Time Series Forcasting report for total industry

Kevork Sulahian

2021-04-23

```
library(readxl)
library(forecast)
```

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

```
df20 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2020')
```

```
## New names:
## * `` -> ...1
## * `` -> ...2
```

```
df19 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2019')
```

```
## New names:
## * `` -> ...1
## * `` -> ...2
```

```
df18 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2018')
```

```
## New names:
## * `` -> ...1
## * `` -> ...2
```

```
df17 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2017')
```

```
## New names:
## * `` -> ...2
## * `` -> ...3
```

```
df16 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2016')
```

```
## New names:
## * `` -> ...2
## * `` -> ...3
```

```
df15 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2015')
```

```
## New names:
## * `` -> ...2
## * `` -> ...3
```

```
df14 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2014')
```

```
## New names:
## * `` -> ...2
## * `` -> ...3
```

```
df13 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2013')
```

```
## New names:
## * `` -> ...2
## * `` -> ...3
```

```
df12 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2012')
```

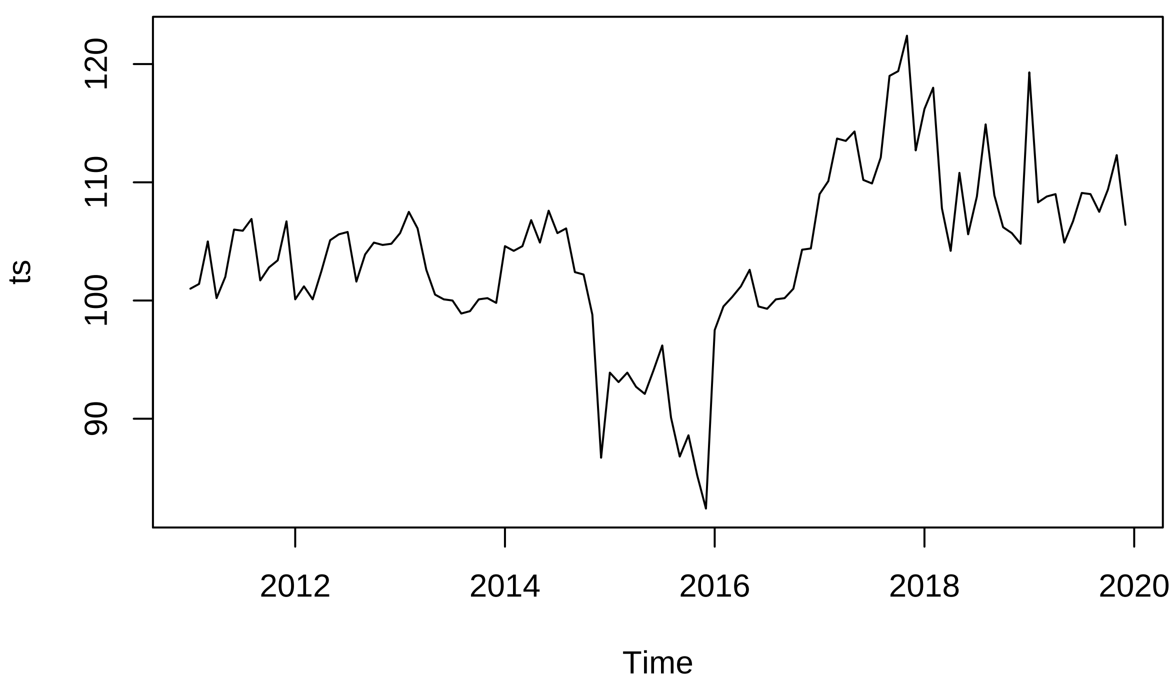
```
## New names:
## * `` -> ...2
## * `` -> ...3
```

```
df11 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2011')
```

```
## New names:
## * `` -> ...2
## * `` -> ...3
```

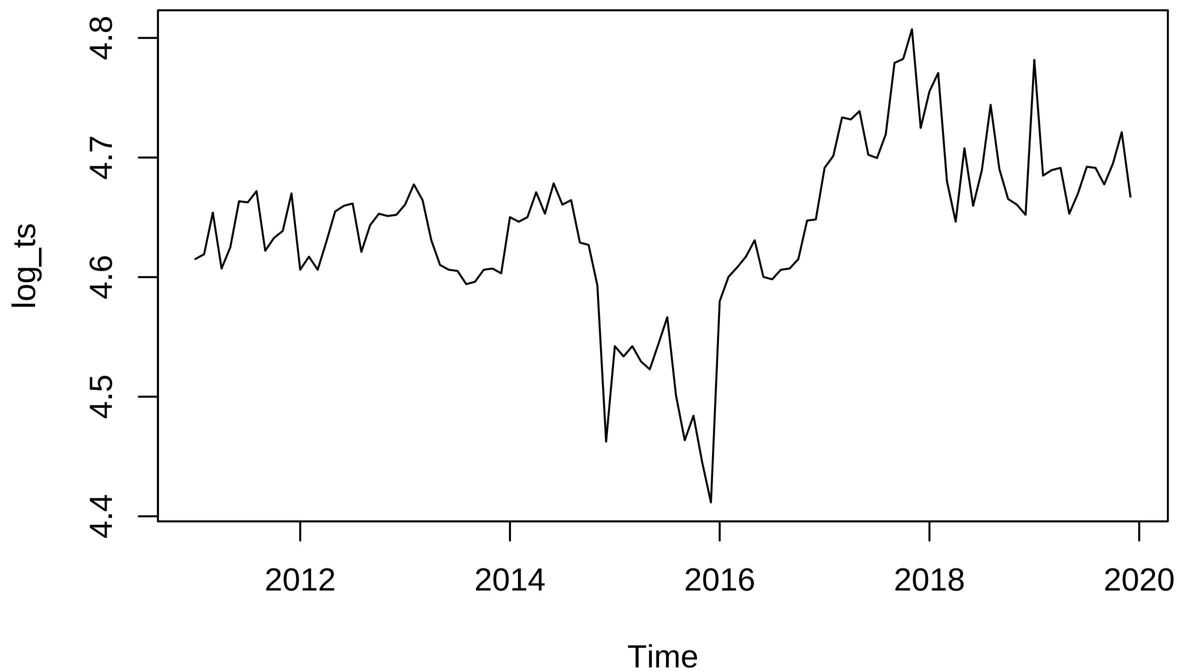
```
# df10 <- read_xlsx('Trade-2000-2020.xlsx', sheet = '2010')

df20 <- df20[,3]
df19 <- df19[,3]
df18 <- df18[,3]
df17 <- df17[,3]
df16 <- df16[,3]
df15 <- df15[,3]
df14 <- df14[,3]
df13 <- df13[,3]
df12 <- df12[,3]
df11 <- df11[,3]
# df10 <- df10[,3]

df20 = df20[-c(1:4),]
df19 = df19[-c(1:4),]
df18 = df18[-c(1:4),]
df17 = df17[-c(1:4),]
df16 = df16[-c(1:4),]
df15 = df15[-c(1:3),]
```

```
df14 = df14[-c(1:4),]
df13 = df13[-c(1:4),]
df12 = df12[-c(1:3),]
df11 = df11[-c(1:4),]

colnames(df20) = "data"
colnames(df19) = "data"
colnames(df18) = "data"
colnames(df17) = "data"
colnames(df16) = "data"
colnames(df15) = "data"
colnames(df14) = "data"
colnames(df13) = "data"
colnames(df12) = "data"
colnames(df11) = "data"



df = rbind(df11,df12,df13,df14,df15,df16,df17,df18,df19)
df = as.numeric(df$data)
ts = ts(df, start = c(2011,1), frequency = c(12))
```



In this case, it appears that an additive model is not appropriate for describing this time series, since the size of the seasonal fluctuations and random fluctuations seem to increase with the level of the time series. Thus, we may need to transform the time series in order to get a transformed time series that can be described using an additive model. For example, we can transform the time series by calculating the natural log of the original data:

```r
log_ts <- log(ts)
plot.ts(log_ts)
```



##Decomposing Time Series

Decomposing a time series means separating it into its constituent components, which are usually a trend component and an irregular component, and if it is a seasonal time series, a seasonal component.

###Decomposing Seasonal Data A seasonal time series consists of a trend component, a seasonal component and an irregular component. Decomposing the time series means separating the time series into these three components: that is, estimating these three components.

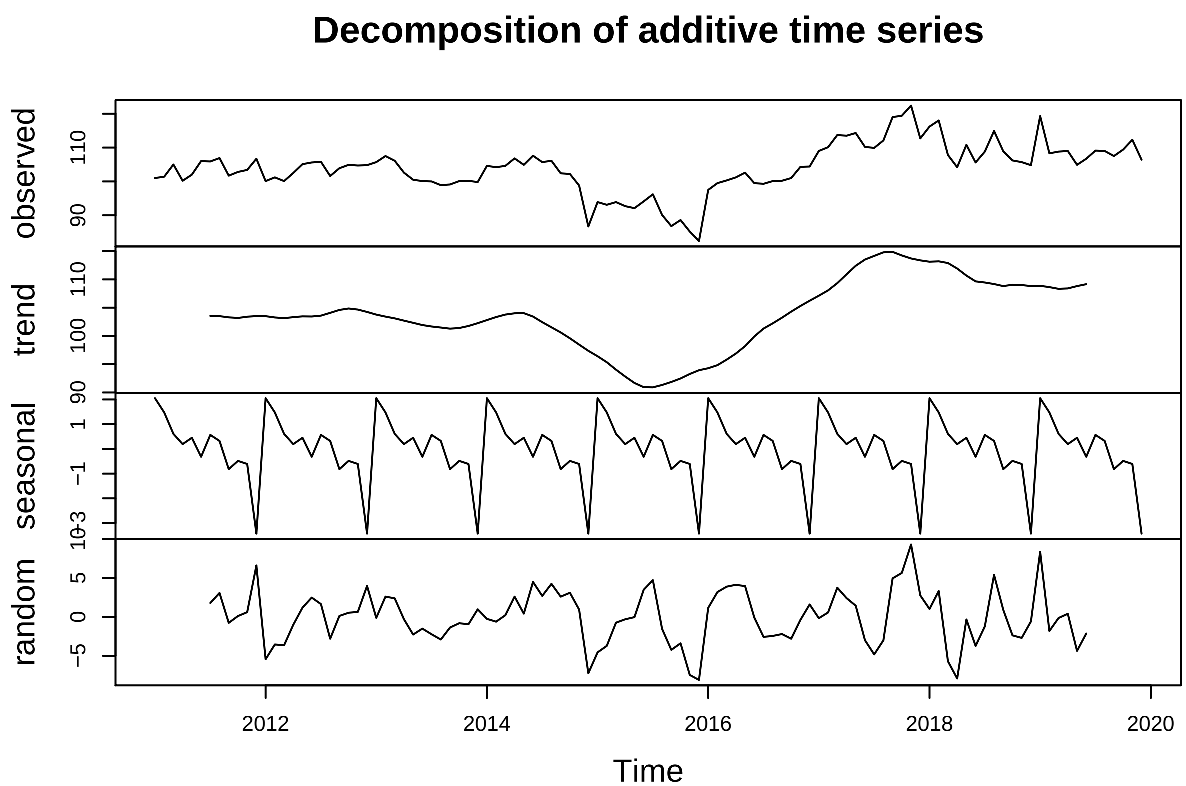
```r
ts_components <- decompose(ts)
```

we can print out the estimated values of the seasonal component

```r
ts_components$seasonal
```

```
##              Jan        Feb        Mar        Apr        May        Jun
## 2011  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
## 2012  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
## 2013  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
## 2014  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
## 2015  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
```

