2382 Quanta Monthly Sales

25, July, 2019 Yen Chun, Liu

Package used

library(readxl)

library(knitr)

library(fpp2)

library(tseries)

library(portes)

Read in the data

```
data <- read_excel("2382MonthlySales.xlsx", sheet="Sheet1")
```

Data informaiton

The data set is Sales of 2382 Quanta company. The duration is from Jan. 2015 to Jun. 2019 monthly data. In general due to year end will have to publish financial statement, when the revenue increases are likely to attract investors, therefore companies do push CRM department at that period, it is also likely to have seasonal effect. Therefore by predicting sales revenue could be one of the factor to predict direction of stock price.

```
head(data)

## # A tibble: 6 x 2
##   Month               Sales
##   <dttm>              <dbl>
## 1 2015-01-01 00:00:00  701.
## 2 2015-02-01 00:00:00  550.
## 3 2015-03-01 00:00:00  800.
## 4 2015-04-01 00:00:00  675.
## 5 2015-05-01 00:00:00  770.
## 6 2015-06-01 00:00:00 1065

tail(data)

## # A tibble: 6 x 2
##   Month               Sales
##   <dttm>              <dbl>
## 1 2019-01-01 00:00:00  849.
## 2 2019-02-01 00:00:00  629.
## 3 2019-03-01 00:00:00  735.
## 4 2019-04-01 00:00:00  781.
## 5 2019-05-01 00:00:00  887.
## 6 2019-06-01 00:00:00  789.
```

Change to time series format

```
to <- ts(data[,2], frequency = 12, start=c(2015))
```

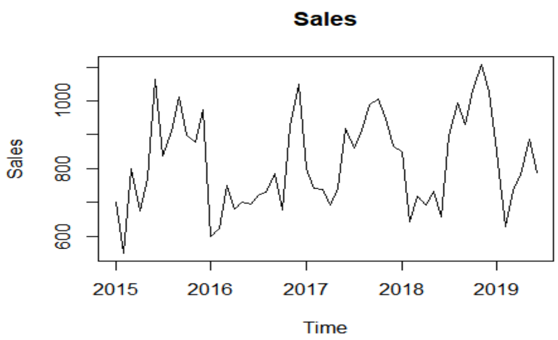
Split train and test

```
train <- window(to, start= c(2015,1), end= c(2018,12))
test <- window(to, start= c(2019,1),end= c(2019,6))
h = length(test)
```

Line plot

From the plot we can see that there's fluctuation from 2015 to 2017.There are four major fluctuation. Through the plot we are not able to see if there is seasonal effect or not. We will build seasonal plot and month plot after.

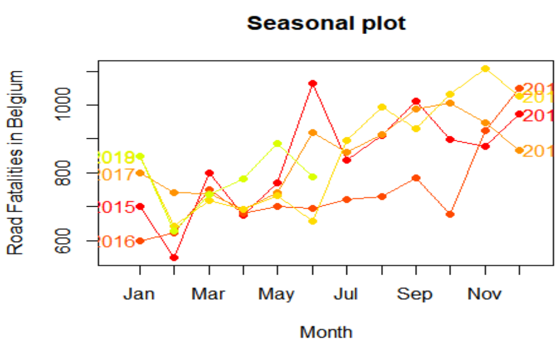
```
plot(to, main="Sales")
```

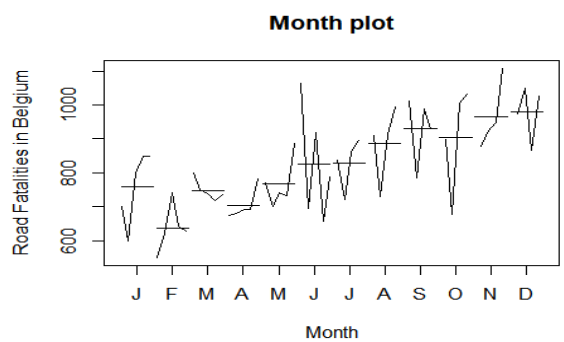


Season and month plot

From the seasonal and month plot we can see that Feb. is generally the lowest and from Feb. will have a upward fluctuation to Dec.

```
seasonplot(to, year.labels=TRUE, year.labels.left=TRUE,
           main="Seasonal plot",
           ylab="Road Fatalities in Belgium",col=rainbow(21), pch=19)
```





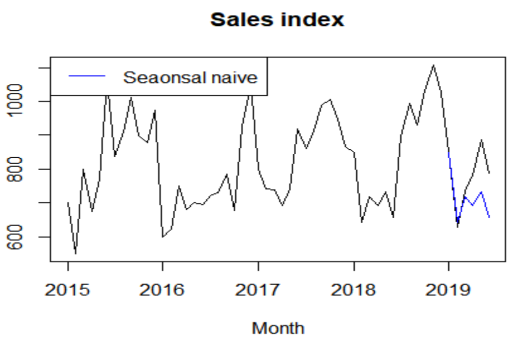

```
monthplot(to, main="Month plot", ylab = "Road Fatalities in Belgium",
          xlab="Month", type="l")
```

Seasonal naive method

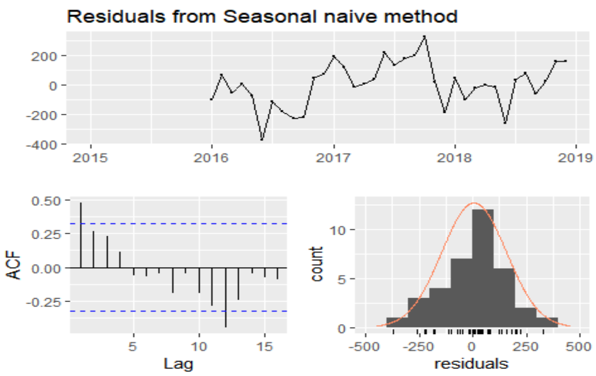
With the plot under we can see that the fluctuation is clearly not enough to the fact. However to judge the model performance we have to compare with other models by RMSE, MAE, MAPE and MAsE.

For white noise series, we expect each autocorrelation to be close to zero. Of course, they will not be exactly equal to zero as there is some random variation. For a white noise series, we expect 95% of the spikes in the ACF to lie within the blue dashed lines above. If one or more large spikes are outside these bounds, or if substantially more than 5% of spikes are outside these bounds, then the series is probably not white noise. If Ljung-Box test p-value is above 0.05 means accept as white noise. The residual diagnostics show that after and below lag 9 the residuals of this method are not white noise.

```
f1 <-snaive(train, h = h)
plot(to,main="Sales index", ylab="",xlab="Month")
lines(f1$mean,col=4)
legend("topleft",lty=1,col=c(4),legend=c("Seaonsal naive"))
```




```
res <- residuals(f1)
checkresiduals(f1)
```

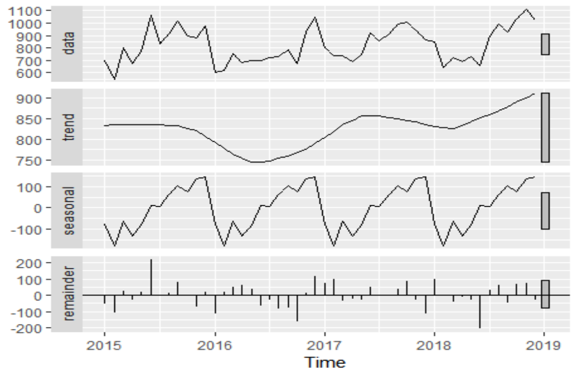



```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 18.659, df = 10, p-value = 0.04482
##
## Model df: 0.   Total lags used: 10

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df      p-value
##     1  8.863397  1 0.0029094561
##     5 14.531333  5 0.0125643150
##     9 16.771076  9 0.0524239418
##    13 37.873044 13 0.0003019289
##    17 38.933447 17 0.0018257592
##    21 42.476217 21 0.0036641907

accuracy(f1)[,c(2,3,5,6)]

##      RMSE       MAE      MAPE      MASE
## 149.00401 115.56667  14.34938   1.00000
```

