Final.R

sarthakmehta

2022-12-06

```
# Final

library(readr)
```

```
## Warning: package 'readr' was built under R version 4.1.2
```

```
library(fpp)
```

```
## Loading required package: forecast

## Warning: package 'forecast' was built under R version 4.1.2

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

## Loading required package: fma

## Loading required package: expsmooth

## Loading required package: lmtest

## Warning: package 'lmtest' was built under R version 4.1.2

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 4.1.2

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: tseries

## Warning: package 'tseries' was built under R version 4.1.2
```

```r
library(fpp2)
```

```
## -- Attaching packages ---------------------------------------------- fpp2 2.4 --
```

```
## v ggplot2 3.3.5
```

```
##
```

```
##
## Attaching package: 'fpp2'
```

```
## The following objects are masked from 'package:fpp':
##
##     ausair, ausbeer, austa, austourists, debitcards, departures,
##     elecequip, euretail, guinearice, oil, sunspotarea, usmelec
```

```r
IPG3113N <- read_csv("/Users/sarthakmehta/Desktop/IPG3113N.csv")
```

```
## Rows: 121 Columns: 2
```

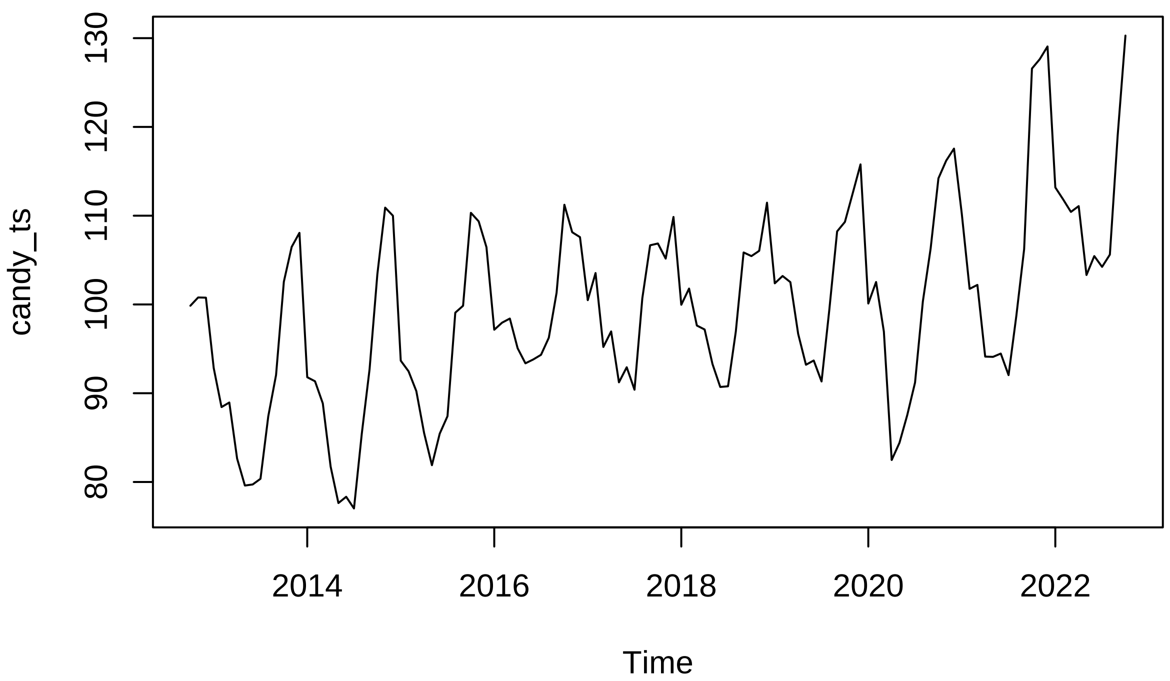
```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (1): IPG3113N
## date (1): DATE
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
candy_ts <- ts(IPG3113N$IPG3113N,frequency = 12,start=c(2012,10))

#Plot and Inference
candy_ts
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2012
## 2013  92.8393  88.4378  88.9572  82.6418  79.6009  79.7216  80.3623  87.3990
## 2014  91.8045  91.3464  88.8527  81.7442  77.6292  78.3359  77.0192  85.4452
## 2015  93.6676  92.4733  90.2388  85.5111  81.8907  85.4442  87.4143  99.0757
## 2016  97.1509  97.9475  98.4111  95.0657  93.3682  93.8015  94.3271  96.2487
## 2017 100.4828 103.5407  95.2095  96.9654  91.2276  92.9268  90.3917 100.6988
## 2018  99.9651 101.7932  97.6257  97.1776  93.2973  90.7030  90.7825  96.9555
## 2019 102.3790 103.2079 102.5133  96.6862  93.2061  93.6885  91.3219  99.4256
## 2020 100.1032 102.5297  96.9412  82.4777  84.4155  87.5411  91.2016 100.3417
## 2021 110.1841 101.7558 102.2030  94.1243  94.0966  94.4684  92.0391  98.7301
## 2022 113.1828 111.8384 110.4263 111.0788 103.3188 105.4497 104.2348 105.6166
##           Sep      Oct      Nov      Dec
## 2012           99.8541 100.7874 100.7643
## 2013  92.0523 102.5585 106.4783 108.0668
## 2014  92.7103 103.4097 110.9082 109.9962
## 2015  99.8525 110.3226 109.3626 106.4516
```

```
## 2016 101.3271 111.2323 108.1348 107.5859
## 2017 106.6639 106.8689 105.1628 109.8611
## 2018 105.8597 105.4468 106.0408 111.4674
## 2019 108.2320 109.2961 112.5268 115.7811
## 2020 106.3095 114.2083 116.2062 117.5581
## 2021 106.2289 126.5753 127.6255 129.0640
## 2022 119.0161 130.2894
```

```
plot(candy_ts) #plotting the candy_ts data
```




```
#Observation: Through the graph we can that the monthly production of candies in the US has been pretty
#can see a clear pattern to it. And there are sharp declines and rises throughout the data, so to cut o
#would not be correct. Hence would be going on with the full data set provided.

#Central Tendency

min(candy_ts)
```

```
## [1] 77.0192
```

```
# [1] 77.0192

max(candy_ts)
```

```
## [1] 130.2894
```

```r
# [1] 130.2894

mean(candy_ts)
```

```
## [1] 99.44454
```

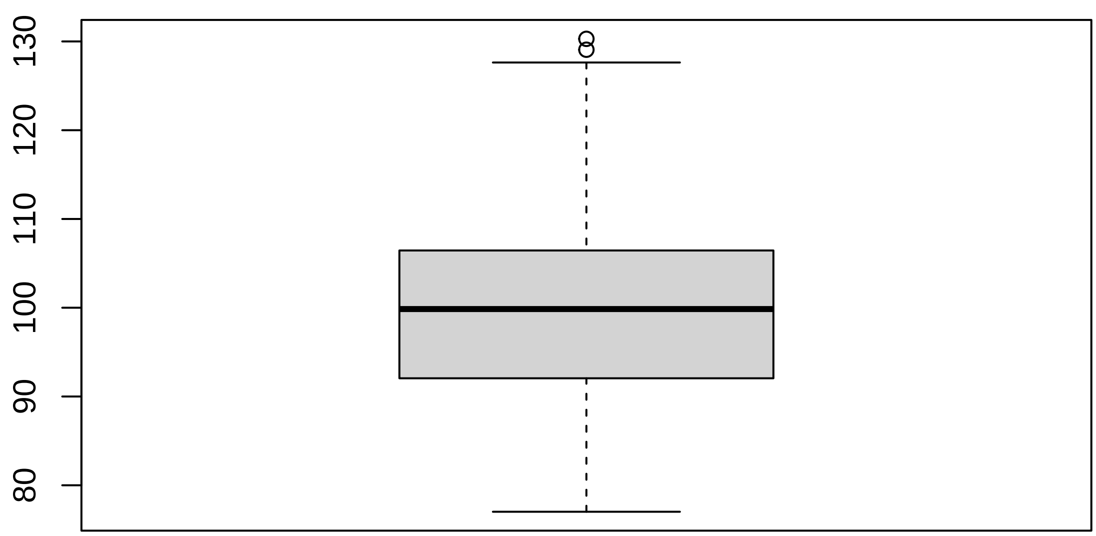
```r
# [1] 99.44454

median(candy_ts)
```

```
## [1] 99.8525
```

```r
# [1] 99.8525

quantile(candy_ts)
```

```
##       0%      25%      50%      75%     100%
##  77.0192  92.0523  99.8525 106.4516 130.2894
```

```r
#   25%      75%
# 92.0523  106.4516

boxplot((candy_ts))
```



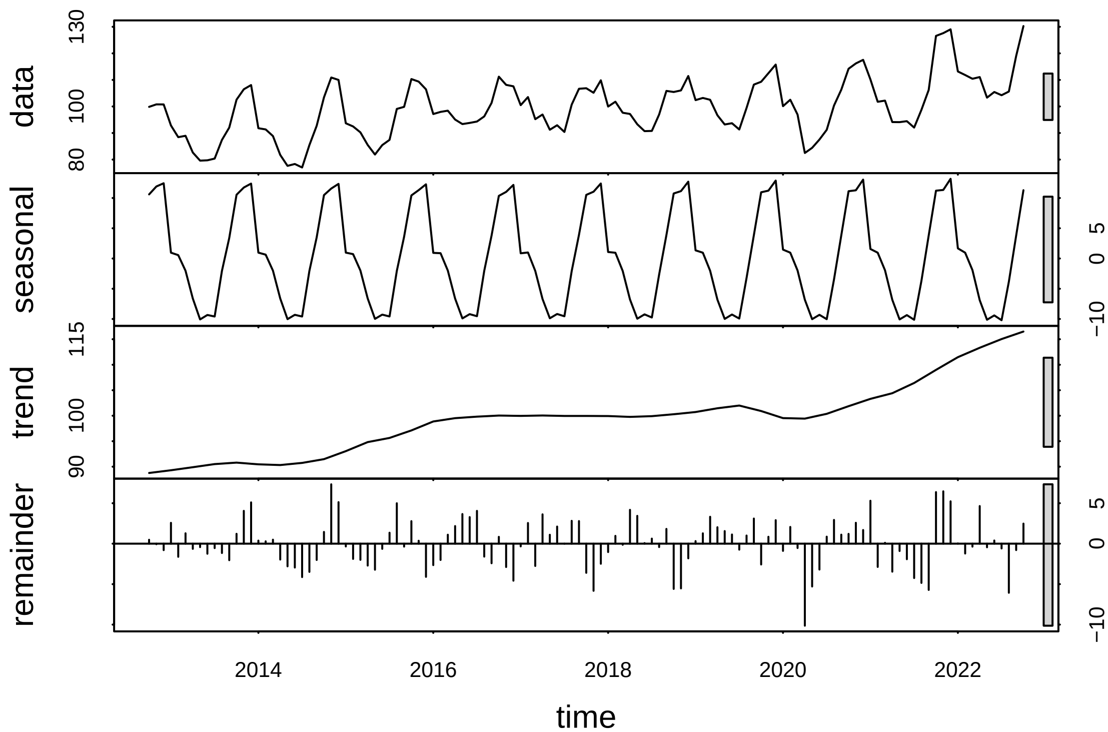
```r
#Summary: Well the summaries show us that the amongst the median values calculated min value of candy p
#period of 2012 to 2022 was 77.0192 and the max value was 130.2894. Furthermore, the mean value was 99.
#the median value was 99.8525. The 1st quartile was 92.0523 and the 3rd quartile was 106.4516. The box p
#that the there are 2 outliers, and the minimum value is a bit more closer to the 1st quartile than the
#the 3rd quatile,and the median is approximate in between the 1st and 3rd quartile. But when we look cl
#median we can see that as mean is a bit lower than the median (2nd quartile line) hence would me that


#Decomposition

stl_decomp <- stl(candy_ts,s.window =12)
stl_decomp
```

```
##  Call:
##  stl(x = candy_ts, s.window = 12)
##
## Components
##             seasonal    trend    remainder
## Oct 2012  10.5991242 88.76885  0.486128404
## Nov 2012  11.9082450 88.94684 -0.067689965
## Dec 2012  12.4396616 89.12484 -0.800204106
## Jan 2013   0.9586825 89.30284  2.577777391
## Feb 2013   0.5607168 89.50068 -1.623601801
## Mar 2013  -2.0268681 89.69853  1.285538278
## Apr 2013  -6.6117581 89.89637 -0.642816560
## May 2013 -10.0729539 90.10035 -0.426498736
## Jun 2013  -9.3367546 90.30433 -1.245976113
## Jul 2013  -9.5950782 90.50831 -0.550930554
## Aug 2013  -2.0496520 90.60483 -1.156182465
## Sep 2013   3.3932326 90.70136 -2.042292762
## Oct 2013  10.5381605 90.79789  1.222453628
## Nov 2013  11.7366414 90.68594  4.055718666
## Dec 2013  12.3879538 90.57399  5.104852222
## Jan 2014   0.9628192 90.46205  0.379632771
## Feb 2014   0.6389851 90.41520  0.292219599
## Mar 2014  -2.0111343 90.36834  0.495491725
## Apr 2014  -6.6100457 90.32149 -1.967244165
## May 2014 -10.0306151 90.46147 -2.801659713
## Jun 2014  -9.3114433 90.60146 -2.954116492
## Jul 2014  -9.5920828 90.74144 -4.130162024
## Aug 2014  -2.0542340 90.98619 -3.486757861
## Sep 2014   3.4924209 91.23094 -2.013059763
## Oct 2014  10.4813530 91.47569  1.452661089
## Nov 2014  11.5692114 91.99909  7.339900617
## Dec 2014  12.3404369 92.52249  5.133272951
## Jan 2015   0.9713533 93.04589 -0.349645527
## Feb 2015   0.7218572 93.63809 -1.886647844
## Mar 2015  -1.9905303 94.23029 -2.000958834
## Apr 2015  -6.6031967 94.82249 -2.708190937
## May 2015  -9.9830309 95.09530 -3.221568015
## Jun 2015  -9.2807779 95.36811 -0.643132352
## Jul 2015  -9.5840192 95.64092  1.357397749
## Aug 2015  -2.0540339 96.13102  4.998711111
```

```
## Sep 2015   3.5954289  96.62112  -0.364053081
## Oct 2015  10.4285374  97.11123   2.782837102
## Nov 2015  11.3025047  97.69738   0.362714208
## Dec 2015  12.2706801  98.28354  -4.102616766
## Jan 2016   0.9304921  98.86969  -2.649284402
## Feb 2016   0.8764821  99.08581  -2.014796319
## Mar 2016  -1.9993279  99.30194   1.108491712
## Apr 2016  -6.6292411  99.51806   2.176882941
## May 2016  -9.9089575  99.61689   3.660265343
## Jun 2016  -9.2001789  99.71573   3.285952692
## Jul 2016  -9.5406504  99.81456   4.053190219
## Aug 2016  -2.0359966  99.88701  -1.602309051
## Sep 2016   3.7863623  99.95945  -2.418713451
## Oct 2016  10.3551207 100.03190   0.845282712
## Nov 2016  11.0123956 100.00919  -2.886783341
## Dec 2016  12.1747194  99.98648  -4.575298361
## Jan 2017   0.8589920  99.96377  -0.339962196
## Feb 2017   0.9960331  99.98942   2.555249318
## Mar 2017  -2.0487293 100.01507  -2.756835789
## Apr 2017  -6.7014192 100.04071   3.626106613
## May 2017  -9.8860464 100.01204   1.101609349
## Jun 2017  -9.1757707  99.98336   2.119209238
## Jul 2017  -9.5560123  99.95469  -0.006973671
## Aug 2017  -2.0792298  99.95389   2.824143474
## Sep 2017   3.9180669  99.95309   2.792746411
## Oct 2017  10.5189014  99.95229  -3.602288426
## Nov 2017  11.0420904  99.94582  -5.825108760
## Dec 2017  12.4069792  99.93935  -2.485228857
## Jan 2018   1.0703083  99.93288  -1.038089254
## Feb 2018   0.9505060  99.87849   0.964202807
## Mar 2018  -2.0618904  99.82410  -0.136510979
## Apr 2018  -6.7759368  99.76971   4.183825153
## May 2018  -9.9575216  99.81734   3.437481672
## Jun 2018  -9.2362093  99.86497   0.074241035
## Jul 2018  -9.7580459  99.91260   0.627949363
## Aug 2018  -2.6602766 100.03749  -0.421708467
## Sep 2018   3.8733267 100.16237   1.823999816
## Oct 2018  10.7524738 100.28726  -5.592935852
## Nov 2018  11.1397392 100.43136  -5.530297287
## Dec 2018  12.7053552 100.57545  -1.813409278
## Jan 2019   1.3426650 100.71955   0.316784921
## Feb 2019   0.9609437 100.96702   1.279936357
## Mar 2019  -2.0251972 101.21449   3.324007384
## Apr 2019  -6.8067104 101.46196   2.030950738
## May 2019  -9.9908230 101.63893   1.557993363
## Jun 2019  -9.2640441 101.81590   1.136644525
## Jul 2019  -9.9317231 101.99287  -0.739246400
## Aug 2019  -3.2172141 101.63460   1.008214976
## Sep 2019   3.8494705 101.27633   3.106200634
## Oct 2019  10.9518486 100.91806  -2.573807143
## Nov 2019  11.2186645 100.45449   0.853641995
## Dec 2019  12.8851579  99.99093   2.905013508
## Jan 2020   1.4674880  99.52736  -0.891651539
## Feb 2020   0.9667147  99.49509   2.067894875
```

```
## Mar 2020  -1.9830794  99.46282  -0.538537889
## Apr 2020  -6.8148633  99.43054 -10.137980898
## May 2020 -10.0329043  99.74307  -5.294666481
## Jun 2020  -9.3135889 100.05560  -3.200908472
## Jul 2020 -10.0346542 100.36812   0.868130191
## Aug 2020  -3.4609873 100.86804   2.934642849
## Sep 2020   3.8269911 101.36796   1.114543976
## Oct 2020  11.1280938 101.86789   1.212320792
## Nov 2020  11.2760261 102.34620   2.583971116
## Dec 2020  13.0449631 102.82452   1.688616745
## Jan 2021   1.5755494 103.30284   5.305713092
## Feb 2021   0.9589601 103.67269  -2.875850112
## Mar 2021  -1.9503607 104.04254   0.110818098
## Apr 2021  -6.8282886 104.41240  -3.459806511
## May 2021 -10.0761009 105.08146  -0.908754977
## Jun 2021  -9.3600919 105.75052  -1.922024797
## Jul 2021 -10.1312918 106.41958  -4.249185771
## Aug 2021  -3.6952153 107.27536  -4.850048264
## Sep 2021   3.8158456 108.13115  -5.718095113
## Oct 2021  11.2130846 108.98694   6.375279931
## Nov 2021  11.3363315 109.81832   6.470852045
## Dec 2021  13.1803233 110.64970   5.233979330
## Jan 2022   1.6748211 111.48108   0.026900529
## Feb 2022   0.9604557 112.09593  -1.217984182
## Mar 2022  -1.9154152 112.71078  -0.369063354
## Apr 2022  -6.8967372 113.32563   4.649908547
## May 2022 -10.1274938 113.89493  -0.448631708
## Jun 2022  -9.4097104 114.46422   0.395188008
## Jul 2022 -10.2055011 115.03352  -0.593218213
## Aug 2022  -3.8402816 115.52604  -6.069163191
## Sep 2022   3.7907091 116.01857  -0.793179360
## Oct 2022  11.2961746 116.51110   2.482129583
```

```
plot(stl_decomp) #Decomposition Plot
```