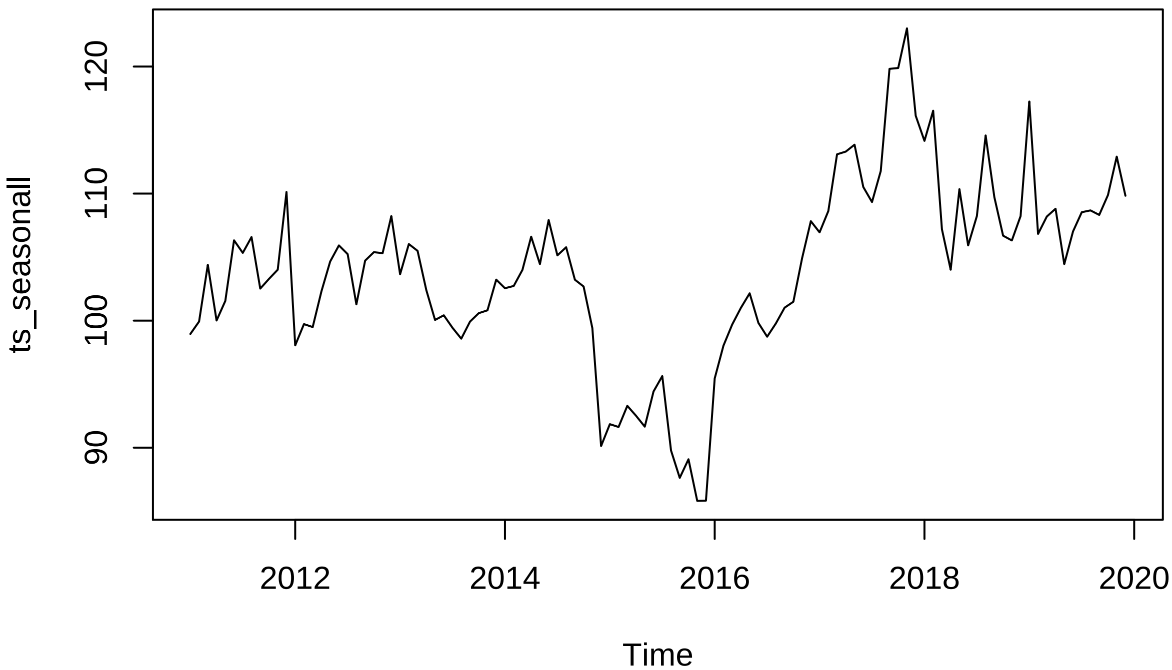
```
## 2016  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
## 2017  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
## 2018  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
## 2019  2.0513455  1.4737413  0.6075955  0.1930122  0.4497830 -0.3200087
##             Jul        Aug        Sep        Oct        Nov        Dec
## 2011  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
## 2012  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
## 2013  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
## 2014  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
## 2015  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
## 2016  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
## 2017  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
## 2018  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
## 2019  0.5664497  0.3226997 -0.8205295 -0.4861545 -0.6095920 -3.4283420
```



The plot above shows the original time series (top), the estimated trend component (second from top), the estimated seasonal component (third from top), and the estimated irregular component (bottom)

Seasonally Adjusting

```
ts_seasonall <- ts - ts_components$seasonal
```



Holt-Winters Exponential Smoothing

```r
ts_forcaste <- HoltWinters(ts)
ts_forcaste
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = ts)
##
## Smoothing parameters:
##  alpha: 0.6986285
##  beta : 0
##  gamma: 1
##
## Coefficients:
##              [,1]
## a   109.885869605
## b    -0.006657925
## s1    5.268864907
## s2    0.588993093
## s3   -0.249034626
## s4   -1.142226312
## s5   -0.873296510
## s6   -1.518809690
## s7    0.405800139
```

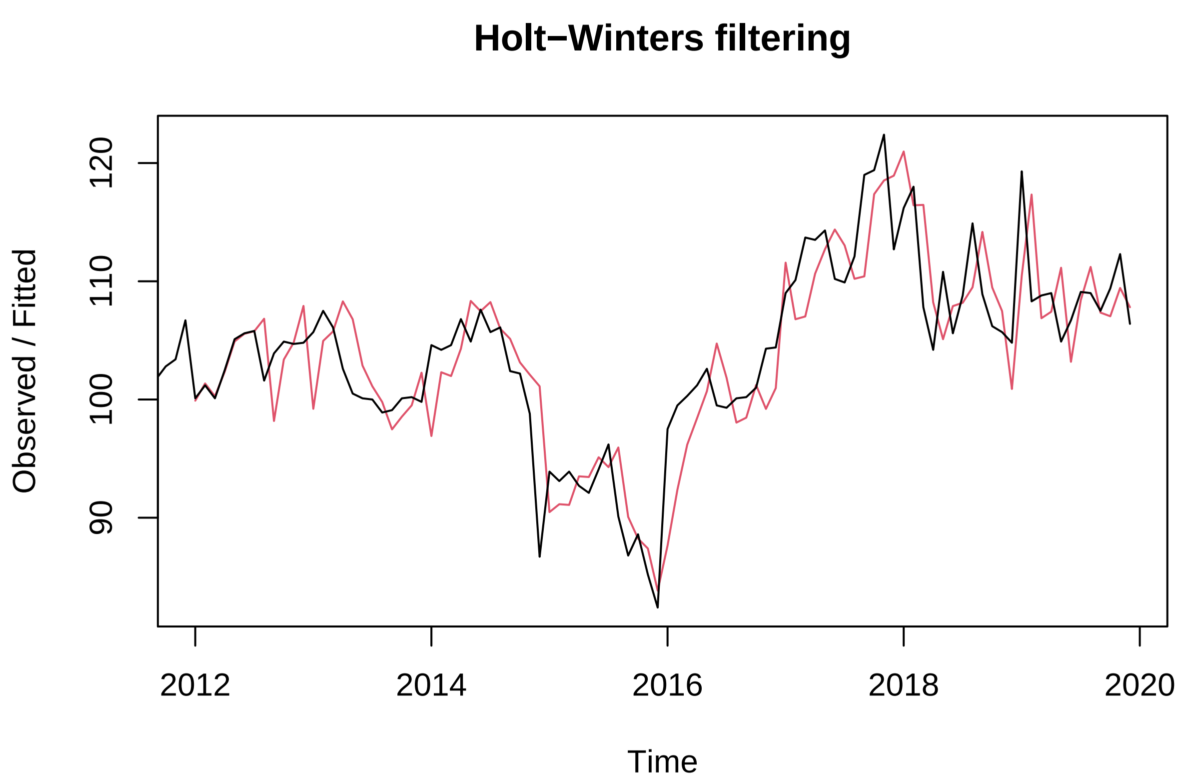
```
## s8    1.859348433
## s9    0.262696512
## s10   0.521573914
## s11   1.425395334
## s12  -3.485869605
```

```
#
```

The value of alpha (0.41) is relatively low, indicating that the estimate of the level at the current time point is based upon both recent observations and some observations in the more distant past. The value of beta is 0.00, indicating that the estimate of the slope b of the trend component is not updated over the time series, and instead is set equal to its initial value. This makes good intuitive sense, as the level changes quite a bit over the time series, but the slope b of the trend component remains roughly the same. In contrast, the value of gamma (0.96) is high, indicating that the estimate of the seasonal component at the current time point is just based upon very recent observations

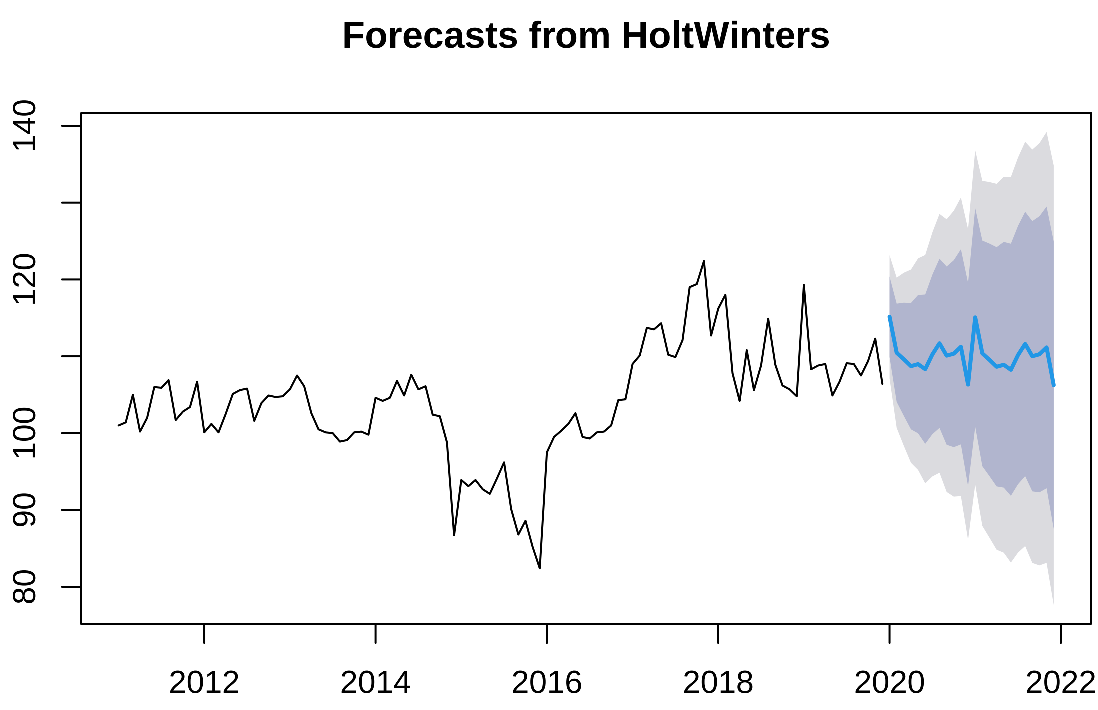
```
ts_forcaste$SSE
```

```
## [1] 1587.967
```




```
ts_forcaste2 = forecast:::forecast.HoltWinters(ts_forcaste, h= 24)
hw_forecaste = forecast:::forecast.HoltWinters(ts_forcaste, h= 24)
(as.data.frame(ts_forcaste2))[1]
```

