The Analysis of Fuel Prices in the city of Ahmedabad

Name: Juilee Sameer Thakur

Date: 2023-05-07

Name of the course: MA 641 - Time Series Analysis

Introduction

Increasing fuel prices have been a constant concern in the city of Ahmedabad, India. In this project, I analysed the daily fuel data using time series analysis methods to better understand the data and its aspects and thus help predict the future prices of fuel.

Importing the dataset

```
price_data <- read.csv("~/Stevens Institute of Technlogy/Spring 2023/MA 641/Project/Non-seasonal/Ahmedabad.csv")
head(price_data)
```

```
##        city       date  rate
## 1 Ahmedabad 2019-04-22 70.32
## 2 Ahmedabad 2019-04-22 70.32
## 3 Ahmedabad 2019-04-22 70.32
## 4 Ahmedabad 2019-04-22 70.32
## 5 Ahmedabad 2019-04-22 70.32
## 6 Ahmedabad 2019-04-22 70.32
```

The data-set contains information of fuel rate of last 4 years in Ahmedabad, India. That data set contains 1024 observations and has 2 major attributes.

- `date` - This column contains the date at which the rate was recorded.
- `rate` - This column contains the rate of the fuel on the corresponding date.

Summarizing the data-set

Below, I summarized the data to see the values of each attribute of the data-set and check for any missing values in the data-set.

```
summary(price_data)
```

```
##      city               date                rate       
##  Length:1024        Length:1024        Min.   : 67.15  
##  Class :character   Class :character   1st Qu.: 70.39  
##  Mode  :character   Mode  :character   Median : 70.81  
##                                        Mean   : 79.45  
##                                        3rd Qu.: 89.00  
##                                        Max.   :106.63  
```

The average fuel rate is Rs. 79.45 with maximum rate being Rs 106.63. I found out that the data-set has no missing values. Hence, we can now proceed with our analysis. I am keen on knowing the future prices of fuel hence, the desired attribute of study is ‘rate’.

Checking for seasonality

To check if the pricedata is seasonal or not, I run the combined test from the “seastests” library. The combined test can be used individually or using the *isSeasonal()* function. So, I ran both tests.

```
library(seastests)
```

```
## Warning: package 'seastests' was built under R version 4.2.3
```

```
# Combined test
combined_test(price_data$rate, freq = 12)
```

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

```
## Test used:  WO
## 
## Test statistic:  0
## P-value:  1 1 0.9374157
```

```
# Seasonal or not Test
isSeasonal(price_data$rate, test = "combined", freq = 12)
```

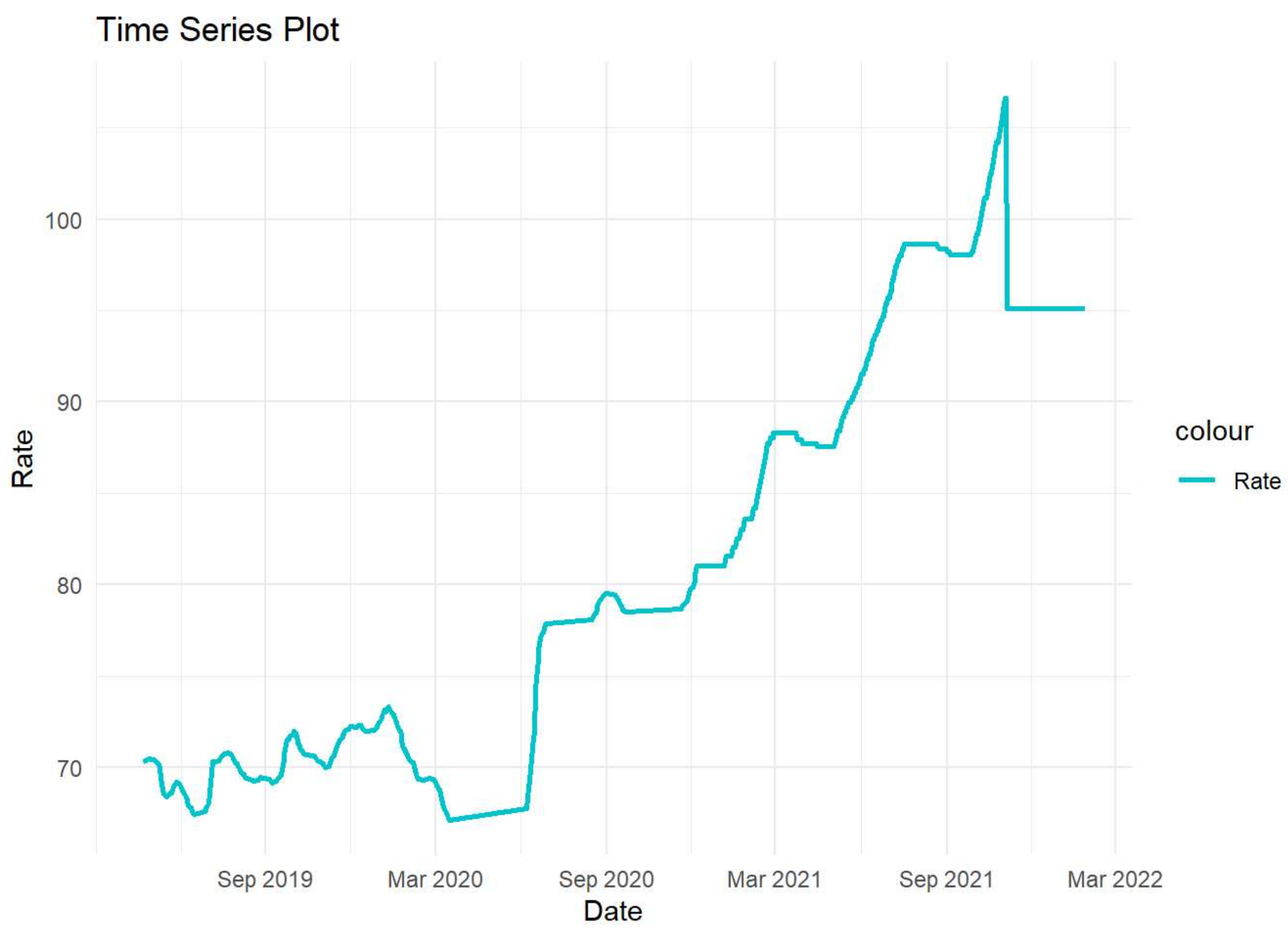
```
## [1] FALSE
```

Since overall p-value of combined test is 0.937 which is considerably close to 1 than 0, we say the data is non-seasonal. Just to get a confirmation, I also applied the isSeasonal test which returned false. Thus, concluded the data-set is Non-seasonal.

Exploratory Data Analysis

I performed some exploratory data analysis to check out the time series plot and see the trend in data over time.

