Modeling Microsoft Stock Price Over Time: Utilizing ARIMA and GARCH

Will Markevitch
wmarkevitch@ucsb.edu
Perm No: 4253167
June 13, 2024

Project Abstract

This project utilizes Microsoft stock price data (MSFT) from the beginning of 2016 to the end of 2020 to forecast its future value. This is done by utilizing time series techniques such as ARIMA to model the trend and GARCH to model the volatility. By analyzing the historical stock prices and applying these advanced statistical models, we aim to provide accurate forecasts and insights into the stock price movements of Microsoft.

1. Introduction

The goal of this project is to analyze and forecast the stock price of Microsoft using time series modeling techniques. The choice of Microsoft as the subject of this analysis is motivated by its significant influence in the tech industry and its substantial stock market presence. Given the company's pivotal role in technology and its robust business strategies, understanding its stock price behavior can provide valuable insights.
In the past, various studies have employed time series models to forecast stock prices. Techniques such as ARIMA (AutoRegressive Integrated Moving Average) have been widely used due to their capability to model the linear dependencies in the data. However, stock prices are also known for their volatility, which is better captured by models like GARCH (Generalized AutoRegressive Conditional Heteroskedasticity).
For this project, we will use ARIMA to model the overall trend and GARCH to capture the volatility in Microsoft's stock prices. The data set spans from April 2015 to April 2021, but for this analysis, we will focus on the period from January 2016 to December 2020, which covers significant events including the rise of cloud computing and the impact of the COVID-19 pandemic.
The analysis will involve data preprocessing, model fitting, and diagnostics to ensure the models are well-specified. The key findings and insights will be discussed, and potential areas for future study will be highlighted to improve the robustness and accuracy of the forecasts.

2. Data Description

The dataset used in this project spans from April 2015 to April 2021, covering Microsoft's daily share prices over a period of six years, with all values expressed in US dollars and non-negative. Comprising 1511 daily records, the dataset offers a detailed view of price trends and fluctuations. It was sourced from Google's public records using the 'GOOGLEFINANCE' command in Google Sheets, ensuring the data's accuracy and reliability. It should be noted that for simplicity of analysis and due to the nature of stock data (no weekends/holidays), I opted to average data for each month and subset from January 1st 2016 to December 31st 2020. This comprehensive dataset is essential for analyzing how Microsoft's stock prices evolved through the late 2010s and through the beginning of the global Covid-19 pandemic. The focus on Microsoft is due to its significant business strategies involving expansions and acquisitions, which directly impact its stock prices. Moreover, as a leading tech company, Microsoft's performance during the pandemic serves as a crucial indicator of the tech sector's resilience. With the future of AI already among us, I wanted to also look into the historical performance of one of the leading companies in AI products. By studying this dataset, we aim to understand the impact of Microsoft's strategic decisions on their stock prices and gauge the overall health and trends of the technology sector during a critical period. The dataset can be accessed on Kaggle via the following link: Microsoft Stock Time Series Analysis.

3. Methodology

3.1 SARIMA (p, d, q) x (P, D, Q) model

Seasonal Auto Regressive Integrated Moving Average (SARIMA) is a time series forecasting method that accounts for seasonality in the data it is provided. It has the capabilities of an ARIMA model, but also incorporates seasonal elements utilizing season differencing and seasonal auto regressive and moving average terms. This makes it a good mixture of capturing short-term patterns, such as week to week patterns or day to day patterns in the data, but also capturing longer term seasonality trends like year over year changes in stock price for our analysis. The general SARIMA model is represented by:

$$\text{SARIMA}(p, d, q) * (P, D, Q)_s$$

Where:
- p: Non-seasonal autoregressive order
- d: Non-seasonal differencing order
- q: Non-seasonal moving average order
- P: Seasonal autoregressive order
- D: Seasonal differencing order
- Q: Seasonal moving average order
- s: Length of the seasonal cycle

Due to the sub-setting of data and potential misrepresentation of seasonal patterns, I opted to remove the seasonality component and employ an Autoregressive Integrated Moving Average (ARIMA) model for forecasting Microsoft's monthly stock prices. This decision ensures a more straightforward and potentially more accurate modeling approach for the given dataset, which might not exhibit strong seasonal characteristics.

