Time Series Project

Contents

---

1. Data Exploration

Describe your time series and the source of the series. The series came from FRED data on beer, wine, and liquor sales - Looking just at the beer sales specifically - Monthly data from 1/1/1992 to 12/1/2018

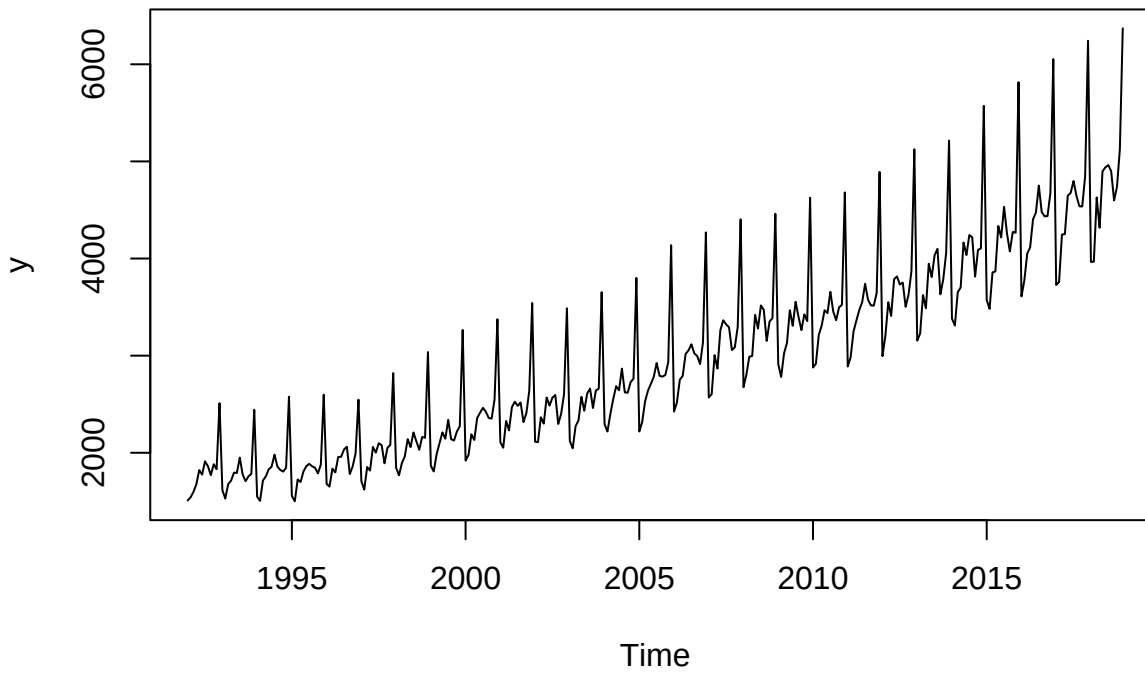
```
library(TTR)
library(forecast)
```

Import the series. Note: Make sure you specify the correct starting time and frequency for the series.

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

```
df <- read.csv("~/Applied Analystics SAS Prog/mymath475/BeerWineLiquor.csv")
y <- ts(df$beer, start = c(1992,1), end = c(2018, 12), freq = 12)
```

```
plot(y)
```



Plot the time series.

Comment on the trend of the series (does the series have trend? upward or downward? from when to when, etc).

- The series seems to have upward trend from start to finish (1992 through 2018)

Comment on the seasonality of the series (does the series have seasonal component? if so, what is the seasonal period, etc.)

- There appears to be a seasonal component of the series, where every year the sales start low, gradually increase throughout the course of the year, but then harshly decrease again at the start of the new year

Comment on the stationality of the series. Notice that trend or seasonal series is not stationary, being stationary implying having constant variance and constant mean

- Because there is a seasonal component to this series, the series cannot be stationary

1.1 Smoothing

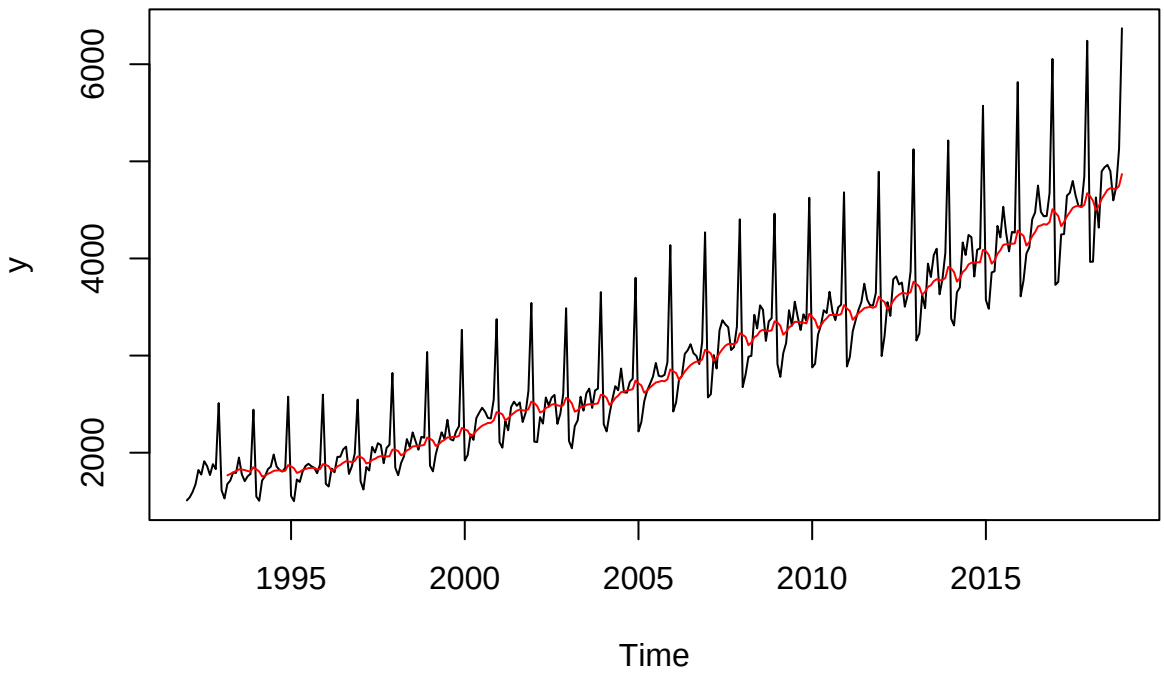
***Moving Average*: Explain the main idea of Moving average. Apply the MA method on the series. Plot the original series and its MA smoothing.**

- Main idea of moving average: Moving Average (MA) creates a new series by taking the average of the most recebt observations from the original series, where a larger k will smooth the series more strongly

```r
plot(y)

# create a moving average series
k = 15
y_sma = SMA(y, n = k)

#plot
lines(y_sma, col = "red")
```

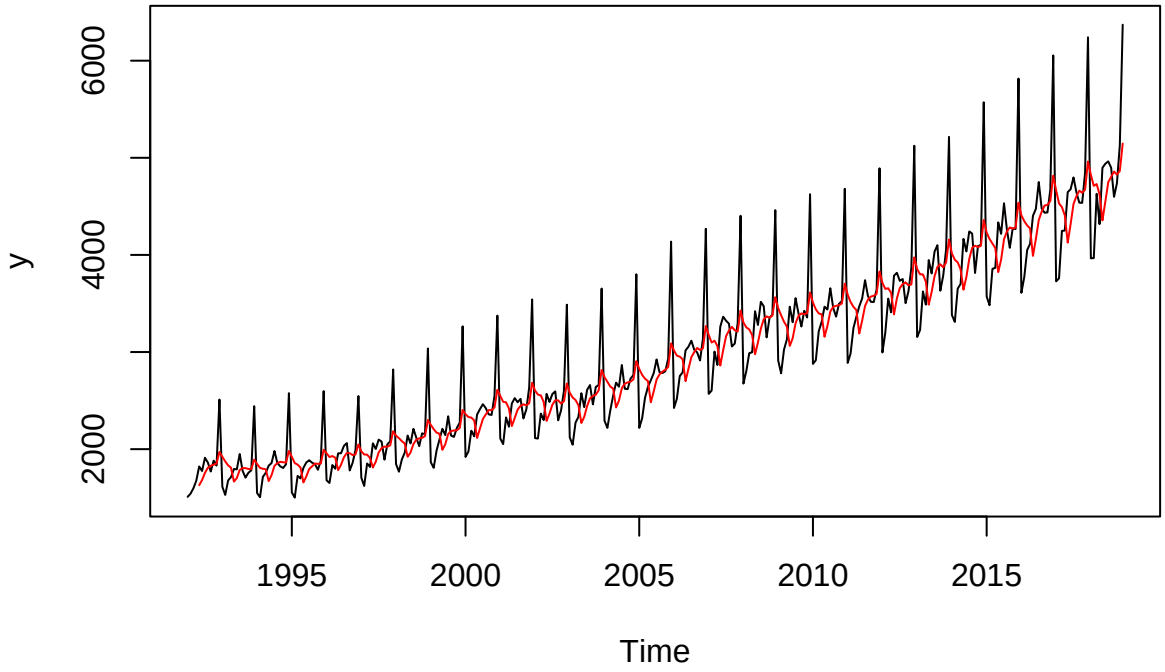



```r
plot(y)

# Try smaller k value
k = 5
y_sma = SMA(y, n = k)

lines(y_sma, col = "red")
```

Try a few different moving average k and comment on the trend of the series through the