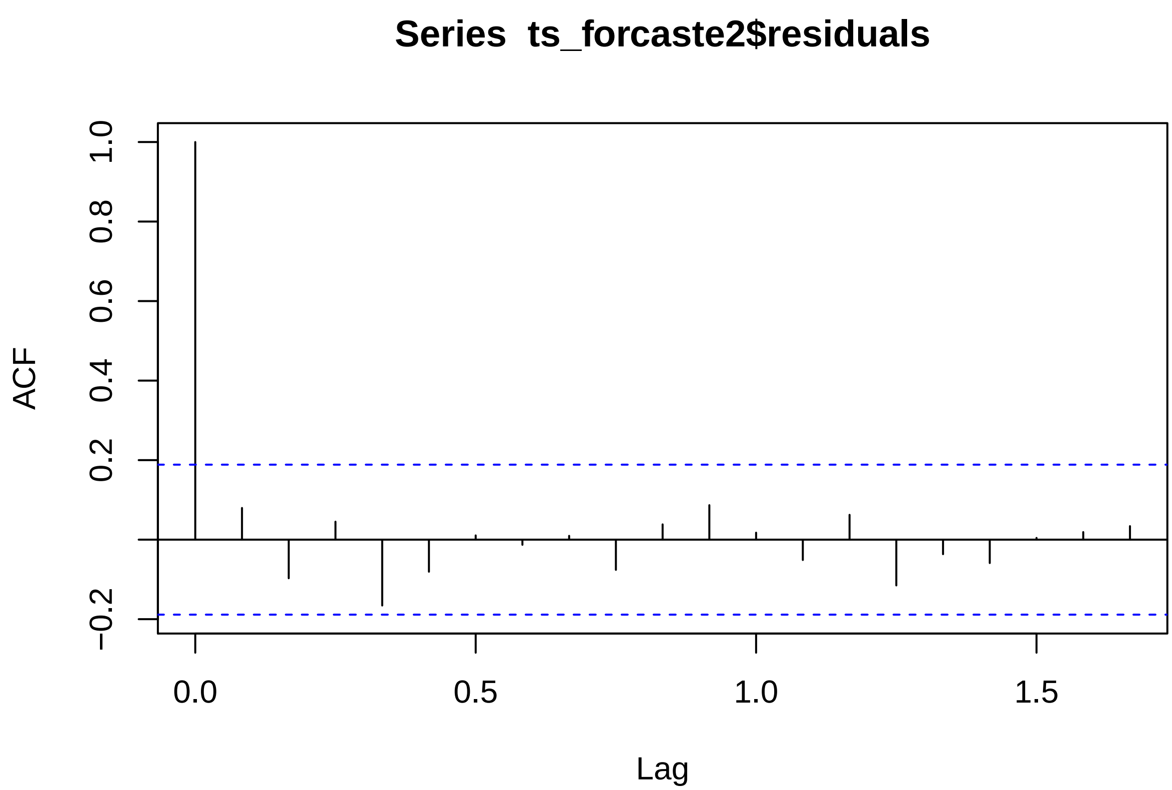
```
##          Point Forecast
## Jan 2020       115.1481
## Feb 2020       110.4615
## Mar 2020       109.6169
## Apr 2020       108.7170
## May 2020       108.9793
## Jun 2020       108.3271
## Jul 2020       110.2451
## Aug 2020       111.6920
## Sep 2020       110.0886
## Oct 2020       110.3409
## Nov 2020       111.2380
## Dec 2020       106.3201
## Jan 2021       115.0682
## Feb 2021       110.3817
## Mar 2021       109.5370
## Apr 2021       108.6371
## May 2021       108.8994
## Jun 2021       108.2472
## Jul 2021       110.1652
## Aug 2021       111.6121
## Sep 2021       110.0087
## Oct 2021       110.2610
## Nov 2021       111.1581
## Dec 2021       106.2402
```



Growth

```
year_2019 <- window(ts, 2019)
year_2020 <- (as.data.frame(ts_forcaste2))[1][c(1:12),]
year_2021 <- (as.data.frame(ts_forcaste2))[1][c(13:24),]

growth_HW_21 <- growth(sum(year_2021),sum(year_2020))
growth_HW_20 <- growth(sum(year_2020),sum(year_2019))
```

We can investigate whether the predictive model can be improved upon by checking whether the in-sample forecast errors show non-zero autocorrelations at lags 1-20, by making a correlogram and carrying out the Ljung-Box test:

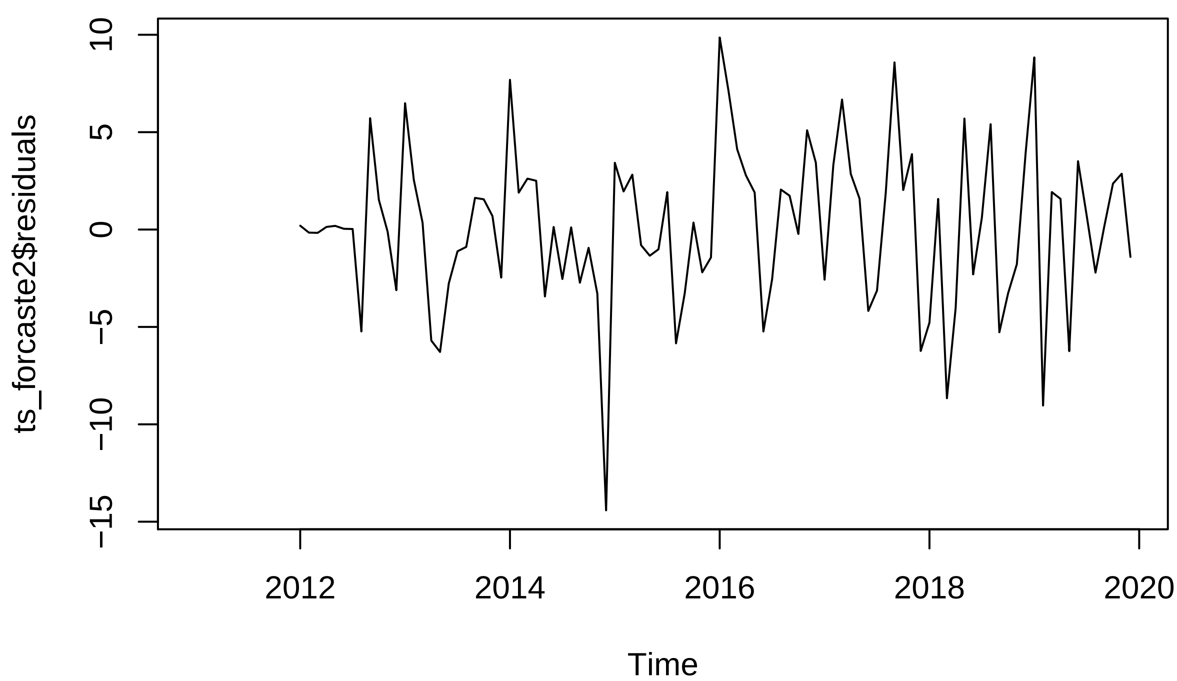



```
##
##  Box-Ljung test
##
## data:  ts_forcaste2$residuals
## X-squared = 9.9957, df = 20, p-value = 0.9682
```

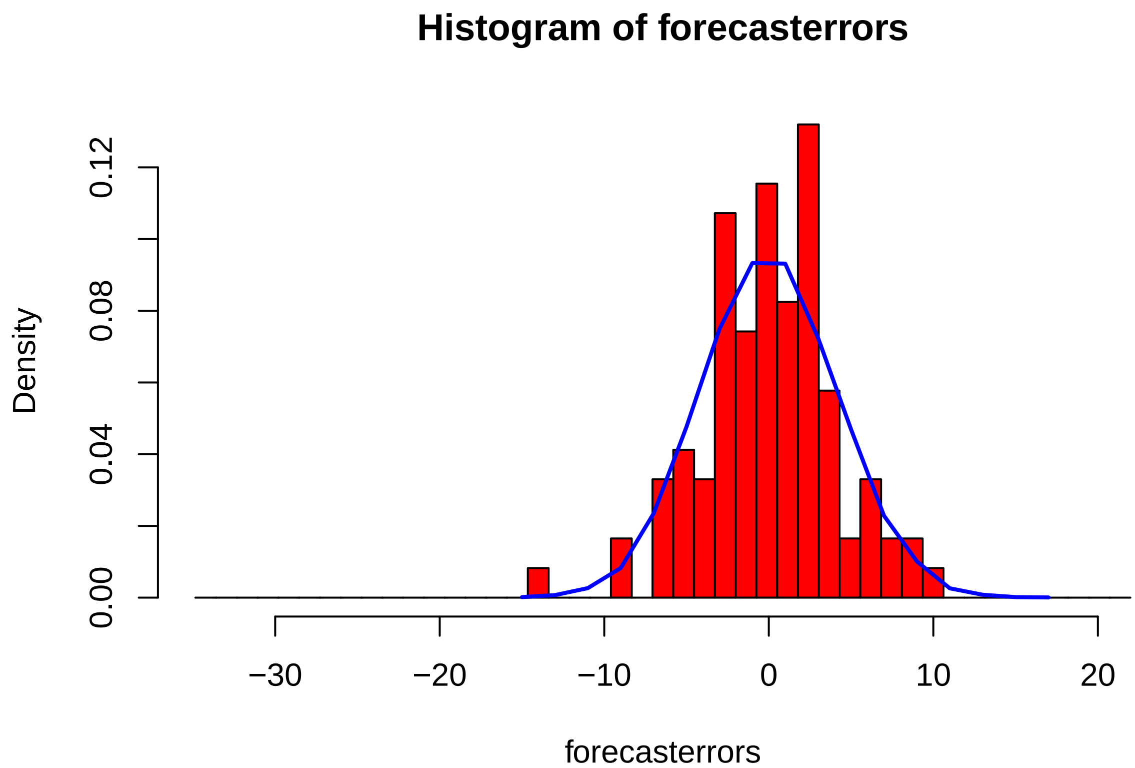
The correlogram shows that the autocorrelations for the in-sample forecast errors do not exceed the significance bounds for lags 1-20. Furthermore, the p-value for Ljung-Box test is 0.2, indicating that there is little evidence of non-zero autocorrelations at lags 1-20.