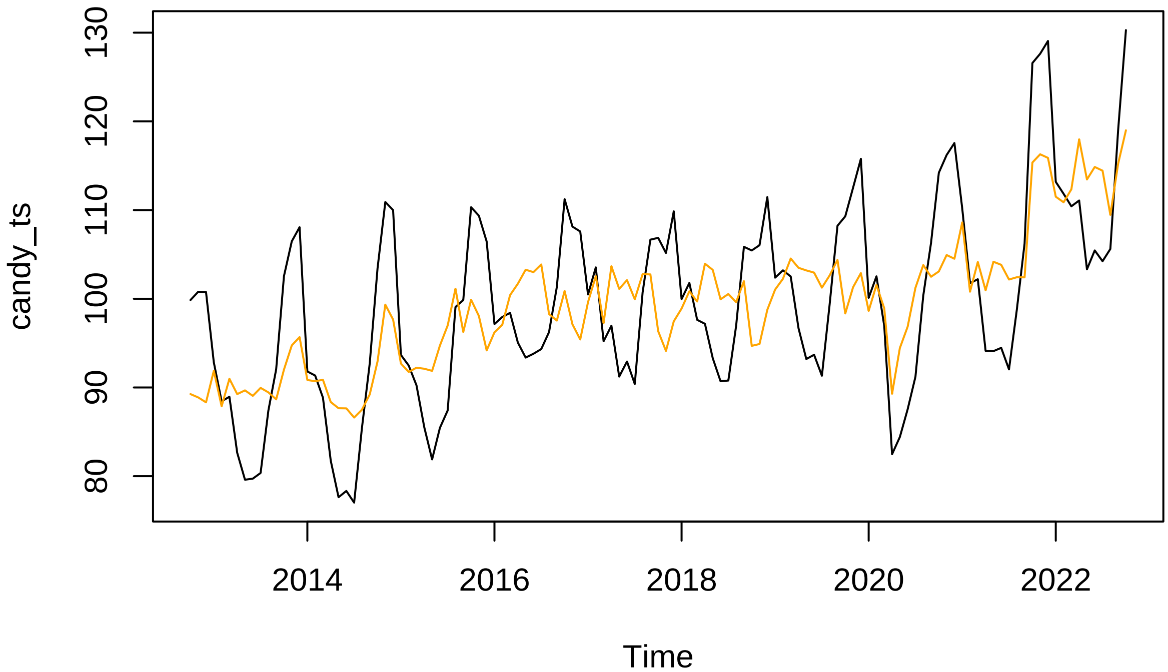



```
#Yes the time series is seasonal, and it is additive seasonal not multiplicative as the magnitude is co
#the month of October - December had pretty high seasonal values and later we also see that this trend
#months of October - December have really high seasonal values. I believe the reason for it is that aro
#we have the holidays season meaning thanksgiving, Christmas and new years so naturally the consumers w
#candies to celebrate the holidays. As for the rest of the years as expected the seasonal values are ne
#is like the holiday season and the production during those months is really high therefore in the mont
#seasonal values tend to become very low or even negative.

seasadj(stl_decomp)
```

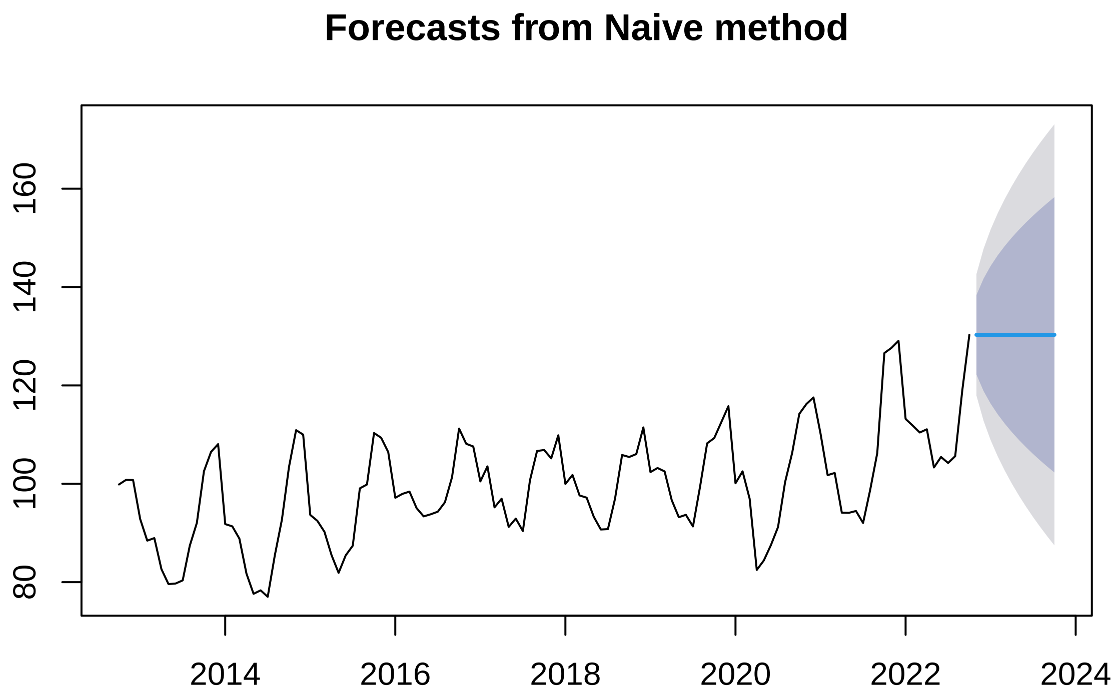
```
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2012
## 2013  91.88062  87.87708  90.98407  89.25356  89.67385  89.05835  89.95738
## 2014  90.84168  90.70741  90.86383  88.35425  87.65982  87.64734  86.61128
## 2015  92.69625  91.75144  92.22933  92.11430  91.87373  94.72498  96.99832
## 2016  96.22041  97.07102 100.41043 101.69494 103.27716 103.00168 103.86775
## 2017  99.62381 102.54467  97.25823 103.66682 101.11365 102.10257  99.94771
## 2018  98.89479 100.84269  99.68759 103.95354 103.25482  99.93921 100.54055
## 2019 101.03634 102.24696 104.53850 103.49291 103.19692 102.95254 101.25362
## 2020  98.63571 101.56299  98.92428  89.29256  94.44840  96.85469 101.23625
## 2021 108.60855 100.79684 104.15336 100.95259 104.17270 103.82849 102.17039
## 2022 111.50798 110.87794 112.34172 117.97554 113.44629 114.85941 114.44030
##            Aug       Sep       Oct       Nov       Dec
## 2012                      89.25498  88.87915  88.32464
## 2013  89.44865  88.65907  92.02034  94.74166  95.67885
## 2014  87.49943  89.21788  92.92835  99.33899  97.65576
```

```
## 2015 101.12973  96.25707  99.89406  98.06010  94.18092
## 2016  98.28470  97.54074 100.87718  97.12240  95.41118
## 2017 102.77803 102.74583  96.35000  94.12071  97.45412
## 2018  99.61578 101.98637  94.69433  94.90106  98.76204
## 2019 102.64281 104.38253  98.34425 101.30814 102.89594
## 2020 103.80269 102.48251 103.08021 104.93017 104.51314
## 2021 102.42532 102.41305 115.36222 116.28917 115.88368
## 2022 109.45688 115.22539 118.99323
```

```
plot(candy_ts)
lines(seasadj(stl_decomp), col="Orange")
```



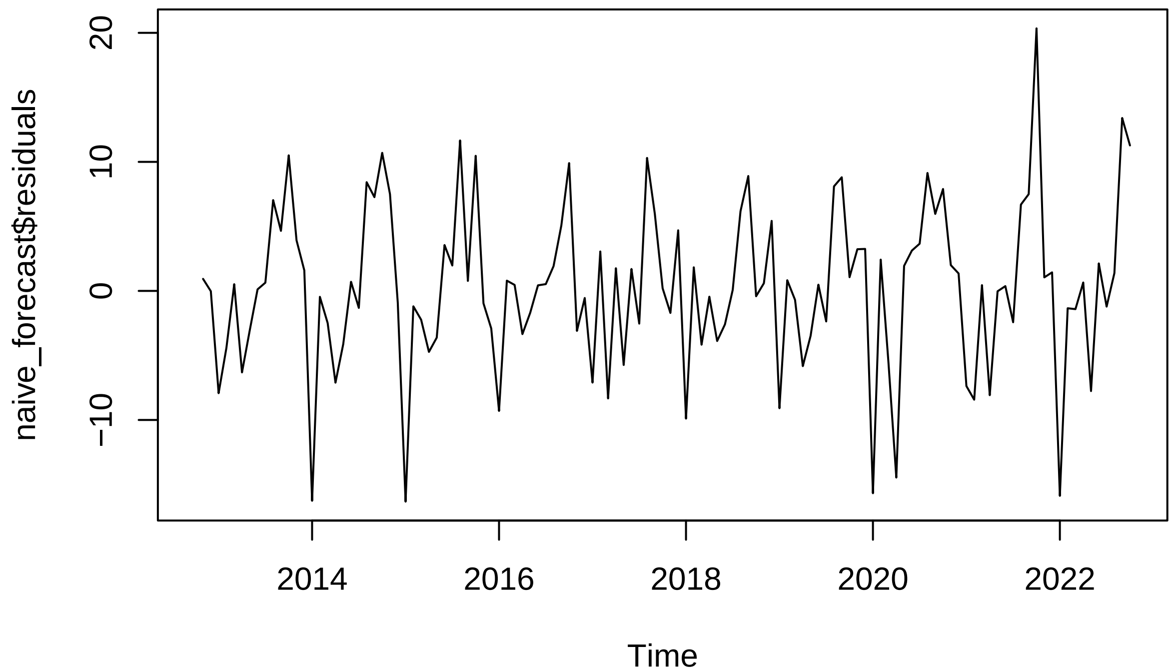

```
#After taking out the seasonal component the graph still looks the same in the overall direction and tr
#shrunk dramatically this means that the magnitude of values in the timeseries is affected but the seas
#seasonality has big fluctuations to the values of the timeseries.


#Naive

naive_forecast <- naive(candy_ts,12) #give me the forecast for the next 12 months using Naive forecast 
plot(naive_forecast)
```



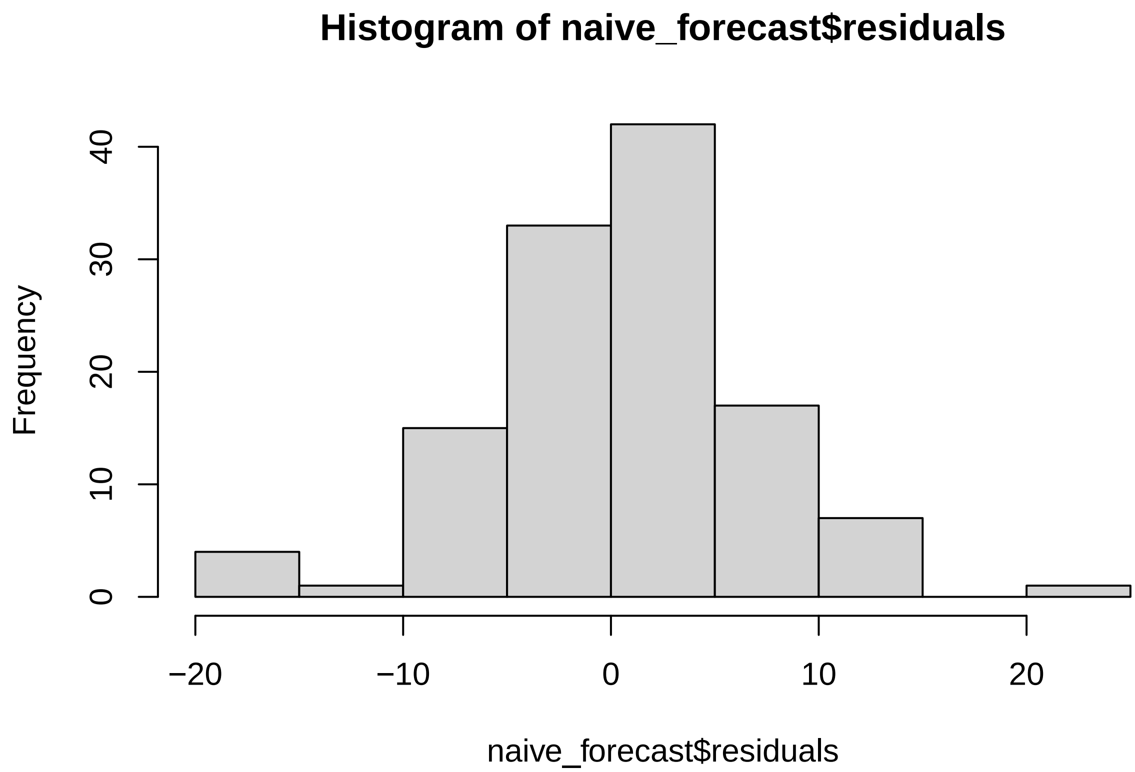

```
attributes(naive_forecast)
```

```
## $names
##  [1] "method"    "model"     "lambda"    "x"         "fitted"    "residuals"
##  [7] "series"    "mean"      "level"     "lower"     "upper"
##
## $class
## [1] "forecast"
```