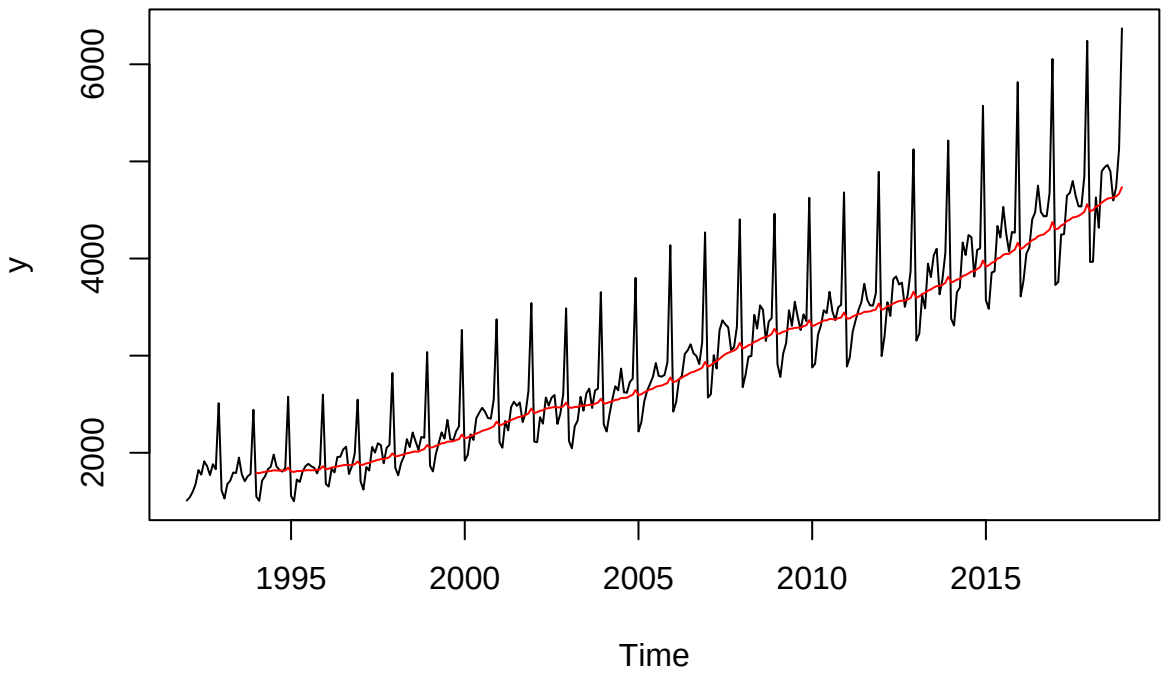


smoothing curve.

```
plot(y)

# Try larger k value
k = 25
y_sma = SMA(y, n = k)

lines(y_sma, col = "red")
```



What values of k best reveal the series pattern?

- The value k=5 best reveals the series pattern

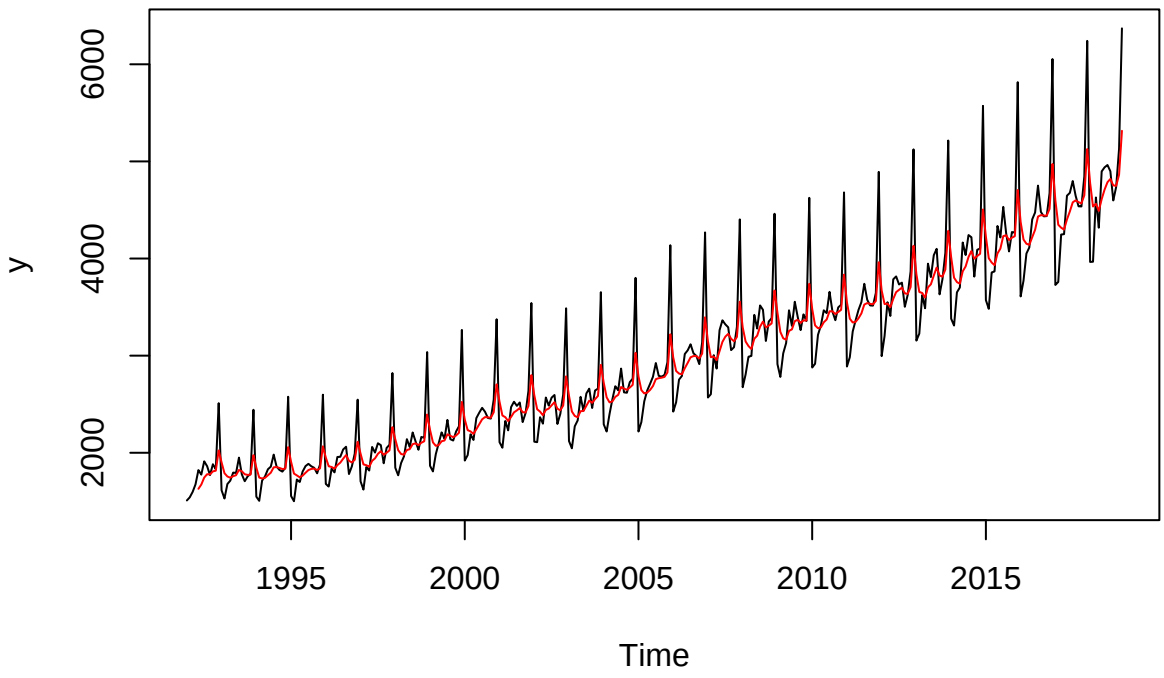
***Exponential Smoothing* (ES): Explain the main idea of exponential smoothing. Apply the ES method on the series. Plot the original series and its ES smoothing.**

- Main idea of exponential smoothing: Exponential Smoothing (ES) controls the weights of the recent observations by weight w
- A smaller w smooths more lightly, where a larger w smooths the series more strongly

```
plot(y)

# create a moving average series
w = .7
y_ema = EMA(y, ratio = 1-w)

# plot the moving average series
lines(y_ema, col = "red")
```

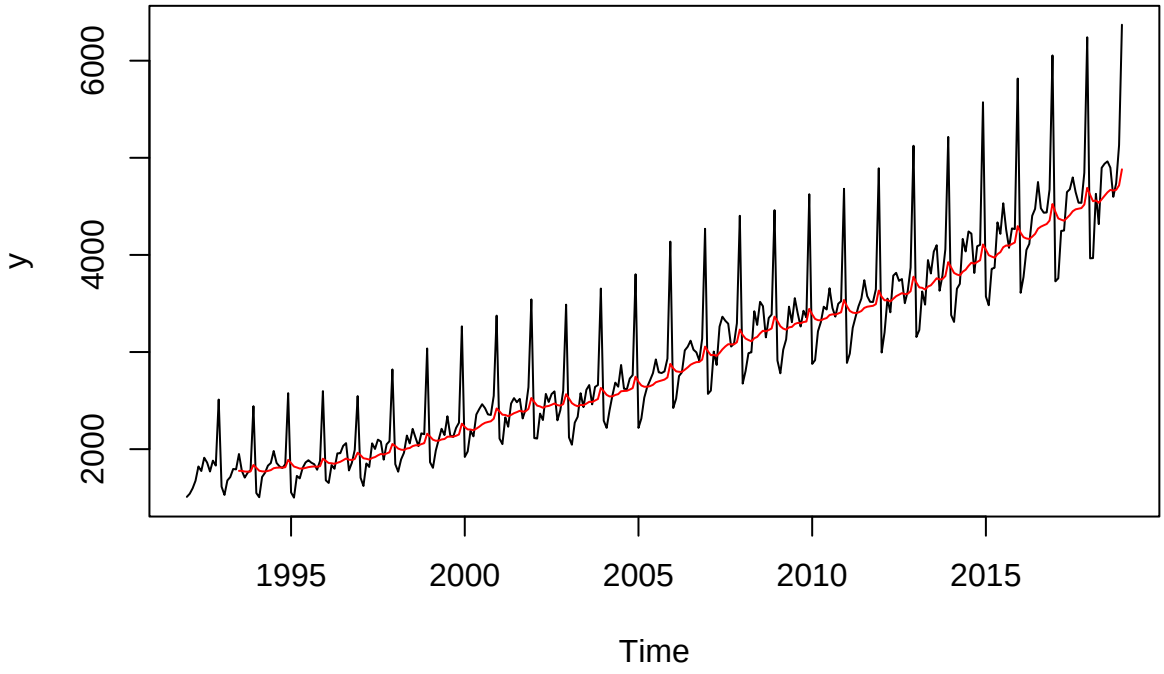



```
plot(y)

# Try larger weight
w = 0.9
y_ema = EMA(y, ratio = 1-w)

# plot
lines(y_ema, col = "red")
```

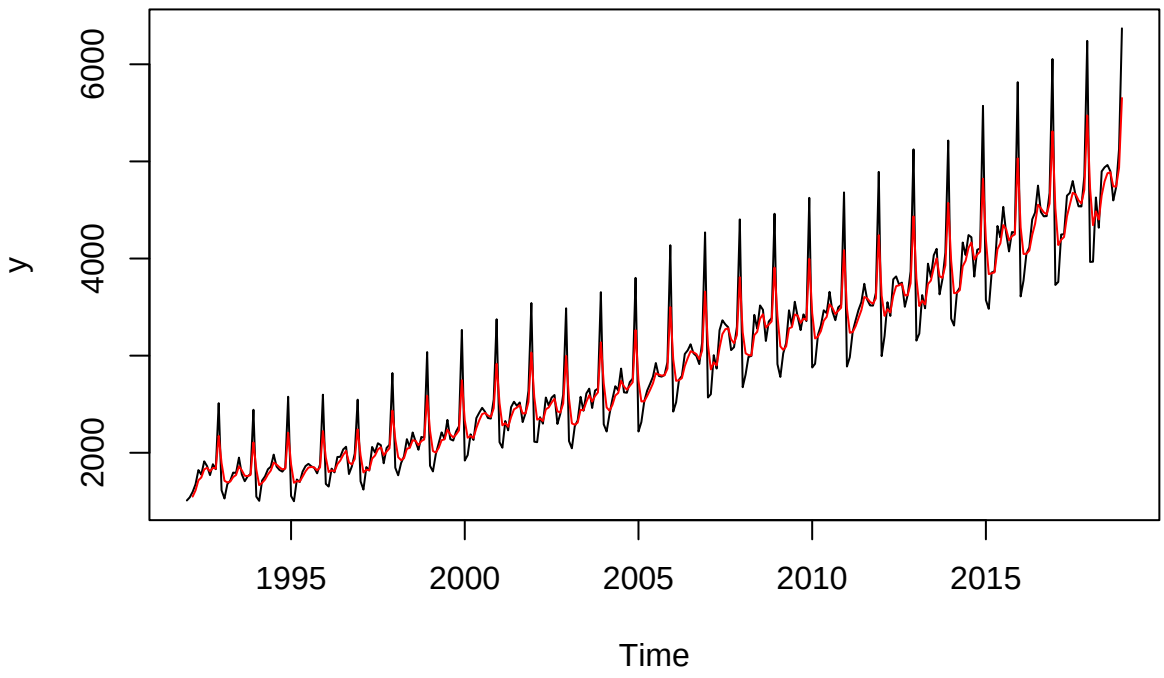
Try a few different weights comment on the trend of the series through the smoothing curve.




```
plot(y)

# Try smaller weight
w = .5
y_ema = EMA(y, ratio = 1-w)

# plot
lines(y_ema, col = "red")
```



What values of the weight best reveal the series' pattern?