We can check whether the forecast errors have constant variance over time, and are normally distributed with mean zero, by making a time plot of the forecast errors and a histogram (with overlaid normal curve):

```r
plot.ts(ts_forcaste2$residuals)
```



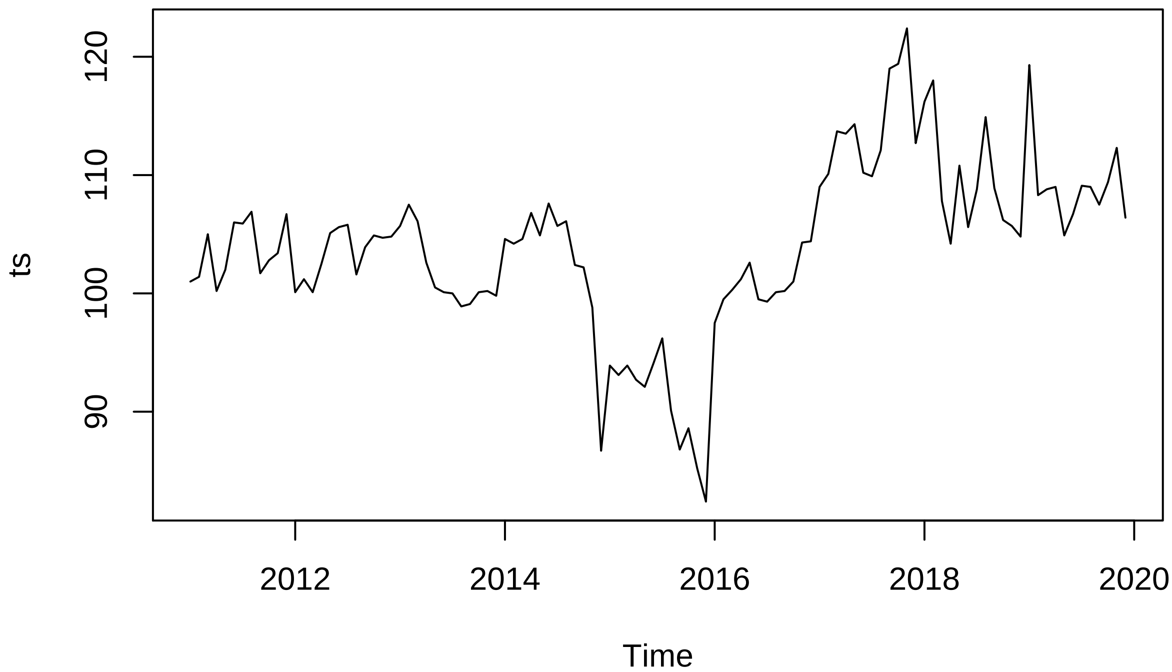

```r
plotForecastErrors(ts_forcaste2$residuals)
```



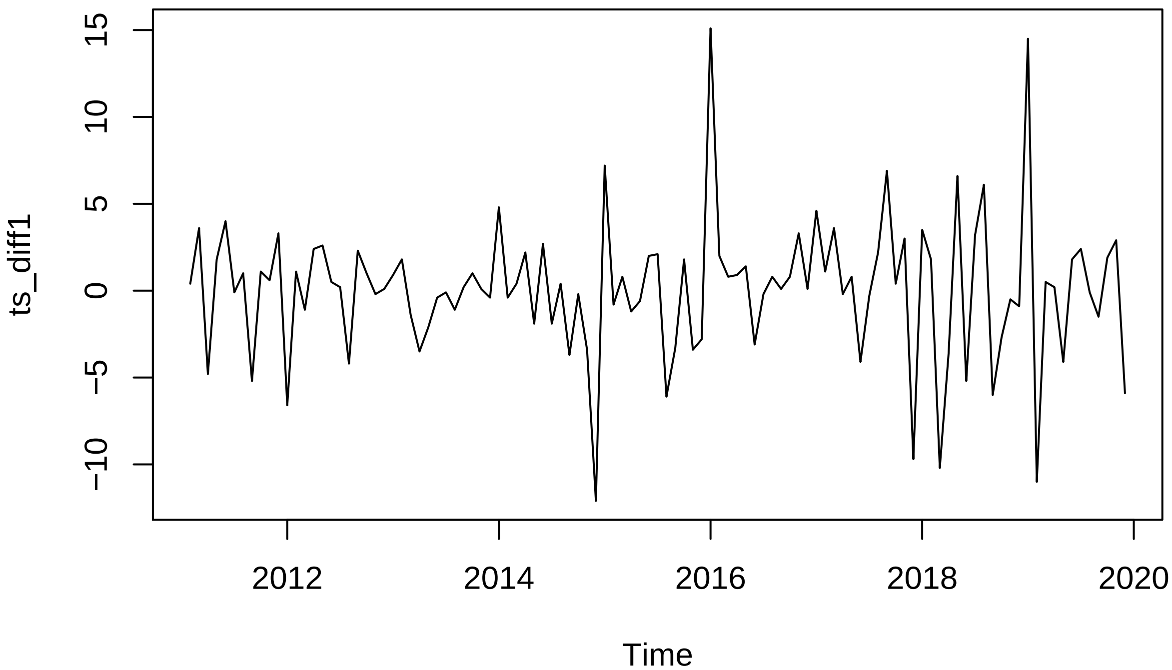
From the time plot, it appears plausible that the forecast errors have constant variance over time. From the histogram of forecast errors, it seems plausible that the forecast errors are normally distributed with mean zero.

Thus,there is little evidence of autocorrelation at lags 1-20 for the forecast errors, and the forecast errors appear to be normally distributed with mean zero and constant variance over time. This suggests that Holt-Winters exponential smoothing provides an adequate predictive model of the log of total productivity, which probably cannot be improved upon. Furthermore, the assumptions upon which the prediction intervals were based are probably valid.

```
plot.ts(ts)
```

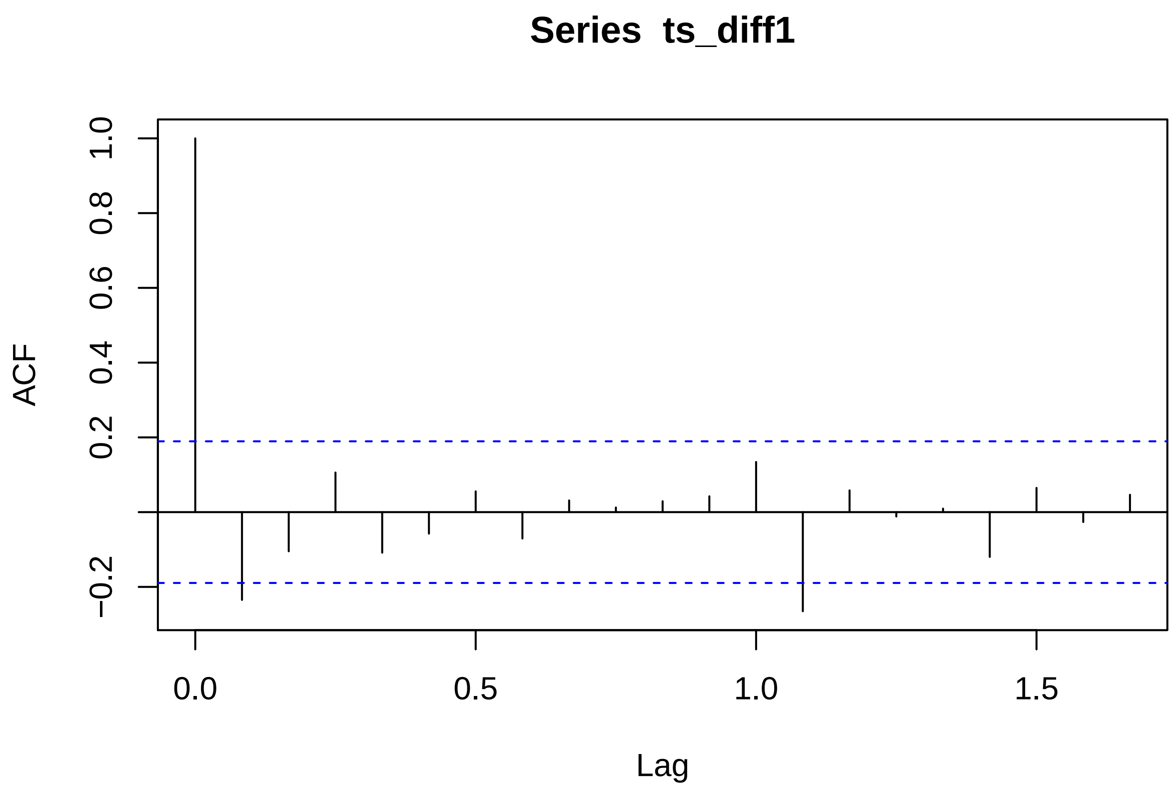



```r
ts_diff1 <-  diff(ts, differences = 1)

plot.ts(ts_diff1)
```



The time series of differences (above) does appear to be stationary in mean and variance, as the level of the series stays roughly constant over time, and the variance of the series appears roughly constant over time

```
acf(ts_diff1, lag.max=20)             # plot a correlogram
```