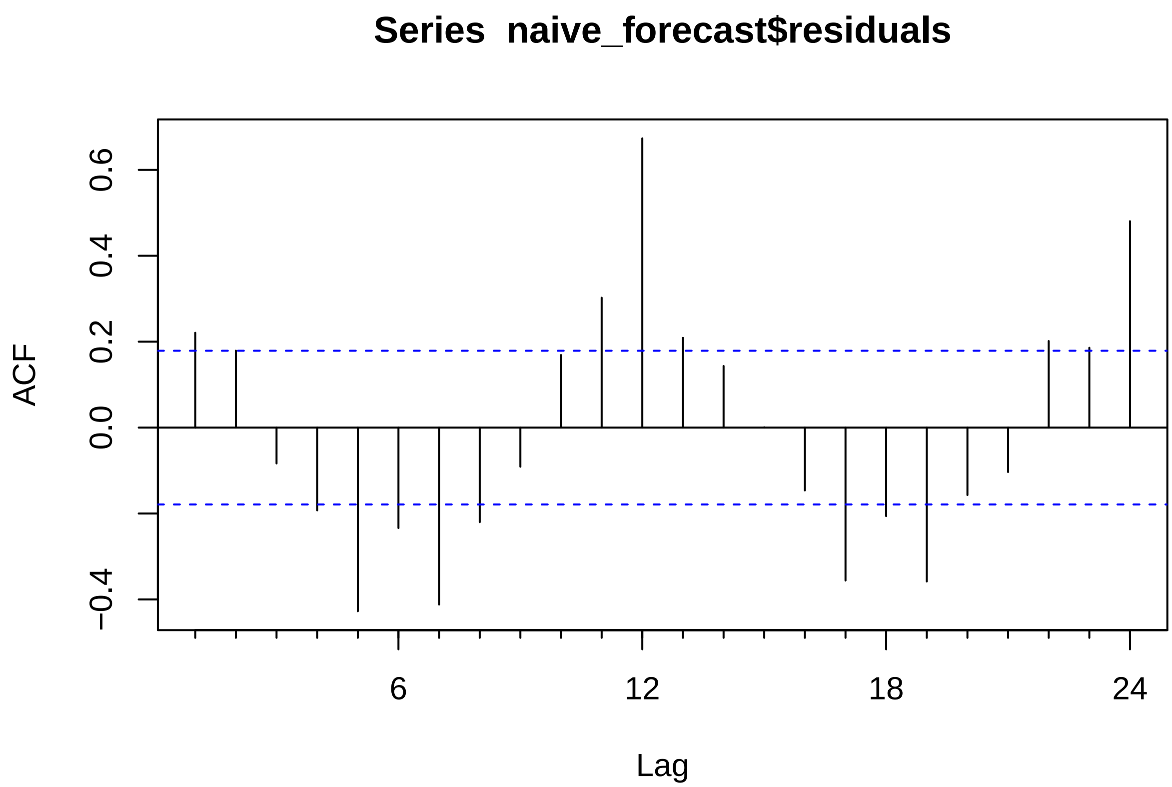
```
#$names
#[1] "method"    "model"     "lambda"    "x"         "fitted"    "residuals" "series"    "mean"      "l
#[11] "upper"

#$class
#[1] "forecast"

plot(naive_forecast$residuals) #the residual plot shows 6 anomalies(not within range from what I can se
```




```
#start of year 2014 where there is a sharp decline in production same goes for the start of year 2015,2
#beginning months of year 2022 and the 6th one being the sharp increase in the production just at the e

hist(naive_forecast$residuals) #shows a pretty decent normal distribution
```



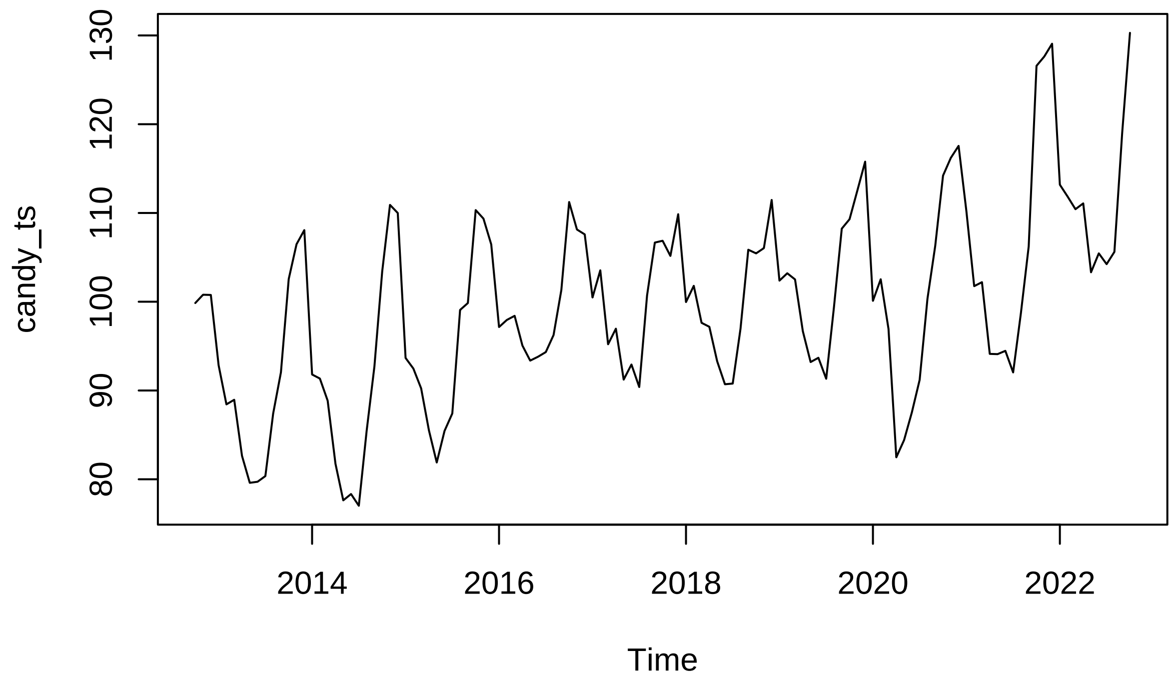

```
Acf(naive_forecast$residuals) #shows most values are out of the threshold region meaning they are depen
```



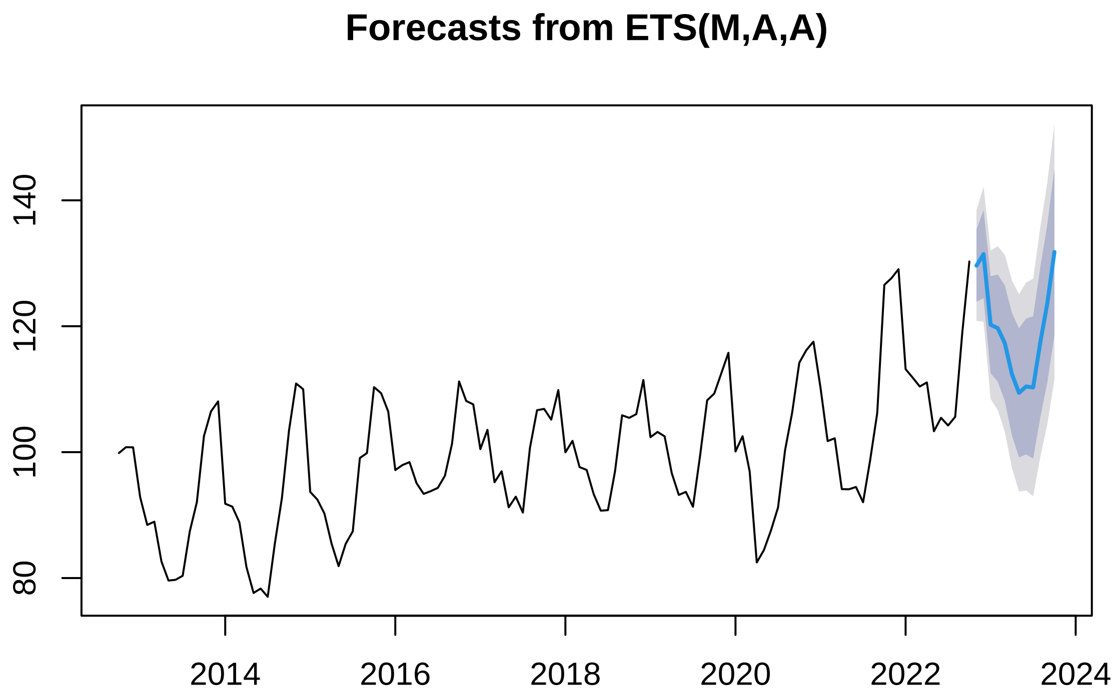

```r
#values (correaltion present).

accuracy(naive_forecast)
```

```
##                     ME     RMSE      MAE        MPE     MAPE    MASE      ACF1
## Training set 0.2536275 6.305638 4.625101 0.02809525 4.635846 1.05288 0.2207691
```

```r
#                   ME     RMSE      MAE        MPE      MAPE        MASE      ACF1
#Training set 0.2536275 6.305638 4.625101 0.02809525 4.635846   1.05288   0.2207691


ets_forecast <- ets(candy_ts)
plot(candy_ts)
```




```r
forecast_ets <- forecast(ets_forecast, h=12) #forecasting for the next 12 months
plot(forecast_ets)
```




```
forecast_ets
```

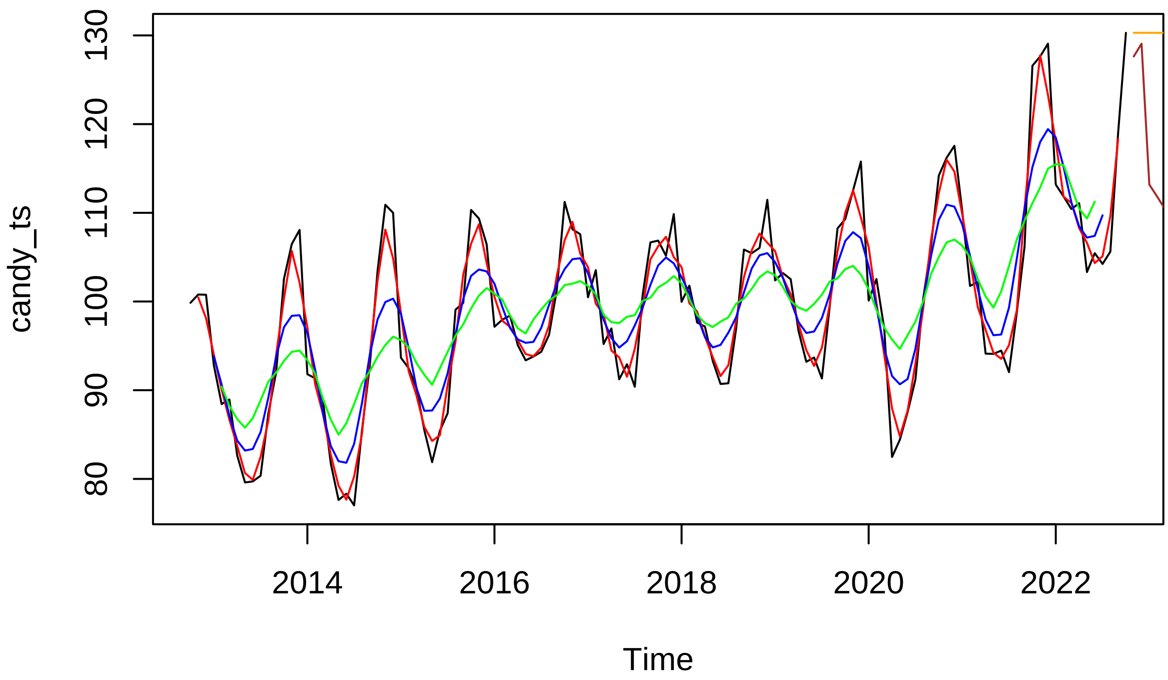
```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Nov 2022       129.6266 123.89521 135.3579 120.86121 138.3919
## Dec 2022       131.4585 124.44955 138.4674 120.73925 142.1777
## Jan 2023       120.2379 112.53103 127.9448 108.45126 132.0245
## Feb 2023       119.7004 111.19327 128.2074 106.68990 132.7108
## Mar 2023       117.3170 108.13076 126.5032 103.26787 131.3661
## Apr 2023       112.4141 102.67755 122.1506  97.52333 127.3049
## May 2023       109.4166  99.16483 119.6684  93.73787 125.0953
## Jun 2023       110.4319  99.63670 121.2271  93.92206 126.9418
## Jul 2023       110.2762  98.97651 121.5760  92.99479 127.5577
## Aug 2023       117.4103 105.49177 129.3289  99.18246 135.6382
## Sep 2023       123.6545 111.09743 136.2116 104.45011 142.8589
## Oct 2023       131.8069 118.54708 145.0667 111.52776 152.0860
```

```
#        Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
#Nov 2022       129.6266 123.89521 135.3579 120.86121 138.3919
#Dec 2022       131.4585 124.44955 138.4674 120.73925 142.1777
#Jan 2023       120.2379 112.53103 127.9448 108.45126 132.0245
#Feb 2023       119.7004 111.19327 128.2074 106.68990 132.7108
#Mar 2023       117.3170 108.13076 126.5032 103.26787 131.3661
#Apr 2023       112.4141 102.67755 122.1506  97.52333 127.3049
#May 2023       109.4166  99.16483 119.6684  93.73787 125.0953
#Jun 2023       110.4319  99.63670 121.2271  93.92206 126.9418
```

```
#Jul 2023       110.2762  98.97651 121.5760  92.99479 127.5577
#Aug 2023       117.4103 105.49177 129.3289  99.18246 135.6382
#Sep 2023       123.6545 111.09743 136.2116 104.45011 142.8589
#Oct 2023       131.8069 118.54708 145.0667 111.52776 152.0860

ets_forecast$mse
```

```
## [1] 10.95792
```