- The weight value w=.7 best reveals the series' pattern

Compare the ES curve and the MA curve above.

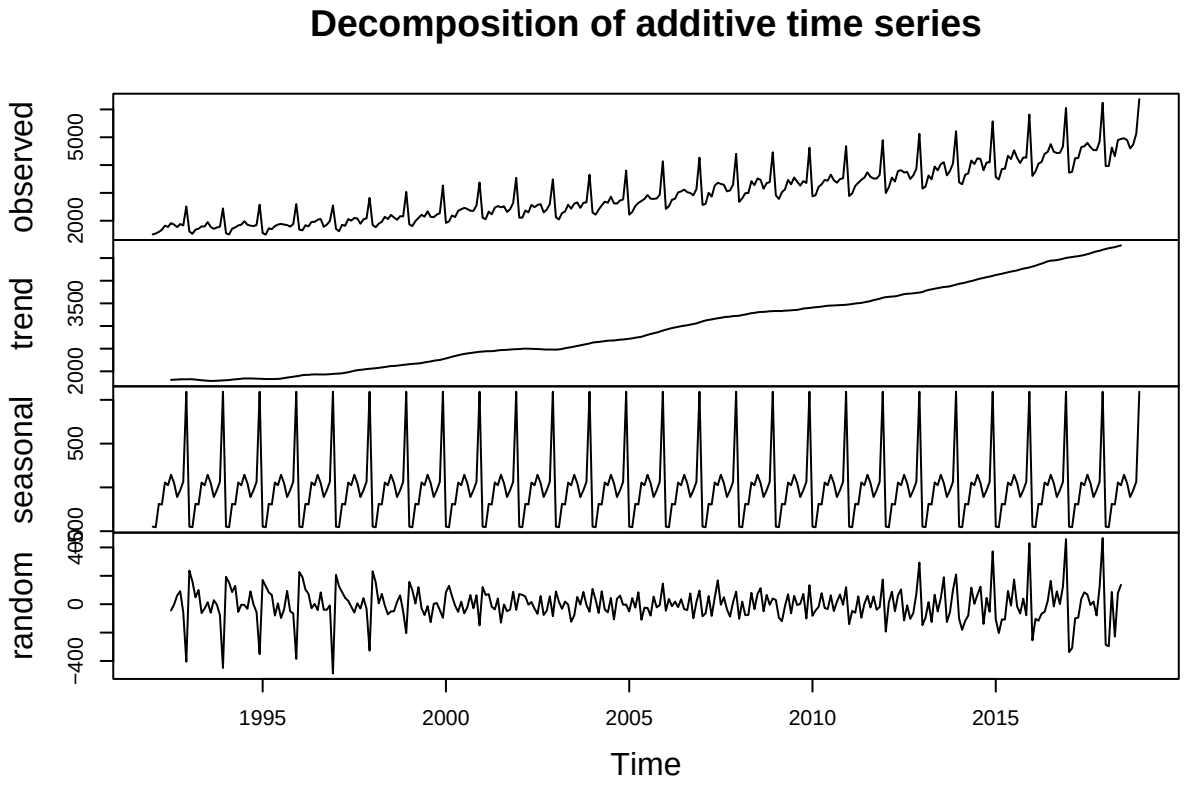
- The ES curve has a bit smoother of a red line, but still follows the peaks and dips of the original curve
- The MA curve has a bit more harsh of a red line, where it follows the peaks and dips of the original curve closely, but has more exaggerated points at the max and min points

1.2 Decomposition

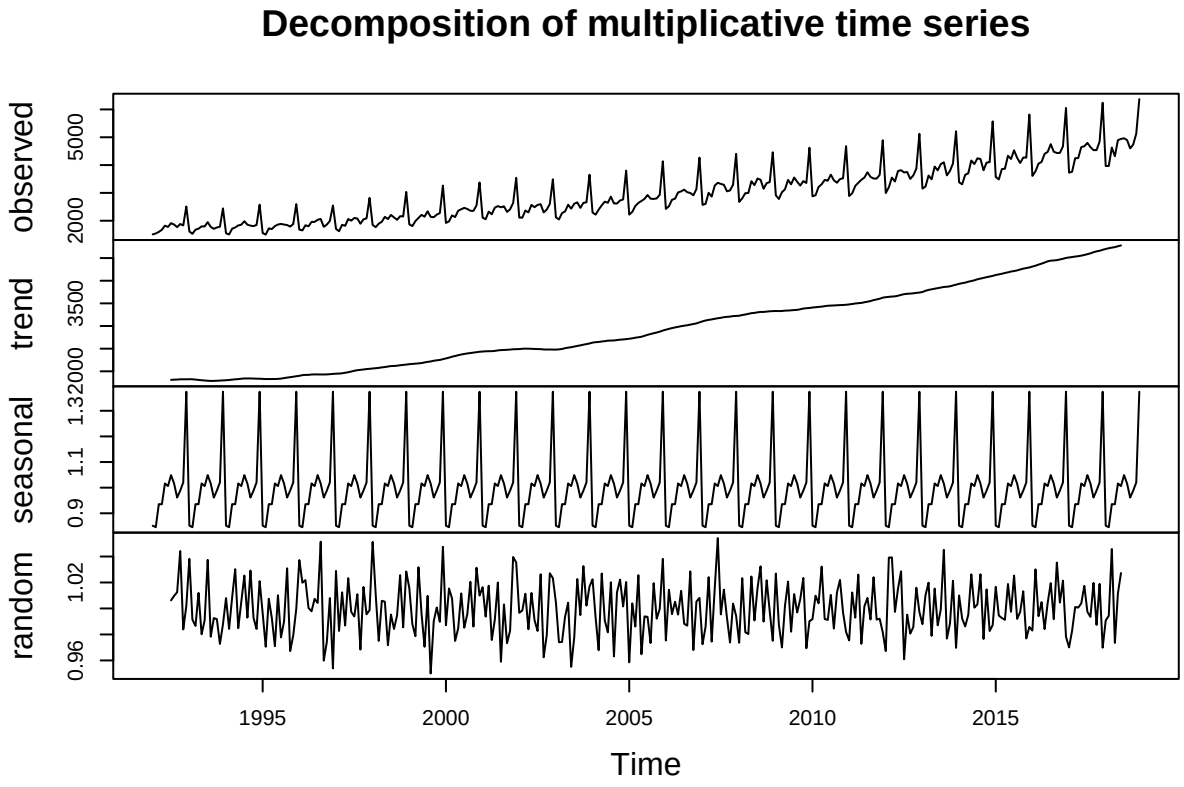
Classical Decomposition. **Explain the idea of Classical Decomposition in your own words. Apply the classical decomposition on the series for additive model and multiplicative model.**

- Idea of classical decomposition: Classical decomposition smooths the data using centered moving average (CMA) of the order equal to the periodicity of the data, calculates the de-trended series, averages the values for each period, and then subtracts from the series the trend and seasonal component estimations.

```
# Additive model
ourDecomposition <- decompose(y, type="additive")
plot(ourDecomposition)
```




```
# Multiplicative model
ourDecomposition <- decompose(y, type="multiplicative")
plot(ourDecomposition)
```



***STL Decomposition*. Explain the idea of STL Decomposition in your own words.**

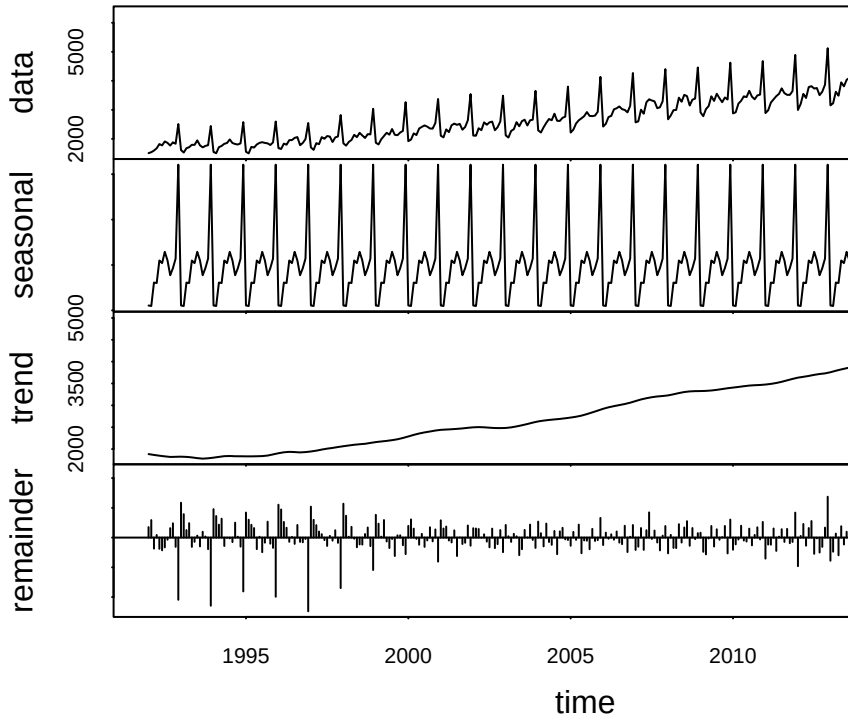
- Idea of STL Decomposition: STL uses Locally Estimated Scatterplot Smoothing (LOESS) rather than the CMA to estimate the trend, and is a versatile method for decomposing time series.