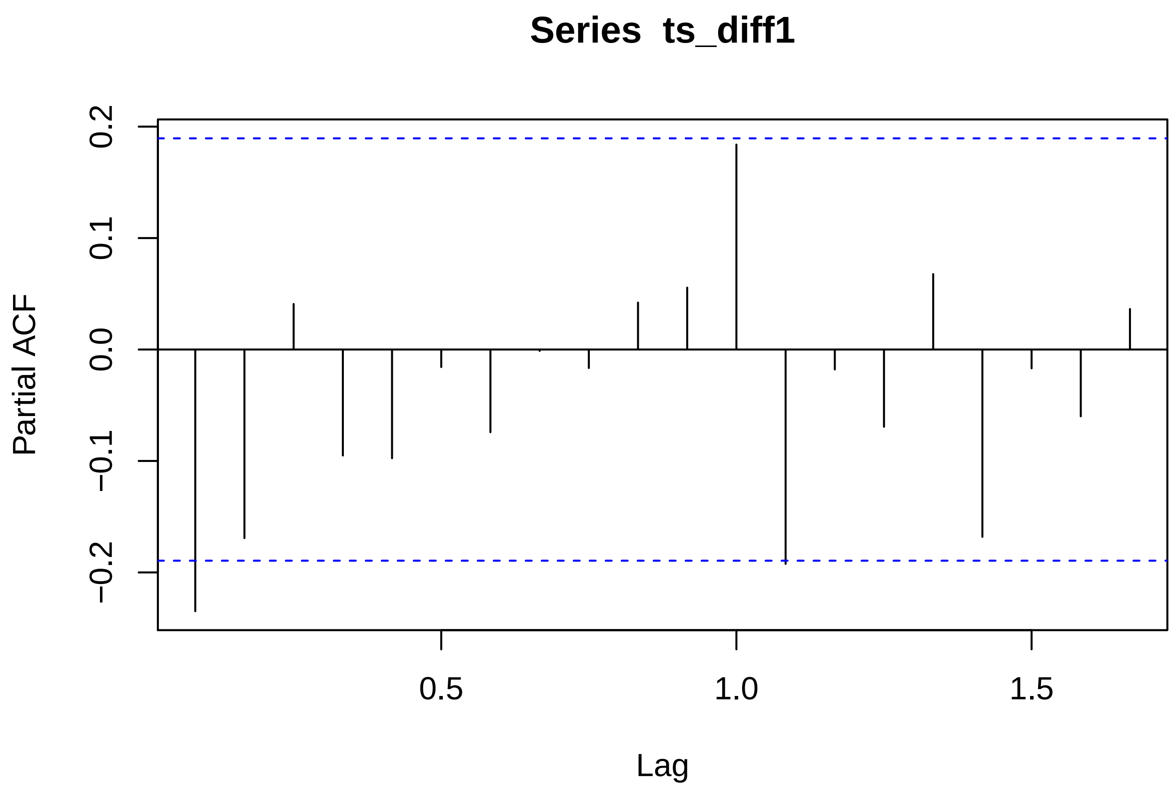


We see from the correlogram that the autocorrelation exceeds the significance bound 3 times but all the others do not exceed

```r
acf(ts_diff1, lag.max=20, plot=FALSE) # get the autocorrelation values
```

```
##
## Autocorrelations of series 'ts_diff1', by lag
##
## 0.0000 0.0833 0.1667 0.2500 0.3333 0.4167 0.5000 0.5833 0.6667 0.7500 0.8333
##  1.000 -0.235 -0.105  0.106 -0.109 -0.058  0.056 -0.071  0.031  0.013  0.029
## 0.9167 1.0000 1.0833 1.1667 1.2500 1.3333 1.4167 1.5000 1.5833 1.6667
##  0.042  0.134 -0.265  0.058 -0.012  0.009 -0.120  0.065 -0.026  0.047
```

```r
pacf(ts_diff1, lag.max=20)             # plot a partial correlogram
```




```r
pacf(ts_diff1, lag.max=20, plot=FALSE) # get the partial autocorrelation values
```

```
##
## Partial autocorrelations of series 'ts_diff1', by lag
##
## 0.0833 0.1667 0.2500 0.3333 0.4167 0.5000 0.5833 0.6667 0.7500 0.8333 0.9167
## -0.235 -0.169  0.041 -0.095 -0.098 -0.016 -0.074 -0.001 -0.017  0.042  0.056
## 1.0000 1.0833 1.1667 1.2500 1.3333 1.4167 1.5000 1.5833 1.6667
##  0.184 -0.192 -0.018 -0.069  0.068 -0.168 -0.017 -0.060  0.036
```

Arima, 1,1,1

```r
ts_arima = Arima(ts, order=c(1,1,1),seasonal = list(order = c(1,1,1)))
ts_arima
```

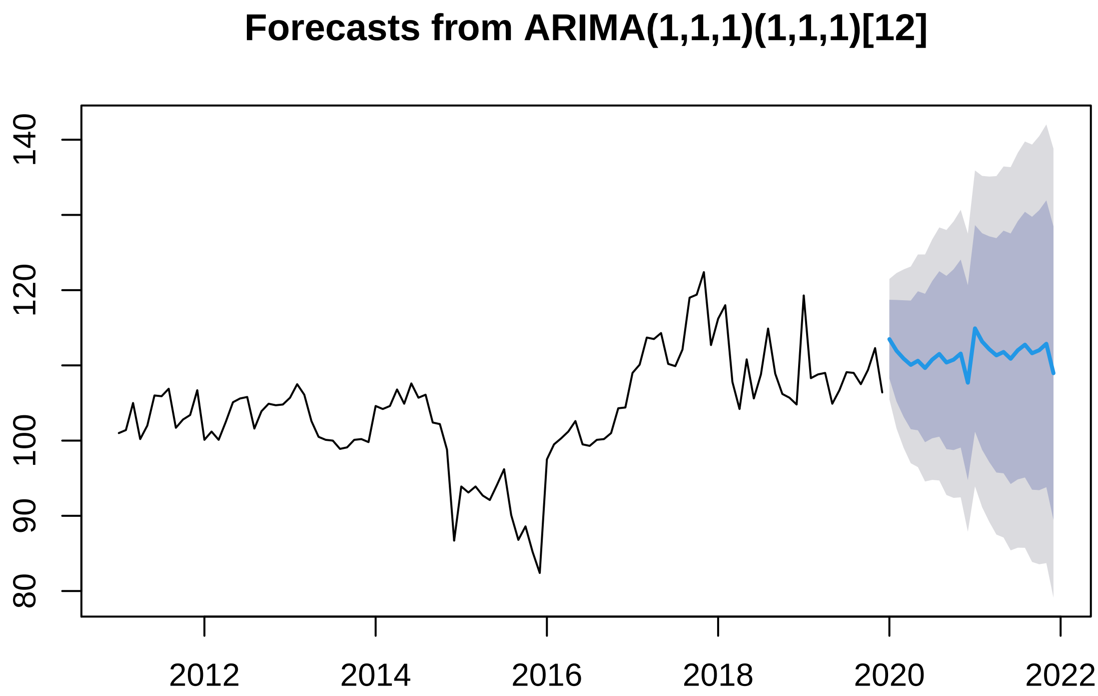
```
## Series: ts
## ARIMA(1,1,1)(1,1,1)[12]
##
## Coefficients:
##          ar1      ma1     sar1     sma1
##       0.4681  -0.6520  -0.0207  -0.7682
## s.e.  0.3818   0.3302   0.1589   0.1838
```

```
##
## sigma^2 estimated as 16.56:  log likelihood=-271.62
## AIC=553.25   AICc=553.92   BIC=566.02
```

```r
ts_arima_forecast = forecast(ts_arima,h = 24)
ts_arima_forecast
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2020       113.4877 108.25585 118.7195 105.48628 121.4891
## Feb 2020       111.9476 105.19771 118.6974 101.62456 122.2706
## Mar 2020       110.8997 103.14628 118.6531  99.04187 122.7575
## Apr 2020       110.0671 101.51676 118.6174  96.99049 123.1437
## May 2020       110.6095 101.36832 119.8508  96.47631 124.7428
## Jun 2020       109.6538  99.78617 119.5215  94.56255 124.7451
## Jul 2020       110.7575 100.30796 121.2070  94.77632 126.7387
## Aug 2020       111.5181 100.52056 122.5156  94.69883 128.3373
## Sep 2020       110.3851  98.86698 121.9031  92.76968 128.0004
## Oct 2020       110.7600  98.74444 122.7755  92.38380 129.1362
## Nov 2020       111.5683  99.07523 124.0613  92.46180 130.6748
## Dec 2020       107.6995  94.74639 120.6526  87.88945 127.5095
## Jan 2021       114.9168 101.18674 128.6469  93.91846 135.9152
## Feb 2021       113.1700  98.77324 127.5667  91.15209 135.1878
## Mar 2021       112.1489  97.14265 127.1552  89.19881 135.0990
## Apr 2021       111.3353  95.75545 126.9151  87.50800 135.1625
## May 2021       111.7805  95.65303 127.9080  87.11564 136.4454
## Jun 2021       110.8813  94.22658 127.5360  85.41010 136.3525
## Jul 2021       112.0115  94.84687 129.1762  85.76046 138.2626
## Aug 2021       112.7542  95.09482 130.4136  85.74652 139.7619
## Sep 2021       111.6135  93.47313 129.7539  83.87019 139.3569
## Oct 2021       112.0200  93.41102 130.6289  83.56003 140.4799
## Nov 2021       112.8715  93.80542 131.9377  83.71242 142.0307
## Dec 2021       108.9607  89.44785 128.4736  79.11837 138.8030
```