3.2 GARCH model

Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) is a time series model designed to estimate the volatility of stock prices. Unlike traditional ARIMA models that focus on the mean of the series, GARCH models are particularly effective in capturing and forecasting the variability or volatility of the data over time. This makes GARCH models particularly useful for financial time series, where volatility clustering (periods of high volatility followed by periods of low volatility) is a common phenomenon.

The GARCH model is defined by two components: the autoregressive (AR) term and the moving average (MA) term, both of which are applied to the variance of the series rather than the series itself. The basic GARCH(p, q) model can be expressed as:

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2$$

Where:

- σ_t^2: The conditional variance at time t.
- α_0: A constant term.
- α_i: The coefficients for the lagged squared residuals (innovations).
- ϵ_{t-i}^2: The lagged squared residuals from the mean equation.
- β_j: The coefficients for the lagged conditional variances.

In our analysis, we used a GARCH(1, 1) model in conjunction with an ARIMA(0, 1, 0) model to capture both the trend and the volatility of Microsoft's monthly stock prices. This combined approach allows us to model the mean and variance of the series effectively, accounting for the persistence in volatility that is often observed in stock prices. In a future study, I could utilize this in the forecasting for a more accurate prediction.

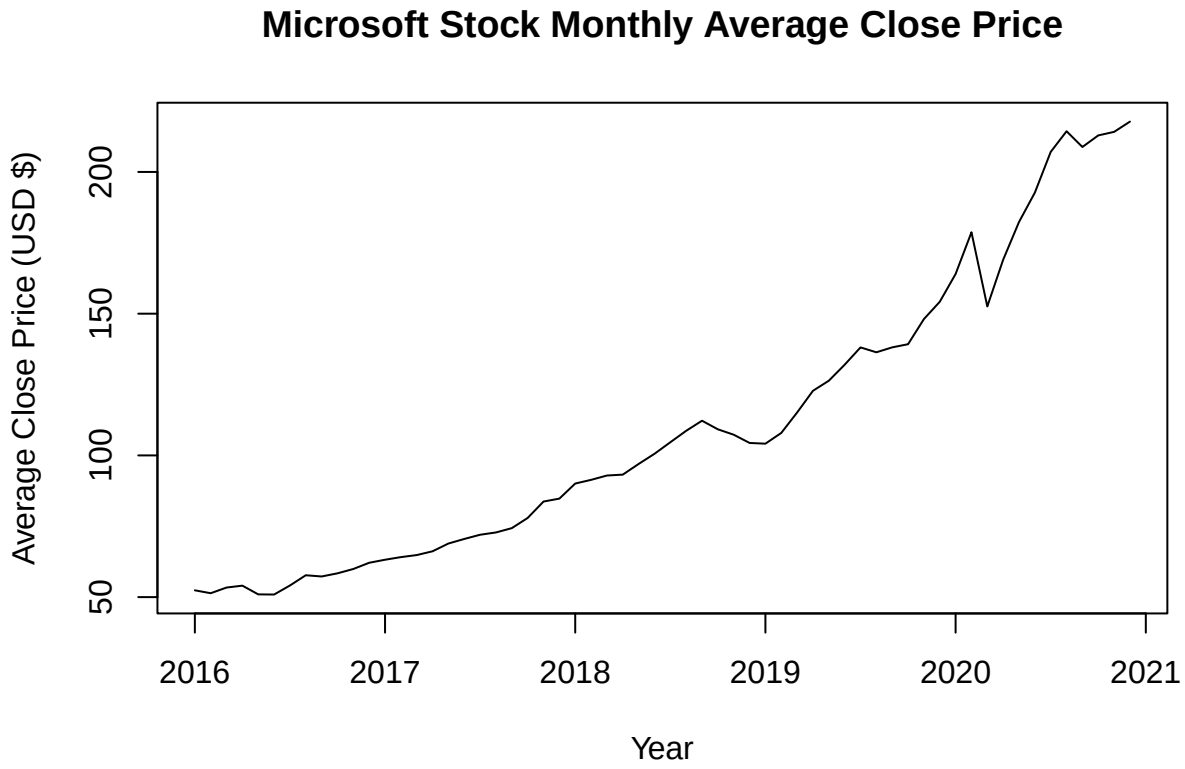
4. Results

Arima Model Fitting

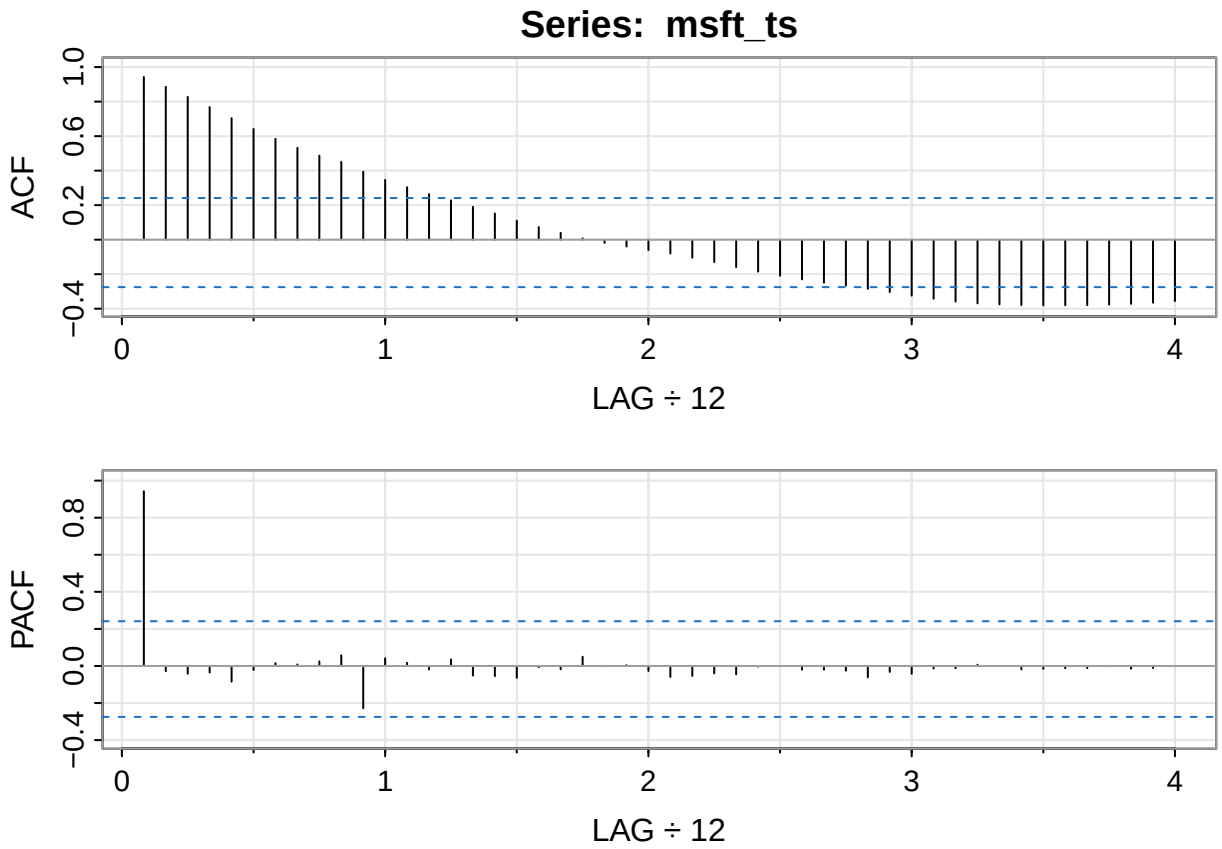
Note: Remember due to the nature of stock data (no weekends/holidays), I opted to average data for each month and subset from January 1st 2016 to December 31st 2020.

Plotting the Original Data and ACF & PACF

```
msft <- read.csv("Microsoft_Stock.csv", stringsAsFactors = FALSE)

msft$Date <- mdy_hms(msft$Date)

msft <- msft %>%
  filter(Date >= as.Date("2016-01-01"), Date <= as.Date("2020-12-31"))

msft_monthly <- msft %>%
  group_by(Year = year(Date), Month = month(Date)) %>%
  summarise(Average.Close = mean(Close))

msft_ts <- ts(msft_monthly$Average.Close, start = c(2016, 1), end = c(2020, 12), frequency = 12)

ts.plot(msft_ts, xlab = "Year", ylab = "Average Close Price (USD $)", main = "Microsoft Stock Monthly A
```



```
acf2(msft_ts)
```