What is the main difference between the classical decomposition and STL Decomposition.

- The main difference is that classical decomposition uses the Central Moving Average to estimate the trend, where the STL decomposition uses the Locally Estimated Scatterplot Smoothing.

```
ourDecomposition <- stl(y, s.window = "periodic")

plot(ourDecomposition)
```



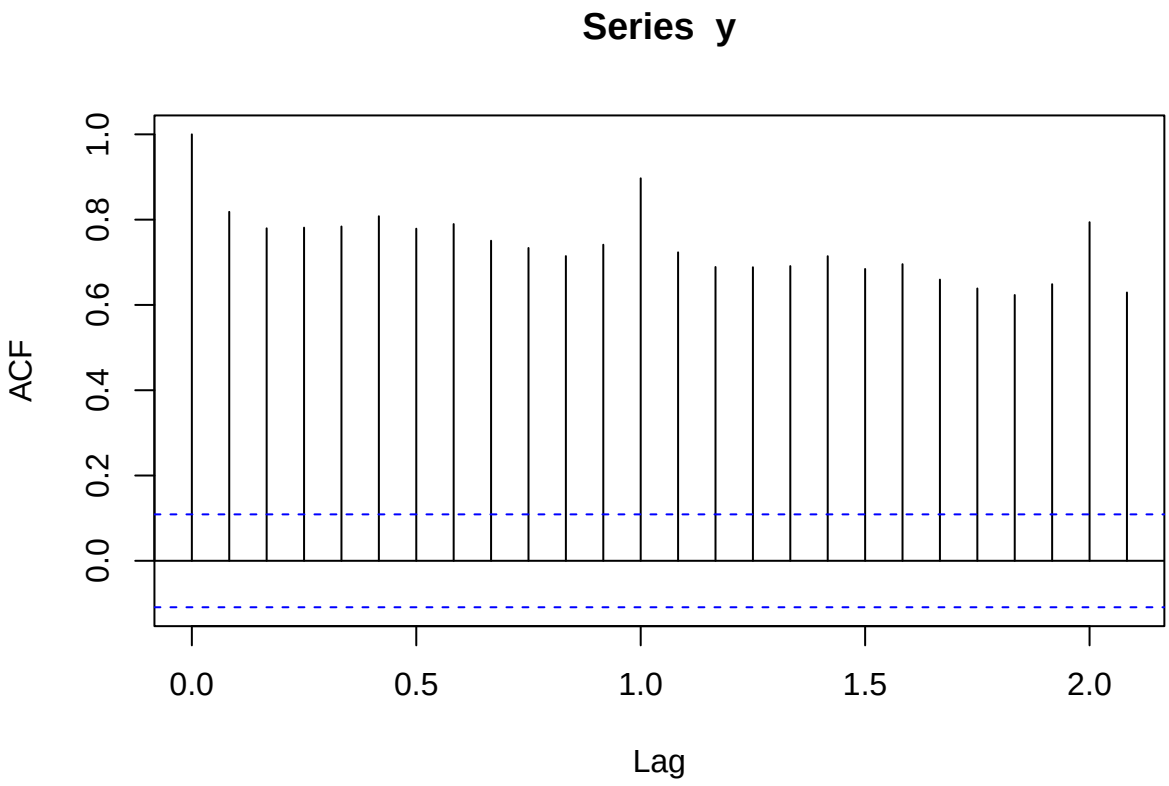
Apply STL Decomposition on the series.

1.3 Auto-correltion

Explain what the Auto-correlation Function (ACF) is and plot the ACF of the series.

- Auto-correlation Function (ACF): This is the function for the autocorrelation lag with lag k, and is the correlation between the time series Yt and Yt-k
- The components of the function include the value of the series at time t (Yt), the mean of the series, and the number of observations in a formula

```
acf(y)
```



What is the first value of the ACF (ACF when the lag is 0)

- The autocorrelation lag with lag 0 is ALWAYS 1

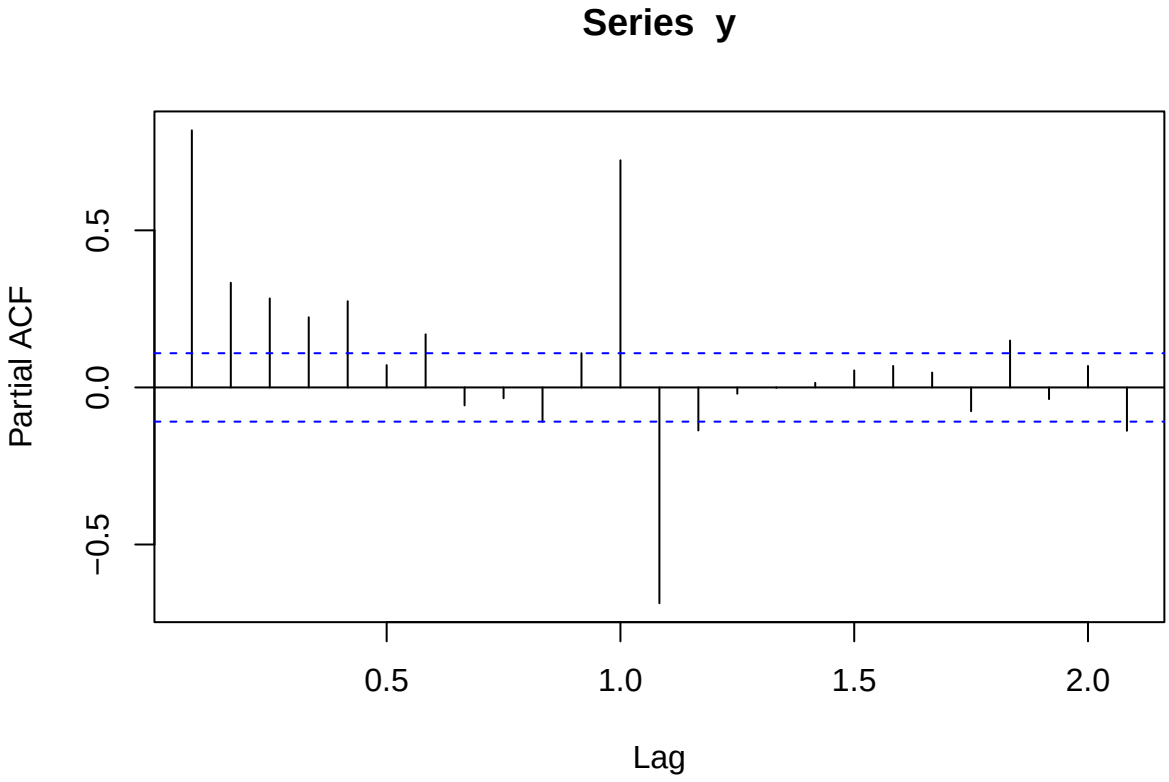
Is the series stationary according to the look of the ACF?

- This series is not stationary according to the look of the ACF, because the value does not quickly get to the value 0
- In addition, there appears to be some seasonality and trend, so it can't be stationary

Explain what the Partial Auto-correlation Function (PACF) is and plot the PACF of the series.

- The PACF is the correlation of a time series with itself at different lags, after removing the effects of the previous lags (where the ACF doesn't remove the effect of the previous).

```
pacf(y)
```



2. Modelling

In addition some baseline models, We will consider two models the HoltWinters and the ARIMA models. The baseline models are

- Mean/Average model: forecast by the average of the training series
- Naive Model: forecast by the last observation of the series
- Seasonal Naive Model: forecast by the values of the last season
- Random Walk with drift Model: Drawing the line from the first to the last observation

To evaluate the models, We will use a machine learning approach, which is use a portion of the data for training and the remainder data for testing. We also perform residual analysis for the models.

2.1 Model Training

Follow the sample codes to.