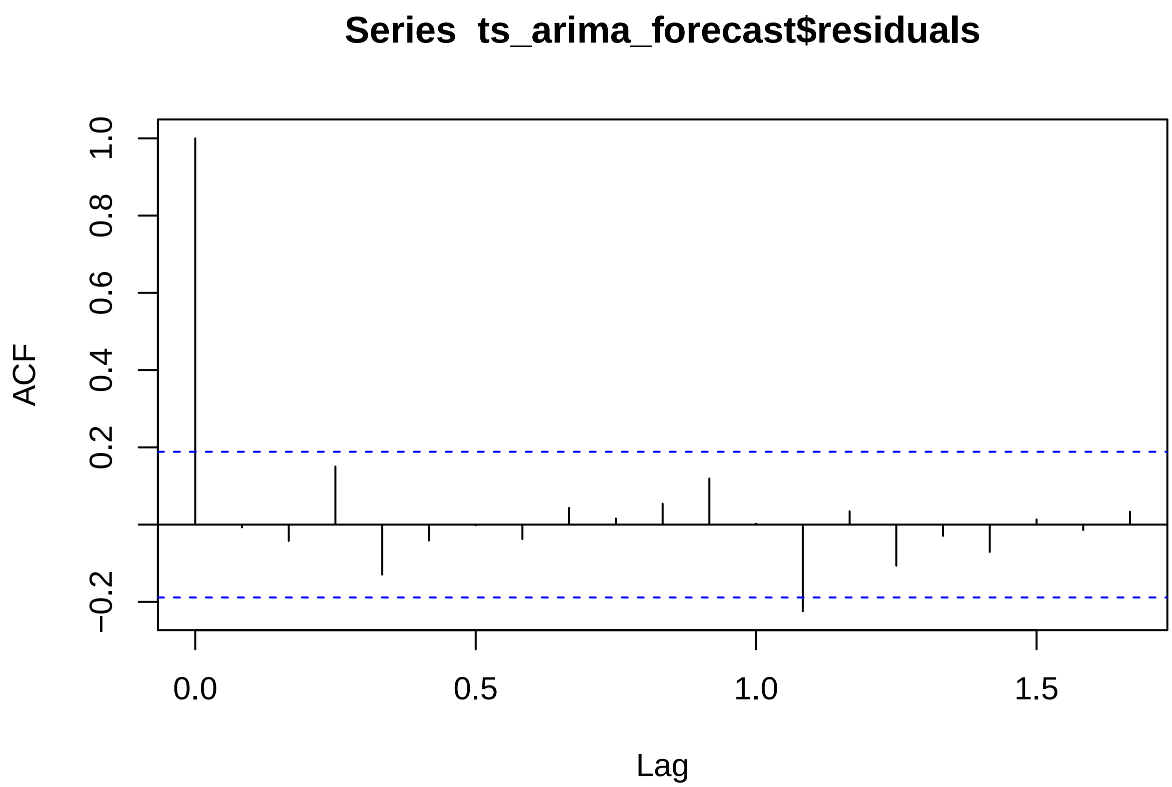
```r
forecast:::plot.forecast(ts_arima_forecast)
```



As in the case of exponential smoothing models, it is a good idea to investigate whether the forecast errors of an ARIMA model are normally distributed with mean zero and constant variance, and whether the are correlations between successive forecast errors.

For example, we can make a correlogram of the forecast errors for our ARIMA(0,1,1) model, and perform the Ljung-Box test for lags 1-20, by typing:

```
acf(ts_arima_forecast$residuals, lag.max=20)
```

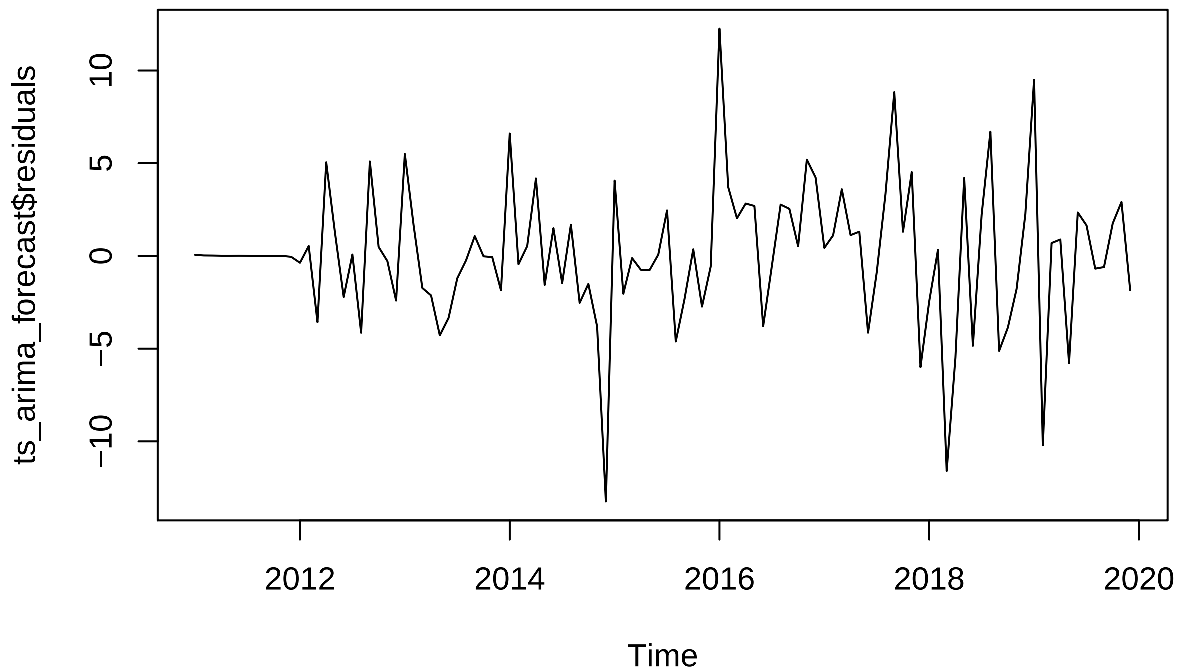



```r
Box.test(ts_arima_forecast$residuals, lag=20, type="Ljung-Box")
```

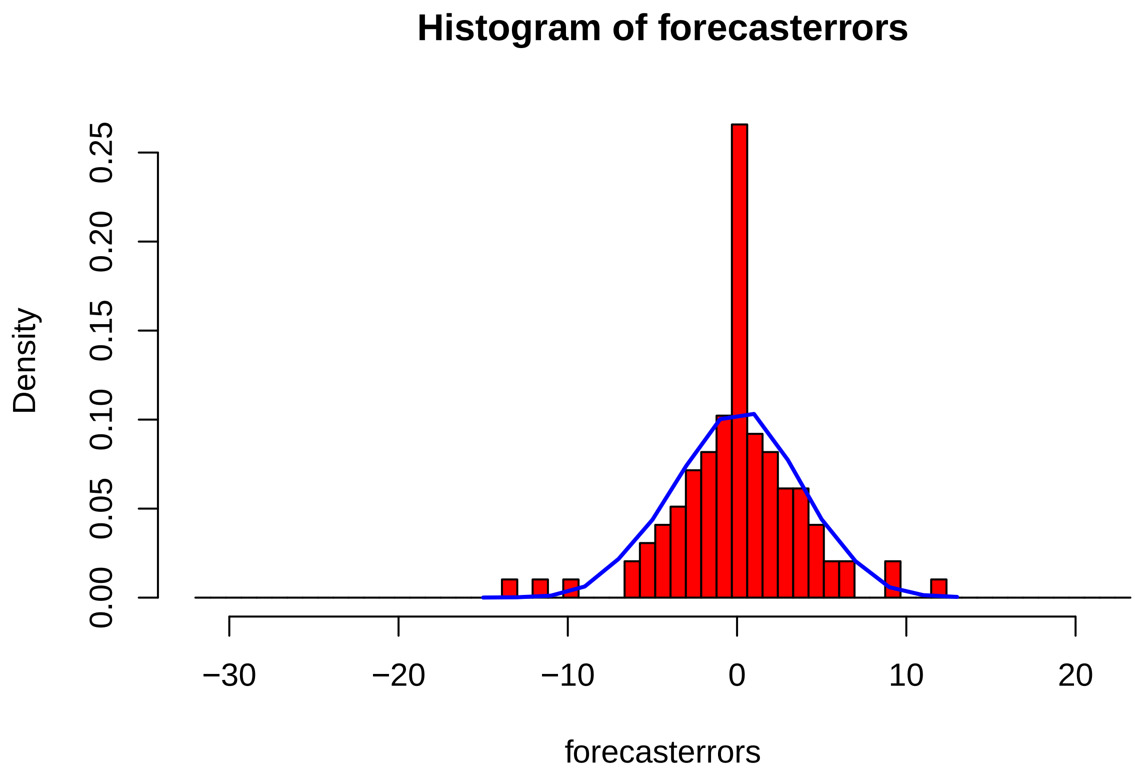
```
##
##  Box-Ljung test
##
## data:  ts_arima_forecast$residuals
## X-squared = 16.243, df = 20, p-value = 0.7014
```

we can reject the null hypothesis, it's rather similar to the HW

```r
plot.ts(ts_arima_forecast$residuals)          # make time plot of forecast errors
```




```r
plotForecastErrors(ts_arima_forecast$residuals)
```

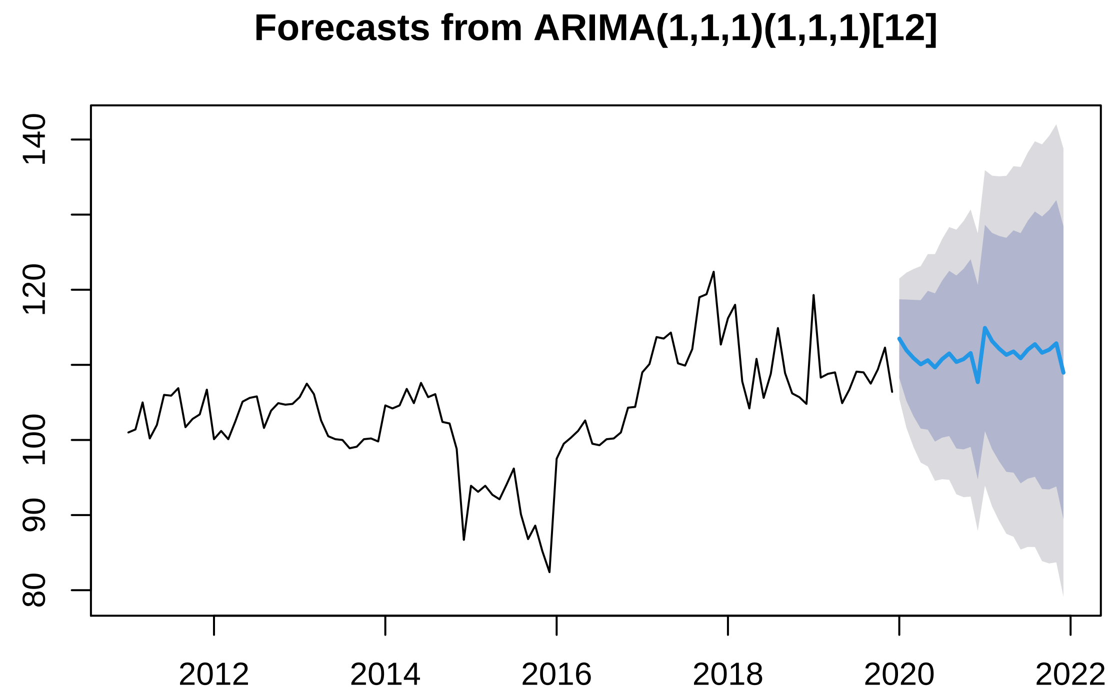


Arima, 0,1,0 as given from the loop

```
ts_arima = Arima(ts, order=c(2,1,1),seasonal = list(order = c(2,1,0)))
ts_arima
```

```
## Series: ts
## ARIMA(2,1,1)(2,1,0)[12]
##
## Coefficients:
##           ar1      ar2     ma1     sar1     sar2
##       -0.8584  -0.2664  0.7067  -0.5915  -0.3605
## s.e.   0.2128   0.1011  0.2094   0.1084   0.1149
##
## sigma^2 estimated as 17.62:  log likelihood=-271.52
## AIC=555.04   AICc=556   BIC=570.37
```