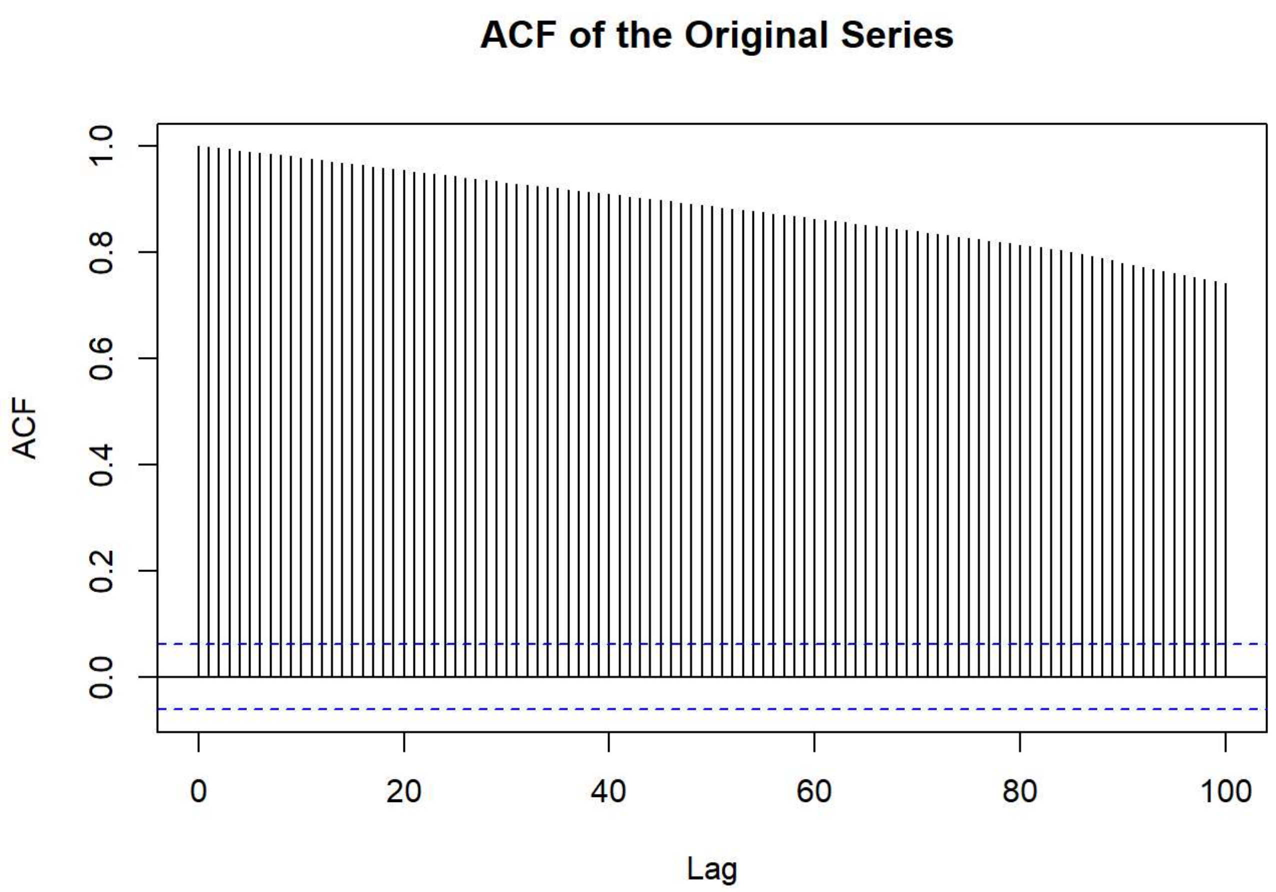
```
options(warn = -1)

library(ggplot2)

price_data$date <- as.Date(price_data$date)

ggplot(price_data, aes(x = date, group = 1)) +
  geom_line(aes(y = rate, color = "Rate"), linewidth = 1) +
  labs(x = "Date", y = "Rate", title = "Time Series Plot") +
  scale_x_date(date_breaks = "6 month", date_labels = "%b %Y") +
  scale_color_manual(values = c("Rate" = "turquoise3")) +
  theme_minimal()
```



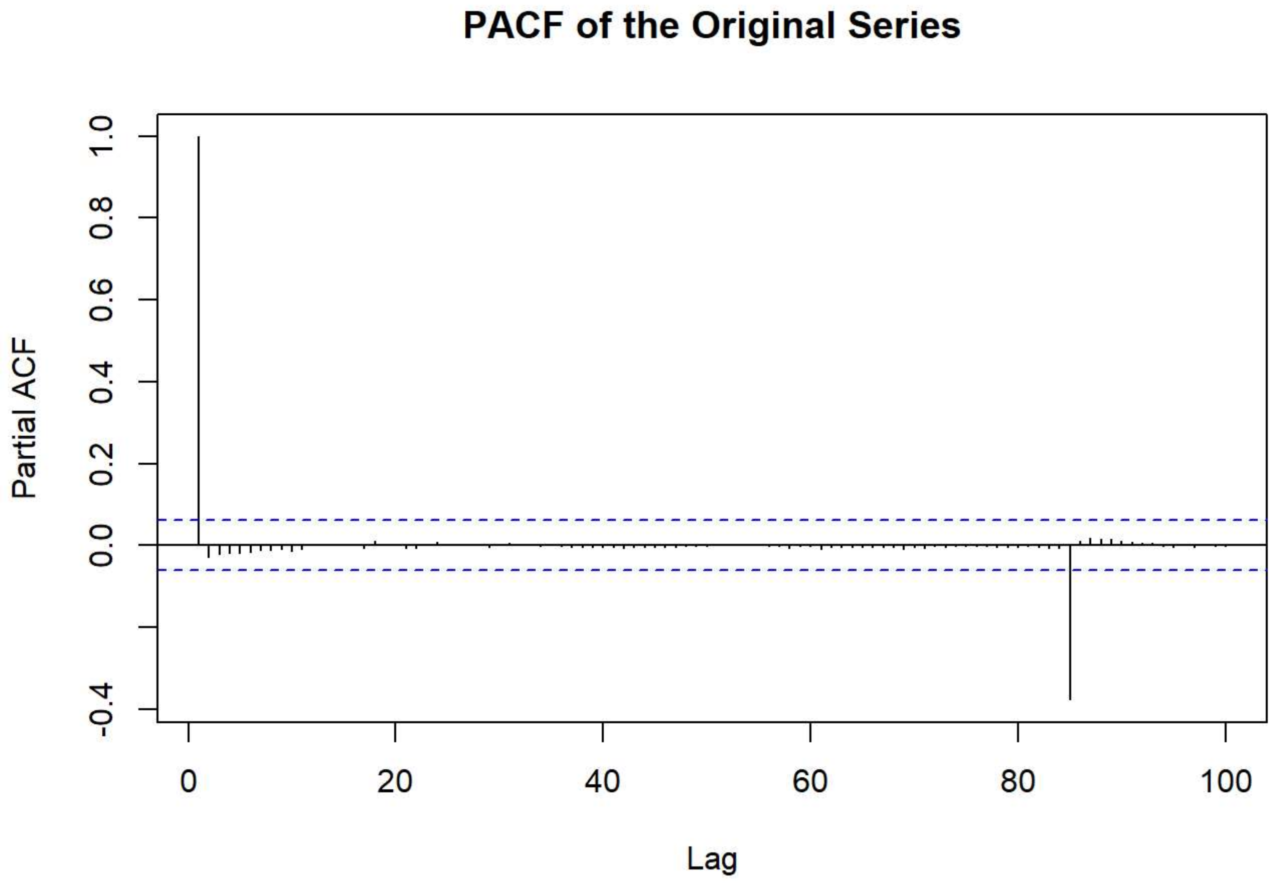
The data seems to follow an upward trend which makes sense as it is the price data for fuels. We now check for the ACF and PACF of the series.

ACF and PACF of the series

```
acf(price_data$rate, lag.max = 100, main = " ACF of the Original Series")
```



```
pacf(price_data$rate, lag.max = 100, main = "PACF of the Original Series")
```



The data shows non-seasonal characteristics but in order to perform further analysis we check for the stationarity of the series.

Checking the stationarity of the series

In order to check the stationarity, I performed the Augmented Dicky-Fuller test from the 'tseries' library.

