Applying ARIMA-GARCH models for time series analysis on Seasonal and Non-Seasonal datasets

Souvik Kumar Misra

Department of Mathematical Sciences, Stevens Institute of Technology

Supervisor: Dr. Hadi Safari Katesari

Abstract

Time series analysis is a methodology for examining data to identify patterns and forecast future outcomes. This project will explore time series analysis on two types of data: seasonal and non-seasonal. The aim is to provide a procedure to analyze and model time series data using the R programming language. The first part of the project focuses on analyzing and forecasting Hourly Traffic data. This dataset contains the number of vehicles recorded every hour over a 20-day period. Techniques such as autoregressive integrated moving average (ARIMA), seasonal ARIMA (SARIMA), autoregressive moving average (ARMA), moving average (MA), and autoregression (AR) will be applied to model this time series data. The second part addresses the time series of Gas Prices, with the goal of analyzing and forecasting the weekly price of gas in the United States. The core of this project is to provide a guide on using ARIMA and ARCH-GARCH models, and to evaluate the combined model's performance in time series modeling and forecasting. Overall, this project presents a comprehensive approach to time series analysis, covering both seasonal and non-seasonal data, and applying various modeling techniques to gain insights and make predictions.

Seasonal Dataset – Hourly Traffic Data

Introduction:

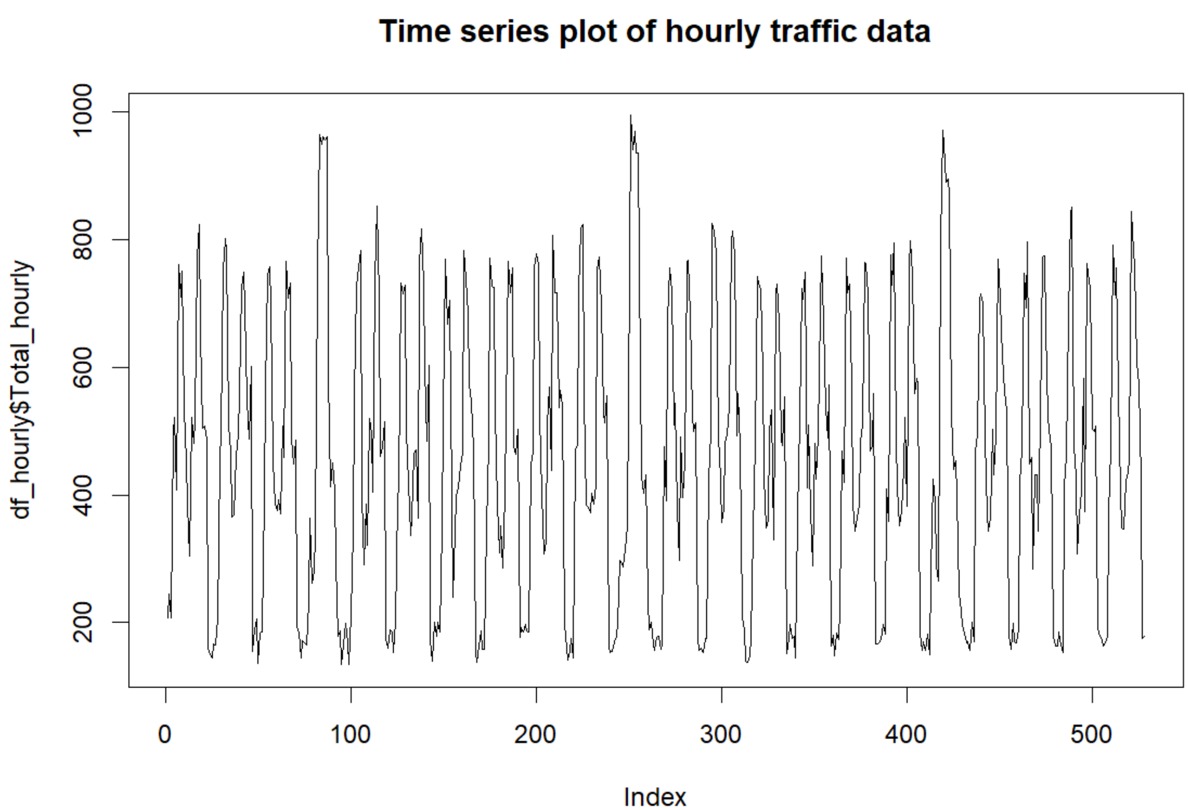
The dataset that we have used consists of hourly traffic data for 10 days. The data was captured by IOT device which captured the vehicle The dataset is stored in a CSV file and includes additional columns such as time in hours, date, days of the week, and counts for each vehicle type (CarCount, BikeCount, BusCount, TruckCount). The "Total" column represents the total count of all vehicle types detected within a 1-hour duration. There are a total of 528 observations consisting of 6 variables. The dataset was already cleaning without any missing values. For our time series analysis, we will be considering the Total_hourly column.

```
> head(df_hourly)
# A tibble: 6 × 6
  hour                CarCount_hourly BikeCount_hourly BusCount_hourly TruckCount_hourly Total_hourly
  <chr>                         <int>            <int>           <int>             <int>        <int>
1 2024-01-10 00:00:00             177                0              12                18          207
2 2024-01-10 01:00:00             180               10              25                29          244
3 2024-01-10 02:00:00             175                0              15                18          208
4 2024-01-10 03:00:00             357               38              98                 5          498
5 2024-01-10 04:00:00             337               90              44                51          522
6 2024-01-10 05:00:00             274               74              18                42          408
```

```
> summary(df_hourly)
     hour            CarCount_hourly BikeCount_hourly BusCount_hourly  TruckCount_hourly  Total_hourly
 Length:528         Min.   : 44.0   Min.   :  0.00   Min.   :  0.00   Min.   :  1.0     Min.   :134.0
 Class :character   1st Qu.:118.8   1st Qu.: 16.75   1st Qu.:  9.00   1st Qu.: 31.0     1st Qu.:209.5
 Mode  :character   Median :265.5   Median : 59.00   Median : 55.00   Median : 63.0     Median :453.5
                    Mean   :273.6   Mean   : 59.91   Mean   : 60.57   Mean   : 61.1     Mean   :455.2
                    3rd Qu.:415.0   3rd Qu.: 85.00   3rd Qu.:105.00   3rd Qu.: 91.0     3rd Qu.:650.8
                    Max.   :677.0   Max.   :251.00   Max.   :195.00   Max.   :149.0     Max.   :995.0
```

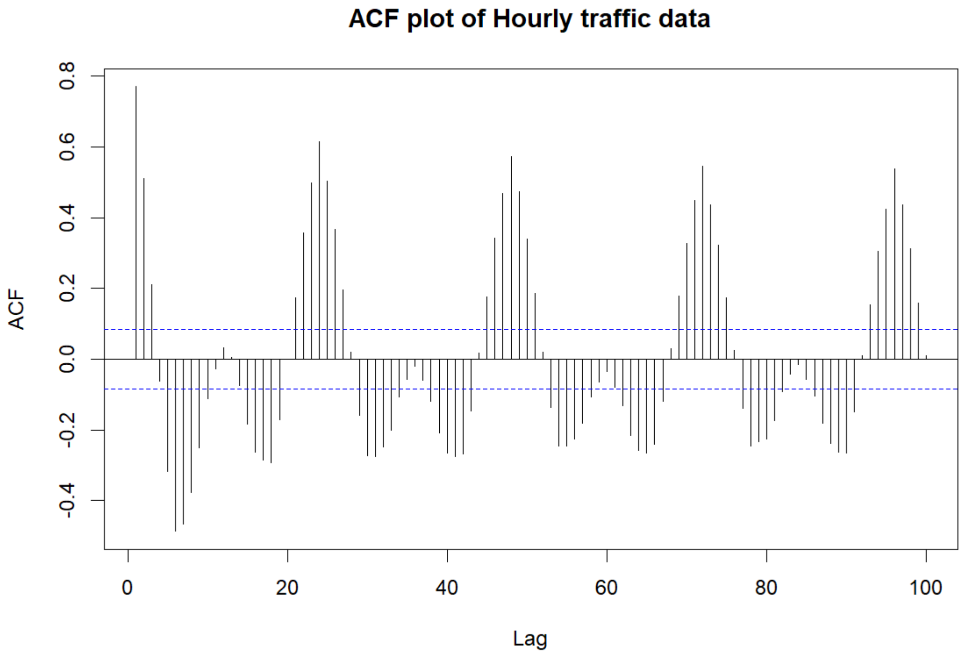
Analysis of Time Series:

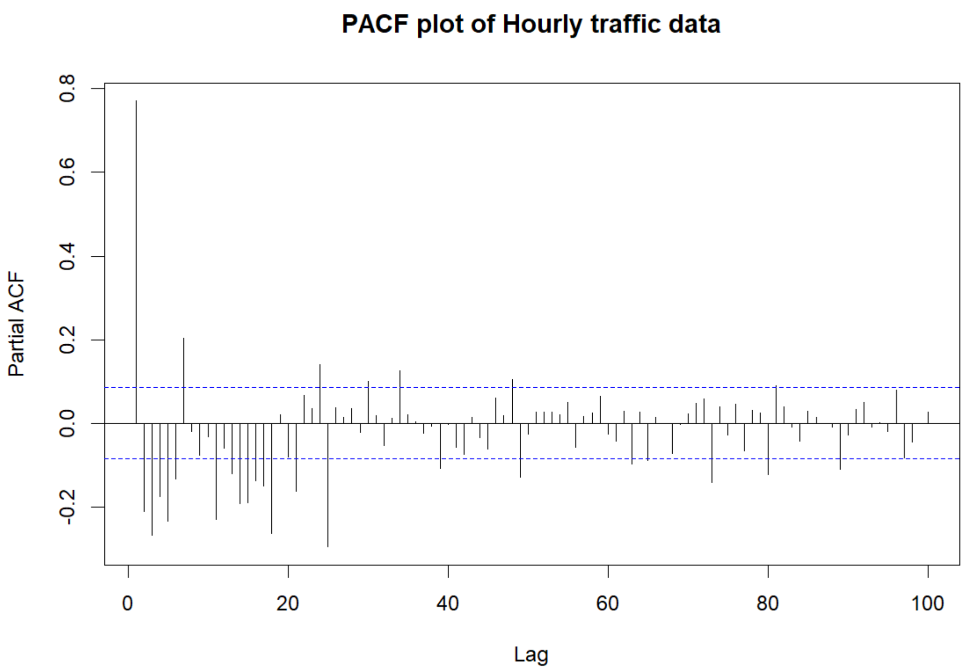
For a better understanding of the time series data, we will plot the data:



Analyzing the plot, we can clearly see the seasonality of dataset. The seasonality is getting repeated every day as we know the traffic increases and decreases continuously within a day. Also, we see a regular 3 spike in no of traffic, that is an indication that for a particular day the traffic is high.