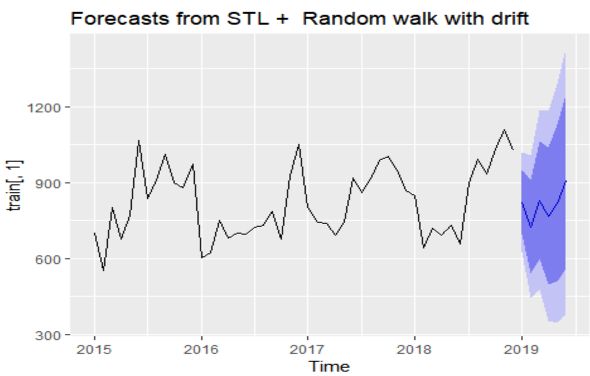
STL decomposition

The two main parameters to be chosen when using STL are the trend-cycle window (t.window) and the seasonal window (s.window). These control how rapidly the trend-cycle and seasonal components can change. Smaller values allow for more rapid changes. Both t.window and s.window should be odd numbers; The residual diagnostics show that the residuals of this method are not white noise.

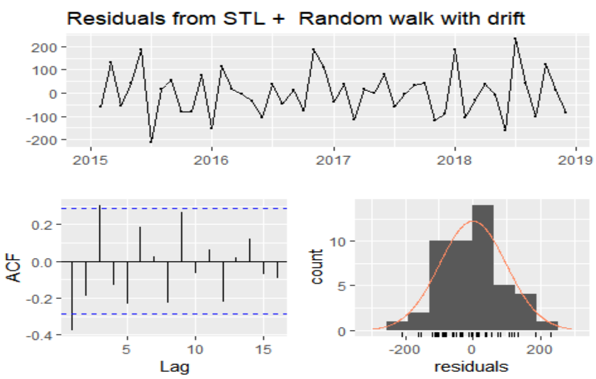
```
f2 <- forecast(stl(train[,1],t.window = 15, s.window=13), method="rwdrift",h=h)
autoplot(stl(train[,1], s.window="periodic"))
```




```
autoplot(f2)
```




```
res <- residuals(f2)
checkresiduals(f2)

## Warning in checkresiduals(f2): The fitted degrees of freedom is based on
## the model used for the seasonally adjusted data.
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  Random walk with drift
## Q* = 26.976, df = 9, p-value = 0.001412
##
## Model df: 1.   Total lags used: 10

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df     p-value
##     1   7.01530  1 0.008081607
```

```
##      5  17.48346  5 0.003668563
##      9  26.69290  9 0.001572030
##     13  30.45075 13 0.004053465
##     17  32.52177 17 0.012951167
##     21  42.69626 21 0.003436164
```

```
accuracy(f2)[,c(2,3,5,6)]
```

```
##        RMSE        MAE       MAPE       MASE
## 97.6713478 77.8764014  9.5779950  0.6738656
```

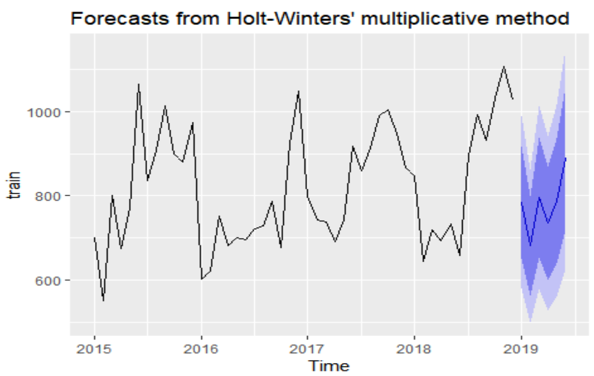
Holt-Winters method

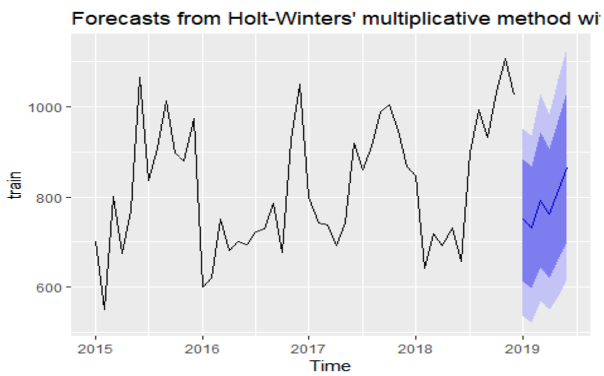
There are two variations to this method that differ in the nature of the seasonal component. The additive method is preferred when the seasonal variations are roughly constant through the series, while the multiplicative method is preferred when the seasonal variations are changing proportional to the level of the series.

WE will apply Holt-Winters method with both additive and multiplicative seasonality and with/without exponential and damped or not to forecast, to see which one have the best accuracy.

The Holt-Winters multiplicative method have the best performance compare to others. Where it does not have an acceptable residual diagnostics. Therefore we have to go with Damped Holt-winters multiplicative method, which it have the second best of accuracy with residual diagnostics that is acceptable.

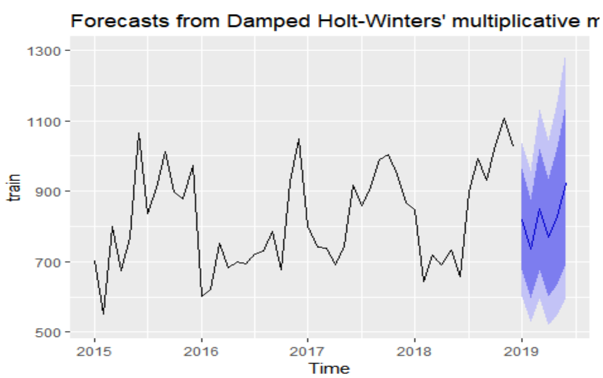
```
f3 <- forecast(hw(train,seasonal="mult", h=h), method="rwdrift", h=h)
autoplot(f3)
```

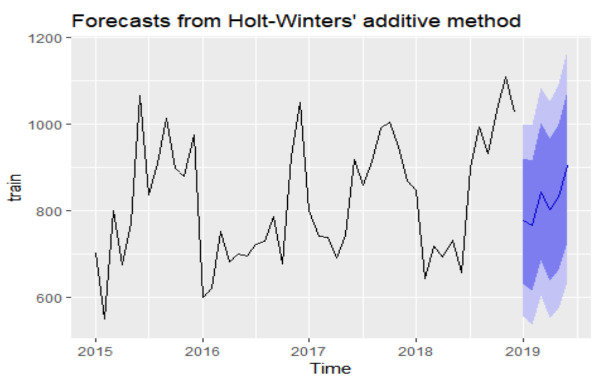





```
f4 <- forecast(hw(train,seasonal="mult",exponential=TRUE, h=h), method="rwdrift", h=h)
autoplot(f4)
```