Following the hypothesis for ADF test:

H_0 : The series is non-stationary

vs

H_1 : The series is stationary

```
options(warn = -1)

library(tseries)
adf.test(price_data$rate)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  price_data$rate
## Dickey-Fuller = -2.011, Lag order = 10, p-value = 0.5737
## alternative hypothesis: stationary
```

Since p-value is greater than 0.05, we accept H_0 and say that the data is not stationary. In order to make the data stationary, I initially took the difference of the data but to make analysis better I took the difference of the log of the series and made it stationary.

```
library(tseries)
diff_price = diff(log(price_data$rate))
adf.test(diff_price)
```

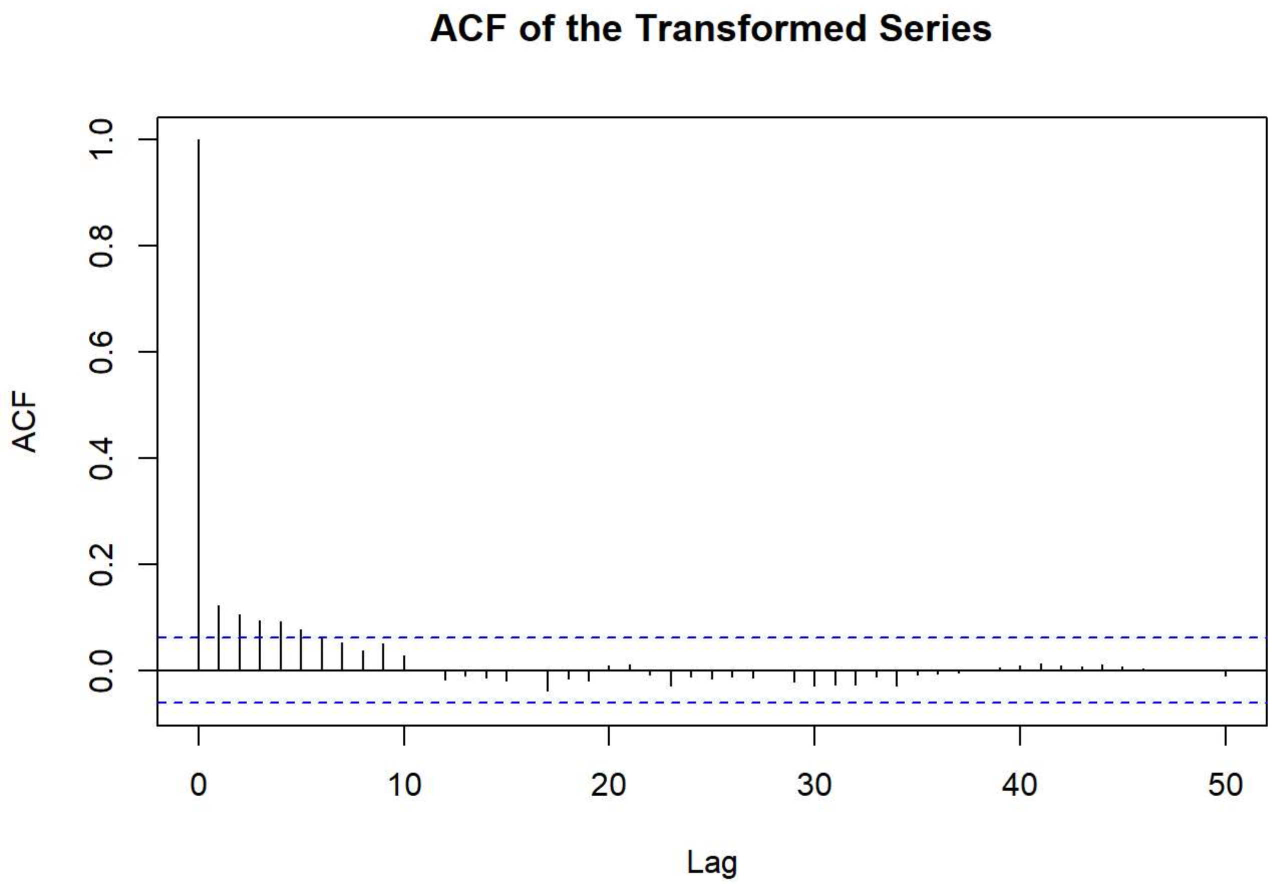
```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff_price
## Dickey-Fuller = -8.1548, Lag order = 10, p-value = 0.01
## alternative hypothesis: stationary
```

After transforming the data, I got the Augmented Dicky-Fuller test results which gave the p-value as 0.01 which in reality was much smaller. Hence, we can say that taking the difference of the logarithm of the data made it stationary. We shall now use this transformed series to perform further analysis.

Finding the model

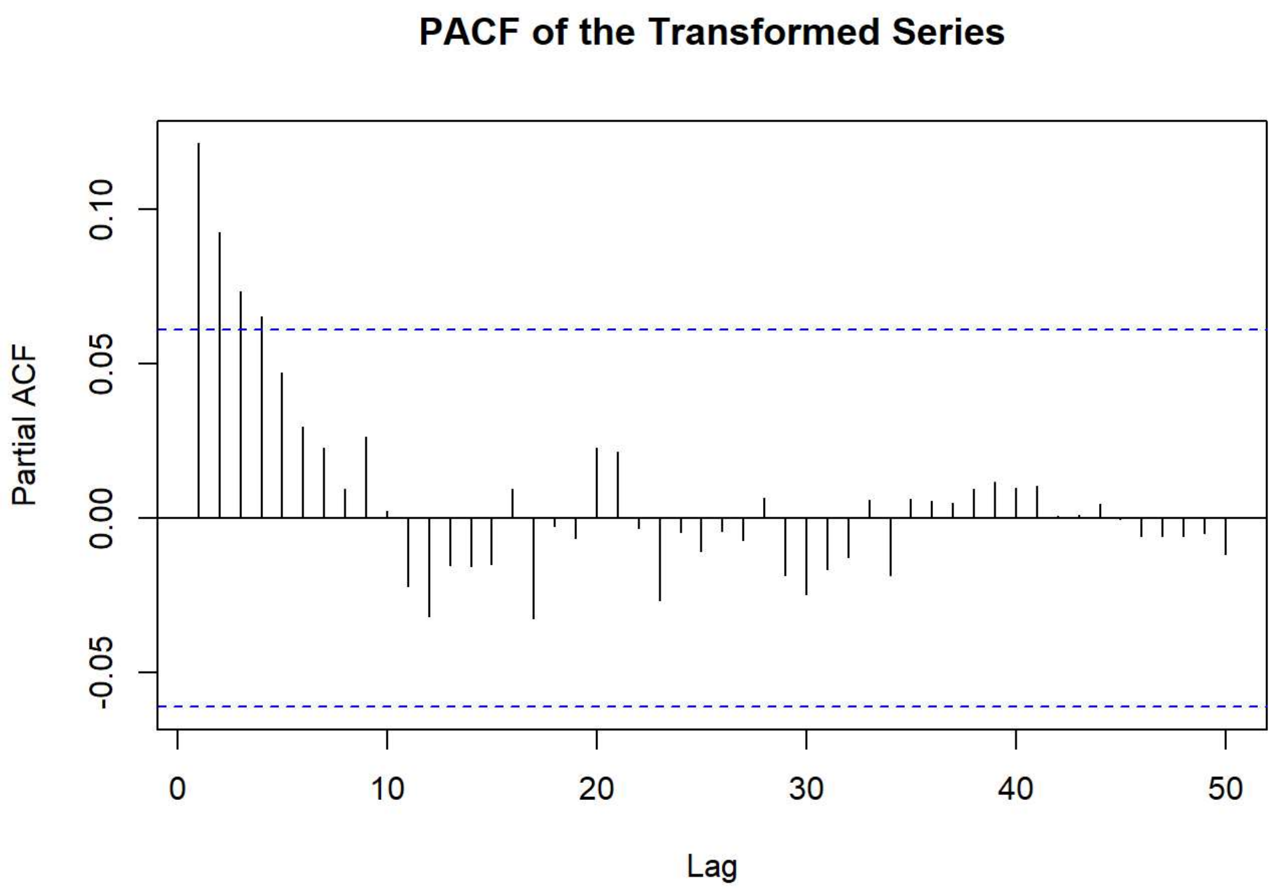
Plotting ACF and PACF of the transformed series.

```
acf(diff_price, lag.max = 50, main = "ACF of the Transformed Series")
```