So, our next step would be to calculate the ACF and PACF plots of the dataset, which will help us understand the strength between the variables over time





From the sample ACF and PACF plots, we can clearly see the seasonality of the data. Also, from the ACF we can se the seasonality is being repeated every 24th lag.

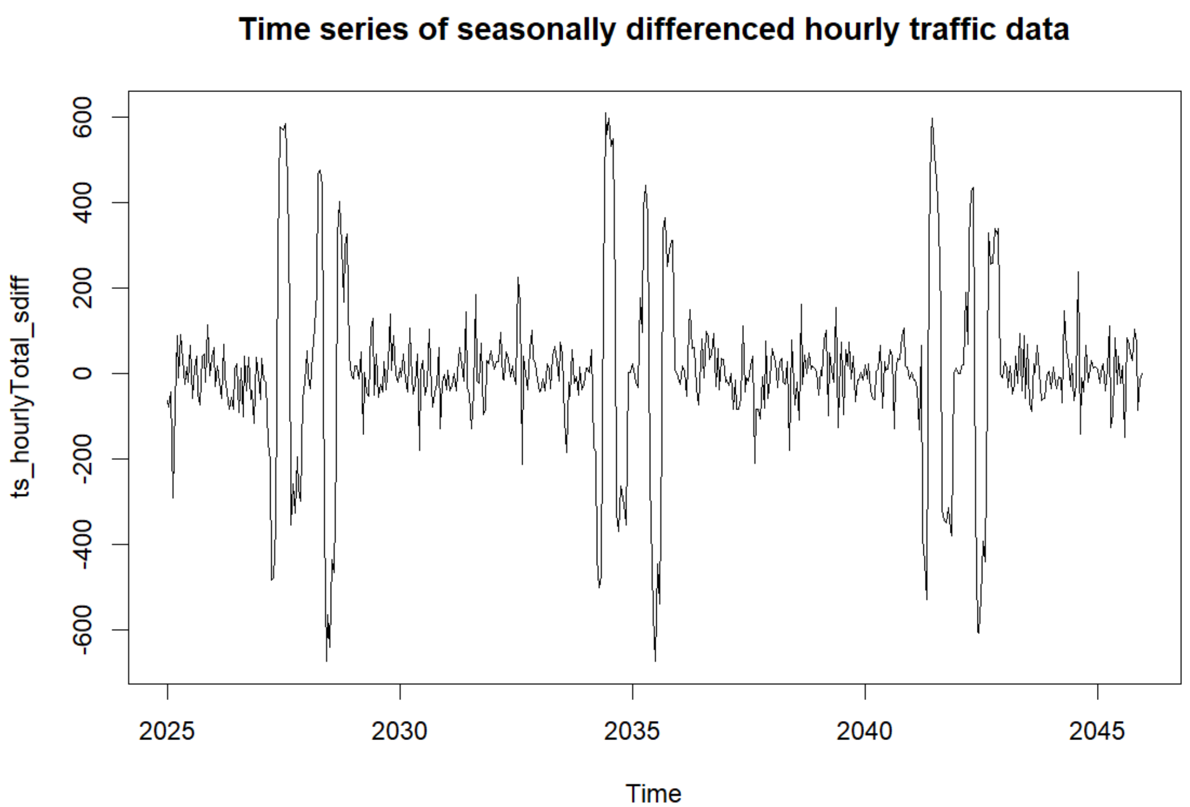
Stationarity of the Data:

So now, our next step would be to check if the data is stationary or not. To see that we need to check if the data requires any differencing.

```
> ndiffs(ts_hourlyTotal)
[1] 0
> nsdiffs(ts_hourlyTotal)
[1] 1
```

The above code snippet shows that for non-seasonal part of the data, there is no differencing required, but if we see the seasonal part of data, 1 differencing is required. Hence we will perform 1 seasonal differencing on lag 24 (Since the seasonality of the data is repeating at 24th lag)

```
ts_hourlyTotal_sdiff <- diff(ts_hourlyTotal, lag=24, difference=1)
par(mfrow = c(1, 1))
plot(ts_hourlyTotal_sdiff, type="l", main="Time series of seasonally differenced hourly traffic data")
```



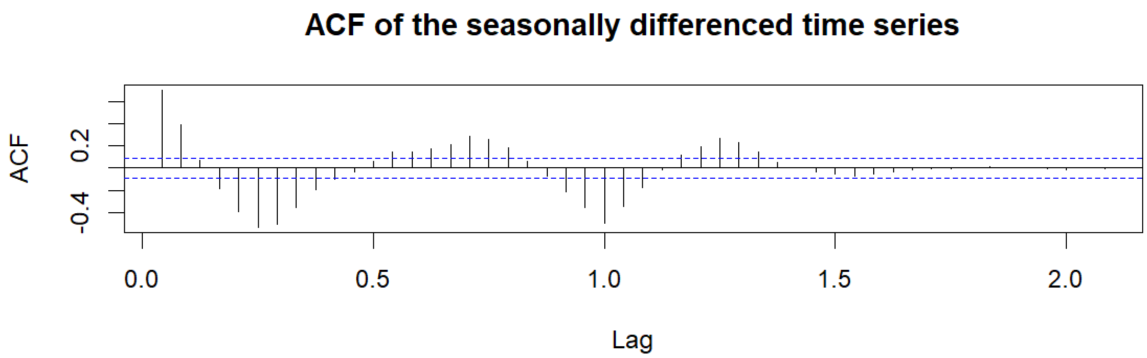
Here we see that after seasonally differencing the data, the plot of the data looks more clearer with significant spike at the seasonal lag. Now lets check if the data is stationary by performing dickey-fuller test.

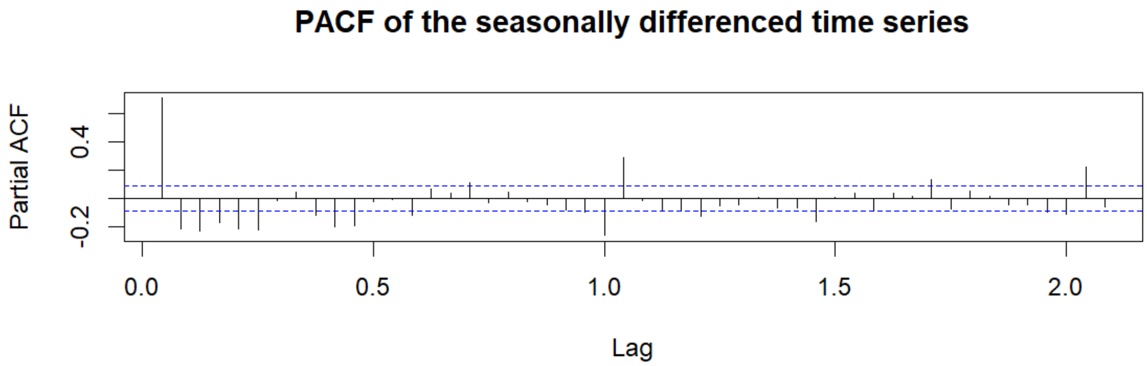
```
> adf.test(ts_hourlyTotal_sdiff)

        Augmented Dickey-Fuller Test

data:  ts_hourlyTotal_sdiff
Dickey-Fuller = -10.175, Lag order = 7, p-value = 0.01
alternative hypothesis: stationary
```

Here in the dickey-fuller test, we see the p-value less than 0.05, so we will reject the null hypothesis and conclude that the data is stationary.

Determining the model specification:

Now, we will proceed with determining which model to use out aur seasonal dataset. Since the data is seasonal, we will first be using SARIMA model. Now, we already know that 1 seasonal differencing was done and also, we know the seasonality is repeating at every 24th lag. So for now we have the SARIMA (P, 0, Q) (p, 1, q) [24].
So now to determine the P, Q, p and q values, we will analyze the ACF, PACF and EACF of the time series.





Here, we observe that in the PACF, each seasonal lag the PACF is decreasing in a constant way, so we are taking q = 0. And after observing the ACF plot, we se that at 2 seasonal lag, the correlations are below the confidence interval. Now lets analyze the EACF of the data

```
> eacf(ts_hourlyTotal_sdiff)
AR/MA
  0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 x x o x x x x x x x o  o  x  x
1 x x x x x x x x o o o  o  x  o
2 x o o o o x x o o o x  o  x  o
3 x x o o x o x x o o x  o  x  o
4 x x x o o o o o o o o  o  x  o
5 x x x x x o o o o o o  o  o  x
6 o x x x x o x o o o o  o  o  x
7 x o x o x o x o o o o  o  o  x
```

From the EACF we have been able to determine the P and Q values of our SARIMA models. So now we need to see which model will have the lowest AIC. The below table shows the SARIMA model specifications and their respective AIC values.

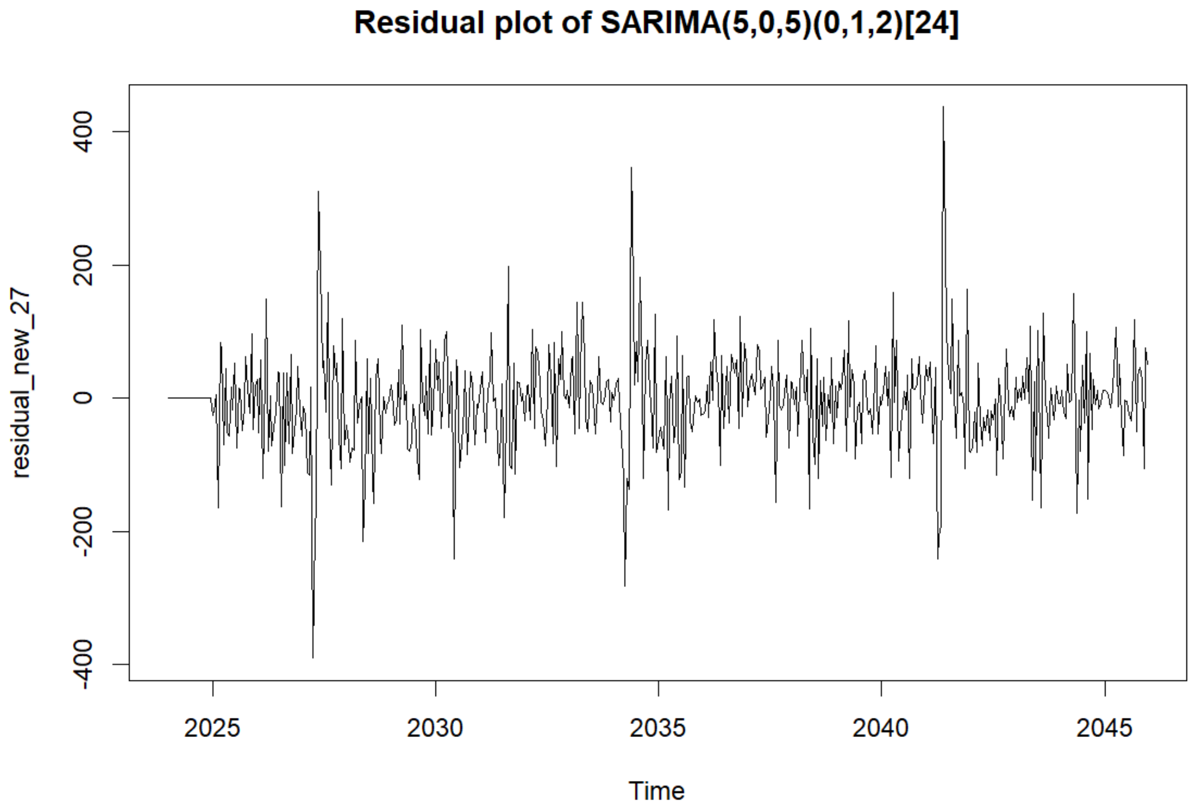
Model	**AIC**
SARIMA(2,0,1)(1,1,0)[24]	6138
SARIMA(2,0,2)(1,1,0)[24]	6253
SARIMA(0,0,2)(1,1,0)[24]	6304
SARIMA(3,0,2)(1,1,0)[24]	6235
SARIMA(3,0,3)(1,1,0)[24]	6142
SARIMA(3,0,5)(1,1,0)[24]	6272
SARIMA(4,0,3)(1,1,0)[24]	6144
SARIMA(5,0,5)(0,1,2)[24]	5940