- Split the original series into the training series and the testing series
- Train all six models

```r
# data partition

# set the proportion of the test set
p = .2

nValid <- round(.2*length(y))
nTrain <- length(y) - nValid
train.ts <- window(y, start = c(1992, 1), end = c(1992, nTrain))
valid.ts <- window(y, start = c(1992, nTrain + 1), end = c(1992, nTrain + nValid))

# Modeling
# baseline models

# average method: forecast by the average of the training series
model1 = meanf(train.ts, h = nValid, level = 0)

# naive: forecast by the last observation of the series
model2 = naive(train.ts, h = nValid, level = 0)

# seasonal naive: forecast by the last season
model3 = snaive(train.ts, h = nValid, level = 0)

# drift: drawing the line from the first to the last observation
model4 = rwf(train.ts, h = nValid, level = 0, drift = TRUE)



# more advanced model
model5 = HoltWinters(train.ts, alpha=TRUE,
                             beta=TRUE,
                             gamma=TRUE)
model6 = auto.arima(train.ts)
```

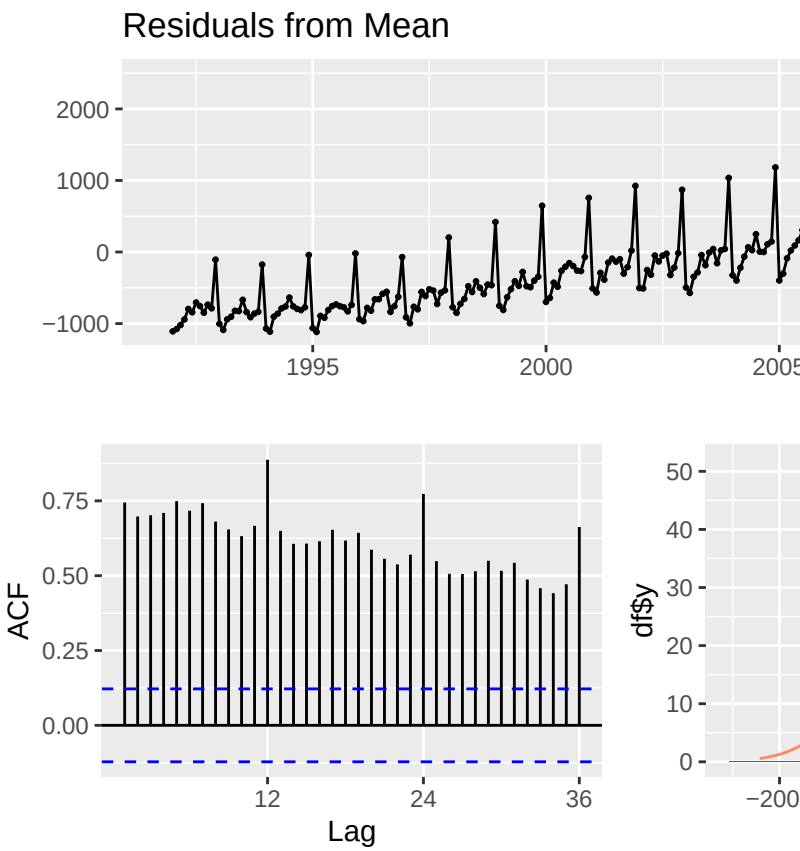
2.2 Residual Analysis

A good model would have residuals looks like white-noise of a mean zero normal distribution, which means

- The residuals should have no autocorrelation.
- The residuals should have a mean zero
- The residuals should have constant variance
- The residuals should be normally distributed

```r

checkresiduals(model1)
```

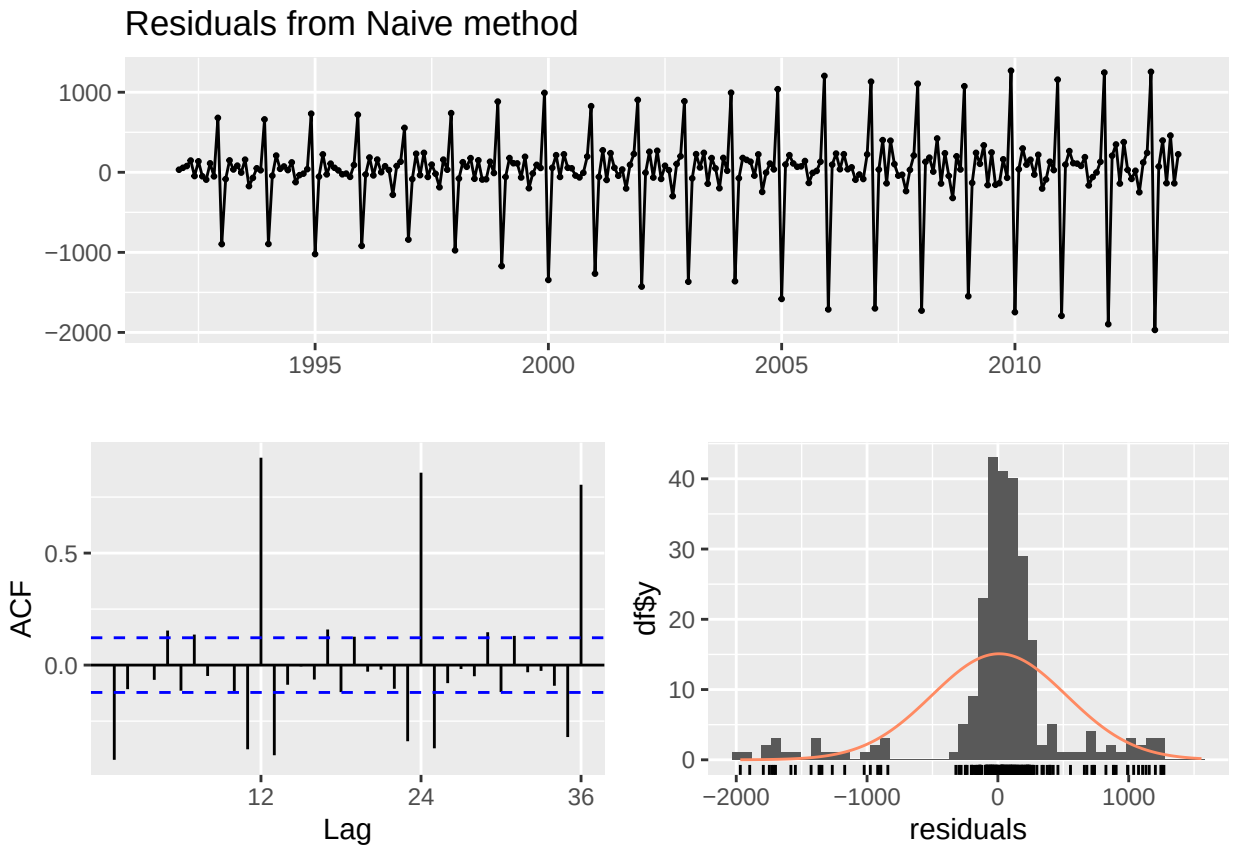


Report the residuals analysis of all the models.

```
##
##  Ljung-Box test
##
## data:  Residuals from Mean
## Q* = 2957.1, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
#This model isn't 'good' because the ACF is not between the blue lines and doesn't seem very normally d
```

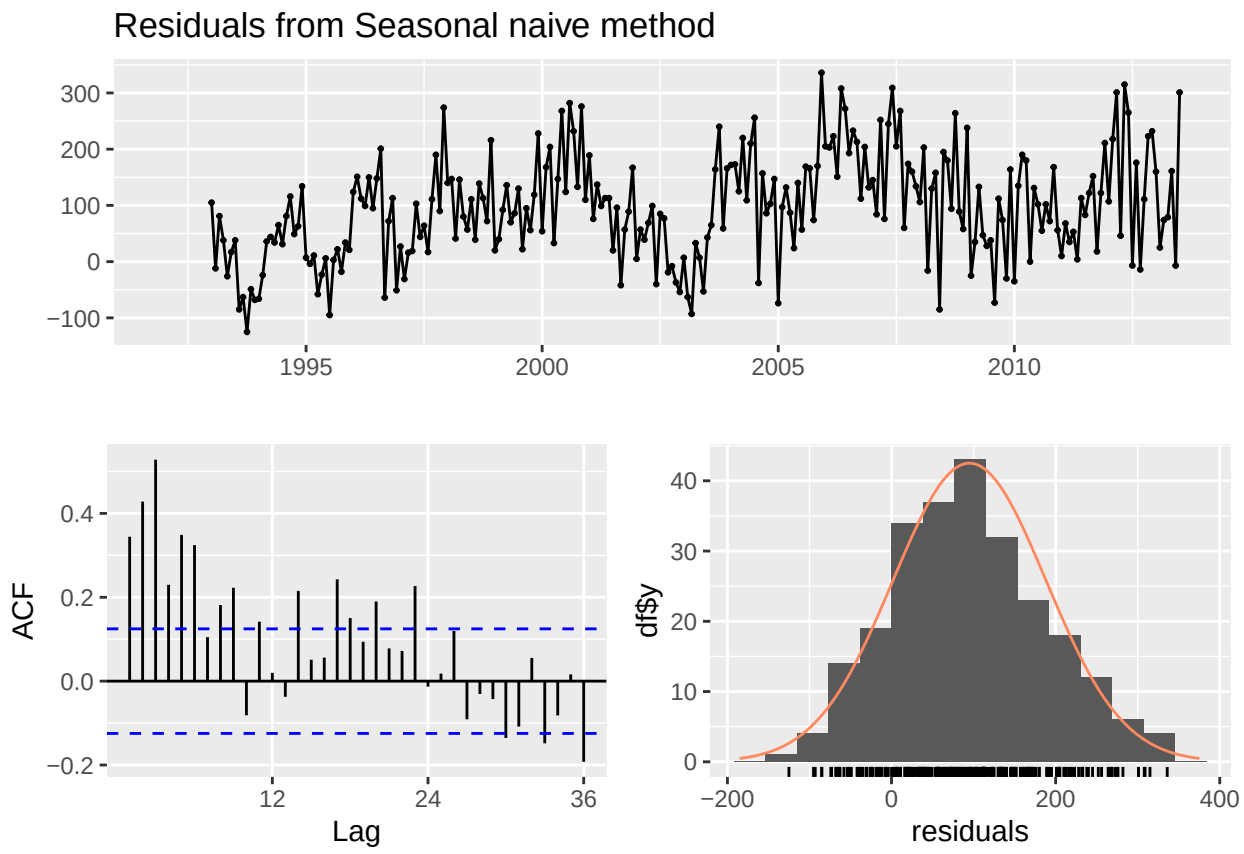
```
checkresiduals(model2)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Naive method
## Q* = 652.9, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
#This model is fairly 'good' because the ACF is mostly between the blue lines and seems to be more norm
```