We can observe 5 significant lags here.

```
pacf(diff_price, lag.max = 50, main = "PACF of the Transformed Series")
```



We can observe 4 significant lags.

Model can be suggested as ARIMA(5,1,4) as we did one differencing so d = 1.

```
library(TSA)
```

```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast
```

```
##
## Attaching package: 'TSA'
```

```
## The following objects are masked from 'package:stats':
##
##     acf, arima
```

```
## The following object is masked from 'package:utils':
##
##     tar
```

```
eacf(diff_price)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x x o o o o o o  o  o  o
## 1 x o o o o o o o o o o  o  o  o
## 2 x x o o o o o o o o o  o  o  o
## 3 x x x o o o o o o o o  o  o  o
## 4 x x x x o o o o o o o  o  o  o
## 5 x x o o x o o o o o o  o  o  o
## 6 x x x o x x o o o o o  o  o  o
## 7 x x x x o x x o o o o  o  o  o
```

We can deduce from the EACF that the model can be ARMA(1,1,1), ARIMA(1,1,2) or ARIMA(2,1,2).

Thus models suggested are:

- Model 1: ARIMA (5,1,4)
- Model 2: ARIMA (1,1,1)
- Model 3: ARIMA(1,1,2)
- Model 4: ARIMA (2,1,2)

Parameter Estimation

Now, for each of the above models, we check the AIC and BIC values.

```
library(forecast)
model_1 <- Arima(diff_price, order = c(5,1,4))
model_2 <- Arima(diff_price, order = c(1,1,1))
model_3 <- Arima(diff_price, order = c(1,1,2))
model_4 <- Arima(diff_price, order = c(2,1,2))
#summary(model_1)
```

```
cat("For Model 1:\n")
```

```
## For Model 1:
```

```
print(paste("AIC is", model_1$bic))
```

```
## [1] "AIC is -8250.82488710856"
```

```
print(paste("BIC is", model_1$bic))
```

```
## [1] "BIC is -8250.82488710856"
```

```
cat("\nFor Model 2:\n")
```

```
##
## For Model 2:
```

```
print(paste("AIC is", model_2$aic))
```

```
## [1] "AIC is -8292.82927131186"
```

```
print(paste("BIC is", model_2$bic))
```

```
## [1] "BIC is -8278.04072099957"
```

```
cat("\nFor Model 3:\n")
```

```
##
## For Model 3:
```

```
print(paste("AIC is", model_3$aic))
```

```
## [1] "AIC is -8311.16130092161"
```

```
print(paste("BIC is", model_3$bic))
```

```
## [1] "BIC is -8291.44323383855"
```

```
cat("\nFor Model 4:\n")
```

```
##
## For Model 4:
```

```
print(paste("AIC is", model_4$aic))
```

```
## [1] "AIC is -8309.35938509031"
```

```
print(paste("BIC is", model_4$bic))
```

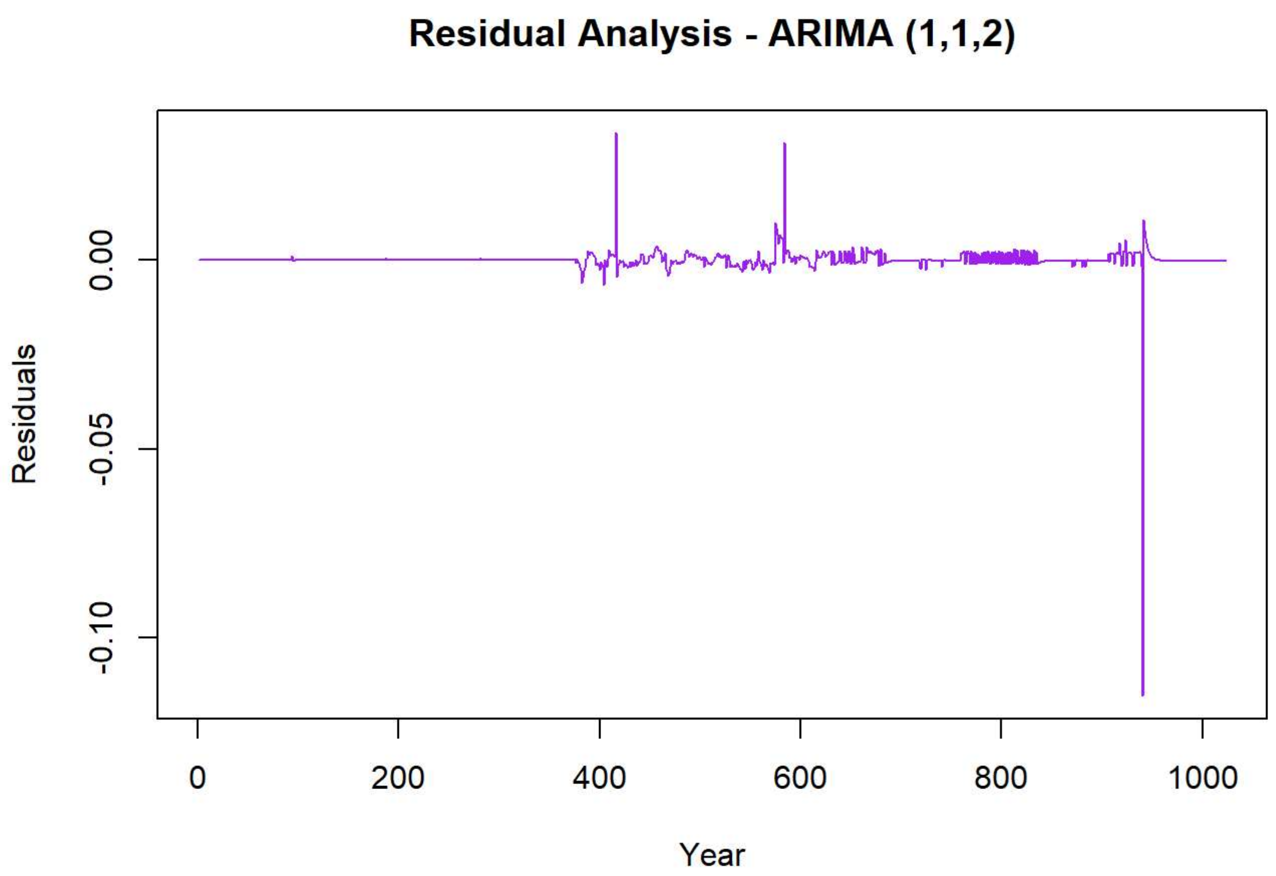
```
## [1] "BIC is -8284.71180123649"
```

The lowest AIC and BIC values are for Model 3 i.e. ARIMA (1,1,2).