Here we see that for SARIMA (4, 0, 5) (2, 1, 0) [24] model, the value of AIC is the lowest. Hence, we will be using the model which has the lowest aic score, which is SARIMA (5, 0, 5) (0, 1, 2) [24].

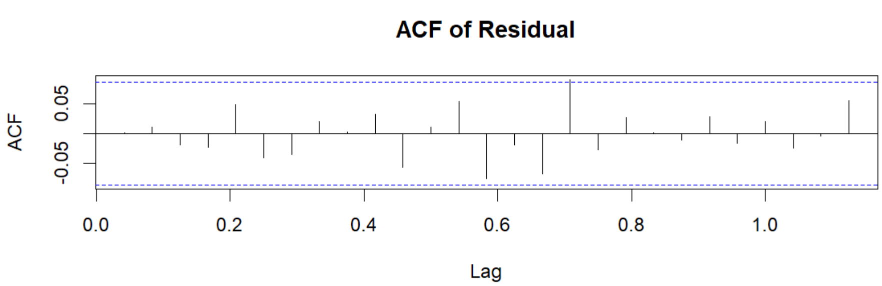
Below contains the details of the model that we will be using.

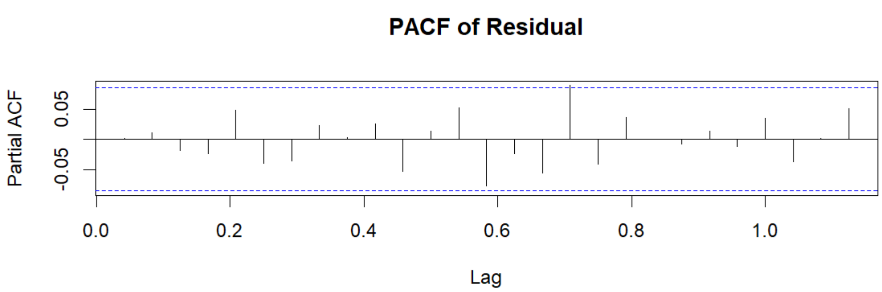
```
> print(model_new_27)

Call:
arima(x = ts_hourlyTotal, order = c(5, 0, 5), seasonal = c(0, 1, 2, 24))

Coefficients:
         ar1      ar2     ar3     ar4      ar5      ma1     ma2     ma3      ma4      ma5     sma1    sma2
      1.7111  -1.3181  0.2764  0.3642  -0.3206  -1.1059  0.6212  0.0422  -0.3259  -0.0687  -1.1210  0.1213
s.e.  0.1932      NaN     NaN     NaN      NaN   0.1929     NaN     NaN      NaN      NaN   0.0705  0.0607

sigma^2 estimated as 6225:  log likelihood = -2958.12,  aic = 5940.23
```

The above model has estimated sigma^2 of 11046 and log likelihood function value of -3065. Also we see the AIC value of the model. Next we will be plotting the residual of the model. Residual of an ARIMA model can be defined as the differences between the observed values of the time series and the values predicted by the ARIMA model.

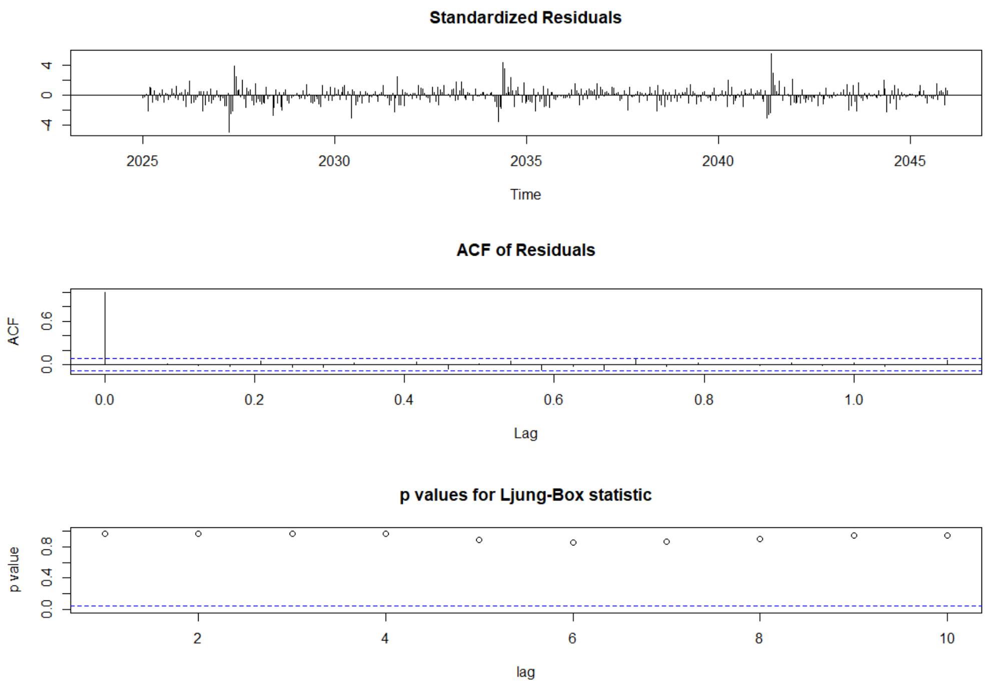


The above residual plot shows the variation of the observed vs the predicted value. Now we will be checking the ACF and PACF of the model

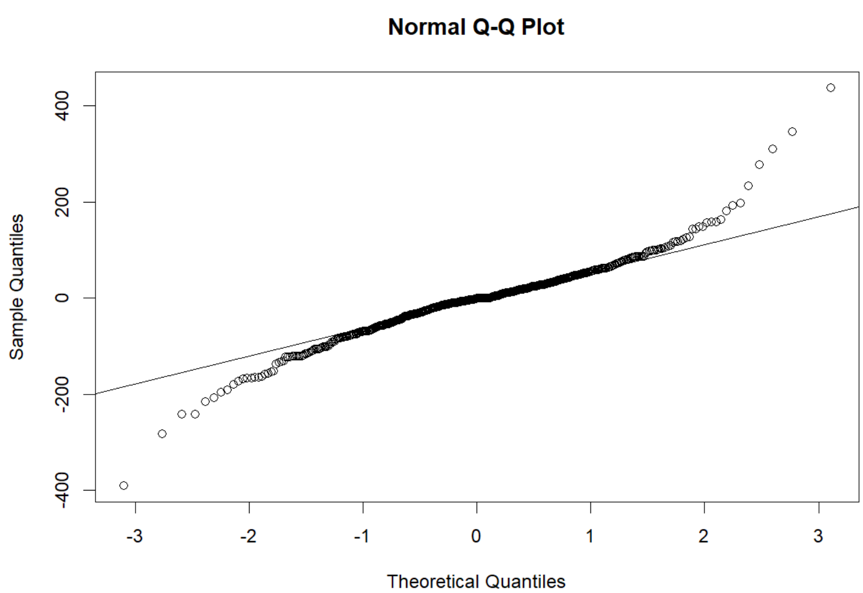




From the ACF and PACF plots of the residual, we can clearly see that the model performs well for the initial prediction of values, but as the time increases, we can see more variations values of observed and the predicted values. Lets perform the Ljung-Box test, which is a statistical test used in time series analysis to check the independence of residuals (or the lack of autocorrelation) in a time series model.



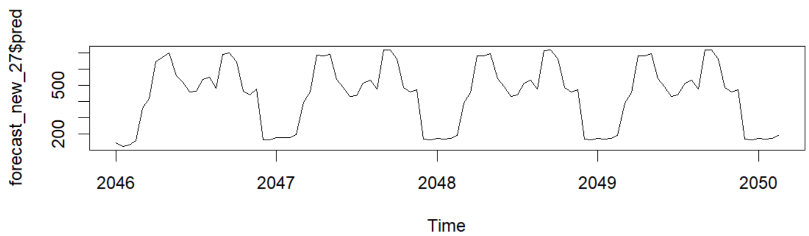
The p values of the Ljung-Box test is above 0.05 for all the lags till 12. So, this suggests that the model performs well for initial values but later as time increases, the model has a high standard deviation.

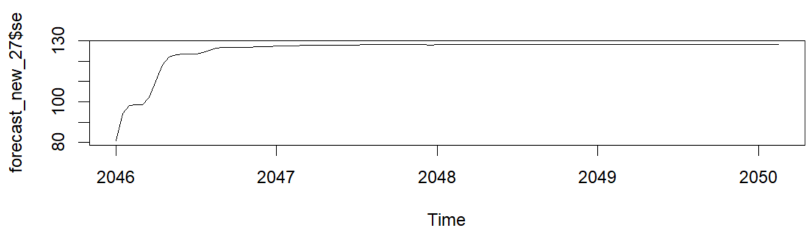


The Normal Q-Q plot shows us that the model has some values out of the Q-Q line, which suggests that the model has a satisfactory result and performs pretty well.

Forecasting:

Now, we will be using the model and forecast the time series based on the model specification.





Here also, we see that the standard error increases with increasing lag, but for initial values, the prediction was good, meaning our model fits well.