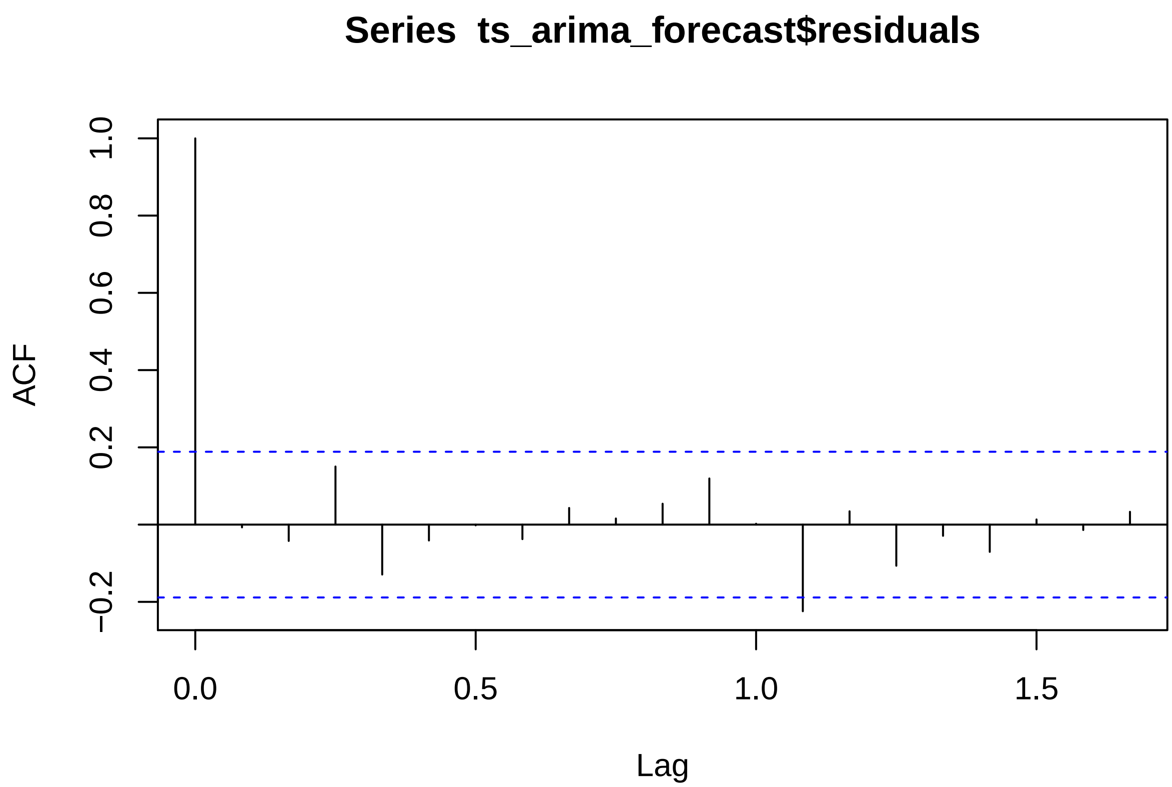
```
ts_arima_forecast2 = forecast(ts_arima,h = 24)
# ts_arima_forecast

forecast:::plot.forecast(ts_arima_forecast)
```



As in the case of exponential smoothing models, it is a good idea to investigate whether the forecast errors of an ARIMA model are normally distributed with mean zero and constant variance, and whether the are correlations between successive forecast errors.

For example, we can make a correlogram of the forecast errors for our ARIMA(0,1,1) model, and perform the Ljung-Box test for lags 1-20, by typing:

```
acf(ts_arima_forecast$residuals, lag.max=20)
```

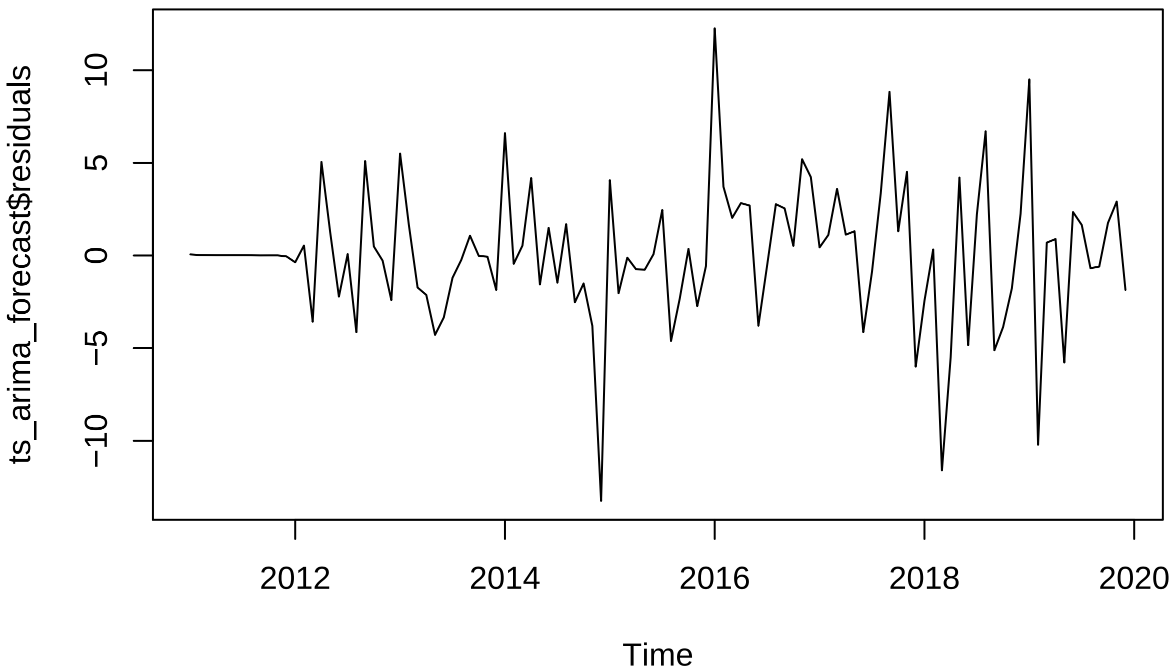



```r
Box.test(ts_arima_forecast$residuals, lag=20, type="Ljung-Box")
```

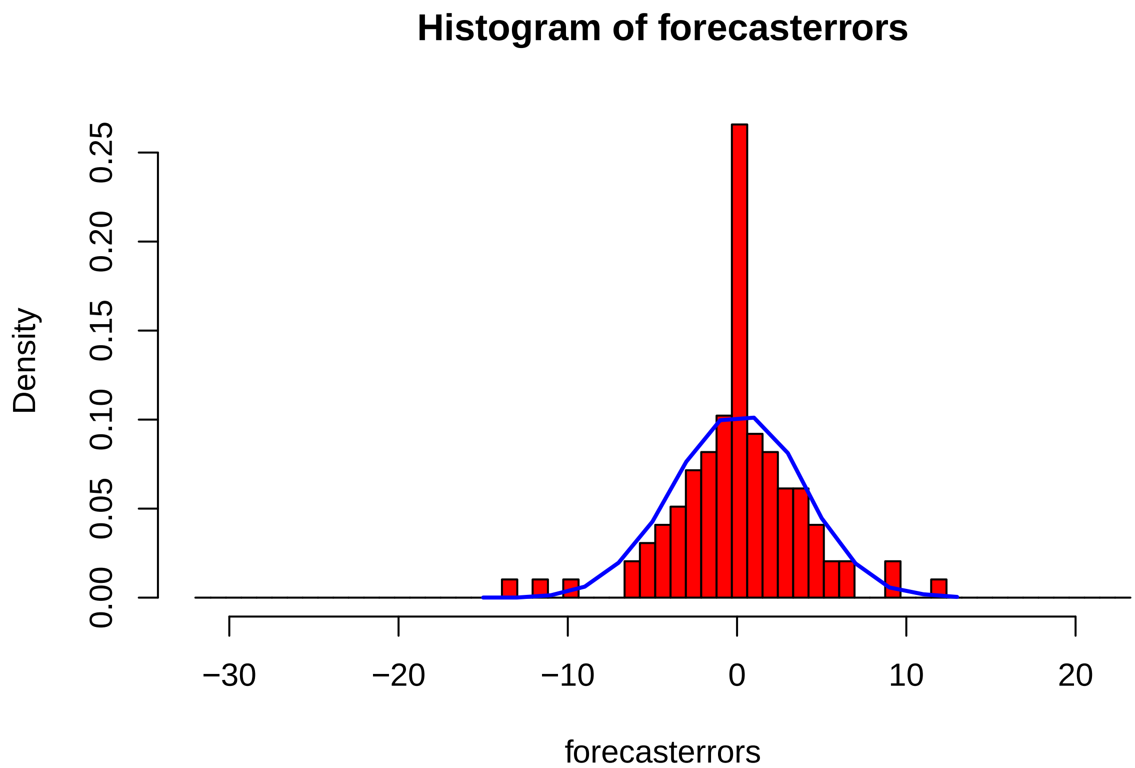
```
##
##  Box-Ljung test
##
## data:  ts_arima_forecast$residuals
## X-squared = 16.243, df = 20, p-value = 0.7014
```

we can reject the null hypothesis, it's rather similar to the HW

```r
plot.ts(ts_arima_forecast$residuals)            # make time plot of forecast errors
```




```r
plotForecastErrors(ts_arima_forecast$residuals)
```