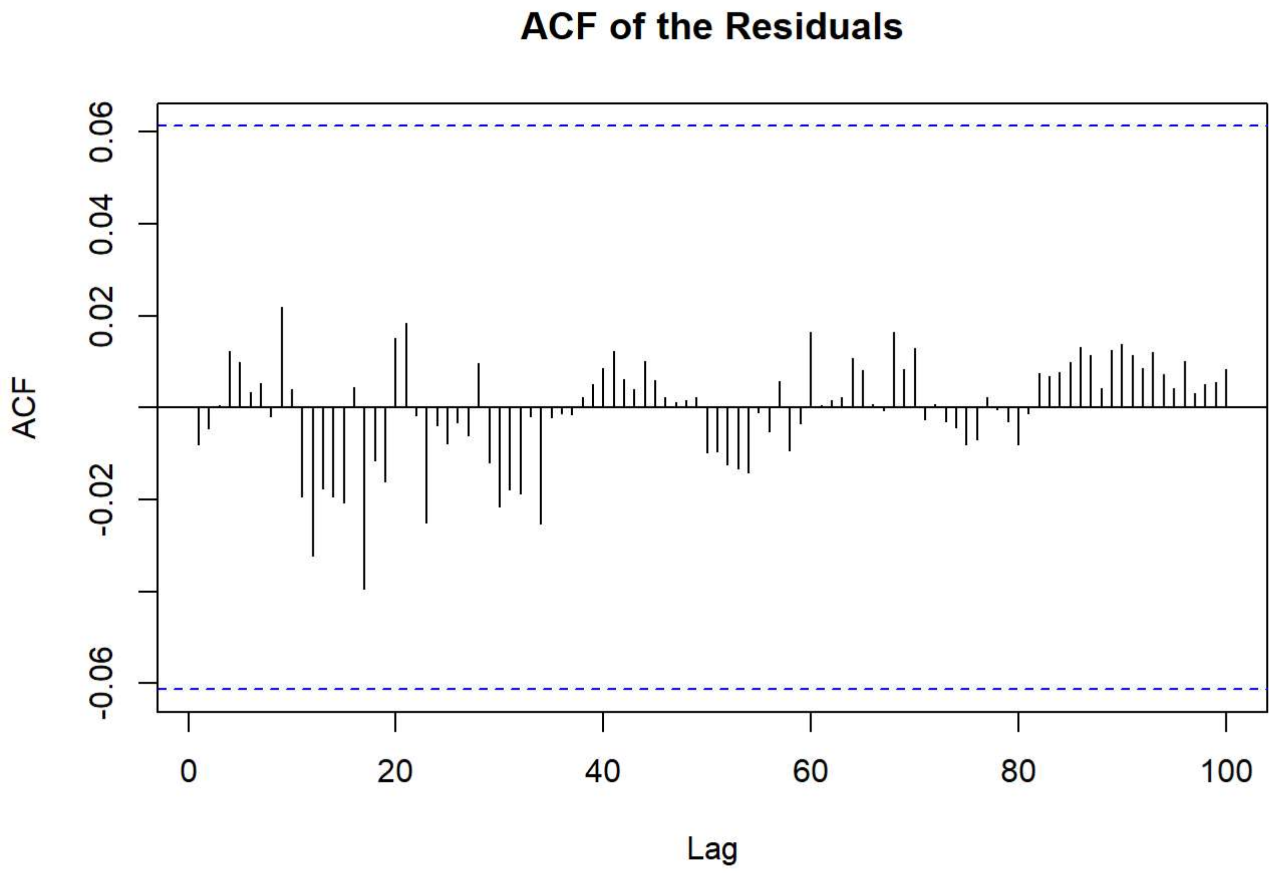
Residual Analysis

Using the above obtained model, I performed the residual analysis and plotted the same along with its ACF and PACF.

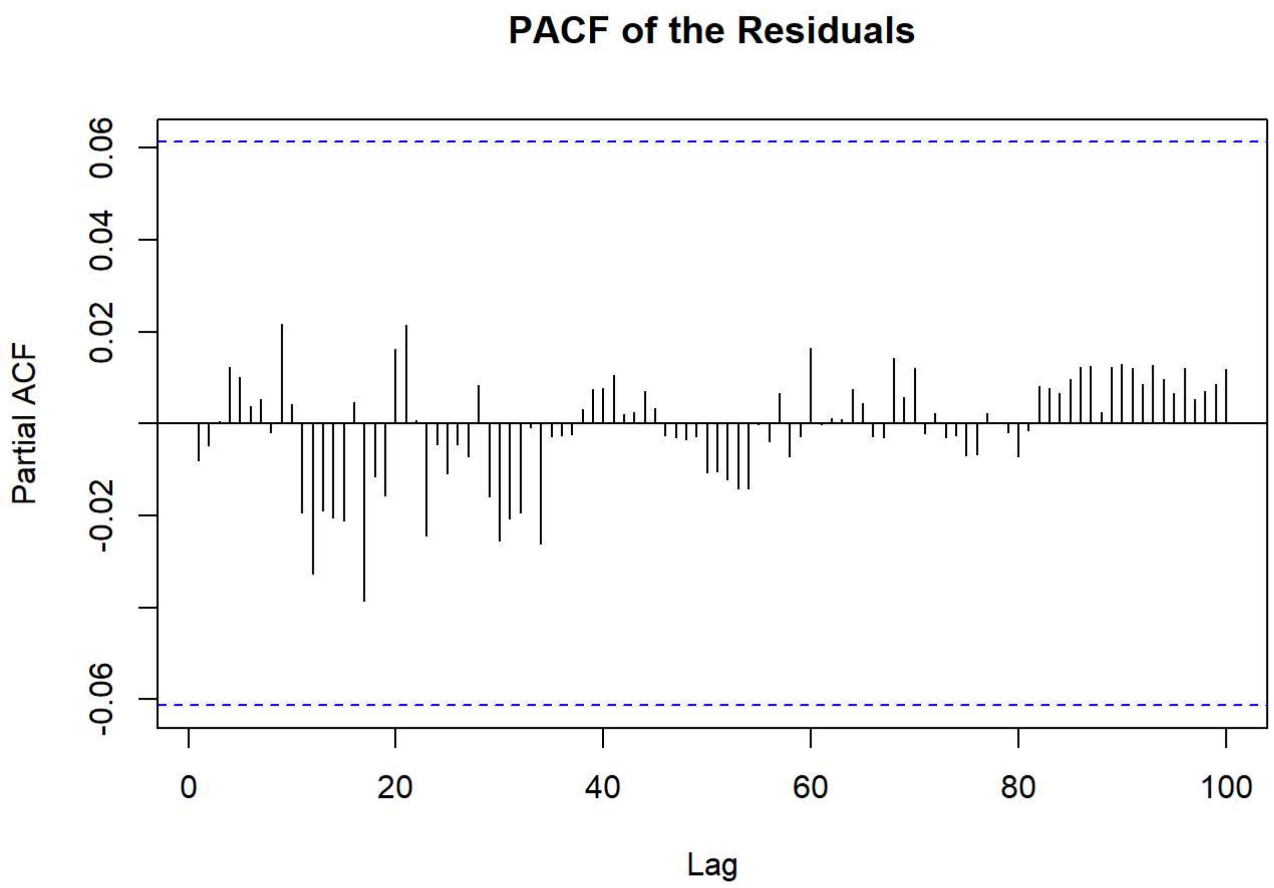
```
ts.plot(model_3$residuals,lwd=1,col="purple",xlab = "Year", ylab = "Residuals",main='Residual Analysis - ARIMA (1,1,2)')
```




```
#plot(residu, xlab = "Year", ylab = "Residuals")

acf(model_3$residuals, lag.max = 100, main = "ACF of the Residuals")
```