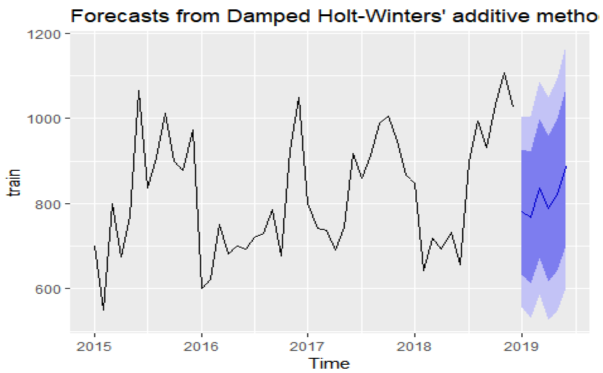
```
f5 <- forecast(hw(train,seasonal="mult",exponential=TRUE, damped=TRUE, h=h), method="rwdrift", h=h)
autoplot(f5)
```



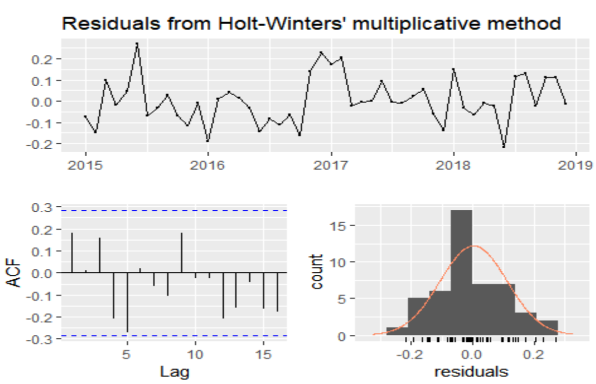


```r
f6 <- forecast(hw(train,seasonal="addi", h=h), method="rwdrift", h=h)
autoplot(f6)
```

```r
f7 <- forecast(hw(train,seasonal="addi",damped=TRUE, h=h), method="rwdrift", h=h)
autoplot(f7)
```

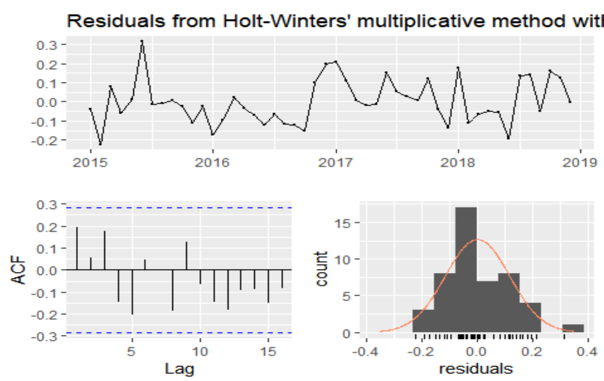


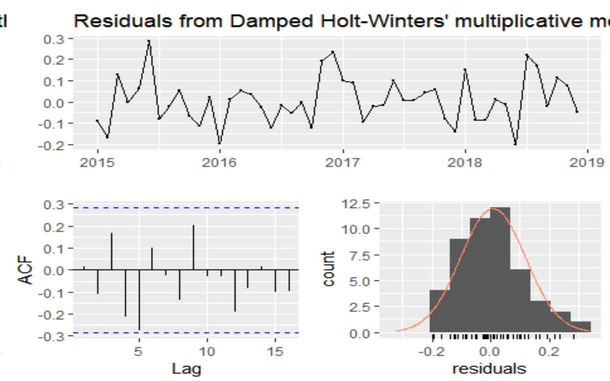

```r
a_fc3 <- accuracy(f3)[,c(2,3,5,6)]
a_fc4 <- accuracy(f4)[,c(2,3,5,6)]
a_fc5 <- accuracy(f5)[,c(2,3,5,6)]
a_fc6 <- accuracy(f6)[,c(2,3,5,6)]
a_fc7 <- accuracy(f7)[,c(2,3,5,6)]
acc <- rbind(a_fc3, a_fc4, a_fc5, a_fc6, a_fc7)
rownames(acc) <- c("a_fc3", "a_fc4", "a_fc5", "a_fc6", "a_fc7")
acc

##             RMSE      MAE     MAPE      MASE
## a_fc3 88.08994 68.37023 8.381014 0.5916085
## a_fc4 92.73950 73.38977 9.104331 0.6350427
## a_fc5 89.41970 70.00359 8.496327 0.6057421
## a_fc6 91.55314 73.16551 9.014213 0.6331022
## a_fc7 91.63767 71.61288 8.722550 0.6196672

res <- residuals(f3)
checkresiduals(f3)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method
## Q* = 23.883, df = 3, p-value = 2.643e-05
##
## Model df: 16.   Total lags used: 19
```

```
res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df    p-value
##     1  1.704552  1 0.19169380
##     5  9.565866  5 0.08851332
##     9 12.493503  9 0.18689527
##    13 17.249846 13 0.18813285
##    17 21.682063 17 0.19725728
##    21 35.352769 21 0.02581757

res <- residuals(f4)
checkresiduals(f4)
```



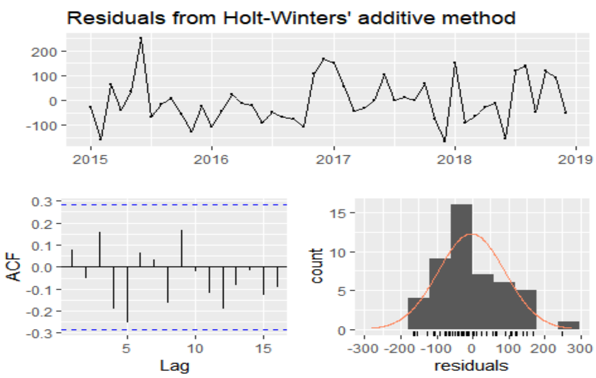


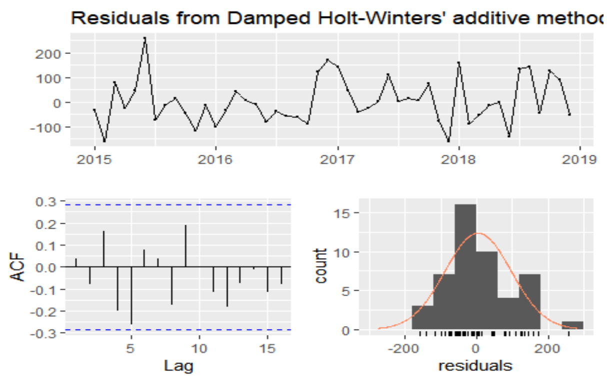

```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method with exponential trend
## Q* = 21.961, df = 3, p-value = 6.645e-05
##
## Model df: 16.   Total lags used: 19

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df   p-value
##     1  1.926916  1 0.1650963
##     5  7.232483  5 0.2039165
##     9 10.408839  9 0.3184119
##    13 14.815101 13 0.3190374
##    17 17.481686 17 0.4222211
##    21 27.123679 21 0.1667968

res <- residuals(f5)
checkresiduals(f5)
```

```
##
##  Ljung-Box test
##
## data:  Residuals from Damped Holt-Winters' multiplicative method with exponential trend
## Q* = 27.295, df = 3, p-value = 5.105e-06
##
## Model df: 17.   Total lags used: 20

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)
```

```
##  lags   statistic df   p-value
##     1  0.01525897  1 0.9016896
##     5  8.96912554  5 0.1103020
##     9 13.27802373  9 0.1504242
##    13 16.32455088 13 0.2320475
##    17 18.01974477 17 0.3875963
##    21 27.32297816 21 0.1604166
```

```
res <- residuals(f6)
checkresiduals(f6)
```






```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 19.613, df = 3, p-value = 0.0002042
##
## Model df: 16.   Total lags used: 19
```

```
res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)
```

```
##  lags  statistic df   p-value
##     1  0.2923946  1 0.5886901
##     5  7.2909407  5 0.1998862
##     9 10.8691512  9 0.2847847
##    13 14.7101682 13 0.3257941
##    17 16.7042757 17 0.4745692
##    21 24.4584509 21 0.2713795
```

```
res <- residuals(f7)
checkresiduals(f7)
```

```
##
##  Ljung-Box test
##
## data:  Residuals from Damped Holt-Winters' additive method
## Q* = 23.977, df = 3, p-value = 2.526e-05
##
## Model df: 17.   Total lags used: 20
```

```
res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)
```

```
##  lags   statistic df   p-value
##     1  0.06962346  1 0.7918858
```

