtitle("Accuracy Metrics for TEST data set based upon date of Train/Test split")
```