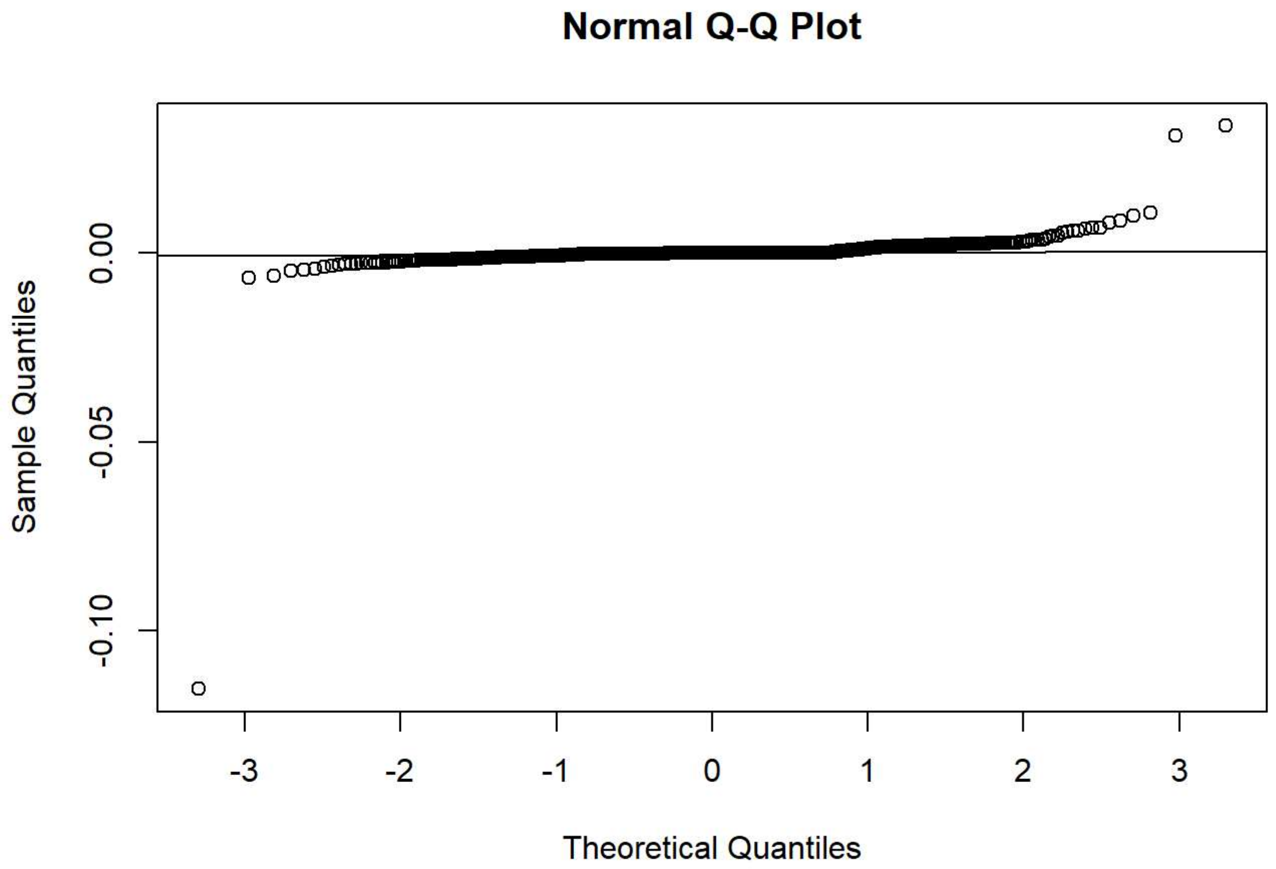

```
pacf(model_3$residuals, lag.max = 100, main = "PACF of the Residuals")
```



The ACF and the PACF plots portray white noise hence indicating ideal for forecasting.

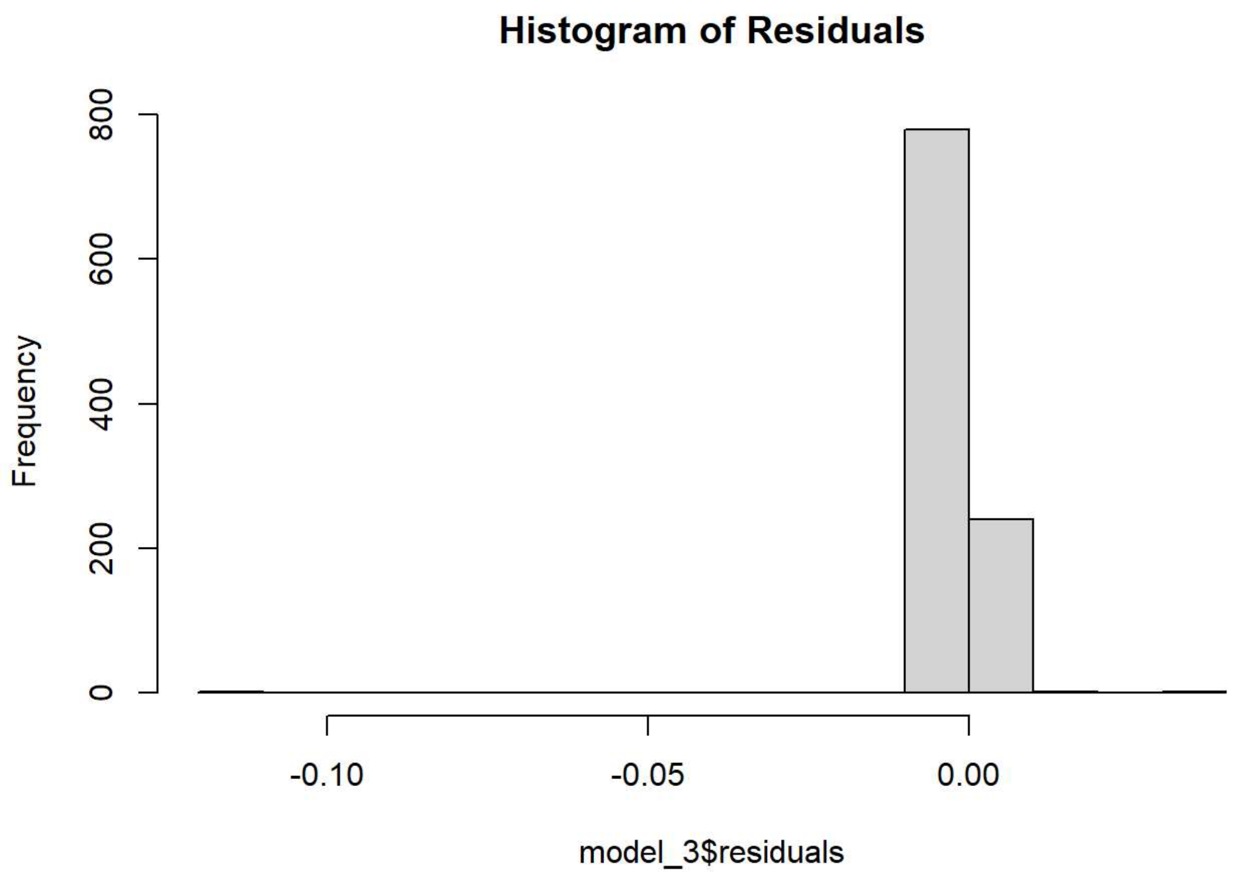
Checking for normality

```
qqnorm(model_3$residuals)
qqline(model_3$residuals)
```



The q-q plot shows partial normality as towards the either ends, the plot seems to distort around the q-q line.

```
hist(model_3$residuals, main = "Histogram of Residuals")
```




```
shapiro.test(model_3$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model_3$residuals
## W = 0.18304, p-value < 2.2e-16
```

The p-value is significantly less than 0.05. Hence, the conclusion is the distribution is non-normal.

```
Box.test(model_3$residuals, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  model_3$residuals
## X-squared = 0.066823, df = 1, p-value = 0.796
```

Thus, we fail to reject the null hypothesis of the test and conclude that the data values are independent.