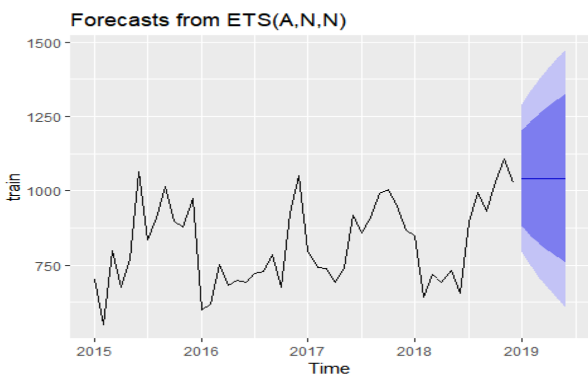
```
##      5  7.86369133  5 0.1639096
##      9 12.28995969  9 0.1974520
##     13 15.69055278 13 0.2662444
##     17 17.25813592 17 0.4370142
##     21 24.03280766 21 0.2914716
```

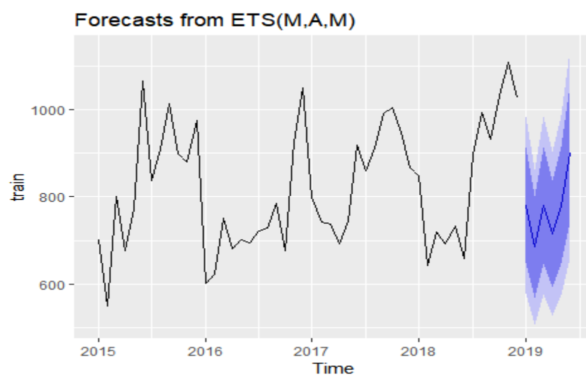
ETS

ETS (Error, Trend, Seasonal) method is an approach method for forecasting time series. Based on the properties of the data, we estimate several ETS models with a trend and a seasonal component. We consider additive and multiplicative errors, and trends with and without damping. The first letter denotes the error type (“A”, “M” or “Z”); the second letter denotes the trend type (“N”,“A”,“M” or “Z”); the third letter denotes the season type (“N”,“A”,“M” or “Z”). In all cases, “N”=none, “A”=additive, “M”=multiplicative and “Z”=automatically selected.

Due to the fact that we do see a continues pattern of fluctuation. We will try MMM and MAM with damped and non damped, to compare with auto ets which auto ets choose among best AIC, but not accuracy. ETS MMM model have the best accuracy among ETS model for this situation. The residual diagnostics doesn’t have an acceptable result. So we have to go with ETS MMM with damped.

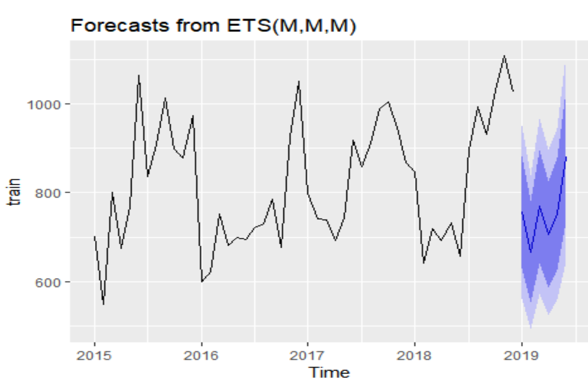
```
f8 <- forecast(ets(train, model = "ZZZ"), method="rwdrift", h=h)
autoplot(f8)
```

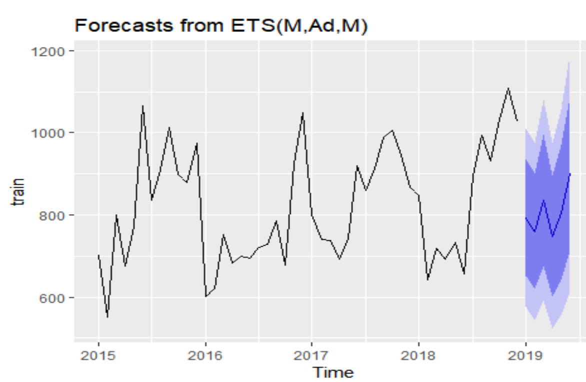





```
f9 <- forecast(ets(train, model = "MAM"), method="rwdrift", h=h)
autoplot(f9)
```

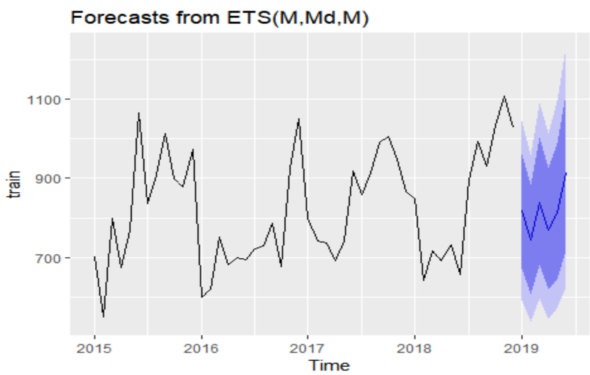
```
f10 <- forecast(ets(train, model = "MMM"), method="rwdrift", h=h)
autoplot(f10)
```