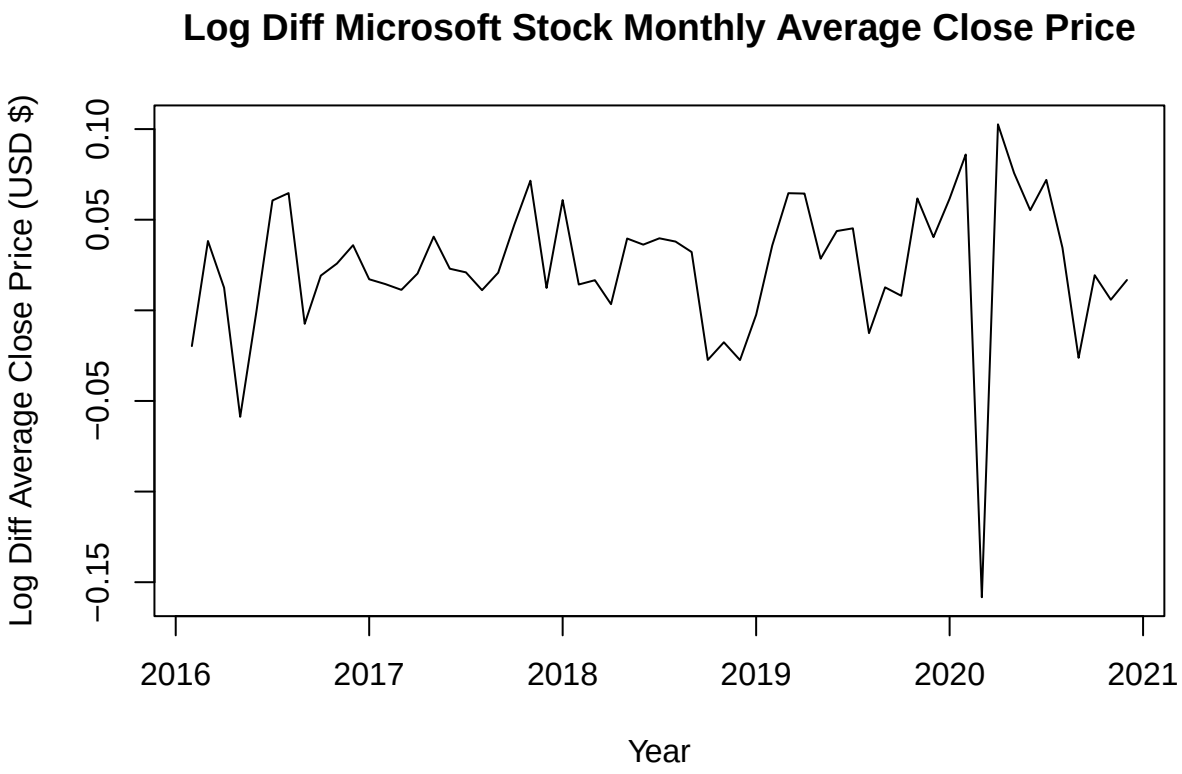



```
##       [,1]  [,2]  [,3]  [,4]  [,5]  [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## ACF   0.94  0.89  0.83  0.77  0.70  0.64 0.58 0.53 0.49  0.45  0.39  0.35  0.30
## PACF 0.94 -0.03 -0.04 -0.04 -0.08 -0.02 0.02 0.01 0.03  0.06 -0.23  0.04  0.02
##       [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25]
## ACF    0.26  0.23  0.19  0.15  0.11  0.07  0.04  0.01 -0.02 -0.04 -0.06 -0.08
## PACF -0.02  0.04 -0.05 -0.05 -0.06 -0.01 -0.02  0.05  0.00  0.00 -0.03 -0.06
##       [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35] [,36] [,37]
## ACF   -0.10 -0.13 -0.16 -0.19 -0.21 -0.23 -0.25 -0.27 -0.28 -0.30 -0.32 -0.34
## PACF -0.05 -0.04 -0.04  0.00  0.00 -0.02 -0.02 -0.03 -0.06 -0.03 -0.04 -0.01
##       [,38] [,39] [,40] [,41] [,42] [,43] [,44] [,45] [,46] [,47] [,48]
## ACF   -0.36 -0.37 -0.37 -0.38 -0.38 -0.38 -0.38 -0.38 -0.37 -0.37 -0.36
## PACF -0.01  0.01  0.00 -0.02 -0.02 -0.01 -0.01  0.00 -0.01 -0.01  0.00
```

We can see that the data has a clear upward trend, indicating that it is non-stationary, so we need to difference our data to try and make it stationary.