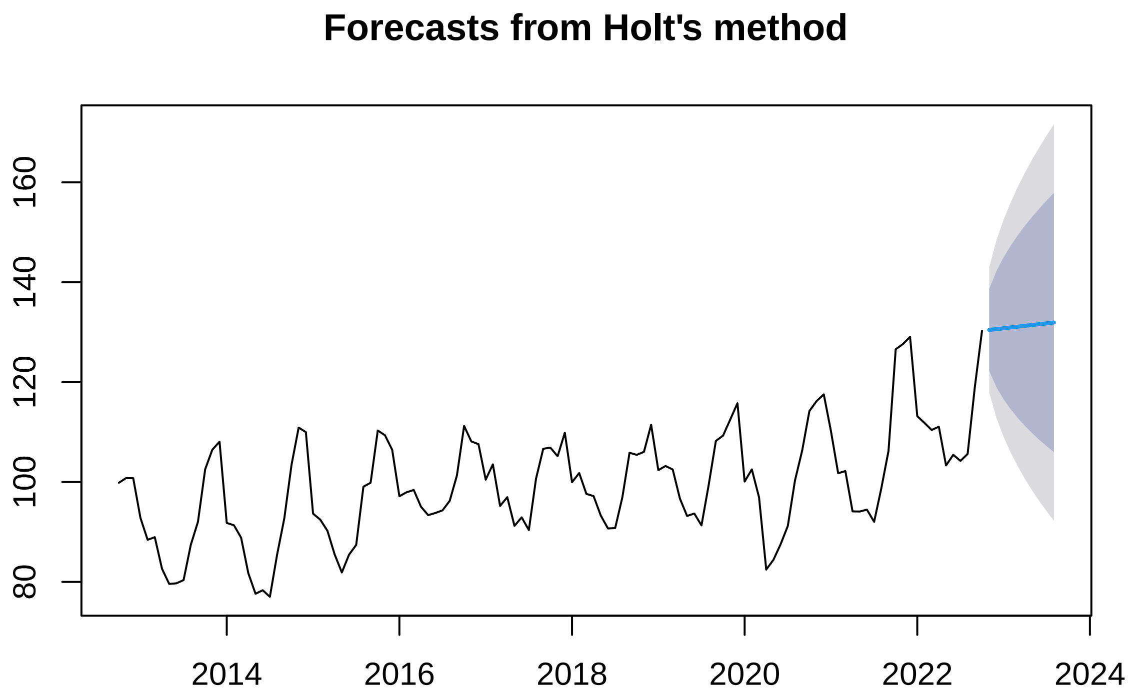
```
#[1] 10.95792
#RSME is 6.305638 on average forecast values were 6.305638 away from the actual
#MPE is 2.8% percenatge of error is around 2.8%


#Simple Moving Average

plot(candy_ts)
MA3_forecast <- ma(candy_ts,order=3) #taking into account the 5 most recent values
MA6_forecast <- ma(candy_ts,order=6) #taking into account the 9 most recent values
MA9_forecast <- ma(candy_ts,order=9) #taking into account the 9 most recent values


rwf_forecast <- rwf(candy_ts,12)
snaive_forecast <- snaive(candy_ts, 12)

plot(candy_ts)
lines(rwf_forecast$mean,col="Orange")
lines(snaive_forecast$mean,col="brown") #works best for my time series: snaive method
lines(MA3_forecast,col="Red")
lines(MA6_forecast,col="Blue")
lines(MA9_forecast,col="Green")
```




```
#as the order goes up the line comes up shorter and shorter from the end, and from the start as well, t
#smaller. And also as the order goes up the trend is somewhat followed but the magnitude shrinks in siz

#Simple Smoothing

SSE_Simple <- holt(candy_ts,gamma=FALSE)
attributes(SSE_Simple)
```

```
## $names
##  [1] "model"      "mean"       "level"      "x"          "upper"      "lower"
##  [7] "fitted"     "method"     "series"     "residuals"
##
## $class
## [1] "forecast"
```

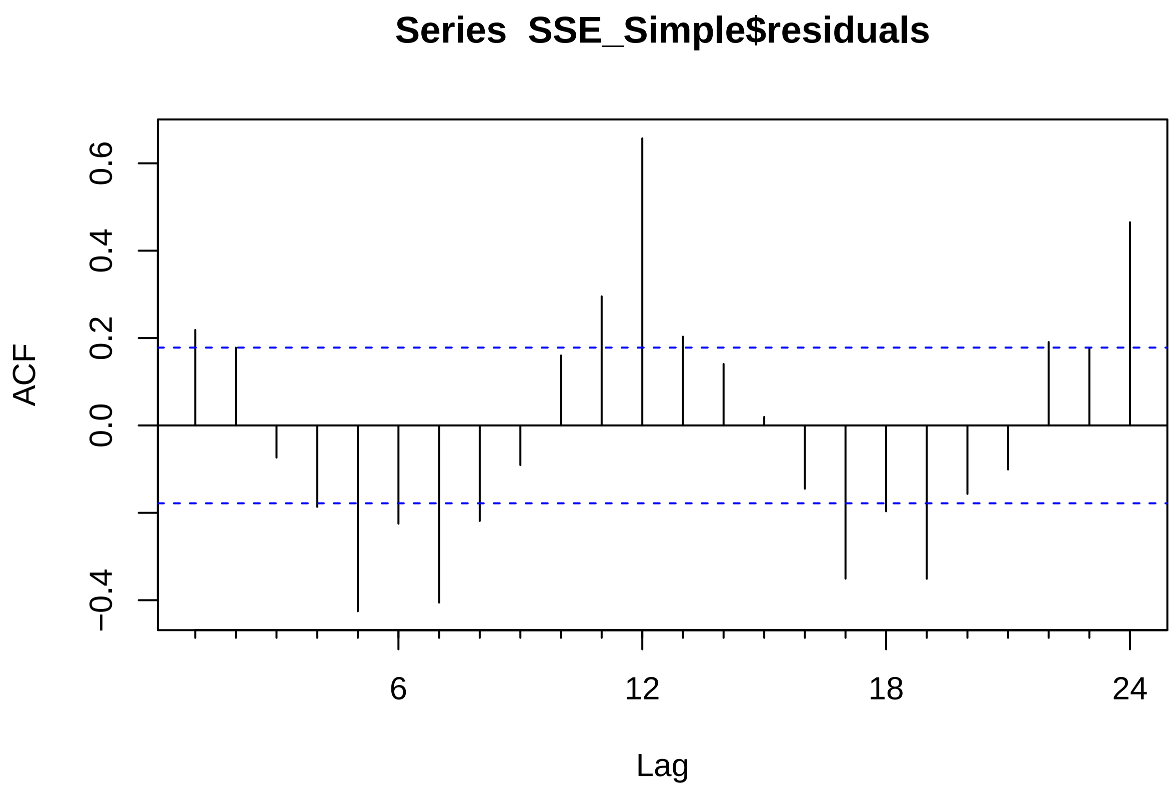
```
plot(SSE_Simple)
```




```
attributes(SSE_Simple)
```

```
## $names
##  [1] "model"     "mean"      "level"     "x"         "upper"     "lower"
##  [7] "fitted"    "method"    "series"    "residuals"
##
## $class
## [1] "forecast"
```

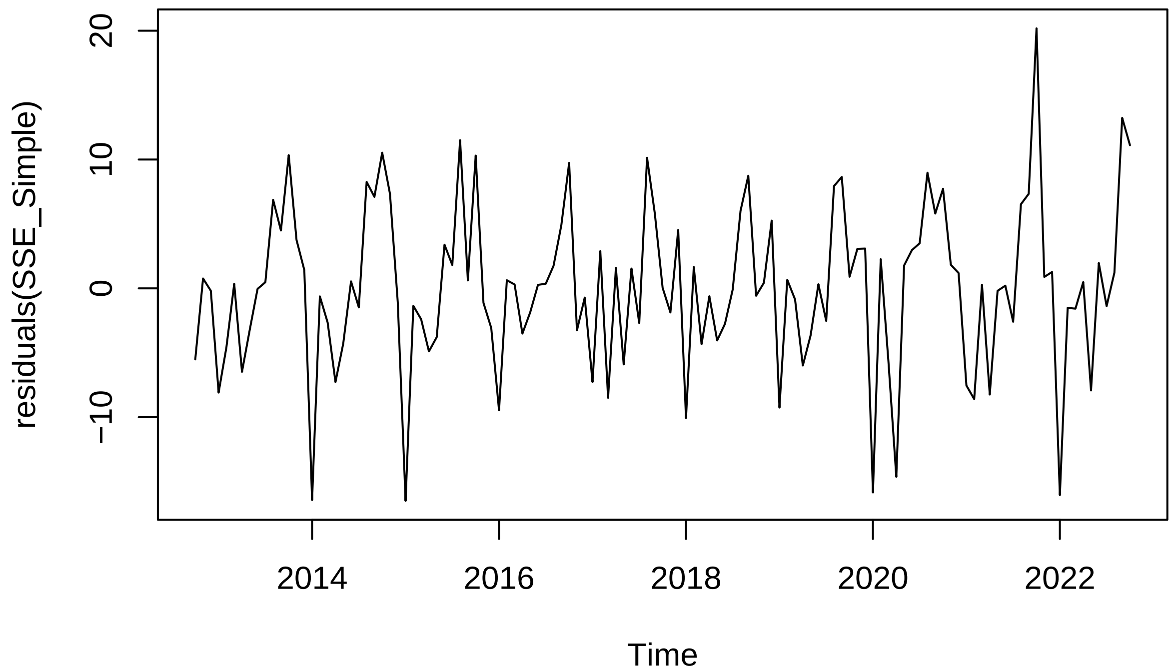
```
#$names
#[1] "model"     "mean"      "level"     "x"         "upper"     "lower"     "fitted"    "method"    "s

#$class
#[1] "forecast"

Acf(SSE_Simple$residuals)
```



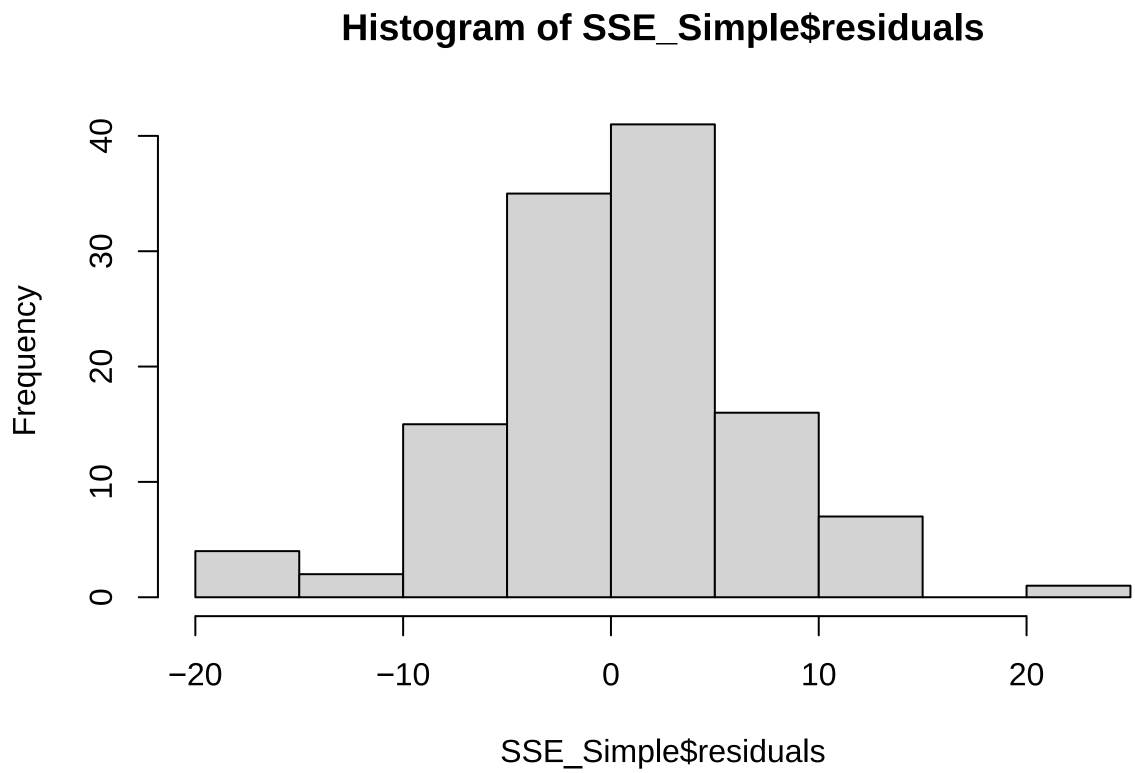

```
Box.test(residuals(SSE_Simple), lag=12, type="Ljung")
```

```
##
##  Box-Ljung test
##
## data:  residuals(SSE_Simple)
## X-squared = 147.8, df = 12, p-value < 2.2e-16
```

```
#data:  residuals(auto_fit)
#X-squared = 13.084, df = 12, p-value < 2.2e-16

#As we can see from the Ljung box statistic that the p-values are < 2.2e-16 which makes them significan
#hypothesis which means that the residual values are dependent

plot.ts(residuals(SSE_Simple))
```




```
hist(SSE_Simple$residuals)  #normal distribution
```



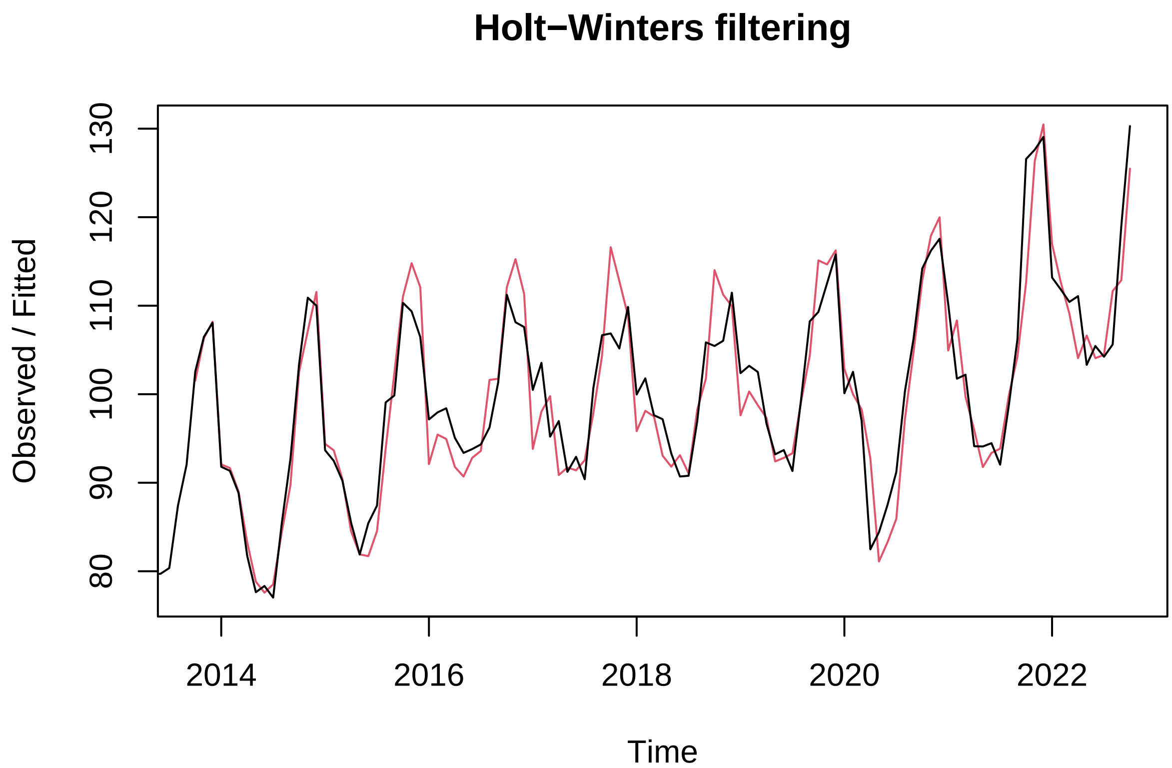

```r
accuracy(SSE_Simple)
```

```
##                      ME    RMSE      MAE        MPE     MAPE     MASE      ACF1
## Training set 0.04457017 6.29552 4.615979 -0.1819999 4.631386 1.050803 0.2185404
```

```r
#                  ME      RMSE       MAE        MPE      MAPE      MASE       ACF1
#Training set 0.04457017 6.29552 4.615979 -0.1819999 4.631386 1.050803 0.2185404

SSE_Simple$mse
```

```
## NULL
```

```r
#[1] NULL
#RSME is 6.29552 on average forecast values were 6.29552 away from the actual
#MPE is 18% percenatge of error is around 18%


#Holt Winters

HW_forecast <- HoltWinters(candy_ts)
plot(HW_forecast)
```