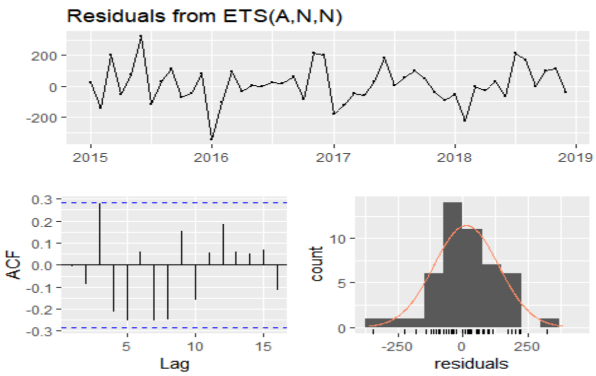


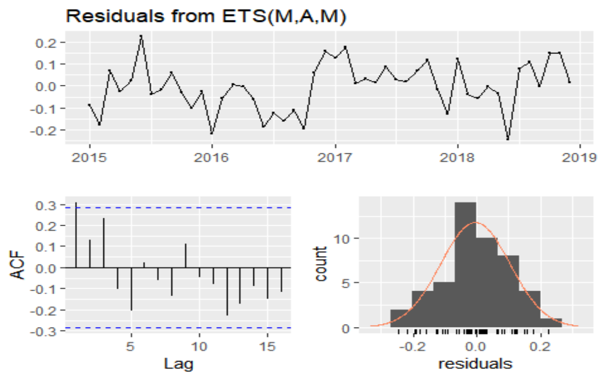

```
f11 <- forecast(ets(train, model = "MAM",damped=TRUE), method="rwdrift", h=h)
autoplot(f11)
```

```
f12 <- forecast(ets(train, model = "MMM",damped=TRUE), method="rwdrift", h=h)
autoplot(f12)
```




```
a_fc8 <- accuracy(f8)[,c(2,3,5,6)]
a_fc9 <- accuracy(f9)[,c(2,3,5,6)]
a_fc10 <- accuracy(f10)[,c(2,3,5,6)]
a_fc11 <- accuracy(f11)[,c(2,3,5,6)]
a_fc12 <- accuracy(f12)[,c(2,3,5,6)]
acc <- rbind(a_fc8, a_fc9, a_fc10, a_fc11, a_fc12)
rownames(acc) <- c("a_fc8", "a_fc9", "a_fc10", "a_fc11", "a_fc12")
acc

##             RMSE      MAE      MAPE      MASE
## a_fc8  123.11504 93.50762 11.554618 0.8091227
## a_fc9   90.03805 69.95839  8.768002 0.6053510
## a_fc10  90.07052 69.62180  8.725973 0.6024384
## a_fc11  91.63382 72.28863  8.869546 0.6255146
## a_fc12  90.61803 71.01396  8.660726 0.6144848

res <- residuals(f8)
checkresiduals(f8)
```






```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,N,N)
## Q* = 21.243, df = 8, p-value = 0.006528
##
## Model df: 2.   Total lags used: 10

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)
```

```
##  lags     statistic df     p-value
##     1  0.001380937  1 0.97035666
##     5 10.573672419  5 0.06051959
##     9 19.655381433  9 0.02016244
##    13 23.927392338 13 0.03180454
##    17 25.940755731 17 0.07554668
##    21 29.223533721 21 0.10873746
```

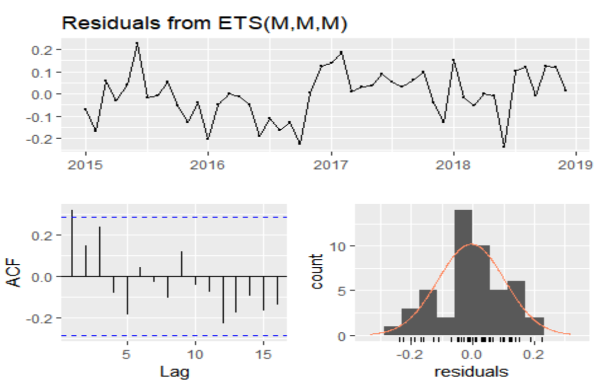
```
res <- residuals(f9)
checkresiduals(f9)
```

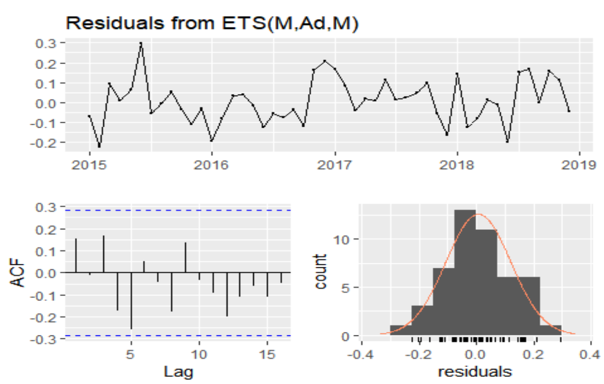
```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,A,M)
## Q* = 24.981, df = 3, p-value = 1.558e-05
##
## Model df: 16.   Total lags used: 19
```

```
res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)
```

```
##  lags statistic df     p-value
##     1  4.805761  1 0.02836474
##     5 11.418403  5 0.04368717
##     9 13.508453  9 0.14091621
##    13 19.541408 13 0.10724991
##    17 22.842690 17 0.15444237
##    21 34.623052 21 0.03104303
```

```
res <- residuals(f10)
checkresiduals(f10)
```





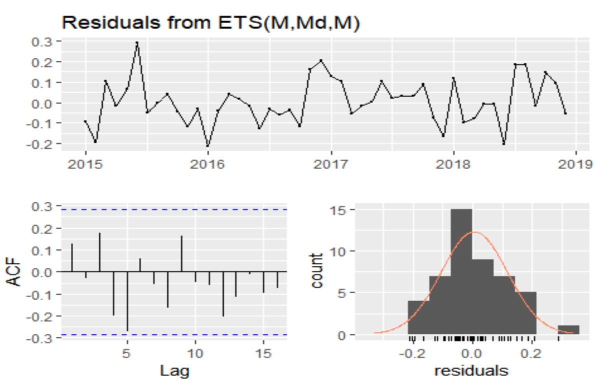

```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,M,M)
## Q* = 25.333, df = 3, p-value = 1.315e-05
##
## Model df: 16.   Total lags used: 19
```

```
res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)
```

```
##  lags statistic df     p-value
##     1  5.123467  1 0.02360441
```

```
##     5 11.483509  5 0.04259286
##     9 13.133990  9 0.15663650
##    13 19.254916 13 0.11540955
##    17 23.351387 17 0.13815682
##    21 33.672965 21 0.03926747

res <- residuals(f11)
checkresiduals(f11)
```

```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,Ad,M)
## Q* = 25.522, df = 3, p-value = 1.201e-05
##
## Model df: 17.   Total lags used: 20

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df   p-value
##     1  1.184131  1 0.2765167
##     5  8.028852  5 0.1546527
##     9 11.332688  9 0.2535923
##    13 15.490751 13 0.2777252
##    17 16.975419 17 0.4560327
##    21 25.816281 21 0.2135410

res <- residuals(f12)
checkresiduals(f12)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,Md,M)
## Q* = 27.494, df = 3, p-value = 4.638e-06
##
## Model df: 17.   Total lags used: 20

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags  statistic df   p-value
##     1  0.8037141  1 0.3699852
##     5  8.8046825  5 0.1171129
##     9 12.4726054  9 0.1879577
```

```
##     13 16.5674410 13 0.2198399
##     17 18.0056151 17 0.3884867
##     21 27.6184792 21 0.1513070
```

ARIMA

The auto.arima procedure results in an ARIMA(1,0,0)(1,1,0) model. This model does not shows satisfactory diagnostics. We will now explore some variations starting from this model, and check model fit and forecast accuracy. The code allows us to gather the results of several models.

The first difference applyed suggest to take zero difference, then the seasonal difference suggest zero differences.