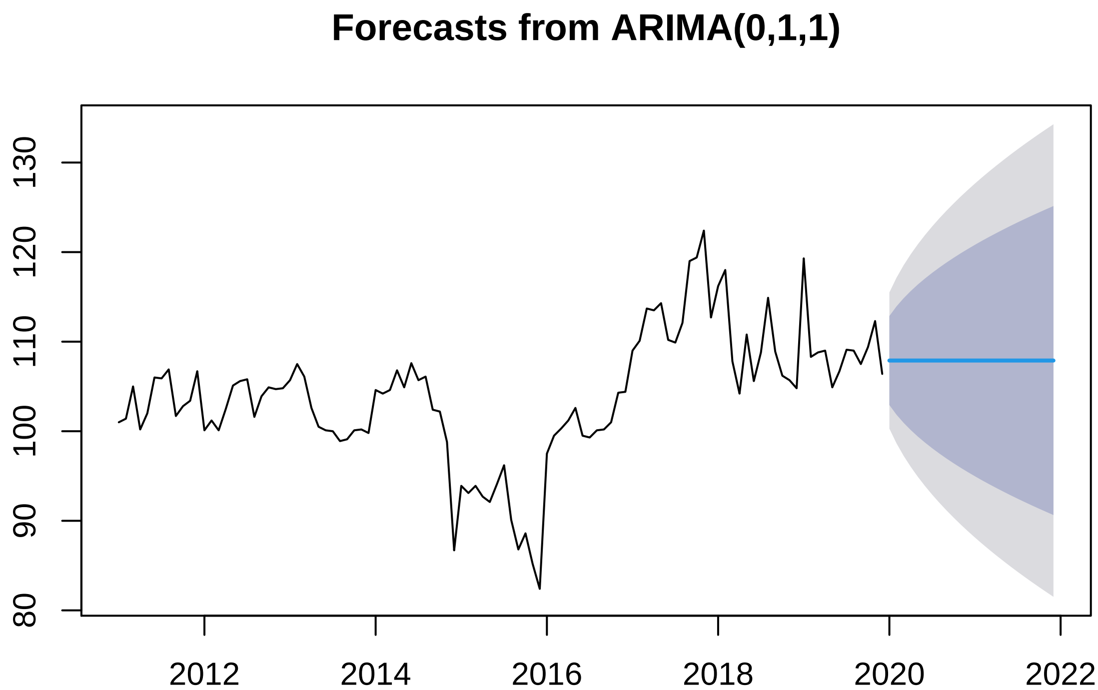


A model chosen automatically

```r
fit <- auto.arima(ts,max.p = 5,max.q = 5,max.P = 5,max.Q = 5,max.d = 3,seasonal = TRUE)
fit
```

```
## Series: ts
## ARIMA(0,1,1)
##
## Coefficients:
##           ma1
##       -0.3067
## s.e.   0.1009
##
## sigma^2 estimated as 15.03:  log likelihood=-296.35
## AIC=596.7   AICc=596.82   BIC=602.05
```

```r
fit_forecast = forecast(fit,h=24)
plot(fit_forecast)
```




```r
# str(fit)
```

Growth

all the growths

```r
hw_forecaste
```

```
##          Point Forecast      Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2020       115.1481 109.91029 120.3859 107.13757 123.1586
## Feb 2020       110.4615 104.07213 116.8510 100.68978 120.2333
## Mar 2020       109.6169 102.25379 116.9799  98.35601 120.8777
## Apr 2020       108.7170 100.49478 116.9392  96.14219 121.2918
## May 2020       108.9793  99.97954 117.9790  95.21537 122.7432
## Jun 2020       108.3271  98.61189 118.0423  93.46896 123.1853
## Jul 2020       110.2451  99.86355 120.6266  94.36791 126.1222
## Aug 2020       111.6920 100.68441 122.6995  94.85737 128.5265
## Sep 2020       110.0886  98.48880 121.6885  92.34822 127.8291
## Oct 2020       110.3409  98.17754 122.5042  91.73866 128.9431
## Nov 2020       111.2380  98.53619 123.9399  91.81224 130.6638
## Dec 2020       106.3201  93.10167 119.5385  86.10425 126.5360
## Jan 2021       115.0682 100.84983 129.2865  93.32310 136.8133
```

```
## Feb 2021       110.3817  95.69997 125.0633  87.92796 132.8353
## Mar 2021       109.5370  94.40614 124.6678  86.39637 132.6776
## Apr 2021       108.6371  93.07010 124.2041  84.82941 132.4448
## May 2021       108.8994  92.90807 124.8907  84.44277 133.3560
## Jun 2021       108.2472  91.84257 124.6519  83.15847 133.3360
## Jul 2021       110.1652  93.35735 126.9730  84.45983 135.8705
## Aug 2021       111.6121  94.41052 128.8136  85.30457 137.9195
## Sep 2021       110.0087  92.42230 127.5952  83.11259 136.9049
## Oct 2021       110.2610  92.29785 128.2241  82.78875 137.7332
## Nov 2021       111.1581  92.82609 129.4902  83.12169 139.1946
## Dec 2021       106.2402  87.54652 124.9339  77.65068 134.8297
```

```
ts_arima_forecast
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2020       113.4877 108.25585 118.7195 105.48628 121.4891
## Feb 2020       111.9476 105.19771 118.6974 101.62456 122.2706
## Mar 2020       110.8997 103.14628 118.6531  99.04187 122.7575
## Apr 2020       110.0671 101.51676 118.6174  96.99049 123.1437
## May 2020       110.6095 101.36832 119.8508  96.47631 124.7428
## Jun 2020       109.6538  99.78617 119.5215  94.56255 124.7451
## Jul 2020       110.7575 100.30796 121.2070  94.77632 126.7387
## Aug 2020       111.5181 100.52056 122.5156  94.69883 128.3373
## Sep 2020       110.3851  98.86698 121.9031  92.76968 128.0004
## Oct 2020       110.7600  98.74444 122.7755  92.38380 129.1362
## Nov 2020       111.5683  99.07523 124.0613  92.46180 130.6748
## Dec 2020       107.6995  94.74639 120.6526  87.88945 127.5095
## Jan 2021       114.9168 101.18674 128.6469  93.91846 135.9152
## Feb 2021       113.1700  98.77324 127.5667  91.15209 135.1878
## Mar 2021       112.1489  97.14265 127.1552  89.19881 135.0990
## Apr 2021       111.3353  95.75545 126.9151  87.50800 135.1625
## May 2021       111.7805  95.65303 127.9080  87.11564 136.4454
## Jun 2021       110.8813  94.22658 127.5360  85.41010 136.3525
## Jul 2021       112.0115  94.84687 129.1762  85.76046 138.2626
## Aug 2021       112.7542  95.09482 130.4136  85.74652 139.7619
## Sep 2021       111.6135  93.47313 129.7539  83.87019 139.3569
## Oct 2021       112.0200  93.41102 130.6289  83.56003 140.4799
## Nov 2021       112.8715  93.80542 131.9377  83.71242 142.0307
## Dec 2021       108.9607  89.44785 128.4736  79.11837 138.8030
```

```
ts_arima_forecast2
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2020       115.4125 110.03340 120.7916 107.18587 123.6391
## Feb 2020       112.0834 105.02976 119.1371 101.29577 122.8711
## Mar 2020       110.7614 102.73468 118.7881  98.48559 123.0372
## Apr 2020       110.2477 100.95795 119.5374  96.04026 124.4551
## May 2020       110.2457 100.07297 120.4184  94.68786 125.8035
## Jun 2020       108.3399  97.26584 119.4139  91.40360 125.2761
## Jul 2020       109.9560  98.06929 121.8427  91.77686 128.1351
## Aug 2020       112.1030  99.46015 124.7458  92.76746 131.4384
## Sep 2020       112.6028  99.23812 125.9674  92.16330 133.0422
## Oct 2020       112.8940  98.85042 126.9376  91.41619 134.3718
```

```
## Nov 2020       115.0467 100.35274 129.7407  92.57423 137.5192
## Dec 2020       108.9314  93.61555 124.2473  85.50781 132.3551
## Jan 2021       117.2236 100.57859 133.8687  91.76722 142.6801
## Feb 2021       113.9723  96.21528 131.7294  86.81525 141.1294
## Mar 2021       109.8703  91.16340 128.5773  81.26054 138.4802
## Apr 2021       108.4089  88.69146 128.1264  78.25366 138.5642
## May 2021       109.8405  89.22622 130.4548  78.31368 141.3673
## Jun 2021       107.6030  86.10275 129.1032  74.72121 140.4848
## Jul 2021       109.9712  87.62633 132.3160  75.79769 144.1446
## Aug 2021       113.0243  89.86702 136.1815  77.60831 148.4402
## Sep 2021       110.7190  86.77416 134.6638  74.09853 147.3394
## Oct 2021       110.3034  85.59803 135.0088  72.51978 148.0871
## Nov 2021       111.6724  86.22824 137.1166  72.75892 150.5859
## Dec 2021       107.4870  81.32520 133.6488  67.47599 147.4980
```

```
fit_forecast
```

```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2020       107.8933 102.92534 112.8612 100.29548 115.4910
## Feb 2020       107.8933 101.84810 113.9384  98.64799 117.1385
## Mar 2020       107.8933 100.93571 114.8508  97.25260 118.5339
## Apr 2020       107.8933 100.12981 115.6567  96.02008 119.7664
## May 2020       107.8933  99.40003 116.3865  94.90399 120.8825
## Jun 2020       107.8933  98.72819 117.0583  93.87649 121.9100
## Jul 2020       107.8933  98.10234 117.6842  92.91933 122.8672
## Aug 2020       107.8933  97.51415 118.2724  92.01978 123.7667
## Sep 2020       107.8933  96.95756 118.8290  91.16855 124.6180
## Oct 2020       107.8933  96.42796 119.3586  90.35859 125.4279
## Nov 2020       107.8933  95.92176 119.8648  89.58443 126.2021
## Dec 2020       107.8933  95.43611 120.3504  88.84170 126.9448
## Jan 2021       107.8933  94.96870 120.8178  88.12685 127.6597
## Feb 2021       107.8933  94.51762 121.2689  87.43698 128.3495
## Mar 2021       107.8933  94.08125 121.7053  86.76962 129.0169
## Apr 2021       107.8933  93.65826 122.1283  86.12271 129.6638
## May 2021       107.8933  93.24748 122.5390  85.49447 130.2921
## Jun 2021       107.8933  92.84791 122.9386  84.88338 130.9031
## Jul 2021       107.8933  92.45868 123.3278  84.28811 131.4984
## Aug 2021       107.8933  92.07903 123.7075  83.70748 132.0790
## Sep 2021       107.8933  91.70828 124.0782  83.14047 132.6461
## Oct 2021       107.8933  91.34584 124.4407  82.58616 133.2004
## Nov 2021       107.8933  90.99116 124.7954  82.04373 133.7428
## Dec 2021       107.8933  90.64378 125.1427  81.51246 134.2741
```