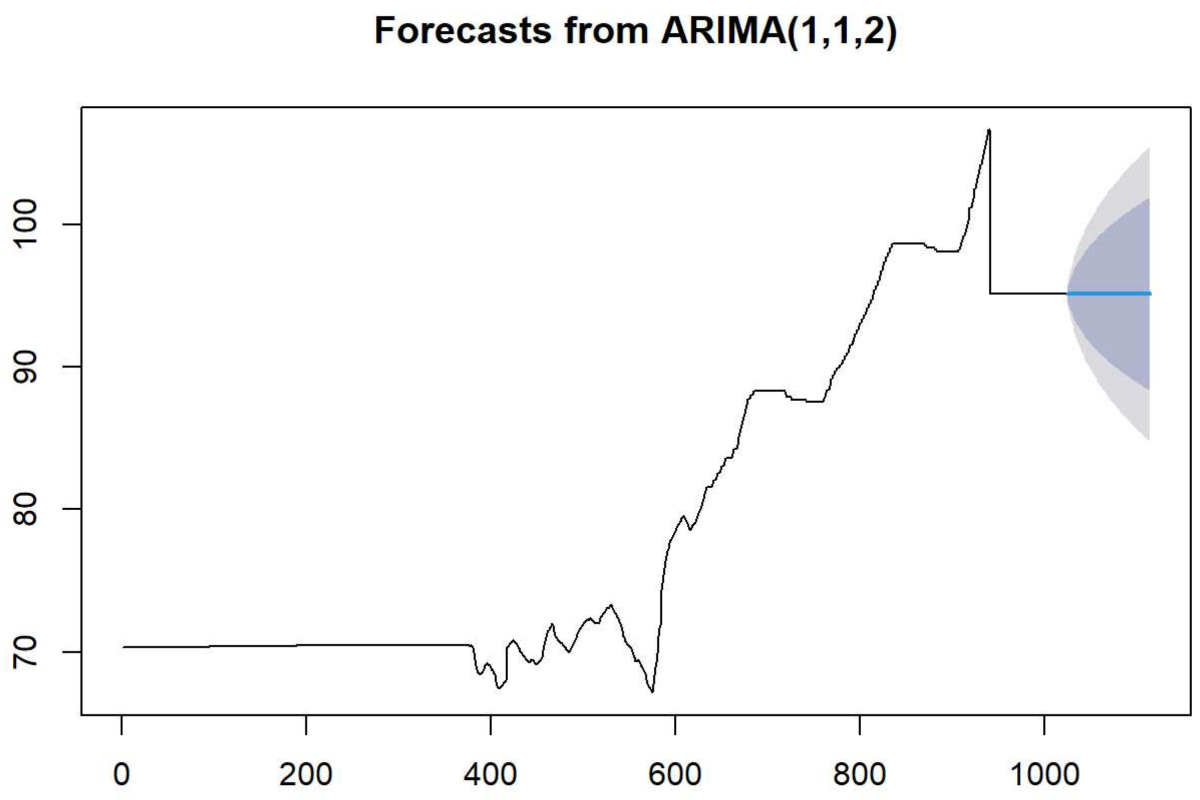
Forecasting

Using the above fitted model ARIMA(1,1,2), we plot the forecasted values on original plot.