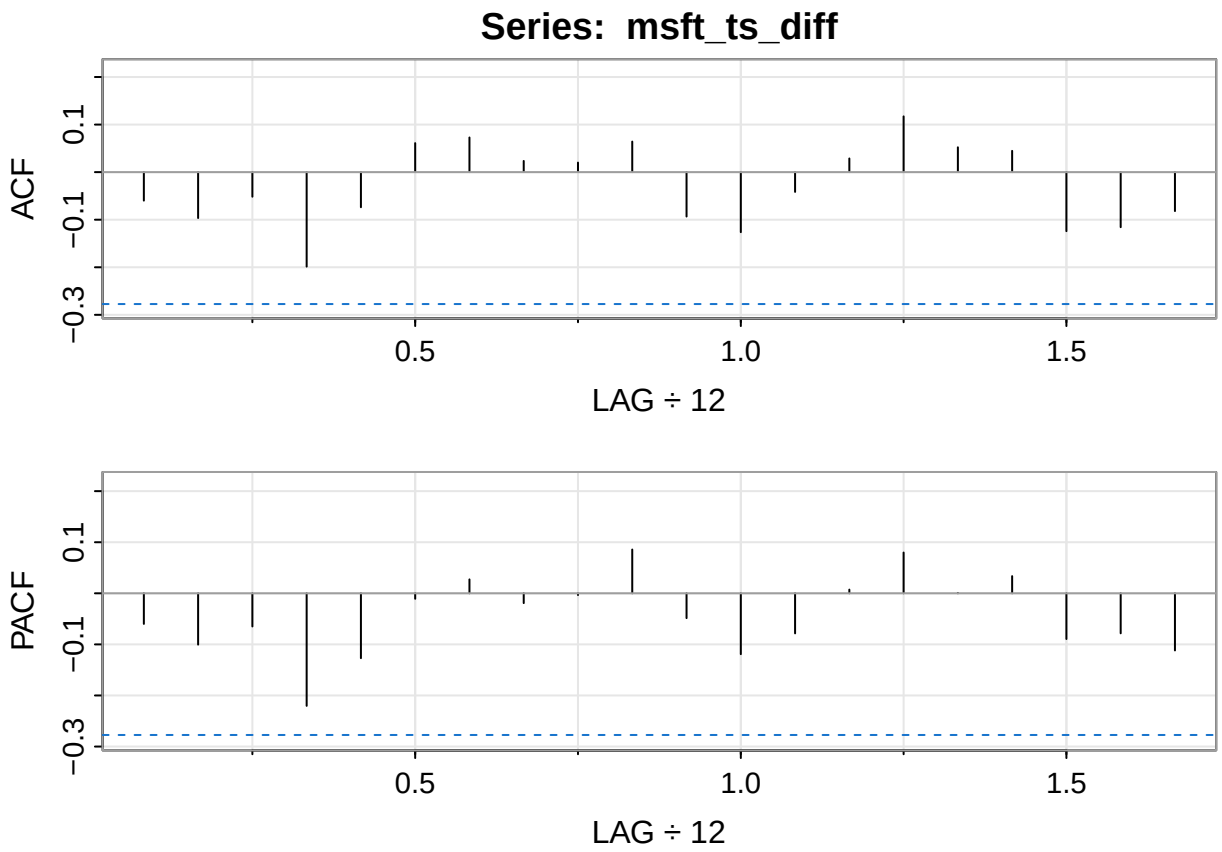
Determining the Difference

```
msft_ts_diff <- diff(log(msft_ts))
ts.plot(msft_ts_diff, xlab = "Year", ylab = "Log Diff Average Close Price (USD $)", main = "Log Diff Mi
```



This looks much more stationary (outlier in early 2020 - covid pandemic), so we can check the ACF and PACF graphs for determining ARIMA models to test.

```
acf2(msft_ts_diff, 20)
```




```
##       [,1] [,2]  [,3]  [,4]  [,5]  [,6] [,7]  [,8] [,9] [,10] [,11] [,12] [,13]
## ACF  -0.06 -0.1 -0.05 -0.20 -0.07  0.06 0.07  0.02 0.02  0.06 -0.09 -0.13 -0.04
## PACF -0.06 -0.1 -0.06 -0.22 -0.13 -0.01 0.03 -0.02 0.00  0.09 -0.05 -0.12 -0.08
##      [,14] [,15] [,16] [,17] [,18] [,19] [,20]
## ACF   0.03  0.12  0.05  0.04 -0.12 -0.12 -0.08
## PACF  0.01  0.08  0.00  0.03 -0.09 -0.08 -0.11
```