Non-Seasonal Dataset – Gasoline Prices in US

Introduction:

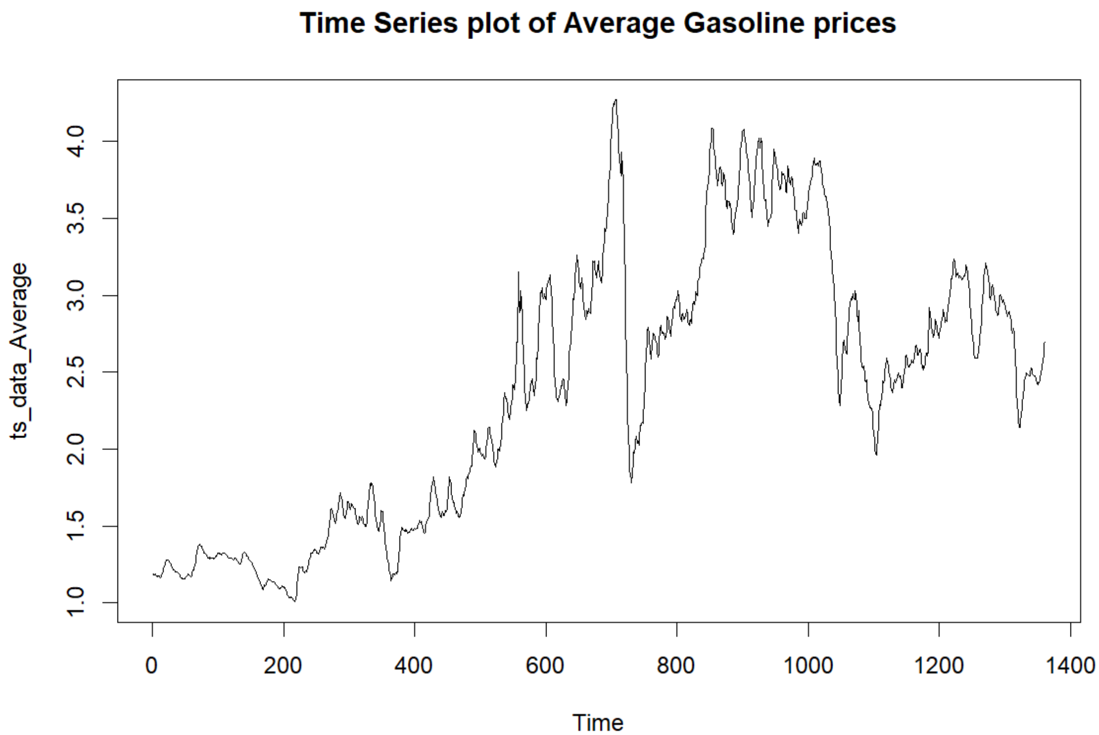
The Dataset that we have used contains weekly Gasoline and diesel prices of the U.S. from 1995 to 2021. The main inspiration of this dataset is to determine what makes the Gasoline prices fluctuate so much. This Dataset is contained in a CSV file with various columns indicating the grade of the Gasoline and their respective prices. Since we are only interested in the time series forecasting of the data, we have created a new column which is the average price of all the grades of Gasoline. The Dataset can be found here - https://www.kaggle.com/datasets/mruanova/us-gasoline-and-diesel-retail-prices-19952021

Now let's look at the structure of the dataset. This dataset has 1361 observations of 14 variables

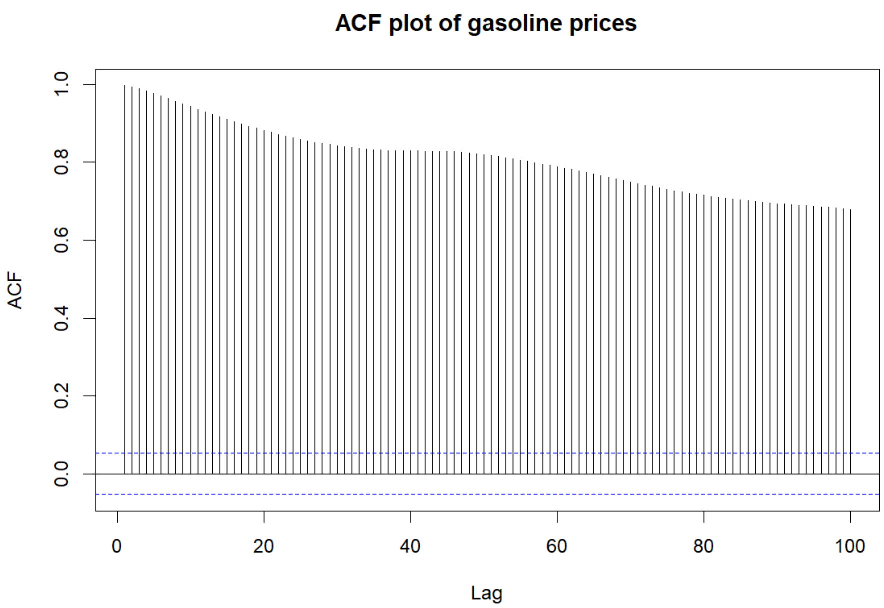
```
> head(Data)
        Date    A1    A2    A3    R1    R2    R3    M1    M2    M3    P1    P2    P3    D1
1 01/02/1995 1.127 1.104 1.231 1.079 1.063 1.167 1.170 1.159 1.298 1.272 1.250 1.386 1.104
2 01/09/1995 1.134 1.111 1.232 1.086 1.070 1.169 1.177 1.164 1.300 1.279 1.256 1.387 1.102
3 01/16/1995 1.126 1.102 1.231 1.078 1.062 1.169 1.168 1.155 1.299 1.271 1.249 1.385 1.100
4 01/23/1995 1.132 1.110 1.226 1.083 1.068 1.165 1.177 1.165 1.296 1.277 1.256 1.378 1.095
5 01/30/1995 1.131 1.109 1.221 1.083 1.068 1.162 1.176 1.163 1.291 1.275 1.255 1.370 1.090
6 02/06/1995 1.124 1.103 1.218 1.076 1.062 1.159 1.169 1.157 1.288 1.270 1.250 1.368 1.086
> summary(Data)
     Date                 A1              A2              A3              R1              R2       
 Length:1361        Min.   :0.949   Min.   :0.926   Min.   :1.039   Min.   :0.907   Min.   :0.885  
 Class :character   1st Qu.:1.461   1st Qu.:1.433   1st Qu.:1.550   1st Qu.:1.421   1st Qu.:1.393  
 Mode  :character   Median :2.326   Median :2.251   Median :2.458   Median :2.237   Median :2.175  
                    Mean   :2.286   Mean   :2.235   Mean   :2.397   Mean   :2.225   Mean   :2.179  
                    3rd Qu.:2.903   3rd Qu.:2.825   3rd Qu.:3.060   3rd Qu.:2.828   3rd Qu.:2.765  
                    Max.   :4.165   Max.   :4.102   Max.   :4.301   Max.   :4.114   Max.   :4.054  
       R3              M1              M2              M3              P1              P2       
 Min.   :0.974   Min.   :1.008   Min.   :0.979   Min.   :1.112   Min.   :1.100   Min.   :1.074  
 1st Qu.:1.489   1st Qu.:1.517   1st Qu.:1.482   1st Qu.:1.616   1st Qu.:1.607   1st Qu.:1.573  
 Median :2.367   Median :2.481   Median :2.404   Median :2.627   Median :2.693   Median :2.640  
 Mean   :2.329   Mean   :2.383   Mean   :2.321   Mean   :2.509   Mean   :2.520   Mean   :2.472  
 3rd Qu.:2.976   3rd Qu.:3.033   3rd Qu.:2.930   3rd Qu.:3.206   3rd Qu.:3.209   3rd Qu.:3.127  
 Max.   :4.247   Max.   :4.229   Max.   :4.153   Max.   :4.387   Max.   :4.344   Max.   :4.283  
       P3              D1       
 Min.   :1.191   Min.   :0.953  
 1st Qu.:1.695   1st Qu.:1.418  
 Median :2.769   Median :2.479  
 Mean   :2.609   Mean   :2.405  
 3rd Qu.:3.318   3rd Qu.:3.070  
 Max.   :4.459   Max.   :4.764  
```

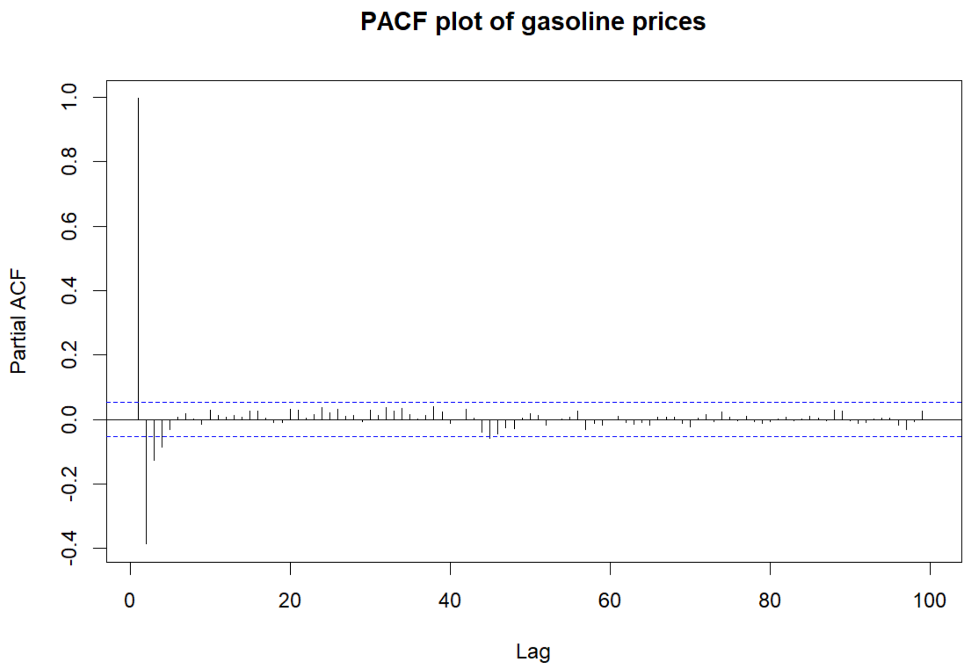
Analysis of Time Series:

For better understanding of the data, we will plot the time series:



Analyzing the plot, we can clearly see that the data is non seasonal as there is no seasonal repetition of the data. We can see the prices of the gasoline going up and down without any seasonality. So, our next step would be to calculate the ACF and PACF plots of the dataset, which will help us understand the strength between the variables over time