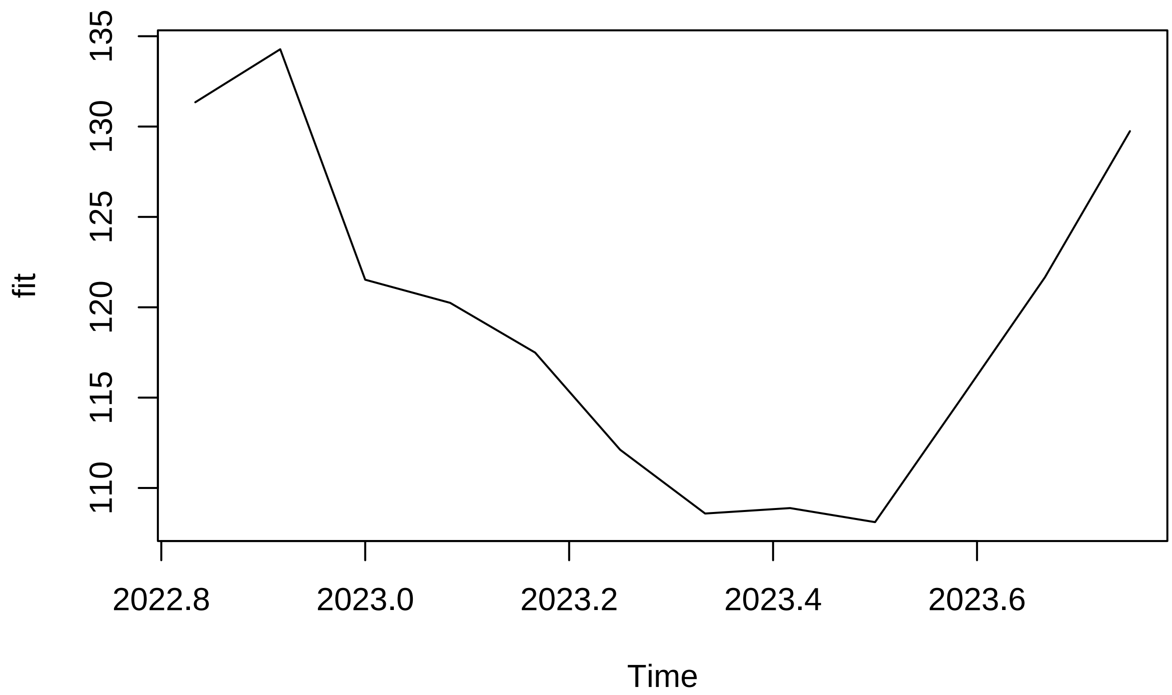



```
HW_forecast
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = candy_ts)
##
## Smoothing parameters:
##  alpha: 0.7137653
##  beta : 0
##  gamma: 0.3677262
##
## Coefficients:
##              [,1]
## a   118.58175151
## b     0.02735169
## s1   12.73645821
## s2   15.64347958
## s3    2.86140879
## s4    1.55347367
## s5   -1.22825112
## s6   -6.63034922
## s7  -10.18821076
## s8   -9.91381693
## s9  -10.71852359
## s10  -3.99912312
```

```
## s11   2.78200916
## s12  10.83734441
```

```
HWForecast <- predict(HW_forecast, 12)
plot(HWForecast)
```




```
HWForecast
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2022
## 2023 121.5252 120.2446 117.4903 112.1155 108.5850 108.8867 108.1094 114.8561
##           Sep      Oct      Nov      Dec
## 2022                   131.3456 134.2799
## 2023 121.6646 129.7473
```

```
HW_forecast #Alpha = 0.7137653 (the newest values does not have maximum weight but has relatively more 
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = candy_ts)
##
## Smoothing parameters:
##  alpha: 0.7137653
```

```
##  beta : 0
##  gamma: 0.3677262
##
## Coefficients:
##             [,1]
## a   118.58175151
## b     0.02735169
## s1   12.73645821
## s2   15.64347958
## s3    2.86140879
## s4    1.55347367
## s5   -1.22825112
## s6   -6.63034922
## s7  -10.18821076
## s8   -9.91381693
## s9  -10.71852359
## s10  -3.99912312
## s11   2.78200916
## s12  10.83734441
```

```
#beta : 0 (this is the coefficient of trend smoothing in HW)
#gamma: 0.3677262 (seasonal model)

ets_forecast #sigma: 0.0345 (standard dev. of residuals)
```

```
## ETS(M,A,A)
##
## Call:
##  ets(y = candy_ts)
##
##   Smoothing parameters:
##     alpha = 0.6829
##     beta  = 1e-04
##     gamma = 1e-04
##
##   Initial states:
##     l = 90.6178
##     b = 0.2533
##     s = 3.0373 -2.9535 -9.8347 -9.4264 -10.1889 -6.9389
##            -1.7831 0.8532 1.6436 13.117 11.537 10.9375
##
##   sigma:  0.0345
##
##      AIC     AICc      BIC
## 894.2047 900.1464 941.7331
```

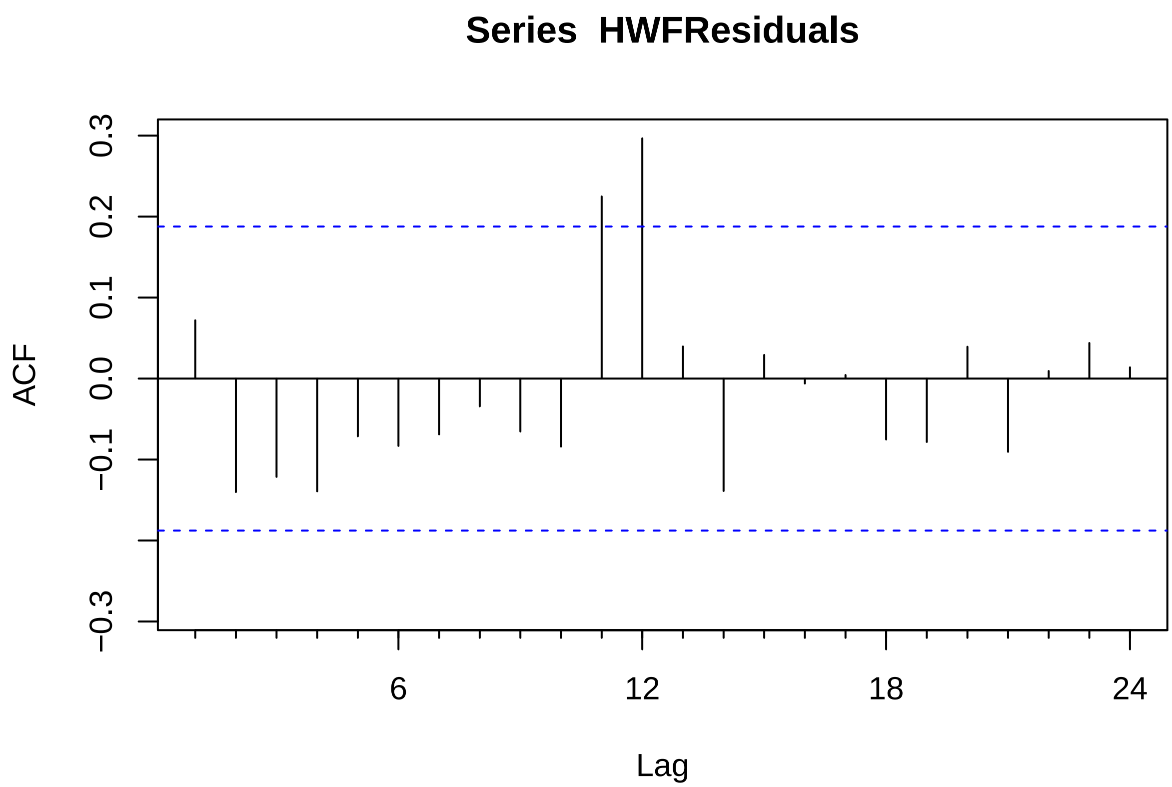
```
#Initial states:l = 90.6178
#b = 0.2533

# AIC     AICc      BIC
#894.2047 900.1464 941.7331

attributes(HW_forecast)
```

```
## $names
## [1] "fitted"       "x"            "alpha"        "beta"         "gamma"
## [6] "coefficients" "seasonal"     "SSE"          "call"
##
## $class
## [1] "HoltWinters"
```

```
#$names
#[1] "fitted"       "x"            "alpha"        "beta"         "gamma"        "coefficients" "seasona
#[9] "call"

#$class
#[1] "HoltWinters"

attributes(HWForecast)
```

```
## $dim
## [1] 12  1
##
## $dimnames
## $dimnames[[1]]
## NULL
##
## $dimnames[[2]]
## [1] "fit"
##
##
## $tsp
## [1] 2022.833 2023.750   12.000
##
## $class
## [1] "ts"
```

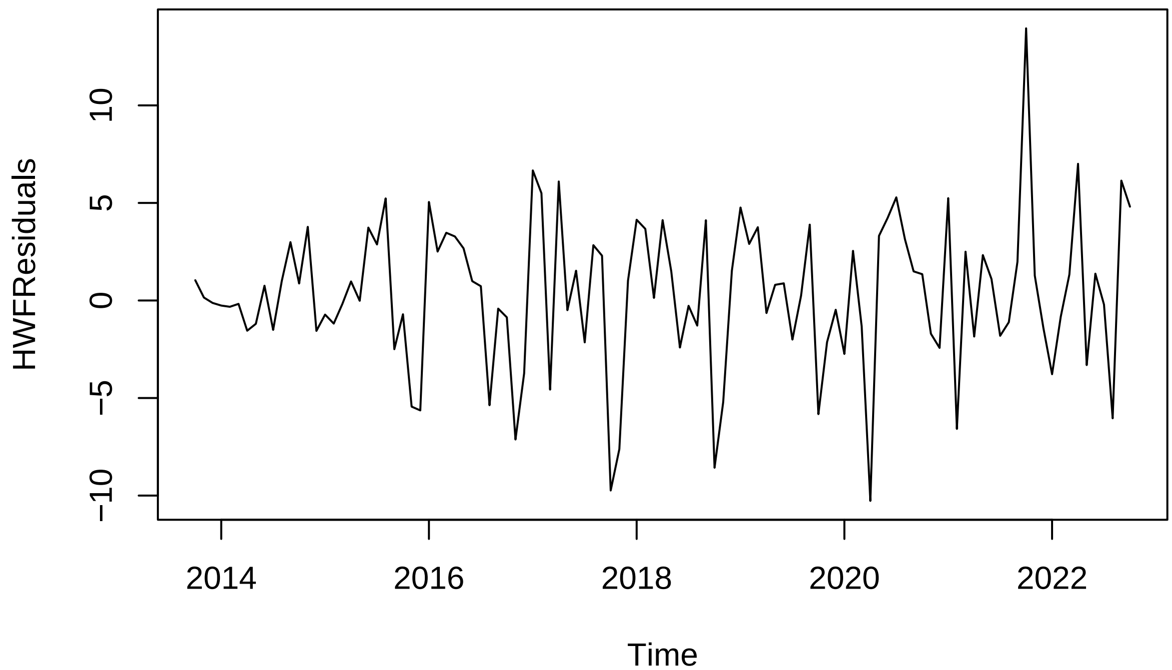
```
#$dim
#[1] 12  1

#$dimnames
#$dimnames[[1]]
#NULL

#$dimnames[[2]]
#[1] "fit"

#$tsp
#[1] 2022.833 2023.750   12.000

#$class
#[1] "ts"

HWFResiduals <-residuals(HW_forecast)

Acf(HWFResiduals) #most values seem to be in the threshold region meaning apart from the 2 values outsi
```



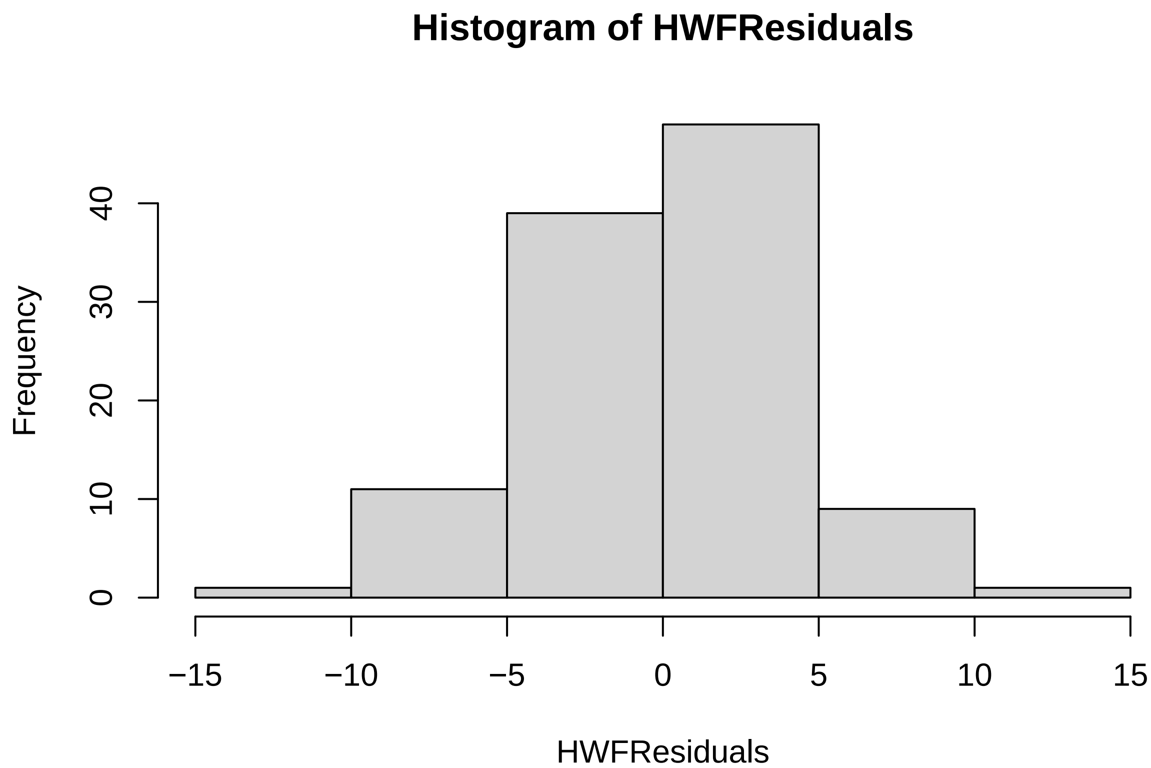

```
#values are dependent on the last value

Box.test(HWFResiduals, lag=12, type="Ljung")
```

```
##
##  Box-Ljung test
##
## data:  HWFResiduals
## X-squared = 27.423, df = 12, p-value = 0.006713
```