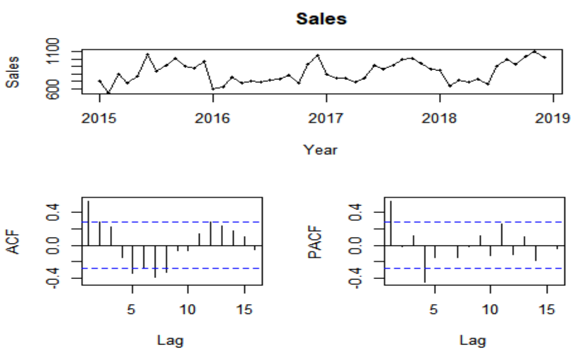
210 110 best AIC

411 112 best MASE training and test set

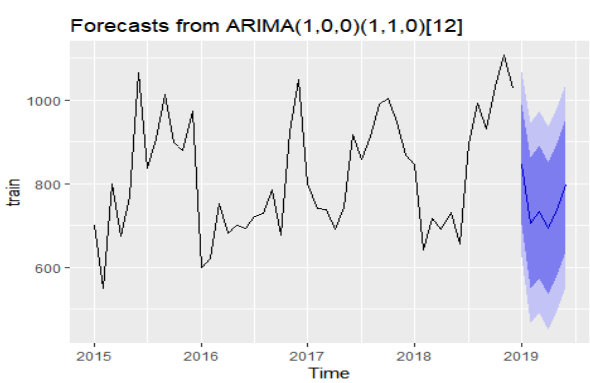
212 112 best RMSE on the training set and 412 012 on test set.

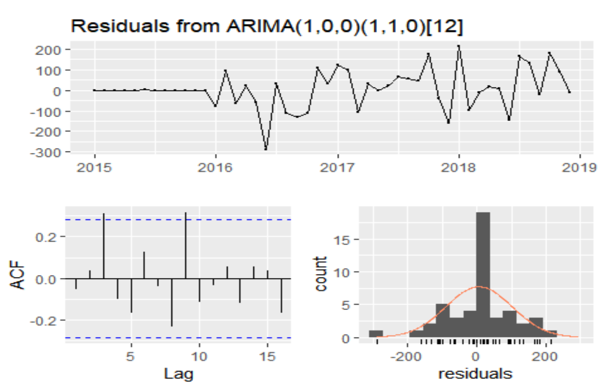
We will compare all test accuracy in the tables below.

```
tsdisplay(train, main="Sales", ylab="Sales", xlab="Year")
```




```
ndiffs(train)

## [1] 0

nsdiffs(diff(train))

## [1] 0

f13 <- forecast(auto.arima(train), h=h)
autoplot(f13)
```






```
accuracy(f13)[,c(2,3,5,6)]
```

```
##       RMSE        MAE       MAPE       MASE
## 94.9414257 66.0963743  8.2727977  0.5719329
```

```r
res <- residuals(f13)
checkresiduals(f13)
```

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(1,1,0)[12]
## Q* = 18.595, df = 8, p-value = 0.01718
##
## Model df: 2.   Total lags used: 10
```

```r
res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)
```

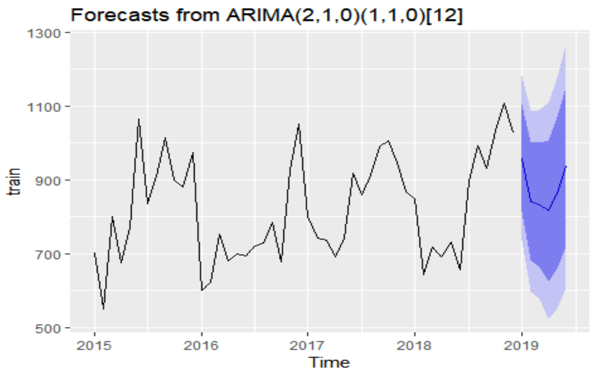
```
##  lags  statistic df    p-value
##     1  0.1425283  1 0.70578041
##     5  7.4268121  5 0.19078299
##     9 17.7348372  9 0.03837754
##    13 19.8779042 13 0.09829679
##    17 22.5493154 17 0.16450570
##    21 33.5829308 21 0.04013984
```

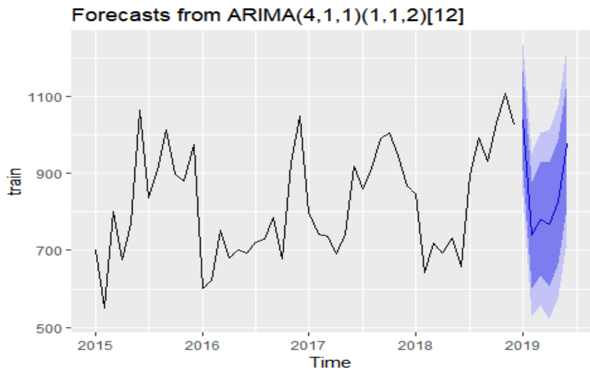
```r
getinfo <- function(x,h,...)
{
  train.end <- time(x)[length(x)-h]
  test.start <- time(x)[length(x)-h+1]
  train <- window(x,end=train.end)
  test <- window(x,start=test.start)
  fit <- Arima(train,...)
  fc <- forecast(fit,h=h)
  a <- accuracy(fc,test)
  result <- matrix(NA, nrow=1, ncol=5)
  result[1,1] <- fit$aicc
  result[1,2] <- a[1,6]
  result[1,3] <- a[2,6]
  result[1,4] <- a[1,2]
  result[1,5] <- a[2,2]
  return(result)
}
mat <- matrix(NA,nrow=54, ncol=5)
modelnames <- vector(mode="character", length=54)
line <- 0
for (i in 2:4){
  for (j in 0:2){
    for (k in 0:1){
      for (l in 0:2){
        line <- line+1
        mat[line,] <- getinfo(train,h=h,order=c(i,1,j),seasonal=c(k,1,l))
        modelnames[line] <- paste0("ARIMA(",i,",1,",j,")(",k,",1,",l,")[12]")
       }
     }
  }
}
colnames(mat) <- c("AICc", "MASE_train", "MASE_test", "RMSE_train", "RMSE_test")
rownames(mat) <- modelnames
```

```
print("best AICc")

## [1] "best AICc"

mat[mat[,1]==min(mat[,1])]

## [1] 368.7597329   0.4403785   1.5515912  80.0502925 204.5689397

which(mat[,1]==min(mat[,1]))

## ARIMA(2,1,0)(1,1,0)[12]
##                        4

print("best MASE_train")

## [1] "best MASE_train"

mat[mat[,2]==min(mat[,2])]

## [1] 388.0836854   0.2791578   1.2130961  51.8914894 167.8901452

which(mat[,2]==min(mat[,2]))

## ARIMA(4,1,2)(1,1,2)[12]
##                       54

print("best MASE_test")

## [1] "best MASE_test"

mat[mat[,3]==min(mat[,3])]

## [1] 388.0836854   0.2791578   1.2130961  51.8914894 167.8901452

which(mat[,3]==min(mat[,3]))

## ARIMA(4,1,2)(1,1,2)[12]
##                       54

print("best RMSE_train")

## [1] "best RMSE_train"

mat[mat[,4]==min(mat[,4])]

## [1] 379.4912117   0.2884214   1.7558909  51.8274642 225.4080770

which(mat[,4]==min(mat[,4]))

## ARIMA(2,1,2)(1,1,2)[12]
##                       18

print("best RMSE_test")

## [1] "best RMSE_test"

mat[mat[,5]==min(mat[,5])]