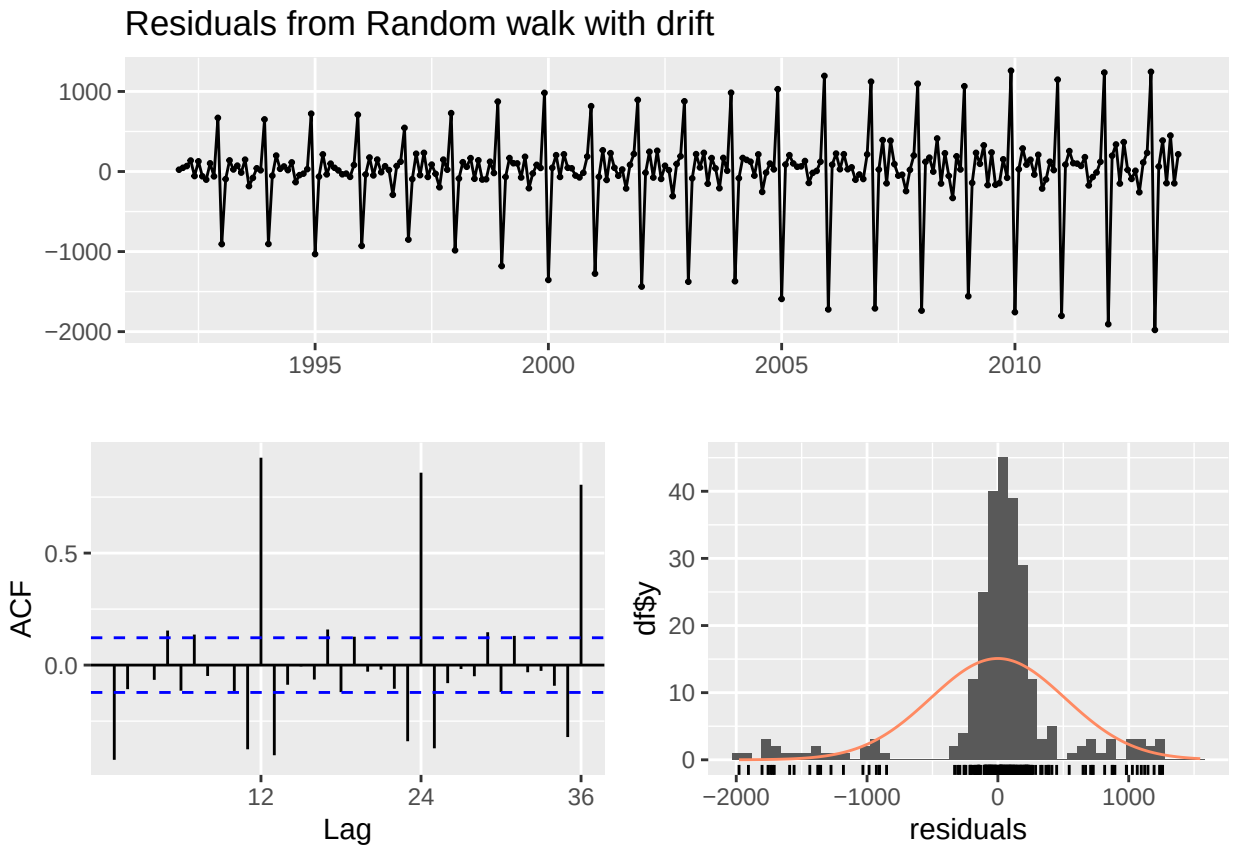
```
checkresiduals(model3)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 313.56, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
#This model is 'good' because the ACF is somewhat between the blue lines and does seem to be very norma
```

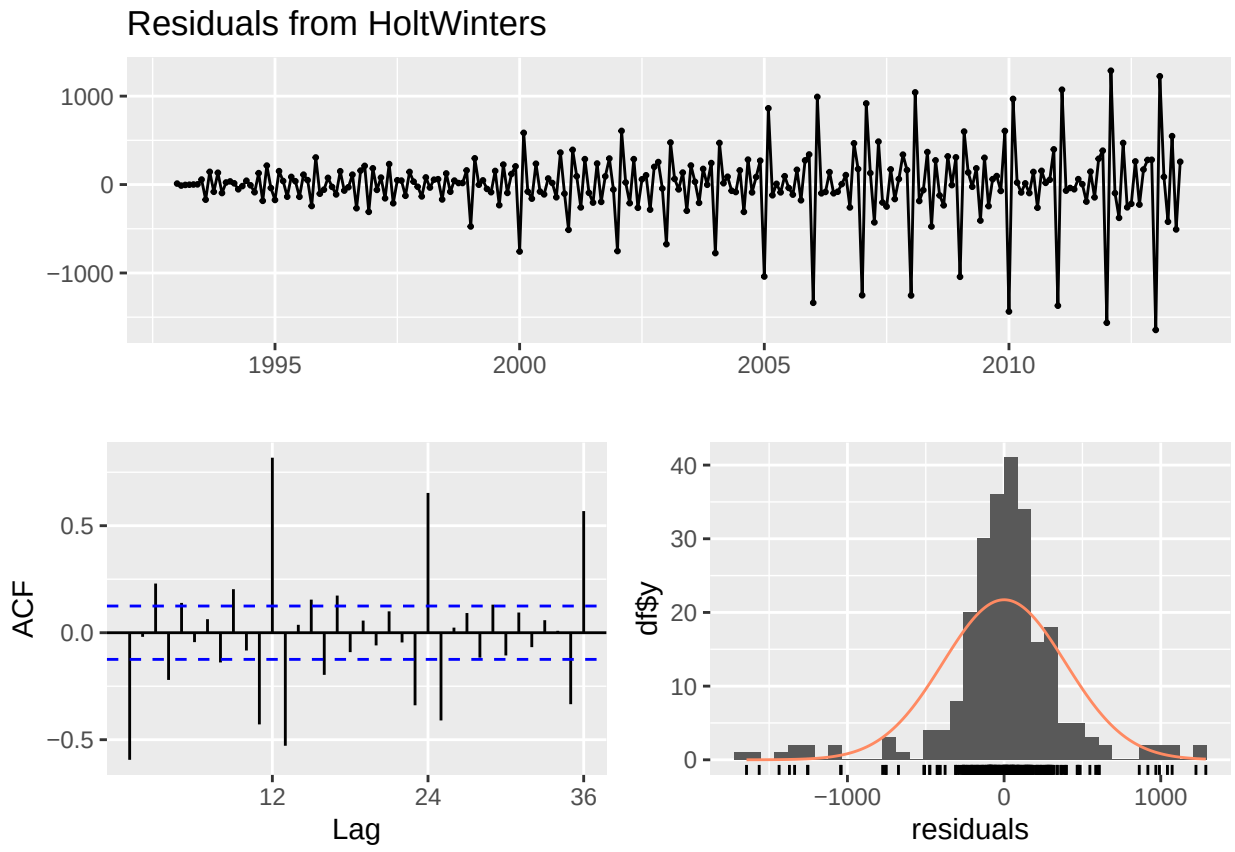
```
checkresiduals(model4)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Random walk with drift
## Q* = 652.9, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
#This model is fairly 'good' because the ACF is somewhat between the blue lines and seems to be centere
```

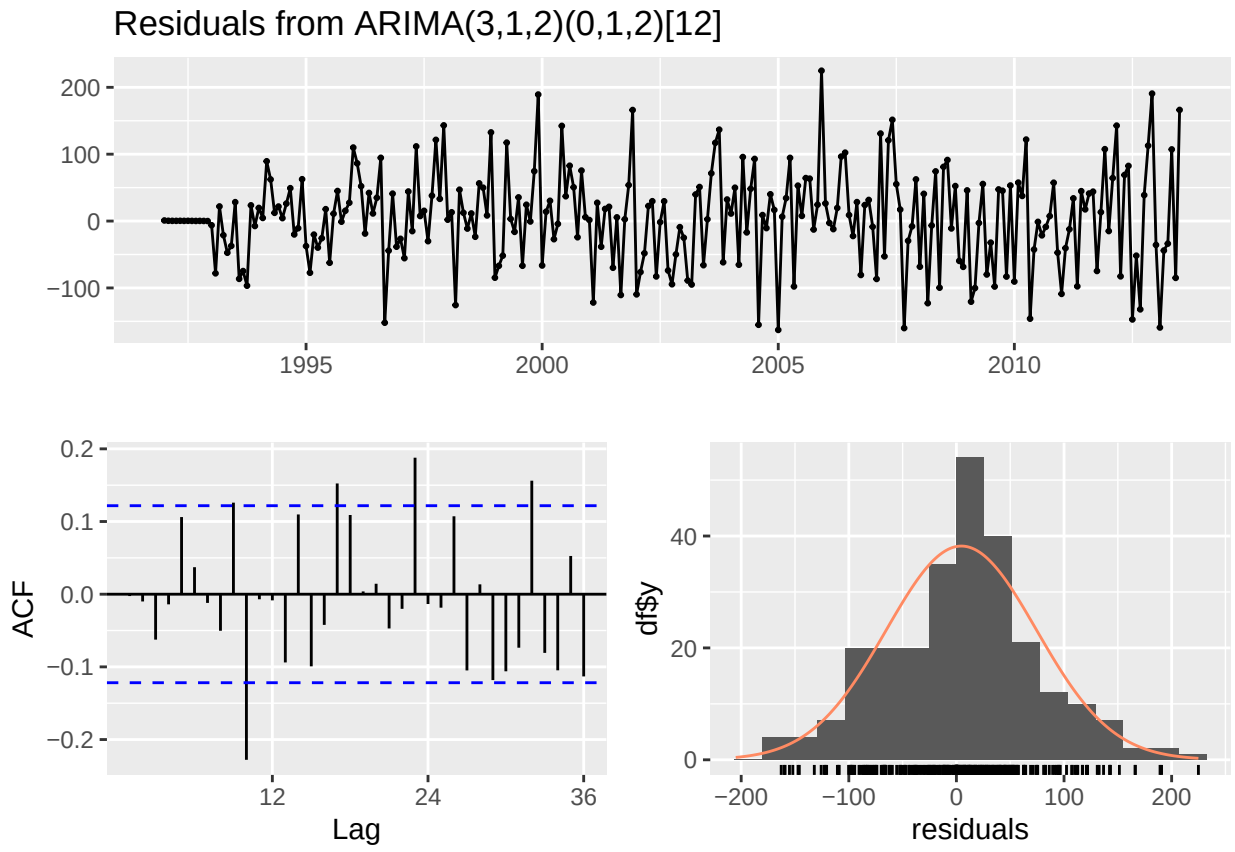
```
checkresiduals(model5)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from HoltWinters
## Q* = 615.58, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
#This model is fairly 'good' because the ACF is mostly between the blue lines and seems to be normally
```

```
checkresiduals(model6)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,1,2)(0,1,2)[12]
## Q* = 53.369, df = 17, p-value = 1.248e-05
##
## Model df: 7.   Total lags used: 24
```

```
#This model is 'good' because the ACF is almost always between the blue lines and does seem to be norma
```