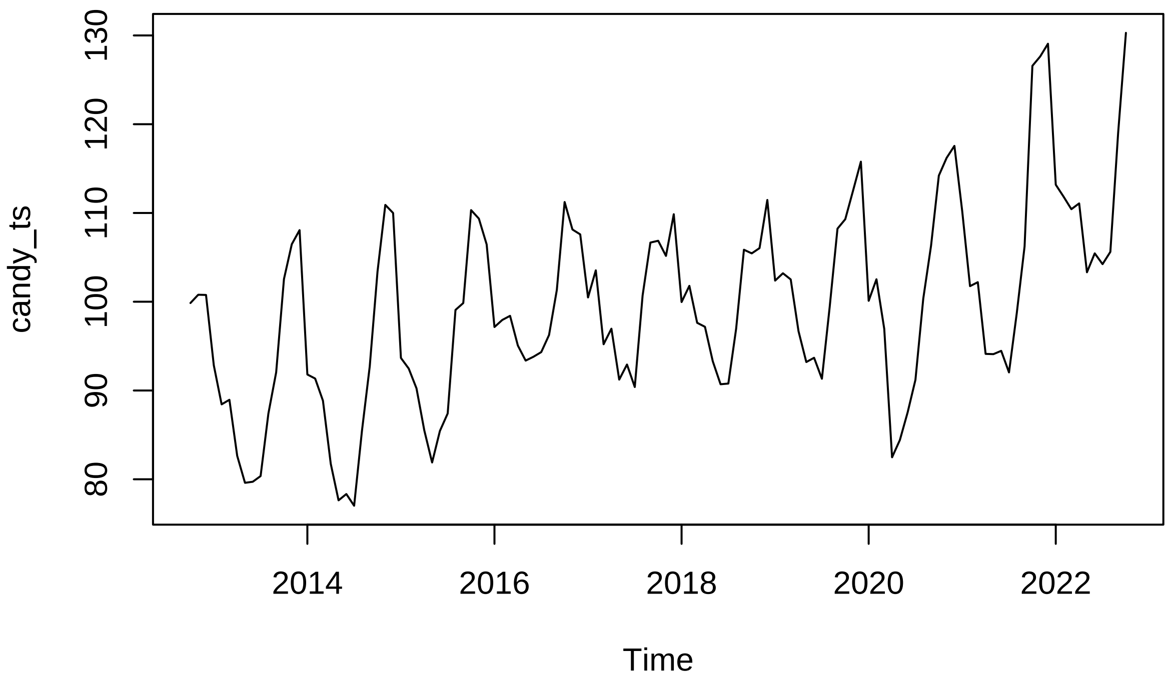
```
#Box-Ljung test
#data:  HWFResiduals
#X-squared = 27.423, df = 12, p-value = 0.006713

#As we can see from the Ljung box statistic that the p-values are 0.006713 which makes them significant
#hypothesis which means that the residual values are dependent.

plot.ts(HWFResiduals)
```




```
hist(HWFResiduals)  #normal distribution
```



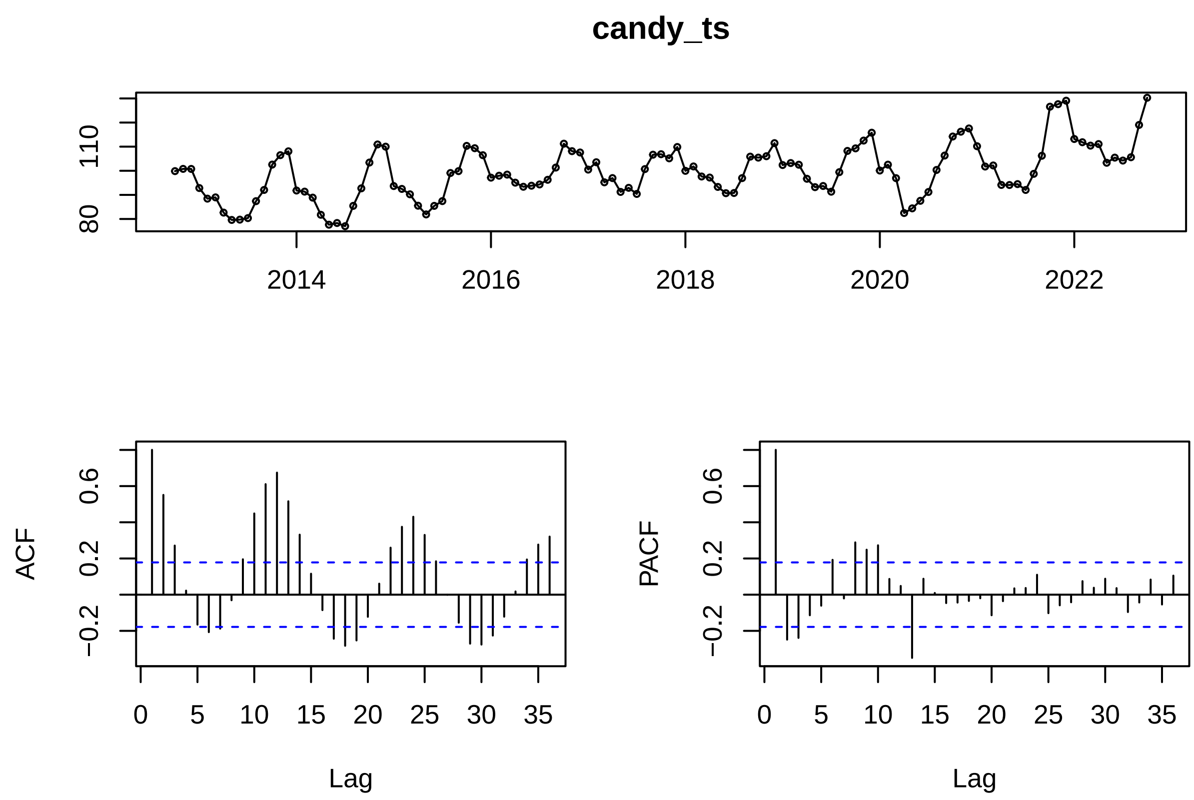

```
#ARIMA or Box-Jenkins

plot(candy_ts)
```



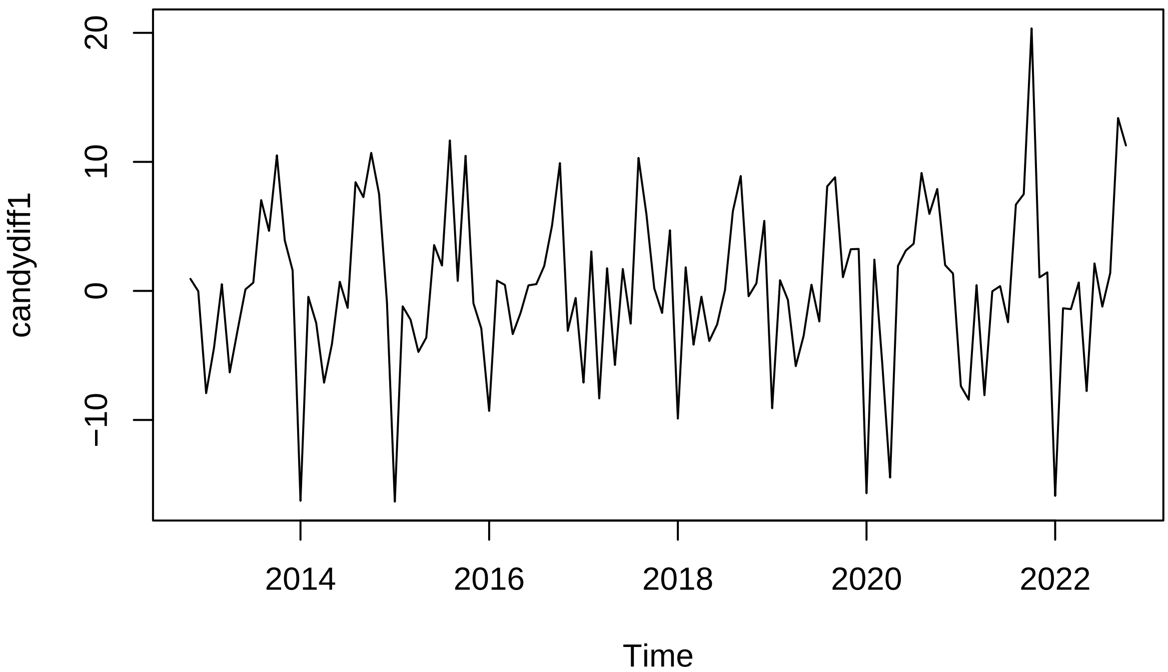

```r
ndiffs(candy_ts) #as the ndiffs function gives out the value 1 it means that the model is not stationar
```