## [1] 385.5585464   0.3438908   1.2136325  63.4844460 160.1524044

which(mat[,5]==min(mat[,5]))
```

```
## ARIMA(4,1,2)(0,1,2)[12]
##                         51
```

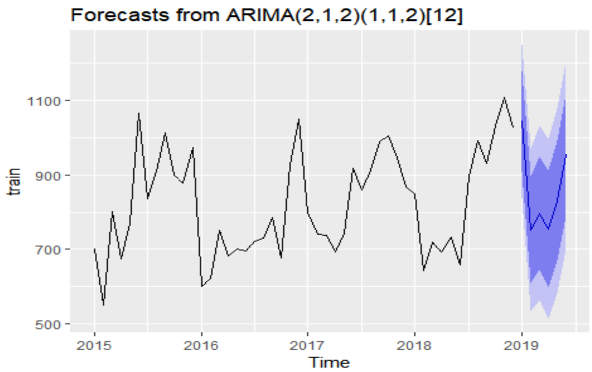
```r
f14 <- forecast(Arima(train, order=c(2,1,0), seasonal=c(1,1,0)), h=h)
autoplot(f14)
```

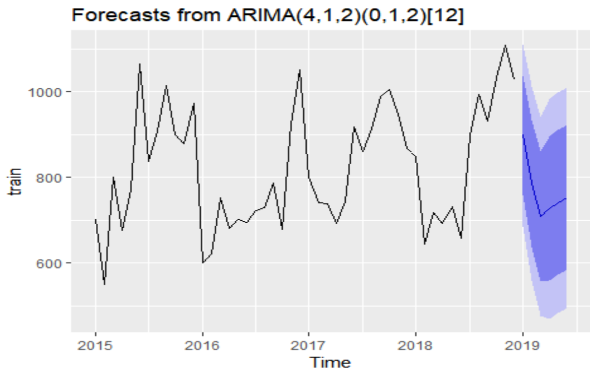





```r
f15 <- forecast(Arima(train, order=c(4,1,1), seasonal=c(1,1,2)), h=h)
autoplot(f15)
```

```r
f16 <- forecast(Arima(train, order=c(2,1,2), seasonal=c(1,1,2)), h=h)
autoplot(f16)
```

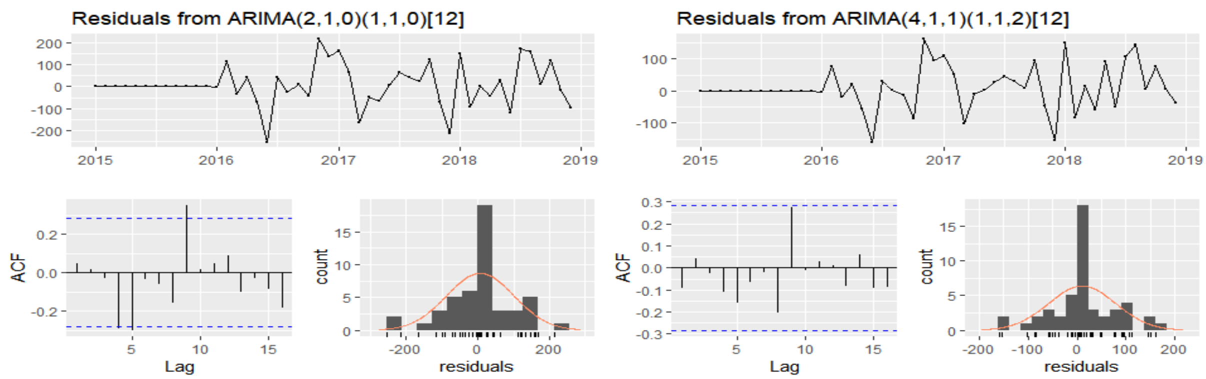





```r
f17 <- forecast(Arima(train, order=c(4,1,2), seasonal=c(0,1,2)), h=h)
autoplot(f17)
```

```r
accuracy(f13)[,c(2,3,5,6)]

##       RMSE        MAE       MAPE       MASE
## 94.9414257 66.0963743  8.2727977  0.5719329

accuracy(f14)[,c(2,3,5,6)]

##       RMSE        MAE       MAPE       MASE
## 93.0237250 63.3558724  7.8029088  0.5482193

accuracy(f15)[,c(2,3,5,6)]

##       RMSE        MAE       MAPE       MASE
## 68.7292105 46.5119244  5.7425639  0.4024683

accuracy(f16)[,c(2,3,5,6)]
```

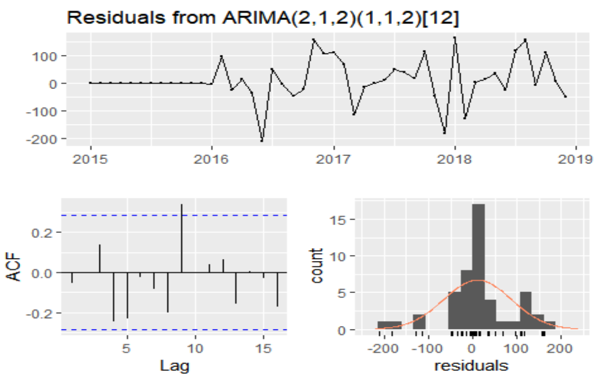
```
##       RMSE        MAE       MAPE       MASE
## 76.2027174 49.2228592  6.0124202  0.4259261

accuracy(f17)[,c(2,3,5,6)]

##       RMSE        MAE       MAPE       MASE
## 66.1188877 45.2832712  5.6061850  0.3918368

res <- residuals(f14)
checkresiduals(f14)
```

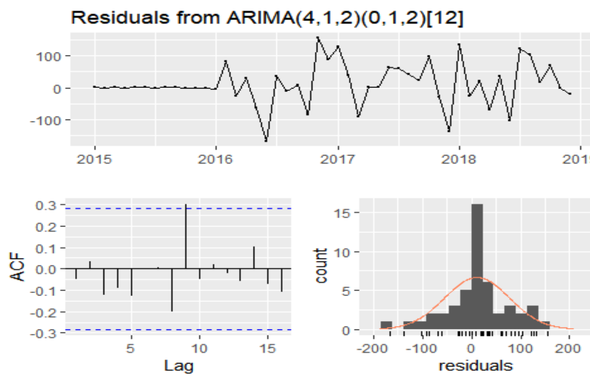



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,0)(1,1,0)[12]
## Q* = 19.077, df = 7, p-value = 0.007948
##
## Model df: 3.   Total lags used: 10

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags  statistic df    p-value
##     1  0.1021332  1 0.74928458
##     5  9.7870618  5 0.08149855
##     9 19.0586724  9 0.02469861
##    13 20.4446014 13 0.08466528
##    17 24.8503354 17 0.09811005
##    21 36.6388943 21 0.01851327

res <- residuals(f15)
checkresiduals(f15)
```

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,1)(1,1,2)[12]
## Q* = 10.104, df = 3, p-value = 0.0177
##
## Model df: 8.   Total lags used: 11

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags  statistic df   p-value
##     1  0.4331007  1 0.5104714
```

```
##     5  2.6478922  5 0.7540758
##     9 10.0406164  9 0.3472071
##    13 10.5766644 13 0.6462344
##    17 12.6517420 17 0.7591696
##    21 24.5454365 21 0.2673877

res <- residuals(f16)
checkresiduals(f16)
```






```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,2)(1,1,2)[12]
## Q* = 17.148, df = 3, p-value = 0.0006589
##
## Model df: 7.   Total lags used: 10

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags  statistic df    p-value
##     1  0.1416295  1 0.70666642
##     5  7.3031934  5 0.19905019
##     9 17.1466119  9 0.04646894
##    13 19.2001022 13 0.11702895
##    17 21.4588714 17 0.20643458
##    21 35.3328812 21 0.02594870

res <- residuals(f17)
checkresiduals(f17)
```

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(0,1,2)[12]
## Q* = 10.517, df = 3, p-value = 0.01464
##
## Model df: 8.   Total lags used: 11

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags  statistic df   p-value
##     1  0.1306813  1 0.7177267
##     5  2.3613619  5 0.7972139
##     9 10.3265771  9 0.3246996
```

```
##    13 10.7721193 13 0.6299012
##    17 15.2027815 17 0.5808782
##    21 24.0135814 21 0.2924009
```