Which model would you consider good models? Some indications of a good model could be

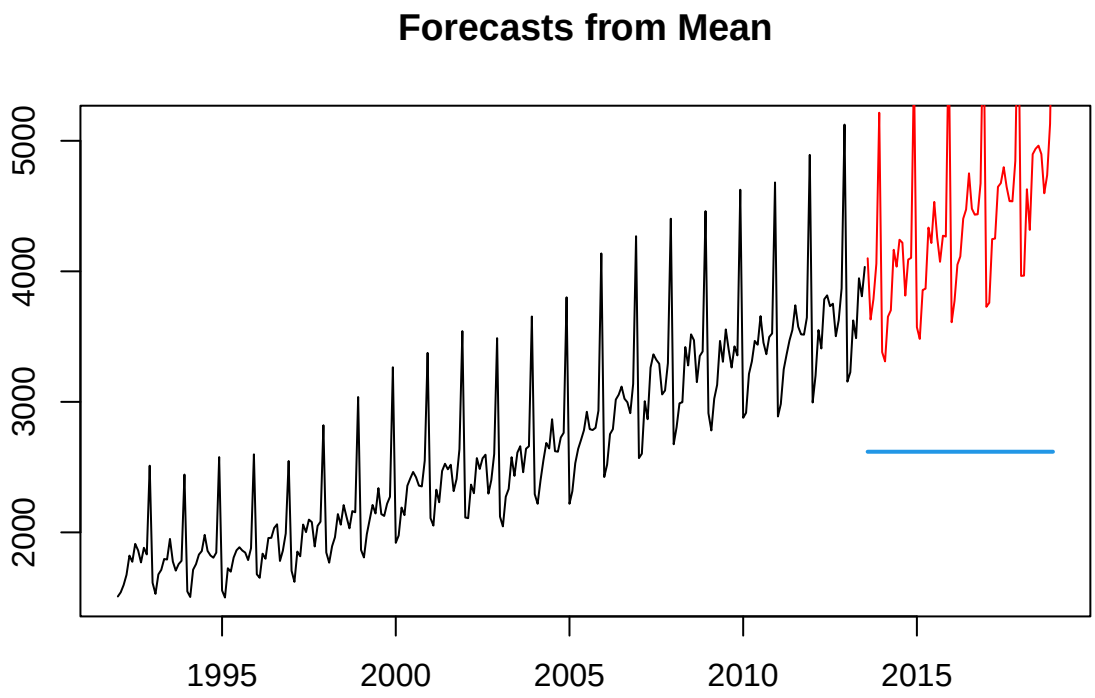
- The ACF is within the blue strip
- The distribution of the residual follows a bell curve
- The p-value of the Ljung-Box test is large
- While model 3 looks to have the distribution of the residual that best follow a bell curve, I would consider model 6 a good model because the ACF is mostly within the blue strip compared to the other models, the distribution of the residual follows the bell curve well, and has a good p-value.

2.3 Testing Accuracy

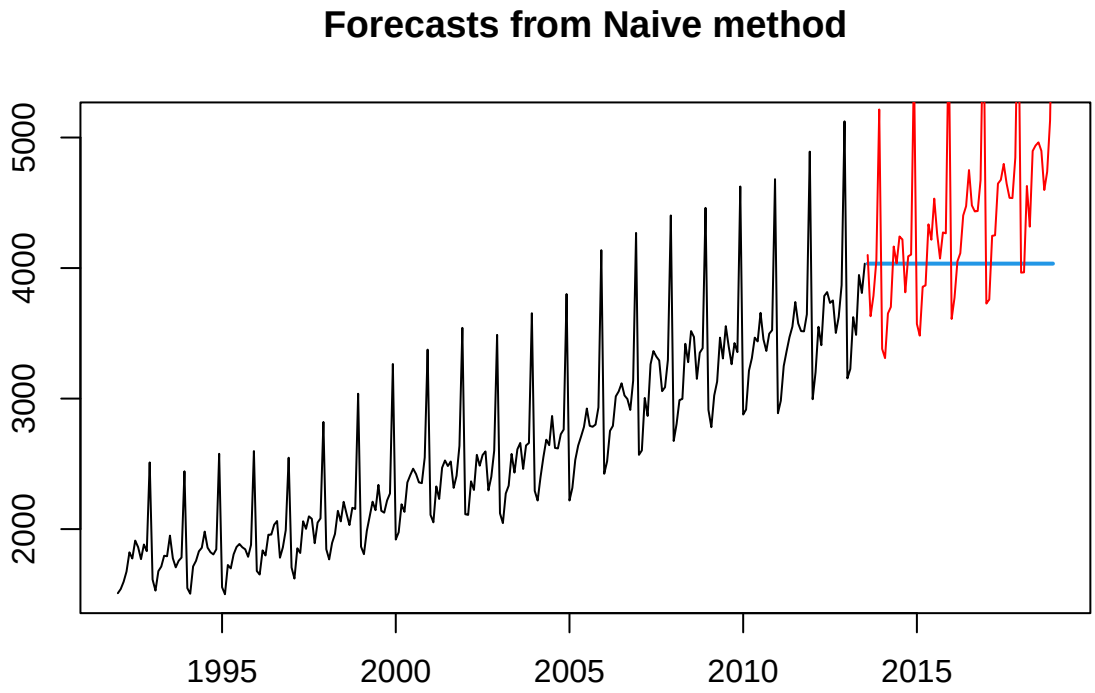
```r
# forecasting
forecast1 = forecast(model1, h = nValid, level = 0)
forecast2 = forecast(model2, h = nValid, level = 0)
forecast3 = forecast(model3, h = nValid, level = 0)
forecast4 = forecast(model4, h = nValid, level = 0)
forecast5 = forecast(model5, h = nValid, level = 0)
forecast6 = forecast(model6, h = nValid, level = 0)


# plotting forecast
plot(forecast1)
lines(valid.ts, col = 'red')
```

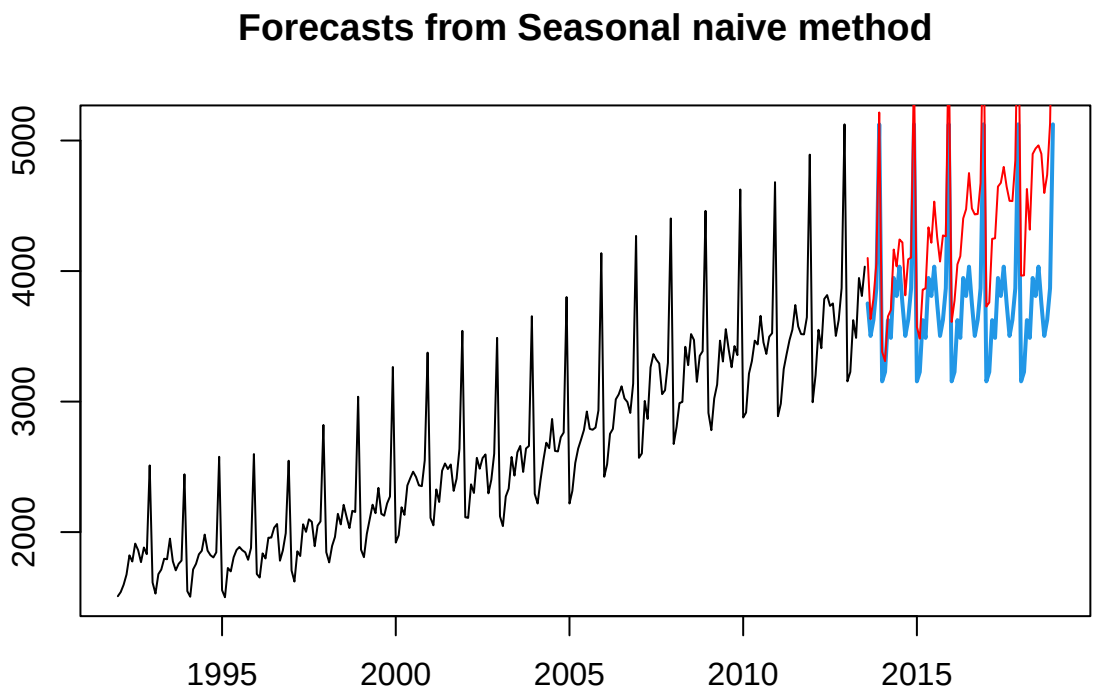
Calculate the forecast of all the models on the validation period. Calculate the Mean absolute percentage error (MAPE) of all the models. Plot the forecast of all models.