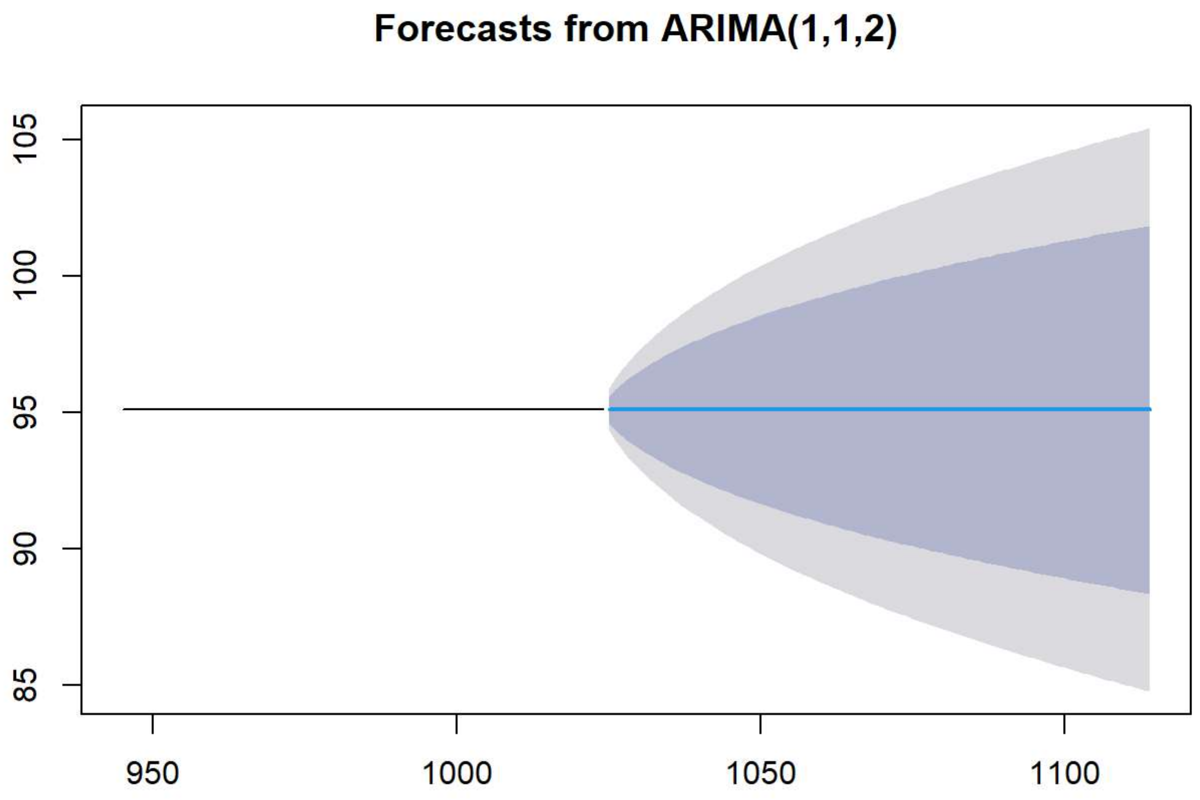
```
library(forecast)
d <- Arima(price_data$rate, order = c(1,1,2))
forecasted_series <- forecast(d, h = 90)

cat("The confidence interval is", forecasted_series$level)
```

```
## The confidence interval is 80 95
```

```
plot(forecasted_series)
```




```
plot(forecasted_series, PI = TRUE, include = 80)
```




```
cat("\nFollowing is the summary of the forecasted series:")
```

```
##
## Following is the summary of the forecasted series:
```

```
summary(forecasted_series)
```

```
##
## Forecast method: ARIMA(1,1,2)
##
## Model Information:
## Series: price_data$rate
## ARIMA(1,1,2)
##
## Coefficients:
##          ar1     ma1      ma2
##       0.8366  -0.765  -0.0006
## s.e.  0.0824   0.088   0.0348
##
## sigma^2 = 0.1559:  log likelihood = -499.43
## AIC=1006.87   AICc=1006.91   BIC=1026.59
##
## Error measures:
##                      ME      RMSE        MAE        MPE       MAPE      MASE
## Training set 0.01694996 0.3940701 0.07540321 0.02022817 0.09151361 0.9435778
##                     ACF1
## Training set -0.00176122
##
## Forecasts:
##      Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
## 1025          95.11 94.60399  95.61601 94.33612  95.88388
## 1026          95.11 94.36835  95.85165 93.97574  96.24426
## 1027          95.11 94.17327  96.04673 93.67740  96.54260
## 1028          95.11 93.99906  96.22094 93.41096  96.80904
## 1029          95.11 93.83861  96.38139 93.16558  97.05442
## 1030          95.11 93.68850  96.53150 92.93600  97.28400
## 1031          95.11 93.54673  96.67327 92.71919  97.50081
## 1032          95.11 93.41204  96.80796 92.51320  97.70680
## 1033          95.11 93.28351  96.93649 92.31663  97.90337
## 1034          95.11 93.16047  97.05953 92.12844  98.09156
## 1035          95.11 93.04235  97.17765 91.94780  98.27220
## 1036          95.11 92.92872  97.29128 91.77402  98.44598
## 1037          95.11 92.81918  97.40082 91.60650  98.61350
## 1038          95.11 92.71342  97.50658 91.44475  98.77525
## 1039          95.11 92.61114  97.60886 91.28832  98.93168
## 1040          95.11 92.51208  97.70792 91.13682  99.08318
## 1041          95.11 92.41600  97.80400 90.98989  99.23011
## 1042          95.11 92.32271  97.89729 90.84721  99.37279
## 1043          95.11 92.23201  97.98799 90.70850  99.51150
## 1044          95.11 92.14374  98.07626 90.57349  99.64651
## 1045          95.11 92.05773  98.16227 90.44195  99.77805
## 1046          95.11 91.97385  98.24615 90.31367  99.90633
## 1047          95.11 91.89196  98.32804 90.18844 100.03156
## 1048          95.11 91.81196  98.40804 90.06608 100.15392
## 1049          95.11 91.73372  98.48628 89.94642 100.27358
## 1050          95.11 91.65715  98.56285 89.82932 100.39068
## 1051          95.11 91.58215  98.63785 89.71463 100.50537
## 1052          95.11 91.50865  98.71135 89.60222 100.61778
## 1053          95.11 91.43657  98.78343 89.49197 100.72803
## 1054          95.11 91.36582  98.85418 89.38377 100.83623
## 1055          95.11 91.29635  98.92365 89.27752 100.94248
## 1056          95.11 91.22808  98.99192 89.17312 101.04688
## 1057          95.11 91.16098  99.05902 89.07049 101.14951
## 1058          95.11 91.09497  99.12503 88.96954 101.25046
## 1059          95.11 91.03001  99.18999 88.87019 101.34981
## 1060          95.11 90.96605  99.25395 88.77238 101.44762
## 1061          95.11 90.90306  99.31694 88.67604 101.54396
## 1062          95.11 90.84099  99.37901 88.58111 101.63889
## 1063          95.11 90.77979  99.44021 88.48752 101.73248
## 1064          95.11 90.71945  99.50055 88.39523 101.82477
## 1065          95.11 90.65991  99.56009 88.30418 101.91582
## 1066          95.11 90.60116  99.61884 88.21432 102.00568
## 1067          95.11 90.54316  99.67684 88.12562 102.09438
## 1068          95.11 90.48588  99.73412 88.03802 102.18198
## 1069          95.11 90.42930  99.79070 87.95149 102.26851
## 1070          95.11 90.37340  99.84660 87.86599 102.35401
## 1071          95.11 90.31814  99.90186 87.78149 102.43851
## 1072          95.11 90.26352  99.95648 87.69795 102.52205
## 1073          95.11 90.20950 100.01050 87.61533 102.60467
## 1074          95.11 90.15607 100.06393 87.53362 102.68638
## 1075          95.11 90.10321 100.11679 87.45277 102.76723
## 1076          95.11 90.05090 100.16910 87.37277 102.84723
## 1077          95.11 89.99913 100.22087 87.29359 102.92641
## 1078          95.11 89.94787 100.27213 87.21520 103.00480
## 1079          95.11 89.89712 100.32288 87.13758 103.08242
## 1080          95.11 89.84685 100.37315 87.06071 103.15929
## 1081          95.11 89.79707 100.42293 86.98457 103.23543
```

```
## 1082          95.11 89.74774 100.47226 86.90913 103.31087
## 1083          95.11 89.69886 100.52114 86.83438 103.38562
## 1084          95.11 89.65043 100.56957 86.76030 103.45970
## 1085          95.11 89.60241 100.61759 86.68687 103.53313
## 1086          95.11 89.55481 100.66519 86.61408 103.60592
## 1087          95.11 89.50762 100.71238 86.54190 103.67810
## 1088          95.11 89.46082 100.75918 86.47033 103.74967
## 1089          95.11 89.41441 100.80559 86.39934 103.82066
## 1090          95.11 89.36837 100.85163 86.32893 103.89107
## 1091          95.11 89.32269 100.89731 86.25908 103.96092
## 1092          95.11 89.27738 100.94262 86.18977 104.03023
## 1093          95.11 89.23241 100.98759 86.12100 104.09900
## 1094          95.11 89.18778 101.03222 86.05275 104.16725
## 1095          95.11 89.14349 101.07651 85.98501 104.23499
## 1096          95.11 89.09953 101.12047 85.91777 104.30223
## 1097          95.11 89.05588 101.16412 85.85102 104.36898
## 1098          95.11 89.01255 101.20745 85.78475 104.43525
## 1099          95.11 88.96952 101.25048 85.71894 104.50106
## 1100          95.11 88.92679 101.29321 85.65360 104.56640
## 1101          95.11 88.88435 101.33565 85.58870 104.63130
## 1102          95.11 88.84221 101.37779 85.52424 104.69576
## 1103          95.11 88.80034 101.41966 85.46021 104.75979
## 1104          95.11 88.75875 101.46125 85.39660 104.82340
## 1105          95.11 88.71743 101.50257 85.33341 104.88659
## 1106          95.11 88.67637 101.54363 85.27062 104.94938
## 1107          95.11 88.63558 101.58442 85.20823 105.01177
## 1108          95.11 88.59504 101.62496 85.14623 105.07377
## 1109          95.11 88.55475 101.66525 85.08461 105.13539
## 1110          95.11 88.51471 101.70529 85.02338 105.19662
## 1111          95.11 88.47491 101.74509 84.96251 105.25749
## 1112          95.11 88.43535 101.78465 84.90200 105.31800
## 1113          95.11 88.39602 101.82398 84.84185 105.37815
## 1114          95.11 88.35692 101.86308 84.78205 105.43795
```

The plot above shows the expected values for rate of fuel for next 90 observations where the rate of fuel prices seem to be constant for the next observations since the prices have been showing no variation towards the end data.