Conclusion

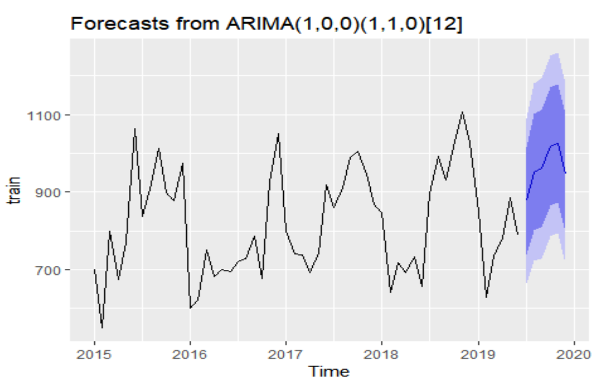
With the table we can see that Arima auto test have best performance overall of MAE, MAPE and MASE. The residual diagnostics results of Arima auto test is also acceptable. Therefore we will use Arima auto test model as final to do forecast to 2020.

```
af1 = accuracy(f1, test)
af2 = accuracy(f2, test)
af3 = accuracy(f3, test)
af4 = accuracy(f4, test)
af5 = accuracy(f5, test)
af6 = accuracy(f6, test)
af7 = accuracy(f7, test)
af8 = accuracy(f8, test)
af9 = accuracy(f9, test)
af10 = accuracy(f10, test)
af11 = accuracy(f11, test)
af12 = accuracy(f12, test)
af13 = accuracy(f13, test)
af14 = accuracy(f14, test)
af15 = accuracy(f15, test)
af16 = accuracy(f16, test)
af17 = accuracy(f17, test)

a.table <- rbind(af1, af2, af3, af4, af5, af6, af7, af8, af9, af10, af11, af12, af13, af14, af15, af16, af
17)
row.names(a.table)<-c("S. Naive training", 'S. Naive test',
                      'STL training', 'STL test',
                      'HW multi train','HW multi test',
                      'HW multi exponential train','HW multi exponential test',
                      'HW damped exponential train','HW damped exponential test',
                      "HW additive train", "HW additive test",
                      'HW addi damped trend train','HW addi damped trend test',
                      'ETS auto training', 'ETS auto test',
                      'ETS MAM training', 'ETS MAM test',
                      'ETS MMM training', 'ETS MMM test',
                      'ETS MAM d training', 'ETS MAM d test',
                      'ETS MMM d training', 'ETS MMM d test',
                      'ARIMA Auto training', 'ARIMA Auto test',
                      'ARIMA 210 110 training', 'ARIMA 210 110 test',
                      'ARIMA 411 112 training', 'ARIMA 411 112 test',
                      'ARIMA 212 112 training', 'ARIMA 212 112 test',
                      'ARIMA 411 012 training', 'ARIMA 412 012 test')
a.table <- as.data.frame(a.table)
print(kable(a.table, caption="Forecast accuracy",digits = 2 ))

##
##
## Table: Forecast accuracy
##
##                                      ME      RMSE       MAE       MPE     MAPE    MASE     ACF1   Theil's U
## ----------------------------  --------  -------  --------  --------  -------  ------  -------  ----------
## S. Naive training                 5.75   149.00    115.57     -1.04    14.35    1.00     0.48          NA
## S. Naive test                    63.23    90.98     68.00      7.62     8.38    0.59     0.62        0.77
## STL training                      0.00    97.67     77.88     -0.77     9.58    0.67    -0.37          NA
## STL test                        -32.66    78.87     69.48     -5.00     9.27    0.60    -0.21        0.68
## HW multi train                    0.77    88.09     68.37     -0.92     8.38    0.59     0.17          NA
```

```
## HW multi test                      -0.35    73.42    70.24    -0.73     8.96  0.61   -0.26          0.59
## HW multi exponential train          0.02    92.74    73.39    -1.31     9.10  0.64    0.19            NA
## HW multi exponential test          -7.72    76.43    71.23    -1.94     9.37  0.62   -0.29          0.55
## HW damped exponential train         5.35    89.42    70.00    -0.40     8.50  0.61    0.02            NA
## HW damped exponential test        -42.56    88.66    76.13    -6.32    10.22  0.66   -0.21          0.77
## HW additive train                  -3.85    91.55    73.17    -1.60     9.01  0.63    0.08            NA
## HW additive test                  -42.33    93.55    84.76    -6.52    11.42  0.73   -0.26          0.78
## HW addi damped trend train          3.52    91.64    71.61    -0.66     8.72  0.62    0.04            NA
## HW addi damped trend test         -35.07    90.01    80.25    -5.61    10.82  0.69   -0.23          0.75
## ETS auto training                  11.75   123.12    93.51    -0.20    11.55  0.81   -0.01            NA
## ETS auto test                    -263.26   275.99   263.26   -35.47    35.47  2.28   -0.07          2.35
## ETS MAM training                   -3.67    90.04    69.96    -1.72     8.77  0.61    0.30            NA
## ETS MAM test                        4.92    79.21    75.33    -0.08     9.58  0.65   -0.25          0.63
## ETS MMM training                   -4.91    90.07    69.62    -1.76     8.73  0.60    0.32            NA
## ETS MMM test                       22.73    84.75    77.58     2.19     9.65  0.67   -0.27          0.65
## ETS MAM d training                  5.01    91.63    72.29    -0.60     8.87  0.63    0.15            NA
## ETS MAM d test                    -29.23    92.02    85.71    -4.83    11.46  0.74   -0.18          0.78
## ETS MMM d training                  3.21    90.62    71.01    -0.75     8.66  0.61    0.13            NA
## ETS MMM d test                    -38.76    87.64    76.30    -5.91    10.25  0.66   -0.20          0.76
## ARIMA Auto training                 6.16    94.94    66.10    -0.04     8.27  0.57   -0.05            NA
## ARIMA Auto test                    25.58    77.08    54.35     2.50     6.98  0.47    0.21          0.67
## ARIMA 210 110 training              6.67    93.02    63.36     0.25     7.80  0.55    0.04            NA
## ARIMA 210 110 test                -96.44   122.63   104.04   -13.42    14.27  0.90    0.08          0.95
## ARIMA 411 112 training              9.98    68.73    46.51     0.79     5.74  0.40   -0.09            NA
## ARIMA 411 112 test                -76.89   122.25   101.50   -10.27    13.11  0.88    0.05          0.75
## ARIMA 212 112 training             10.66    76.20    49.22     0.87     6.01  0.43   -0.05            NA
## ARIMA 212 112 test                -77.26   121.49   104.88   -10.39    13.63  0.91    0.16          0.74
## ARIMA 411 012 training             10.96    66.12    45.28     0.90     5.61  0.39   -0.05            NA
## ARIMA 412 012 test                  9.73    94.39    78.44     0.17    10.43  0.68    0.33          0.77
```

Final model

```r
train <- window(to, start=c(2015,1),end=c(2019,6))

f <- forecast(auto.arima(train), h=6)
autoplot(f)
```




```r
f$mean
```

```
##             Jul       Aug       Sep       Oct       Nov       Dec
## 2019  877.6836  952.2301  961.1981 1018.3185 1026.5181  946.1785
```