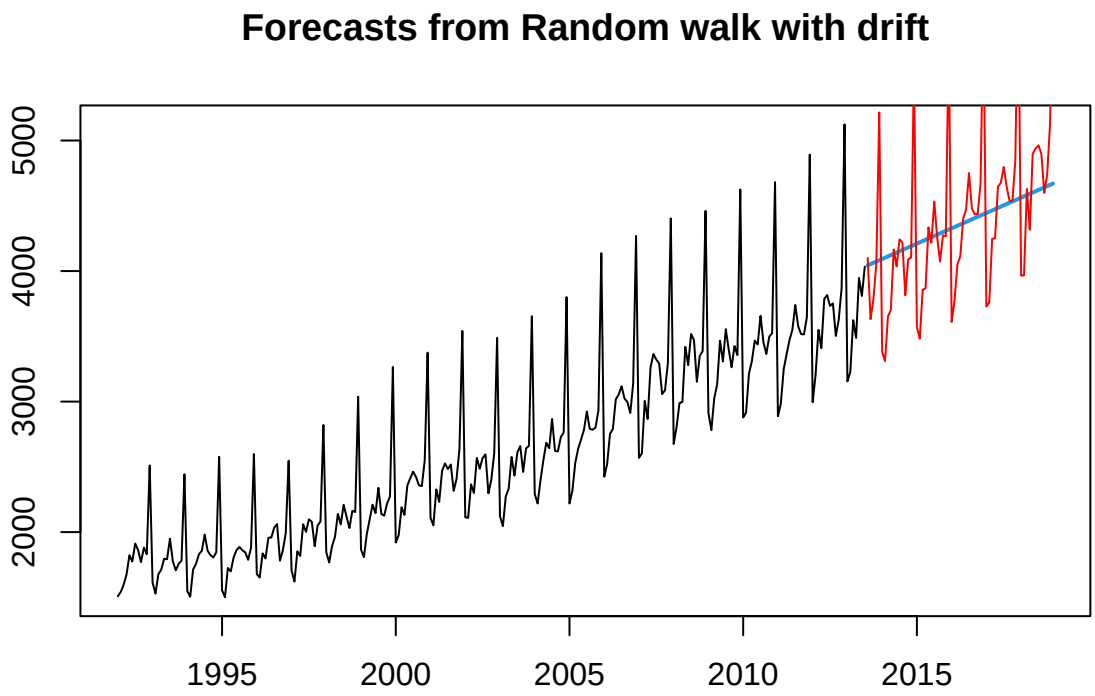

```r
plot(forecast2)
lines(valid.ts, col = 'red')
```



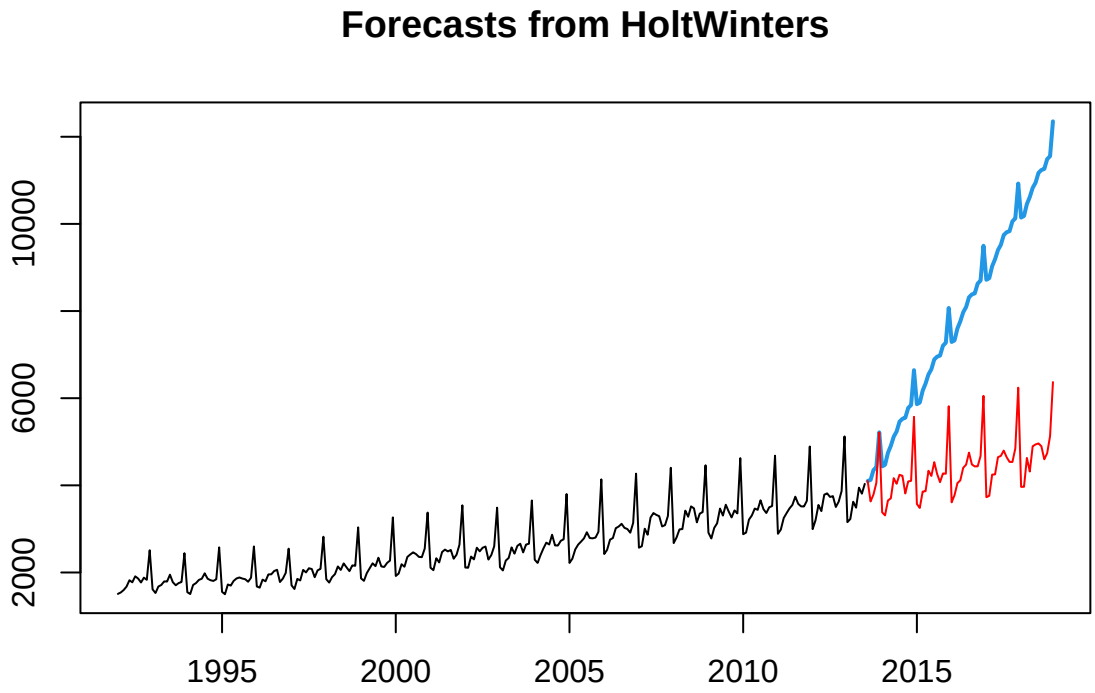

```
plot(forecast3)
lines(valid.ts, col = 'red')
```



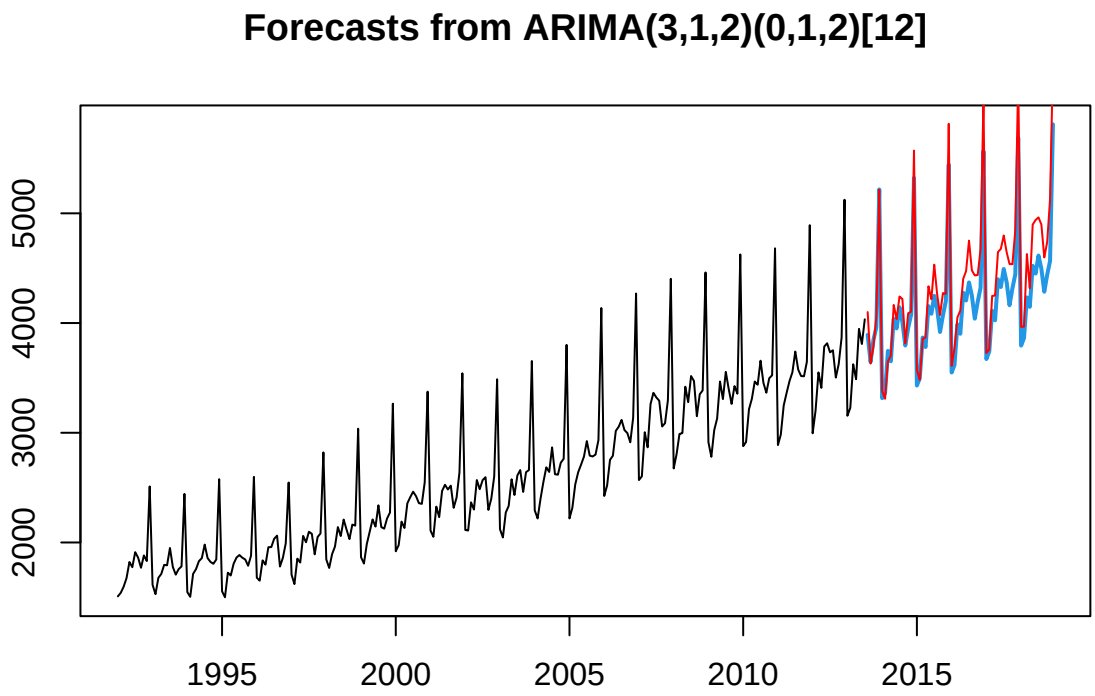

```
plot(forecast4)
lines(valid.ts, col = 'red')
```




```
plot(forecast5)
lines(valid.ts, col = 'red')
```




```
plot(forecast6)
lines(valid.ts, col = 'red')
```




```
# accuracy
a1 = accuracy(forecast1$mean, valid.ts)
a2 = accuracy(forecast2$mean, valid.ts)
a3 = accuracy(forecast3$mean, valid.ts)
a4 = accuracy(forecast4$mean, valid.ts)
a5 = accuracy(forecast5$mean, valid.ts)
a6 = accuracy(forecast6$mean, valid.ts)

rbind(a1, a2, a3,a4, a5, a6)
```

Which model gives the lowest MAPE? Which model gives the greatest MAPE?

```
##                     ME      RMSE       MAE        MPE      MAPE      ACF1
## Test set  1769.99691 1882.6269 1769.9969  39.169783 39.169783 0.2185321
## Test set   353.80000  732.5090  529.8308   6.261439 11.163434 0.2185321
## Test set   608.35385  688.7382  608.3538  13.448195 13.448195 0.8566482
## Test set    30.83488  568.8391  381.5753  -0.922978  8.462362 0.0336548
## Test set -3490.45064 3992.5495 3490.4788 -78.574232 78.574920 0.9344789
## Test set   203.07998  258.4554  210.6509   4.307779  4.517138 0.5074249
##          Theil's U
## Test set 2.8315383
## Test set 1.0698961
## Test set 1.0079979
```

```
## Test set 0.8531500
## Test set 5.8974308
## Test set 0.3751979
```

- The model with the lowest MAPE is model 6, and the model that gives the greatest MAPE is model 1.

3. Forecasting

Based on the MAPE above, decide the best model. Retrain the best model on the entire series. Use the retrained model to forecast the next values. Plot the series and the forecast values.

- The model with the lowest MAPE is model 6, which is the ARIMA model, so this will be considered the best model

```
selected_model =  auto.arima(y)

new_forecast = forecast(selected_model)

plot(new_forecast)
```