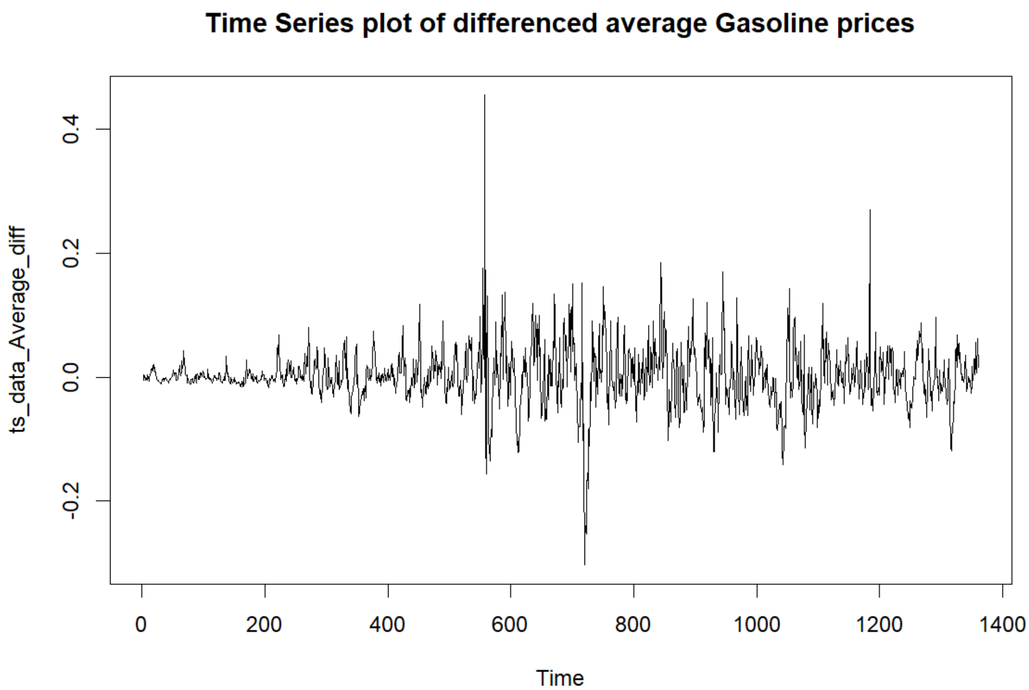
From the ACF and PACF plot we can clearly see that the data is non stationary as the correlations are reducing in a constant way.

Stationarity of the Data:

So now, our next step would be to check if the data is stationary or not. To see that we need to check if the data requires any differencing. So for this, we will be using Augmented Dickey-Fuller test.

```
> adf.test(ts_data_Average)

        Augmented Dickey-Fuller Test

data:  ts_data_Average
Dickey-Fuller = -2.852, Lag order = 11, p-value = 0.2176
alternative hypothesis: stationary
```

The Augmented dickey-fuller test shows us that the data is non stationary since the value of p is greater than 0.05. So, the data needs to be differenced in order to make it stationary. Let’s plot the differenced time series



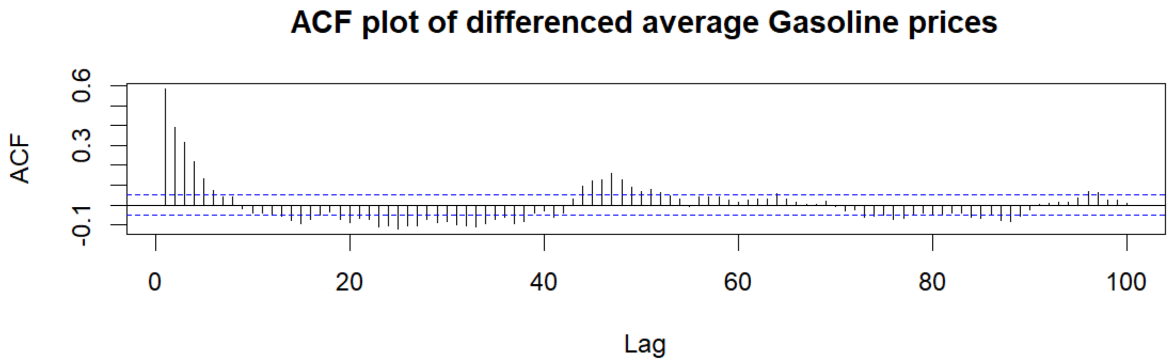
Here the data appears to be stationary now, but we will confirm this by using the Augmented Dickey-Fuller Test.

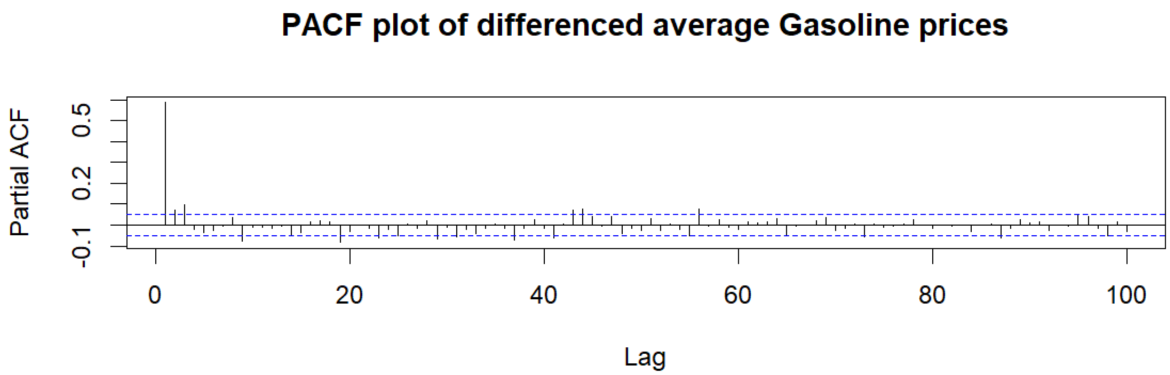
```
> adf.test(ts_data_Average_diff)

        Augmented Dickey-Fuller Test

data:  ts_data_Average_diff
Dickey-Fuller = -9.9735, Lag order = 11, p-value = 0.01
alternative hypothesis: stationary
```

Now we can say that the data is now stationary as the value of p is less than 0.05.

Determining the model specification:

Now, we will proceed with determining which model to use for our non-seasonal dataset. Since the data is non stationary, first we will be using ARIMA model. Since we have to difference the data, so value of d = 1. Now to determine the p and q values of the ARIMA(p, 1 q) model, we need to analyze the ACF, PACF and EACF of the difference time series data.






```
> eacf(ts_data_Average_diff)
AR/MA
  0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 x x x x x x o o o o o  o  x  x
1 x x x o o o o o o o o  o  o  o
2 x x x o o o o o x o o  o  o  o
3 x x x o o o o x x x o  o  o  o
4 x x x o o o o x o o o  o  o  o
5 x x o x o o o x o x o  o  o  o
6 x x o o x x o o o o o  o  o  o
7 x x x x x x o o o o o  o  o  o
```

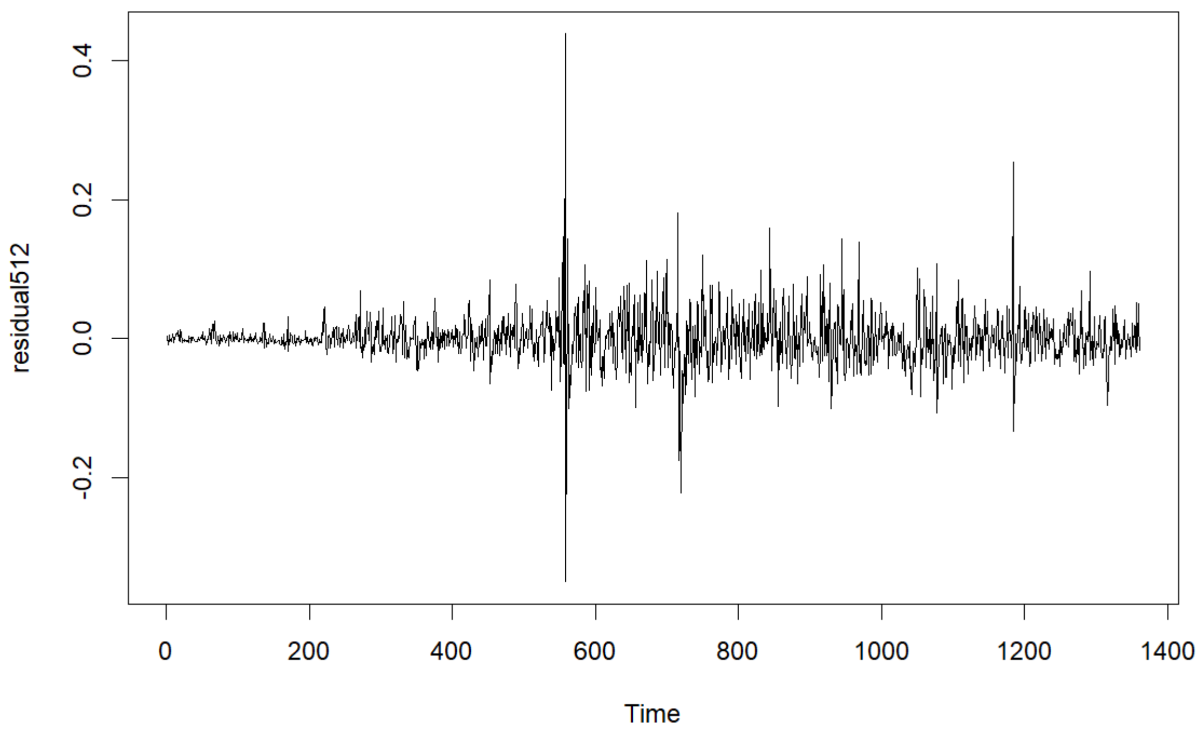
Observing the ACF plot, see that the approximately 4 lags are above the confidence interval. Also, for PACF plot we see 1 lag being above the confidence interval. So our initial guess is to take ARIMA(4, 1, 1) and ARIMA(5, 1, 2) since 2 lags are above the confidence interval in the PACF plot and 5 lags are above confidence interval in the ACF plot

```
> print(aic_results)
  p q       AIC
1 4 0 -5061.094
2 4 1 -5071.881
3 4 2 -5069.938
4 5 0 -5060.806
5 5 1 -5069.951
6 5 2 -5073.009
```

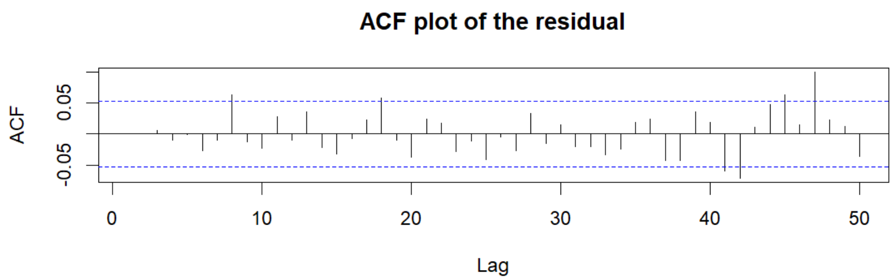
Here we have compared the AIC values of models using for loop and we see for ARIMA (5, 1, 2) model has the lowest AIC value. Hence, we will be using this model for fitting the data.
Below contains the details of the model that we will be using.