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

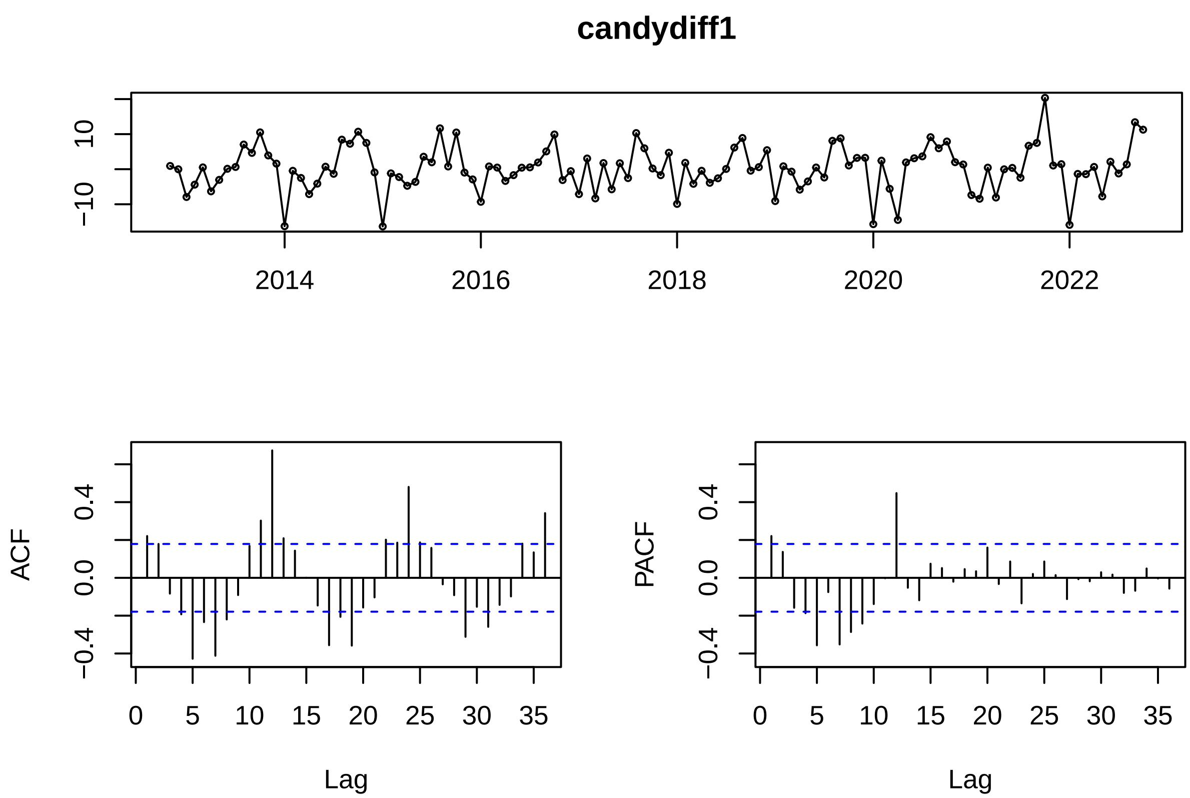
```r
#that we need 1 difference to get the stationary value.
#Seasonality is not needed for this.

tsdisplay(candy_ts)
```




```
candydiff1 <- diff(candy_ts, differences=1)
plot(candydiff1)
```




```
tsdisplay(candydiff1)
```




```r
auto_fit <- auto.arima(candy_ts, trace=TRUE, stepwise = FALSE)
```

```
##
##  ARIMA(0,0,0)(0,1,0)[12]                    : 695.7015
##  ARIMA(0,0,0)(0,1,0)[12] with drift         : 670.7642
##  ARIMA(0,0,0)(0,1,1)[12]                    : 695.999
##  ARIMA(0,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,0)(0,1,2)[12]                    : 696.3967
##  ARIMA(0,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,0)(1,1,0)[12]                    : 696.3465
##  ARIMA(0,0,0)(1,1,0)[12] with drift         : 672.2067
##  ARIMA(0,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(0,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,0)(1,1,2)[12]                    : Inf
##  ARIMA(0,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,0)(2,1,0)[12]                    : 697.5285
##  ARIMA(0,0,0)(2,1,0)[12] with drift         : 666.0752
##  ARIMA(0,0,0)(2,1,1)[12]                    : Inf
##  ARIMA(0,0,0)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,0)(2,1,2)[12]                    : Inf
##  ARIMA(0,0,0)(2,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(0,1,0)[12]                    : 654.6188
##  ARIMA(0,0,1)(0,1,0)[12] with drift         : 639.2968
##  ARIMA(0,0,1)(0,1,1)[12]                    : 656.1893
##  ARIMA(0,0,1)(0,1,1)[12] with drift         : Inf
```

```
##  ARIMA(0,0,1)(0,1,2)[12]                    : 657.6255
##  ARIMA(0,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(1,1,0)[12]                    : 656.284
##  ARIMA(0,0,1)(1,1,0)[12] with drift         : 637.5419
##  ARIMA(0,0,1)(1,1,1)[12]                    : 657.9208
##  ARIMA(0,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,1)(1,1,2)[12]                    : 659.7173
##  ARIMA(0,0,1)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(2,1,0)[12]                    : 657.8574
##  ARIMA(0,0,1)(2,1,0)[12] with drift         : 634.4358
##  ARIMA(0,0,1)(2,1,1)[12]                    : 659.7902
##  ARIMA(0,0,1)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,1)(2,1,2)[12]                    : Inf
##  ARIMA(0,0,1)(2,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(0,1,0)[12]                    : 636.3448
##  ARIMA(0,0,2)(0,1,0)[12] with drift         : 626.0602
##  ARIMA(0,0,2)(0,1,1)[12]                    : 637.898
##  ARIMA(0,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,2)(0,1,2)[12]                    : 636.7653
##  ARIMA(0,0,2)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(1,1,0)[12]                    : 638.0881
##  ARIMA(0,0,2)(1,1,0)[12] with drift         : 625.6669
##  ARIMA(0,0,2)(1,1,1)[12]                    : 638.334
##  ARIMA(0,0,2)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,2)(1,1,2)[12]                    : 638.992
##  ARIMA(0,0,2)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(2,1,0)[12]                    : 637.8508
##  ARIMA(0,0,2)(2,1,0)[12] with drift         : 621.3809
##  ARIMA(0,0,2)(2,1,1)[12]                    : 638.4469
##  ARIMA(0,0,2)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(0,1,0)[12]                    : 629.9275
##  ARIMA(0,0,3)(0,1,0)[12] with drift         : 622.7562
##  ARIMA(0,0,3)(0,1,1)[12]                    : 629.7434
##  ARIMA(0,0,3)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(0,1,2)[12]                    : 629.0504
##  ARIMA(0,0,3)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,3)(1,1,0)[12]                    : 630.4963
##  ARIMA(0,0,3)(1,1,0)[12] with drift         : 620.9963
##  ARIMA(0,0,3)(1,1,1)[12]                    : 629.6219
##  ARIMA(0,0,3)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(2,1,0)[12]                    : 630.7565
##  ARIMA(0,0,3)(2,1,0)[12] with drift         : 618.3351
##  ARIMA(0,0,4)(0,1,0)[12]                    : 629.5131
##  ARIMA(0,0,4)(0,1,0)[12] with drift         : 623.8854
##  ARIMA(0,0,4)(0,1,1)[12]                    : 628.2313
##  ARIMA(0,0,4)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,4)(1,1,0)[12]                    : 629.5068
##  ARIMA(0,0,4)(1,1,0)[12] with drift         : 621.9546
##  ARIMA(0,0,5)(0,1,0)[12]                    : 630.6005
##  ARIMA(0,0,5)(0,1,0)[12] with drift         : 625.8286
##  ARIMA(1,0,0)(0,1,0)[12]                    : 622.9559
##  ARIMA(1,0,0)(0,1,0)[12] with drift         : 619.2546
##  ARIMA(1,0,0)(0,1,1)[12]                    : 616.096
##  ARIMA(1,0,0)(0,1,1)[12] with drift         : Inf
```

```
##  ARIMA(1,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(1,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,0)(1,1,0)[12]                    : 620.045
##  ARIMA(1,0,0)(1,1,0)[12] with drift         : 615.4554
##  ARIMA(1,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(1,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,0)(1,1,2)[12]                    : Inf
##  ARIMA(1,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(1,0,0)(2,1,0)[12]                    : 619.4956
##  ARIMA(1,0,0)(2,1,0)[12] with drift         : 613.3055
##  ARIMA(1,0,0)(2,1,1)[12]                    : Inf
##  ARIMA(1,0,0)(2,1,1)[12] with drift         : Inf
##  ARIMA(1,0,0)(2,1,2)[12]                    : Inf
##  ARIMA(1,0,0)(2,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(0,1,0)[12]                    : 621.839
##  ARIMA(1,0,1)(0,1,0)[12] with drift         : 619.5991
##  ARIMA(1,0,1)(0,1,1)[12]                    : 615.267
##  ARIMA(1,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,1)(0,1,2)[12]                    : Inf
##  ARIMA(1,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(1,1,0)[12]                    : 618.9704
##  ARIMA(1,0,1)(1,1,0)[12] with drift         : 616.1254
##  ARIMA(1,0,1)(1,1,1)[12]                    : Inf
##  ARIMA(1,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,1)(1,1,2)[12]                    : Inf
##  ARIMA(1,0,1)(1,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(2,1,0)[12]                    : 618.2582
##  ARIMA(1,0,1)(2,1,0)[12] with drift         : 614.2405
##  ARIMA(1,0,1)(2,1,1)[12]                    : Inf
##  ARIMA(1,0,1)(2,1,1)[12] with drift         : Inf
##  ARIMA(1,0,2)(0,1,0)[12]                    : 623.7407
##  ARIMA(1,0,2)(0,1,0)[12] with drift         : 620.8486
##  ARIMA(1,0,2)(0,1,1)[12]                    : Inf
##  ARIMA(1,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,2)(0,1,2)[12]                    : Inf
##  ARIMA(1,0,2)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,2)(1,1,0)[12]                    : 621.1253
##  ARIMA(1,0,2)(1,1,0)[12] with drift         : 618.2083
##  ARIMA(1,0,2)(1,1,1)[12]                    : Inf
##  ARIMA(1,0,2)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,2)(2,1,0)[12]                    : 620.3847
##  ARIMA(1,0,2)(2,1,0)[12] with drift         : 616.1228
##  ARIMA(1,0,3)(0,1,0)[12]                    : 624.0376
##  ARIMA(1,0,3)(0,1,0)[12] with drift         : 622.9934
##  ARIMA(1,0,3)(0,1,1)[12]                    : Inf
##  ARIMA(1,0,3)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,3)(1,1,0)[12]                    : 621.5939
##  ARIMA(1,0,3)(1,1,0)[12] with drift         : 620.4402
##  ARIMA(1,0,4)(0,1,0)[12]                    : 625.2055
##  ARIMA(1,0,4)(0,1,0)[12] with drift         : 624.5141
##  ARIMA(2,0,0)(0,1,0)[12]                    : 621.502
##  ARIMA(2,0,0)(0,1,0)[12] with drift         : 619.1749
##  ARIMA(2,0,0)(0,1,1)[12]                    : 615.6749
##  ARIMA(2,0,0)(0,1,1)[12] with drift         : Inf
```

```
##  ARIMA(2,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(2,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(2,0,0)(1,1,0)[12]                    : 619.0898
##  ARIMA(2,0,0)(1,1,0)[12] with drift         : 616.0103
##  ARIMA(2,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(2,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(2,0,0)(1,1,2)[12]                    : Inf
##  ARIMA(2,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(2,0,0)(2,1,0)[12]                    : 618.3519
##  ARIMA(2,0,0)(2,1,0)[12] with drift         : 614.0563
##  ARIMA(2,0,0)(2,1,1)[12]                    : Inf
##  ARIMA(2,0,0)(2,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(0,1,0)[12]                    : 623.5522
##  ARIMA(2,0,1)(0,1,0)[12] with drift         : 621.0785
##  ARIMA(2,0,1)(0,1,1)[12]                    : Inf
##  ARIMA(2,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(0,1,2)[12]                    : Inf
##  ARIMA(2,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(2,0,1)(1,1,0)[12]                    : 619.2936
##  ARIMA(2,0,1)(1,1,0)[12] with drift         : 618.2342
##  ARIMA(2,0,1)(1,1,1)[12]                    : Inf
##  ARIMA(2,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(2,1,0)[12]                    : 617.3422
##  ARIMA(2,0,1)(2,1,0)[12] with drift         : Inf
##  ARIMA(2,0,2)(0,1,0)[12]                    : 625.7487
##  ARIMA(2,0,2)(0,1,0)[12] with drift         : 623.077
##  ARIMA(2,0,2)(0,1,1)[12]                    : Inf
##  ARIMA(2,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(2,0,2)(1,1,0)[12]                    : 621.2392
##  ARIMA(2,0,2)(1,1,0)[12] with drift         : 620.4897
##  ARIMA(2,0,3)(0,1,0)[12]                    : 625.4338
##  ARIMA(2,0,3)(0,1,0)[12] with drift         : 624.7418
##  ARIMA(3,0,0)(0,1,0)[12]                    : 623.565
##  ARIMA(3,0,0)(0,1,0)[12] with drift         : 620.9597
##  ARIMA(3,0,0)(0,1,1)[12]                    : 617.5374
##  ARIMA(3,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(3,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(3,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(3,0,0)(1,1,0)[12]                    : 621.2407
##  ARIMA(3,0,0)(1,1,0)[12] with drift         : 618.2282
##  ARIMA(3,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(3,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(3,0,0)(2,1,0)[12]                    : Inf
##  ARIMA(3,0,0)(2,1,0)[12] with drift         : 616.1571
##  ARIMA(3,0,1)(0,1,0)[12]                    : Inf
##  ARIMA(3,0,1)(0,1,0)[12] with drift         : Inf
##  ARIMA(3,0,1)(0,1,1)[12]                    : Inf
##  ARIMA(3,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(3,0,1)(1,1,0)[12]                    : Inf
##  ARIMA(3,0,1)(1,1,0)[12] with drift         : 620.2729
##  ARIMA(3,0,2)(0,1,0)[12]                    : Inf
##  ARIMA(3,0,2)(0,1,0)[12] with drift         : Inf
##  ARIMA(4,0,0)(0,1,0)[12]                    : 625.6611
##  ARIMA(4,0,0)(0,1,0)[12] with drift         : 623.1987
```

```
##  ARIMA(4,0,0)(0,1,1)[12]                    : 619.4666
##  ARIMA(4,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(4,0,0)(1,1,0)[12]                    : 623.4068
##  ARIMA(4,0,0)(1,1,0)[12] with drift         : 620.4931
##  ARIMA(4,0,1)(0,1,0)[12]                    : Inf
##  ARIMA(4,0,1)(0,1,0)[12] with drift         : Inf
##  ARIMA(5,0,0)(0,1,0)[12]                    : 626.6308
##  ARIMA(5,0,0)(0,1,0)[12] with drift         : 624.8284
##
##
##
##  Best model: ARIMA(1,0,0)(2,1,0)[12] with drift
```

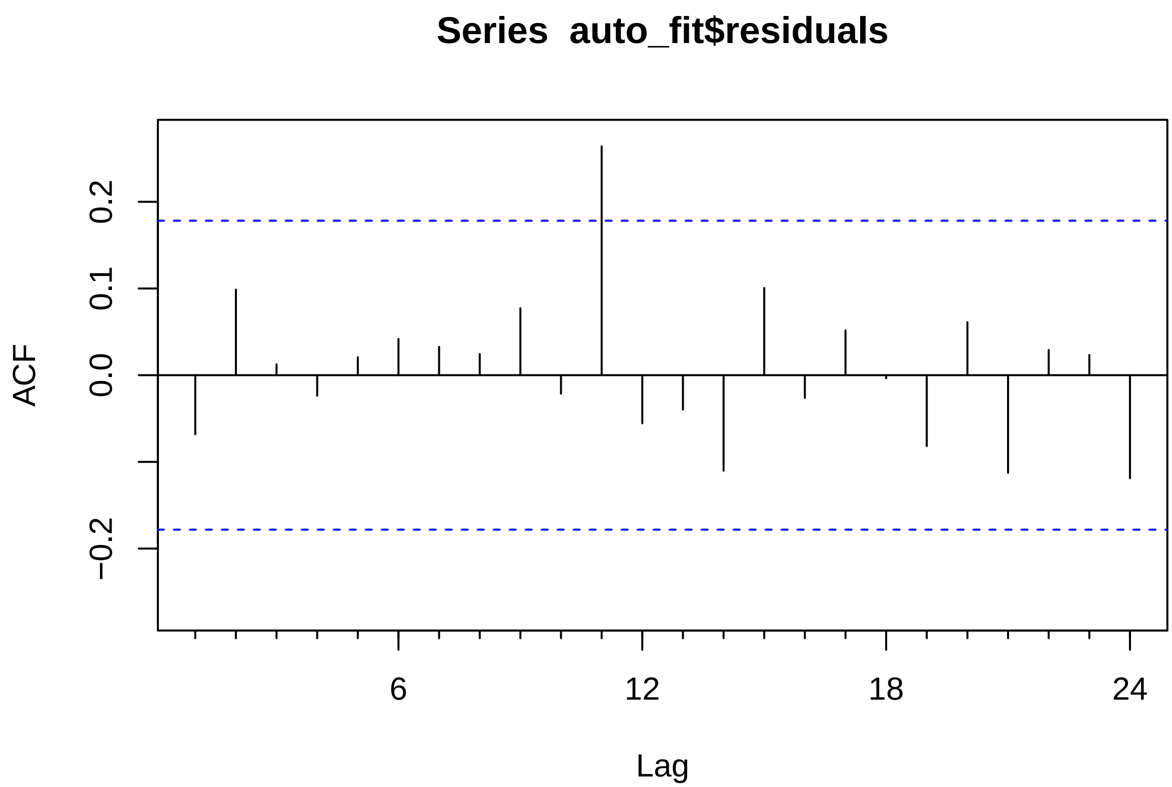
```
# Best model: ARIMA(1,0,0)(2,1,0)[12] with drift

auto_fit
```

```
## Series: candy_ts
## ARIMA(1,0,0)(2,1,0)[12] with drift
##
## Coefficients:
##          ar1     sar1     sar2   drift
##       0.6441  -0.3042  -0.2351  0.2083
## s.e.  0.0742   0.1040   0.1099  0.0579
##
## sigma^2 = 14.95:  log likelihood = -301.36
## AIC=612.72   AICc=613.31   BIC=626.18
```

```
# Series: candy_ts
#ARIMA(1,0,0)(2,1,0)[12] with drift

#Coefficients:
#        ar1      sar1      sar2   drift
#      0.6441   -0.3042   -0.2351  0.2083
#s.e.  0.0742    0.1040    0.1099  0.0579

#sigma^2 = 14.95:  log likelihood = -301.36
#AIC=612.72   AICc=613.31   BIC=626.18


attributes(auto_fit)
```

```
## $names
##  [1] "coef"      "sigma2"    "var.coef"  "mask"      "loglik"    "aic"
##  [7] "arma"      "residuals" "call"      "series"    "code"      "n.cond"
## [13] "nobs"      "model"     "xreg"      "bic"       "aicc"      "x"
## [19] "fitted"
##
## $class
## [1] "forecast_ARIMA" "ARIMA"          "Arima"
```

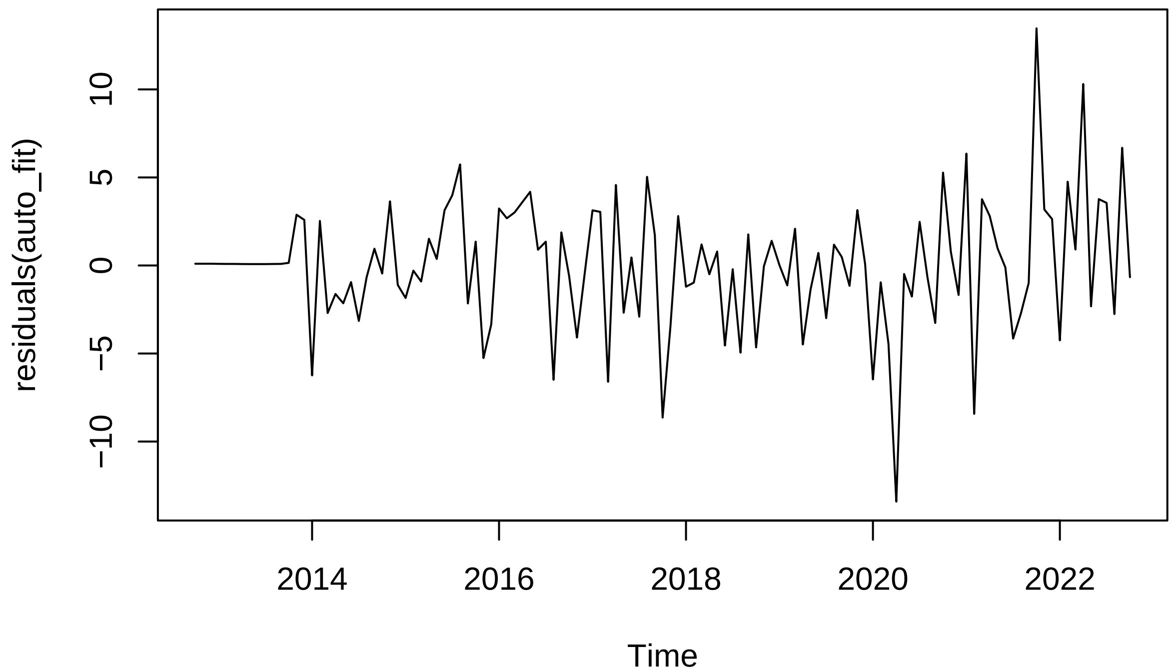
```
#$names
#[1] "coef"      "sigma2"    "var.coef"  "mask"      "loglik"    "aic"       "arma"      "residuals" "c
#[11] "code"      "n.cond"    "nobs"      "model"     "xreg"      "bic"       "aicc"      "x"         "

#$class
#[1] "forecast_ARIMA" "ARIMA"          "Arima"


Acf(auto_fit$residuals)
```