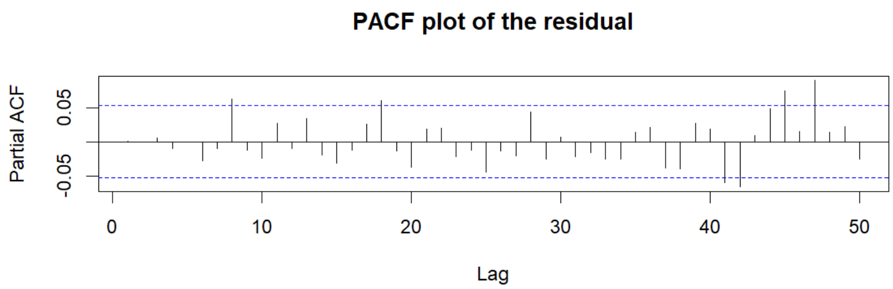
```
> print(model512)

Call:
arima(x = ts_data_Average, order = c(5, 1, 2))

Coefficients:
         ar1     ar2      ar3      ar4      ar5     ma1      ma2
      0.5218  0.9635  -0.3961  -0.0077  -0.1295  0.0108  -0.9456
s.e.  0.0362  0.0420   0.0411   0.0310   0.0274  0.0253   0.0251

sigma^2 estimated as 0.001389:  log likelihood = 2543.5,  aic = -5073.01
```

The above model has estimated sigma^2 of 0.0014 and log likelihood function value of 2543.5. Also, we see the AIC value of the model. Next we will be plotting the residual of the model. Residual of an ARIMA model can be defined as the differences between the observed values of the time series and the values predicted by the ARIMA model.



Here we can see that the residuals are all near zero, indicating that our model performed really well. Now we will see the ACF and PACF plots of the residual.

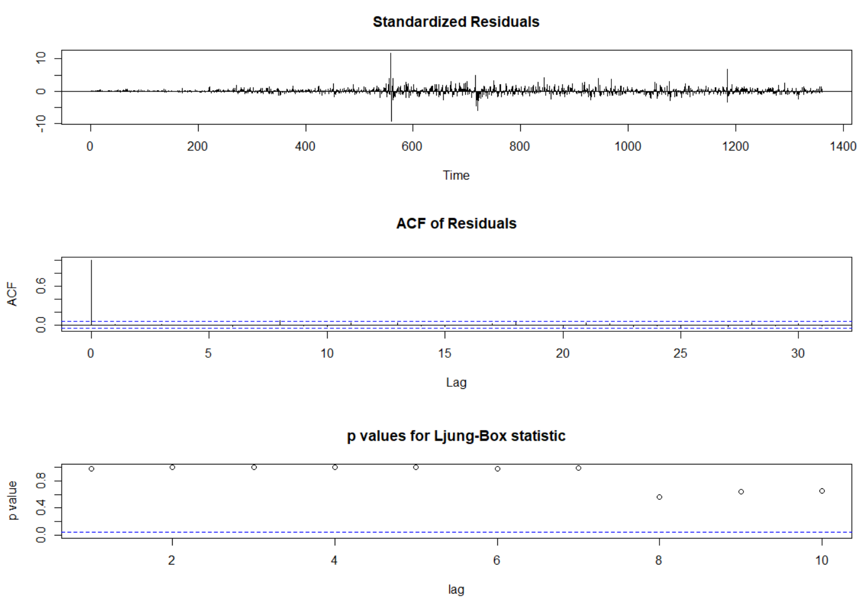




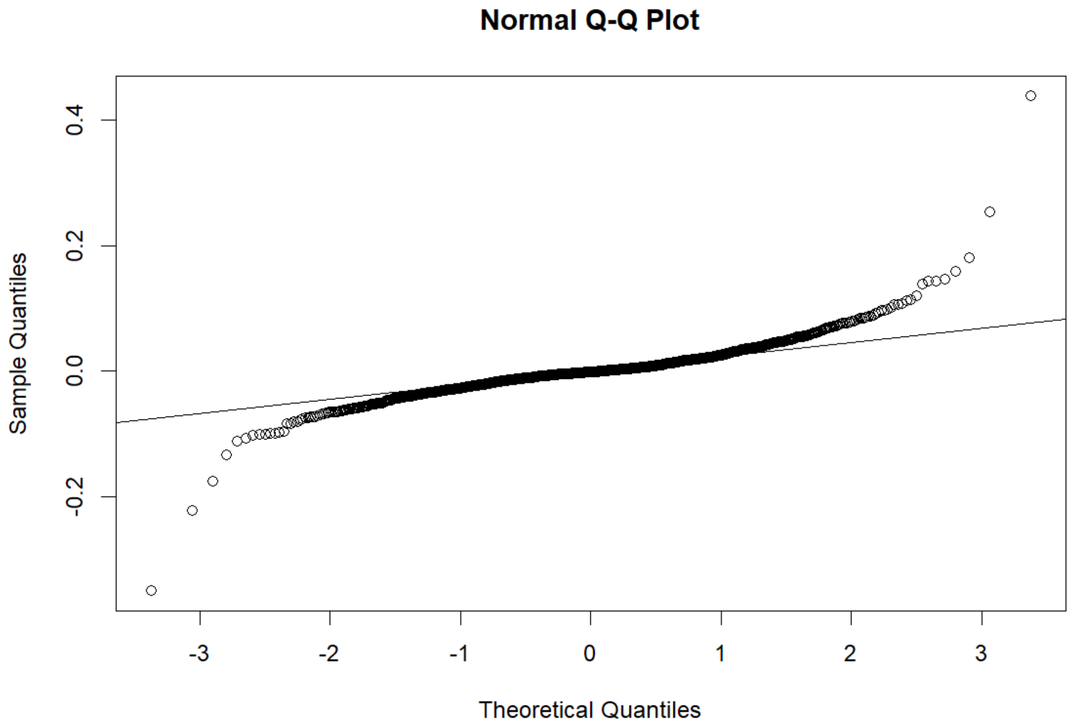
Observing the ACF and PACF plots, we see that volatility of the residual is less as most of the lags are below the significance level. Now we will perform the Ljung-Box test which is a statistical test used in time series analysis to check the independence of residuals (or the lack of autocorrelation) in a time series model.

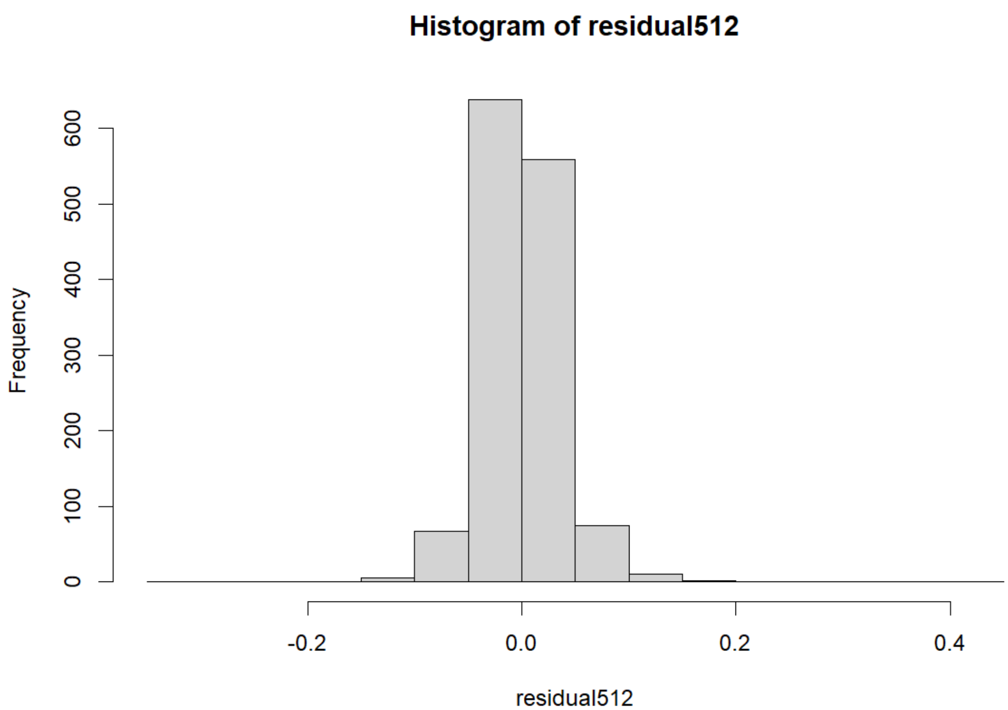
```
> Box.test(residual512, lag = 12, type = "Ljung-Box")

        Box-Ljung test

data:  residual512
X-squared = 8.9213, df = 12, p-value = 0.7096
```

The Ljung-Box test showed us that the p value is significantly greater than 0.05, which is a really good score. We will see a convenient way to visualize and assess the adequacy of a fitted time series model by generating a set of diagnostic plots.



Here we see that the model performs really well and the p values of Ljung-Box test are significantly greater than 0.05.



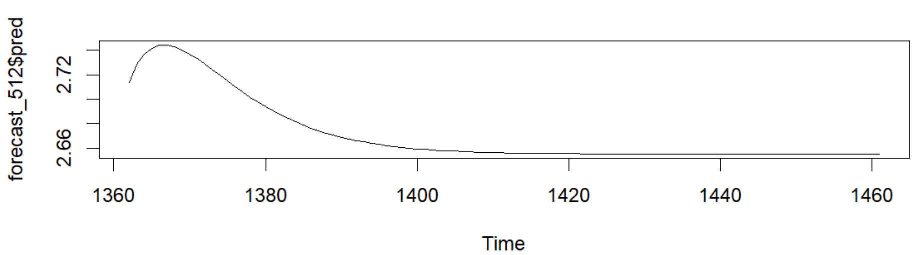


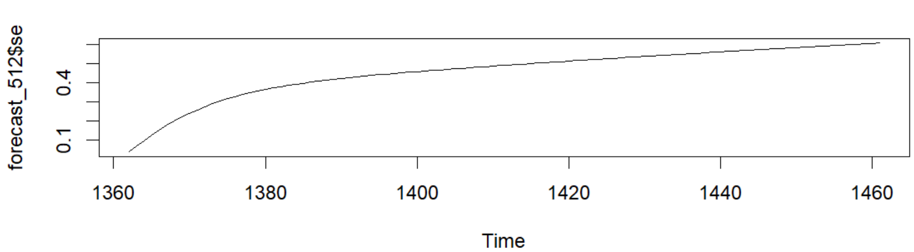
Forecast of ARIMA model:

We will now forecast the data for next 100 values.

```
> forecast_512 <- predict(model512, n.ahead = 100, type = "both")
> print(forecast_512)
$pred
Time Series:
Start = 1362
End = 1461
Frequency = 1
  [1] 2.713799 2.728775 2.737048 2.741821 2.744136 2.744487 2.743006 2.740546 2.737062 2.733158 2.728703
 [12] 2.724208 2.719463 2.714901 2.710269 2.705947 2.701655 2.697735 2.693892 2.690444 2.687087 2.684121
 [23] 2.681241 2.678735 2.676300 2.674213 2.672176 2.670459 2.668771 2.667374 2.665984 2.664858 2.663721
 [34] 2.662822 2.661895 2.661184 2.660430 2.659874 2.659262 2.658831 2.658335 2.658004 2.657602 2.657352
 [45] 2.657025 2.656840 2.656572 2.656438 2.656219 2.656125 2.655943 2.655881 2.655729 2.655691 2.655564
 [56] 2.655545 2.655436 2.655431 2.655338 2.655344 2.655263 2.655277 2.655205 2.655226 2.655161 2.655187
 [67] 2.655128 2.655157 2.655103 2.655134 2.655084 2.655116 2.655070 2.655103 2.655059 2.655093 2.655051
 [78] 2.655085 2.655045 2.655079 2.655041 2.655075 2.655038 2.655071 2.655036 2.655068 2.655034 2.655066
 [89] 2.655033 2.655065 2.655032 2.655063 2.655032 2.655062 2.655032 2.655061 2.655032 2.655061 2.655032
[100] 2.655060

$se
Time Series:
Start = 1362
End = 1461
Frequency = 1
  [1] 0.03727515 0.06821473 0.09643049 0.12420249 0.15108962 0.17611165 0.19944655 0.22089766 0.24067418
 [10] 0.25869476 0.27522288 0.29024190 0.30400221 0.31651084 0.32799240 0.33845807 0.34810204 0.35693182
 [19] 0.36511237 0.37264482 0.37966826 0.38617722 0.39228894 0.39799213 0.40338607 0.40845482 0.41328306
 [28] 0.41785132 0.42223258 0.42640479 0.43043154 0.43428905 0.43803335 0.44163951 0.44515750 0.44856173
 [37] 0.45189724 0.45513815 0.45832543 0.46143320 0.46449908 0.46749738 0.47046292 0.47337033 0.47625207
 [46] 0.47908319 0.48189415 0.48466051 0.48741097 0.49012166 0.49281977 0.49548200 0.49813419 0.50075369
 [55] 0.50336512 0.50594645 0.50852124 0.51106807 0.51360952 0.51612481 0.51863561 0.52112177 0.52360412
 [64] 0.52606311 0.52851881 0.53095227 0.53338282 0.53579213 0.53819881 0.54058511 0.54296899 0.54533328
 [73] 0.54769532 0.55003846 0.55237946 0.55470222 0.55702291 0.55932595 0.56162699 0.56391093 0.56619291
 [82] 0.56845830 0.57072175 0.57296910 0.57521453 0.57744433 0.57967220 0.58188487 0.58409562 0.58629160
 [91] 0.58848564 0.59066530 0.59284303 0.59500676 0.59716854 0.59931669 0.60146288 0.60359579 0.60572672
[100] 0.60784472
```

We see that the standard error increases with increasing lags, but for the initial values, the model performed really well. Let's visualize the predicted value and their standard error.





Applying ARIMA-GARCH model:

Since the model's residual has some volatility, we will try to use ARIMA-GARCH model. ARIMA models are used to capture the linear dependencies in the mean of a time series, while GARCH models are used to capture the conditional heteroscedasticity (time-varying volatility) in the variance of the time series.

To fit an ARIMA-GARCH model we have taken the square of the residual and tried to fit ARMA model to determine the p and q values of the GARCH model. So, after doing all these, we get ARMA (3, 0) as the lowest AIC value of -9997.616.

So we will fit ARIMA-GARCH model with ARIMA(4,1,1) and GARCH(3,0)

```
> spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(3, 0)),
+                    mean.model = list(armaOrder = c(4, 1, 1), include.mean = TRUE,
+                                      archm = FALSE, archpow = 1))
> fit <- ugarchfit(spec, ts_data_Average_log)
> print(fit)

*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model     : sGARCH(3,0)
Mean Model      : ARFIMA(4,0,1)
Distribution    : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error    t value Pr(>|t|)
mu      0.167636    0.003059    54.7948  0.0e+00
ar1     2.532563    0.000432  5869.0882  0.0e+00
ar2    -2.062278    0.000368 -5606.5564  0.0e+00
ar3     0.477835    0.000126  3805.8380  0.0e+00
ar4     0.051891    0.000050  1028.5999  0.0e+00
ma1    -0.884790    0.017554   -50.4035  0.0e+00
omega   0.000067    0.000007    10.2080  0.0e+00
alpha1  0.409390    0.066544     6.1522  0.0e+00
alpha2  0.183741    0.046301     3.9684  7.2e-05
alpha3  0.172034    0.034878     4.9325  1.0e-06

Robust Standard Errors:
        Estimate  Std. Error   t value Pr(>|t|)
mu      0.167636    0.002489   67.3421 0.000000
ar1     2.532563    0.007296  347.1329 0.000000
ar2    -2.062278    0.005939 -347.2688 0.000000
ar3     0.477835    0.001537  310.8050 0.000000
ar4     0.051891    0.000257  201.6355 0.000000
ma1    -0.884790    0.143693   -6.1575 0.000000
omega   0.000067    0.000018    3.8319 0.000127
alpha1  0.409390    0.231791    1.7662 0.077362
alpha2  0.183741    0.076530    2.4009 0.016355
alpha3  0.172034    0.122496    1.4044 0.160196

LogLikelihood : 4055

Information Criteria
------------------------------------

Akaike       -5.9442
Bayes        -5.9058
Shibata      -5.9443
Hannan-Quinn -5.9298
```

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                    0.08545  0.7700
Lag[2*(p+q)+(p+q)-1][14]  5.29332  1.0000
Lag[4*(p+q)+(p+q)-1][24] 11.81184  0.5768
d.o.f=5
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.2708  0.6028
Lag[2*(p+q)+(p+q)-1][8]    3.0719  0.6723
Lag[4*(p+q)+(p+q)-1][14]   8.0043  0.3836
d.o.f=3

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[4]    0.7120 0.500 2.000  0.3988
ARCH Lag[6]    0.8935 1.461 1.711  0.7785
ARCH Lag[8]    3.0752 2.368 1.583  0.5295

Nyblom stability test
------------------------------------
Joint Statistic:  3.9676
Individual Statistics:
mu     0.003947
ar1    0.141799
ar2    0.143035
ar3    0.144244
ar4    0.145731
ma1    0.047539
omega  1.960814
alpha1 0.126529
alpha2 0.149294
alpha3 0.129048

Asymptotic Critical Values (10% 5% 1%)
Joint Statistic:         2.29 2.54 3.05
Individual Statistic:    0.35 0.47 0.75

Sign Bias Test
------------------------------------
                   t-value   prob sig
Sign Bias           1.4619 0.1440
Negative Sign Bias  0.9177 0.3590
Positive Sign Bias  0.1023 0.9186
Joint Effect        7.0221 0.0712   *
```

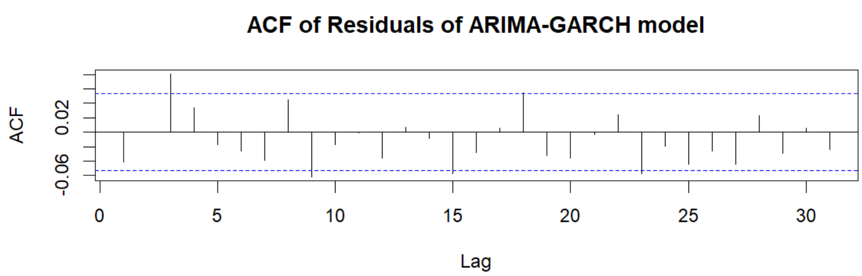
```
Adjusted Pearson Goodness-of-Fit Test:
------------------------------------
  group statistic p-value(g-1)
1    20     104.6    7.855e-14
2    30     134.0    1.903e-15
3    40     138.8    4.097e-13
4    50     142.3    5.013e-11
```

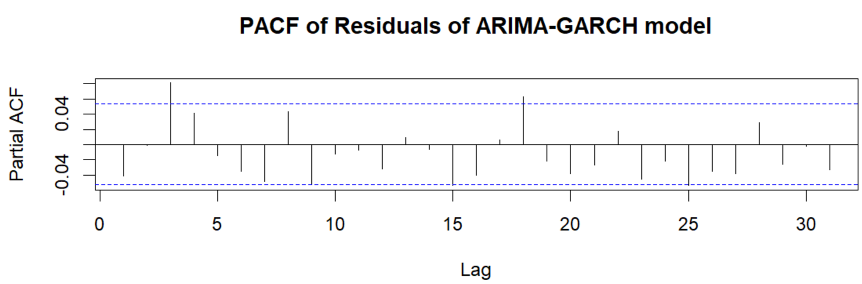
From the model specification, we see that Ljung-Box test has all p values greater than 0.05, specifying good fit. But if we look at the Adjusted Pearson Goodness-of-Fit test, we see the p values less than 0.05. So we need to fit a different model. Now we will be using the Skewed Generalized Error Distribution.

```
> spec2 <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(3, 0)),
+                     mean.model = list(armaOrder = c(4, 1, 1), include.mean = TRUE,
+                                       archm = FALSE, archpow = 1),
+                     distribution.model = "sged")
> fit2 <- ugarchfit(data = ts_data_Average_log, spec = spec2, solver ='hybrid')
> print(fit2)

*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model     : sGARCH(3,0)
Mean Model      : ARFIMA(4,0,1)
Distribution    : sged

Optimal Parameters
------------------------------------
        Estimate  Std. Error   t value Pr(>|t|)
mu      0.174336    0.002433   71.6545 0.000000
ar1     1.478208    0.000251 5899.5937 0.000000
ar2    -0.399620    0.001557 -256.6593 0.000000
ar3    -0.055273    0.002348  -23.5381 0.000000
ar4    -0.023577    0.001725  -13.6653 0.000000
ma1     0.127461    0.022980    5.5467 0.000000
omega   0.000061    0.000006    9.4495 0.000000
alpha1  0.381474    0.064057    5.9552 0.000000
alpha2  0.186126    0.048845    3.8105 0.000139
alpha3  0.188930    0.041588    4.5429 0.000006
skew    1.245268    0.034907   35.6736 0.000000
shape   1.180758    0.060061   19.6592 0.000000

Robust Standard Errors:
        Estimate  Std. Error   t value Pr(>|t|)
mu      0.174336    0.000423  412.4880 0.000000
ar1     1.478208    0.000383 3861.2106 0.000000
ar2    -0.399620    0.001542 -259.1830 0.000000
ar3    -0.055273    0.002025  -27.2970 0.000000
ar4    -0.023577    0.000577  -40.8841 0.000000
ma1     0.127461    0.019890    6.4082 0.000000
omega   0.000061    0.000009    6.4466 0.000000
alpha1  0.381474    0.069028    5.5264 0.000000
alpha2  0.186126    0.050295    3.7006 0.000215
alpha3  0.188930    0.044760    4.2209 0.000024
skew    1.245268    0.038587   32.2716 0.000000
shape   1.180758    0.072489   16.2889 0.000000