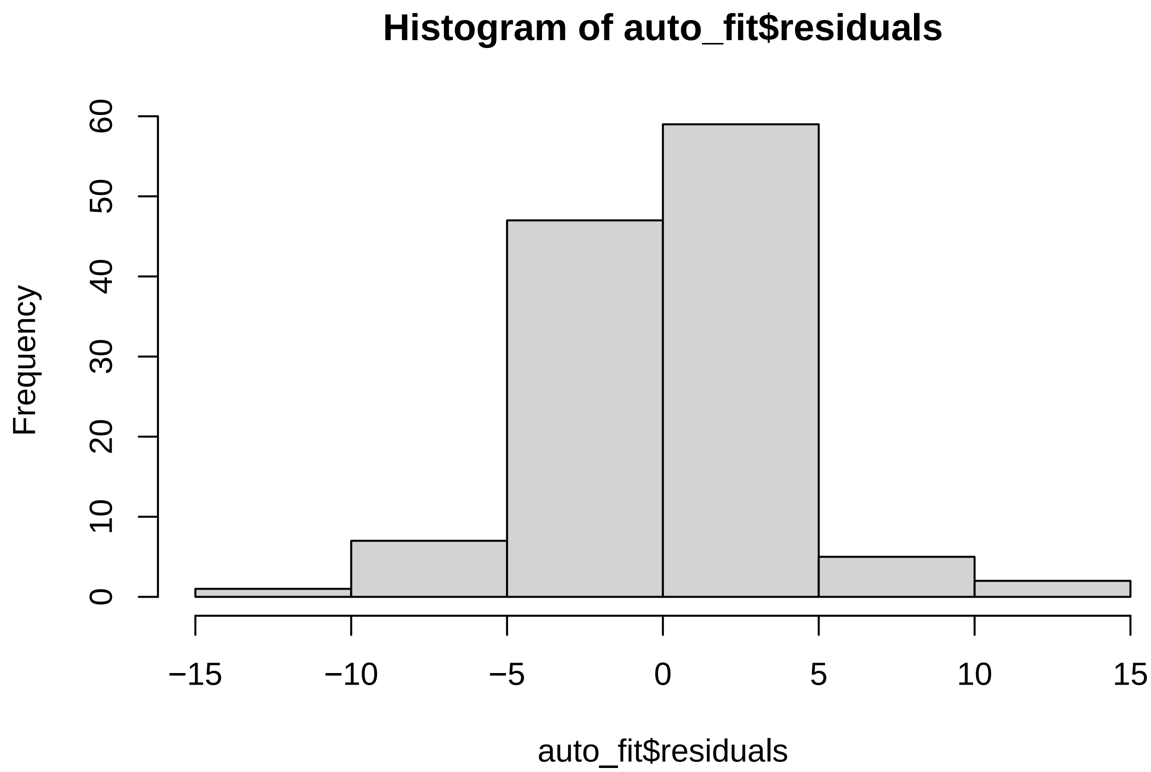

```
Box.test(residuals(auto_fit), lag=12, type="Ljung")
```

```
##
##  Box-Ljung test
##
## data:  residuals(auto_fit)
## X-squared = 13.084, df = 12, p-value = 0.363
```

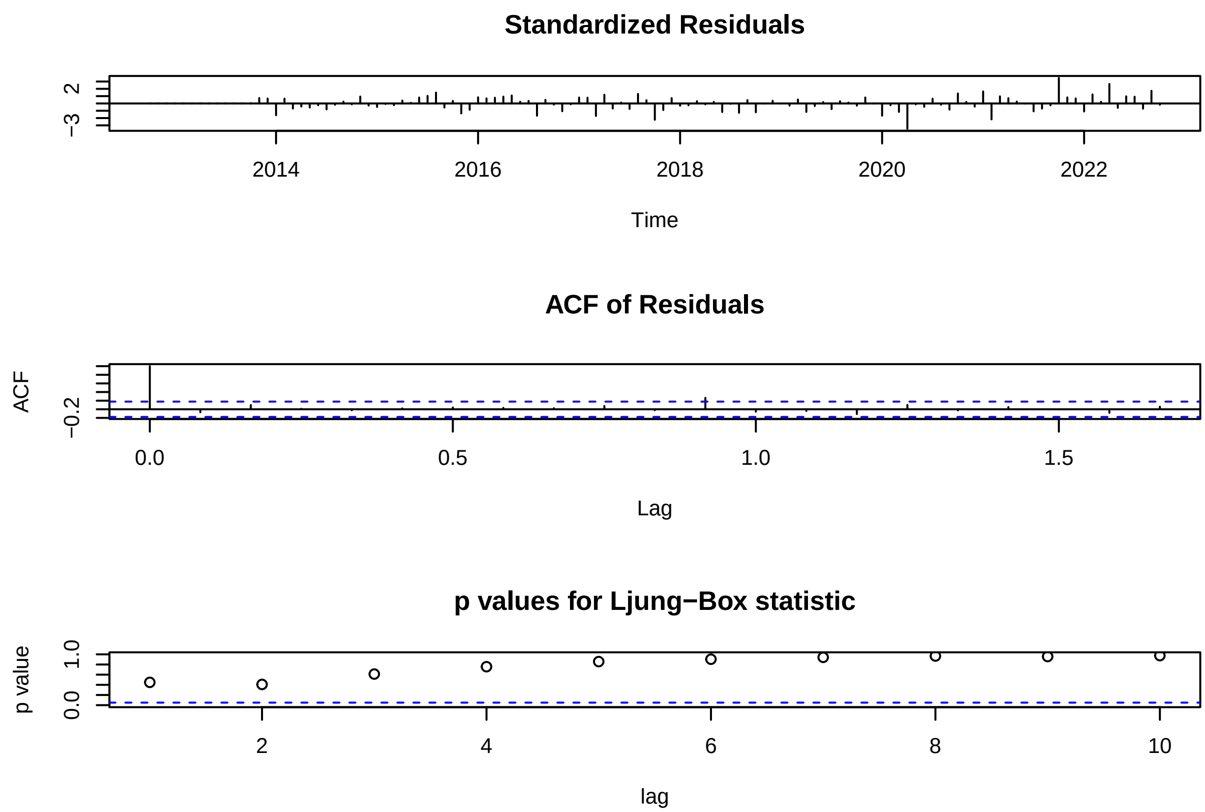
```
#data:  residuals(auto_fit)
#X-squared = 13.084, df = 12, p-value = 0.363

#As we can see from the Ljung box statistic that the p-values are 0.363 which makes them non-significan
#hypothesis which means that the residual values are independent

plot.ts(residuals(auto_fit))
```




```
hist(auto_fit$residuals)   #normal distribution
```



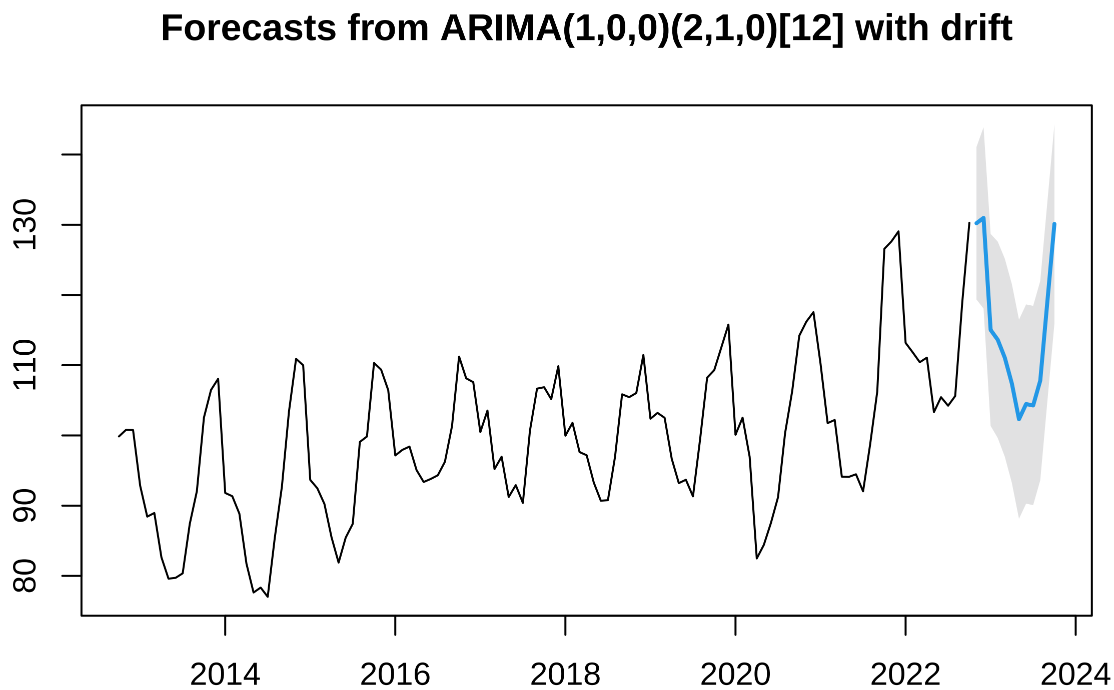

```
tsdiag(auto_fit) #ACF test shows no relative significance amongst residuals as well same as the p-values
```




```r
accuracy(auto_fit)
```

```
##                        ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -0.004444951 3.602064 2.583457 -0.1100037 2.571757 0.5881103
##                     ACF1
## Training set -0.06809078
```

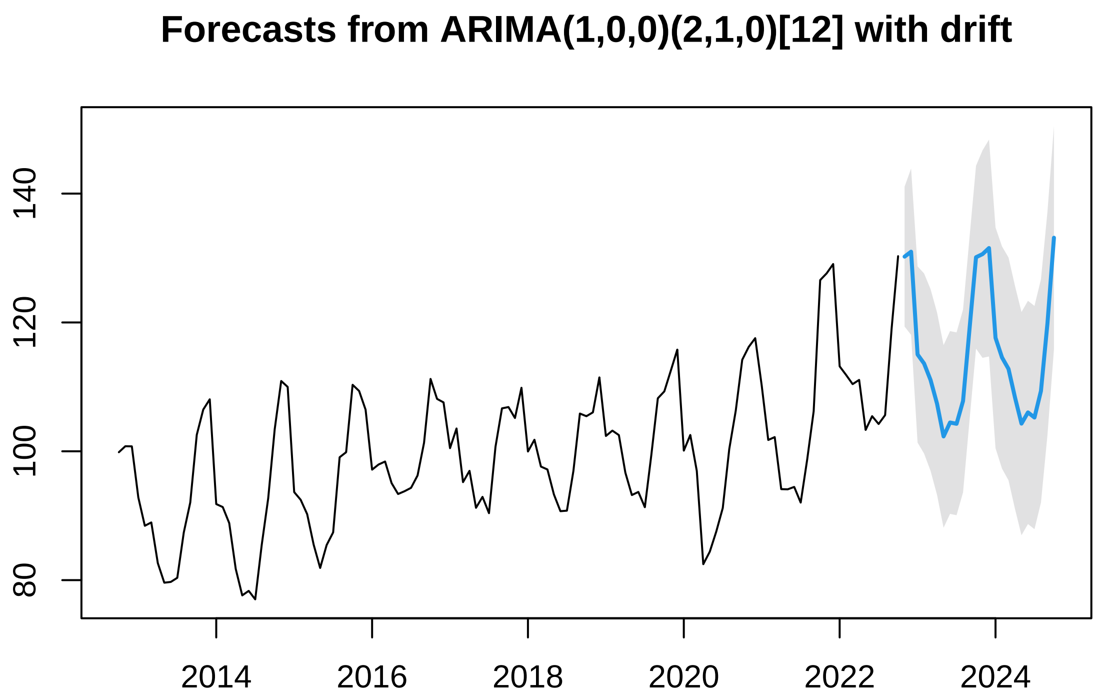
```r
#                     ME         RMSE       MAE        MPE        MAPE       MASE          ACF1
#Training set -0.004444951 3.602064 2.583457 -0.1100037 2.571757 0.5881103 -0.06809078

ARIMAF12 <- plot(forecast(auto_fit,h=12,level=c(99.5)))
```




```
ARIMAF12
```

```
## $mean
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2022
## 2023 115.0259 113.6240 111.0655 107.3725 102.3049 104.4700 104.2670 107.8069
##           Sep      Oct      Nov      Dec
## 2022                   130.2152 130.9781
## 2023 119.0309 130.1239
##
## $lower
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2022
## 2023 101.35264  99.64659  96.96380  93.21964  88.13085  90.28708  90.08048
##            Aug       Sep       Oct       Nov       Dec
## 2022                             119.36102 118.06743
## 2023  93.61892 104.84220 115.93496
##
## $upper
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2022
## 2023 128.6992 127.6015 125.1672 121.5254 116.4790 118.6528 118.4535 121.9950
##           Sep      Oct      Nov      Dec
## 2022                   141.0694 143.8888
## 2023 133.2195 144.3128
```

```r
ARIMAF24 <-plot(forecast(auto_fit,h=24,level=c(99.5)))
```




```r
ARIMAF24
```

```
## $mean
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2022
## 2023 115.0259 113.6240 111.0655 107.3725 102.3049 104.4700 104.2670 107.8069
## 2024 117.6141 114.5620 112.7877 108.3628 104.2936 106.0342 105.2377 109.3693
##           Sep      Oct      Nov      Dec
## 2022                   130.2152 130.9781
## 2023 119.0309 130.1239 130.6058 131.5482
## 2024 119.8675 133.1484
##
## $lower
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2022
## 2023 101.35264  99.64659  96.96380  93.21964  88.13085  90.28708  90.08048
## 2024 100.49160  97.31920  95.49535  91.04988  86.97211  88.70925  87.91127
##            Aug       Sep       Oct       Nov       Dec
## 2022                               119.36102 118.06743
## 2023  93.61892 104.84220 115.93496 114.50585 114.71910
## 2024  92.04224 102.54024 115.82103
##
## $upper
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2022
## 2023 128.6992 127.6015 125.1672 121.5254 116.4790 118.6528 118.4535 121.9950
## 2024 134.7366 131.8047 130.0801 125.6757 121.6150 123.3592 122.5642 126.6963
##           Sep      Oct      Nov      Dec
## 2022                   141.0694 143.8888
## 2023 133.2195 144.3128 146.7058 148.3774
## 2024 137.1948 150.4758
```

```
auto_fit$mse
```

```
## NULL
```

```
#[1] NULL
#RSME is 3.602064 on average forecast values were 3.602064 away from the actual
#MPE is 11% percenatge of error is around 11%


#accuracy summary

accuracy(ets_forecast)
```

```
##                       ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -0.03996037 3.310275 2.498396 -0.1249896 2.482994 0.5687464
##                    ACF1
## Training set 0.07788466
```

```
accuracy(naive_forecast)
```

```
##                     ME     RMSE      MAE        MPE     MAPE    MASE      ACF1
## Training set 0.2536275 6.305638 4.625101 0.02809525 4.635846 1.05288 0.2207691
```

```
accuracy(rwf_forecast)
```

```
##                     ME     RMSE      MAE        MPE     MAPE    MASE      ACF1
## Training set 0.2536275 6.305638 4.625101 0.02809525 4.635846 1.05288 0.2207691
```

```
accuracy(snaive_forecast)
```

```
##                    ME     RMSE      MAE      MPE     MAPE MASE      ACF1
## Training set 2.731406 5.829941 4.392811 2.525765 4.342721    1 0.6254568
```

```
accuracy(SSE_Simple)
```

```
##                      ME    RMSE      MAE        MPE     MAPE     MASE      ACF1
## Training set 0.04457017 6.29552 4.615979 -0.1819999 4.631386 1.050803 0.2185404
```

```r
accuracy(auto_fit)
```

```
##                        ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -0.004444951 3.602064 2.583457 -0.1100037 2.571757 0.5881103
##                     ACF1
## Training set -0.06809078
```

```r
#I would choose ets_forecast as it has the lowest MAPE. I chose MAPE because the units are in Percent,
#when observations are large numbers and also It can be used to compare same or different techniques as


#Best MAPE Forecast

accuracy(ets_forecast)
```

```
##                       ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -0.03996037 3.310275 2.498396 -0.1249896 2.482994 0.5687464
##                    ACF1
## Training set 0.07788466
```

```r
#worst MAPE Forecast

accuracy(naive_forecast)
```

```
##                     ME     RMSE      MAE        MPE     MAPE    MASE      ACF1
## Training set 0.2536275 6.305638 4.625101 0.02809525 4.635846 1.05288 0.2207691
```

```r
#conclusion

#based on my analysis the value of time series will be cyclical as it has been so far in 1 and 2 both y
```