LogLikelihood : 4143.512
Information Criteria
------------------------------------

Akaike       -6.0713
Bayes        -6.0253
Shibata      -6.0714
Hannan-Quinn -6.0541

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                      1.180  0.2773
Lag[2*(p+q)+(p+q)-1][14]    7.715  0.3506
Lag[4*(p+q)+(p+q)-1][24]   15.223  0.1512
d.o.f=5
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.2456  0.6202
Lag[2*(p+q)+(p+q)-1][8]    3.2078  0.6473
Lag[4*(p+q)+(p+q)-1][14]   7.2663  0.4719
d.o.f=3
```

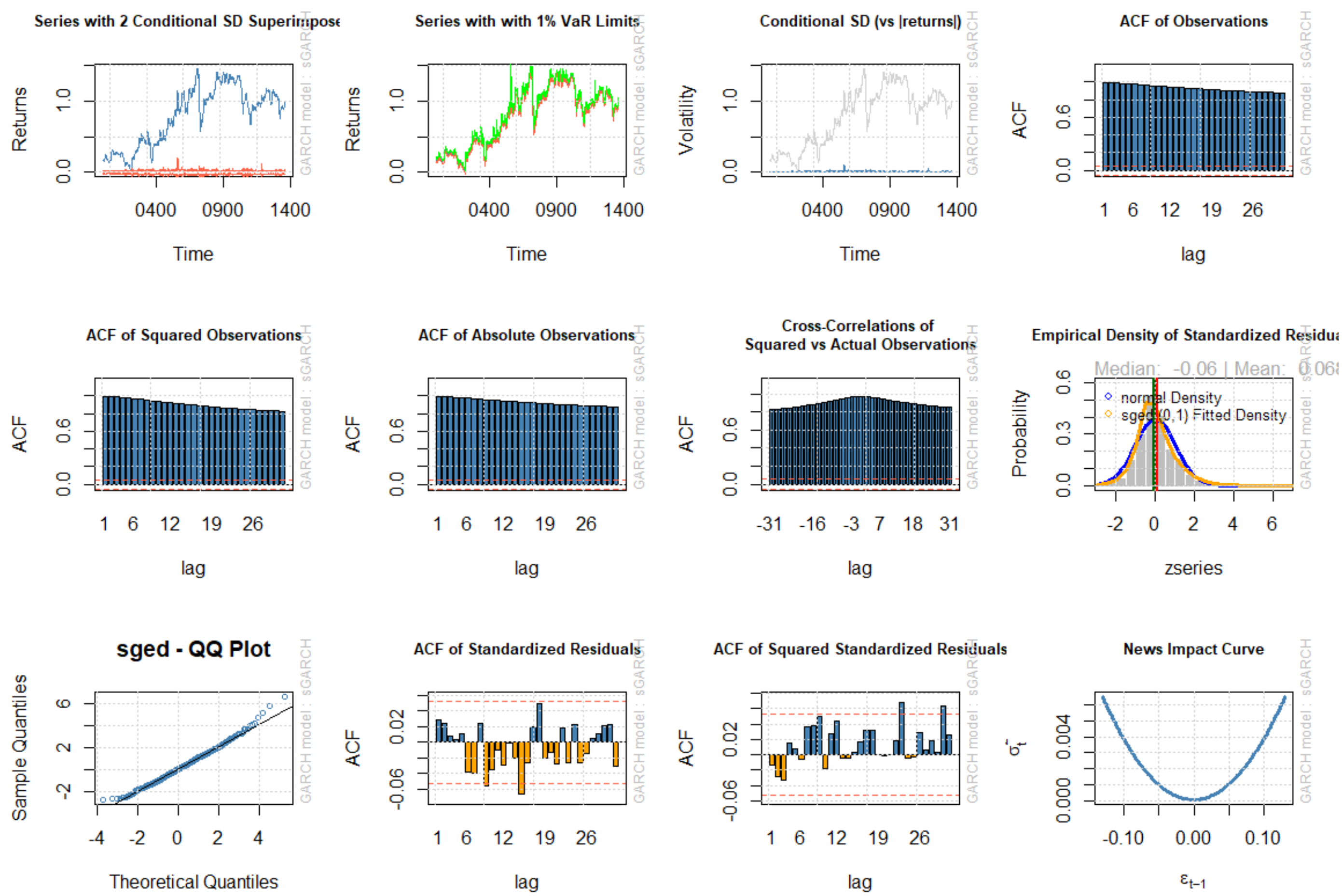
```
Weighted ARCH LM Tests
------------------------------------
             Statistic Shape Scale P-Value
ARCH Lag[4]     0.2985 0.500 2.000  0.5848
ARCH Lag[6]     0.4046 1.461 1.711  0.9181
ARCH Lag[8]     2.4743 2.368 1.583  0.6457

Nyblom stability test
------------------------------------
Joint Statistic:  7.7767
Individual Statistics:
mu     0.0007157
ar1    0.0207012
ar2    0.0208112
ar3    0.0209629
ar4    0.0209876
ma1    0.2310737
omega  4.4515007
alpha1 0.2507352
alpha2 0.2123985
alpha3 0.2919097
skew   0.0546241
shape  1.9391539

Asymptotic Critical Values (10% 5% 1%)
Joint Statistic:         2.69 2.96 3.51
Individual Statistic:    0.35 0.47 0.75

Sign Bias Test
------------------------------------
                   t-value   prob sig
Sign Bias          1.22216 0.2219
Negative Sign Bias 1.08583 0.2777
Positive Sign Bias 0.08042 0.9359
Joint Effect       6.50670 0.0894   *


Adjusted Pearson Goodness-of-Fit Test:
------------------------------------
  group statistic p-value(g-1)
1    20     26.08       0.1279
2    30     26.16       0.6167
3    40     38.24       0.5045
4    50     43.59       0.6914
```

Here we see that for both Ljung-Box test and Adjusted Pearson Goodness-of-Fit test has p values greater than 0.05, which is a good fit. Now we will plot all the visualizing in order to understand the model better.





The ACF and PACF of the residual of ARIMA-GARCH model shows even less volatility in the data, meaning the model has performed well.

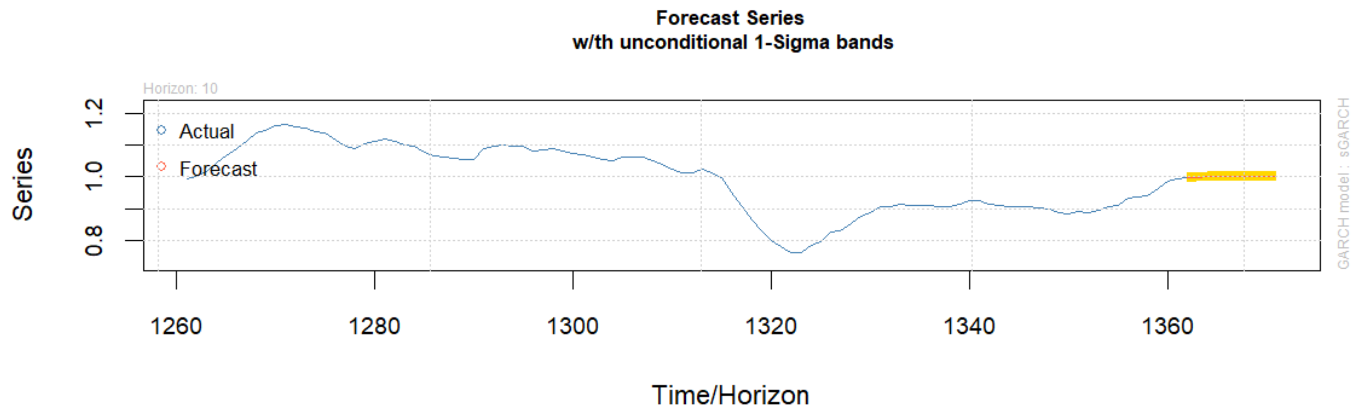


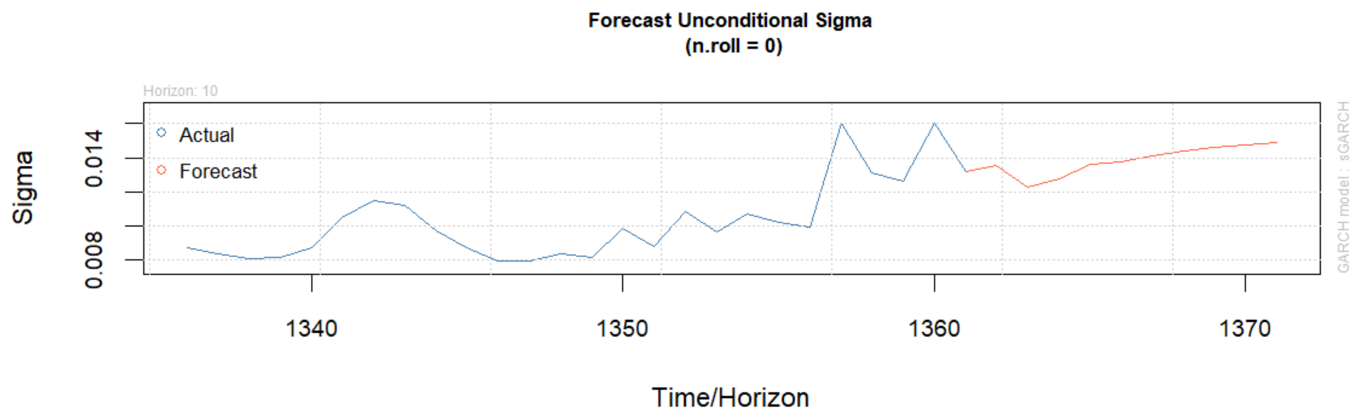
The diagnostic plots generated for the model show that the Q-Q (Quantile-Quantile) plot indicates the residuals are more closely aligned with the theoretical normal distribution. This is evidenced by the residuals forming a distribution that more closely follows the straight line in the Q-Q plot

Forecasting the ARIMA-GARCH model:

Now we will try to forecast the data using the ugarchforecast() method.

```
> forecast_fit2 <- ugarchforecast(fit2, n.ahead = 10)
> fitted(forecast_fit2)
      1361-01-01
T+1    0.9964937
T+2    0.9992015
T+3    1.0007410
T+4    1.0015697
T+5    1.0019346
T+6    1.0019938
T+7    1.0018534
T+8    1.0015826
T+9    1.0012264
T+10   1.0008145
```





The above plots shows the forecasting the next 10 values. In the above figure, we see the forecast results on the line graph in yellow. If we look at the forecast values and visualization, it can be called a satisfactory result. So, the model ARIMA(4,1,1)-GARCH(3,0) is a good fit to the data and we are getting adequate forecast results.