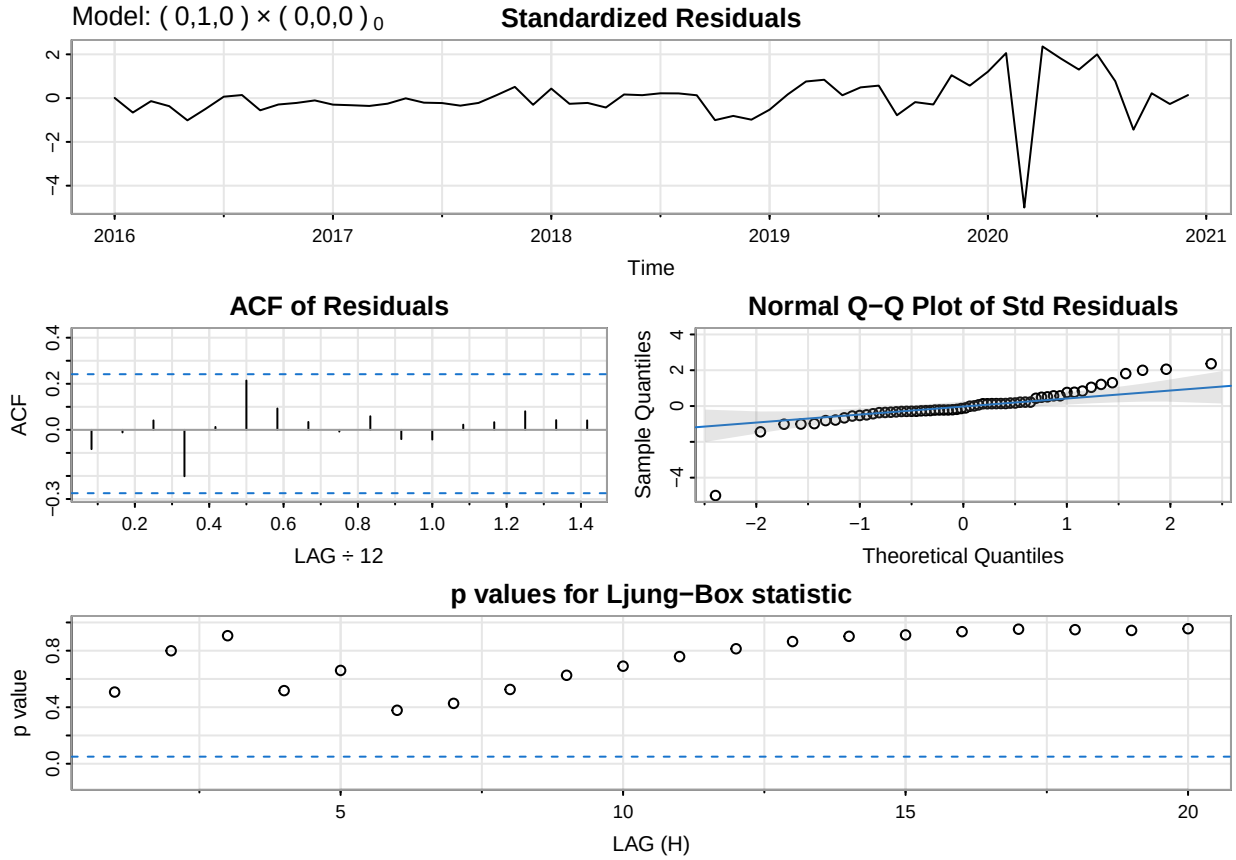
From the ACF and PACF graphs, it isn't clearly decipherable which values for p and q should be used for the ARIMA modeling. So instead, I'll test various values and check for alignment with the qq normality plot and lowest AIC and BIC.

Model Selection: Estimation Results and Diagnostics

```
sarima(msft_ts, 0, 1, 0, 0, 0, 0, 0) # Non-seasonal ARIMA(0, 1, 0) with zero mean
```

```
## initial  value 1.757110
## iter   1 value 1.757110
## final  value 1.757110
## converged
## initial  value 1.757110
## iter   1 value 1.757110
## final  value 1.757110
## converged
```

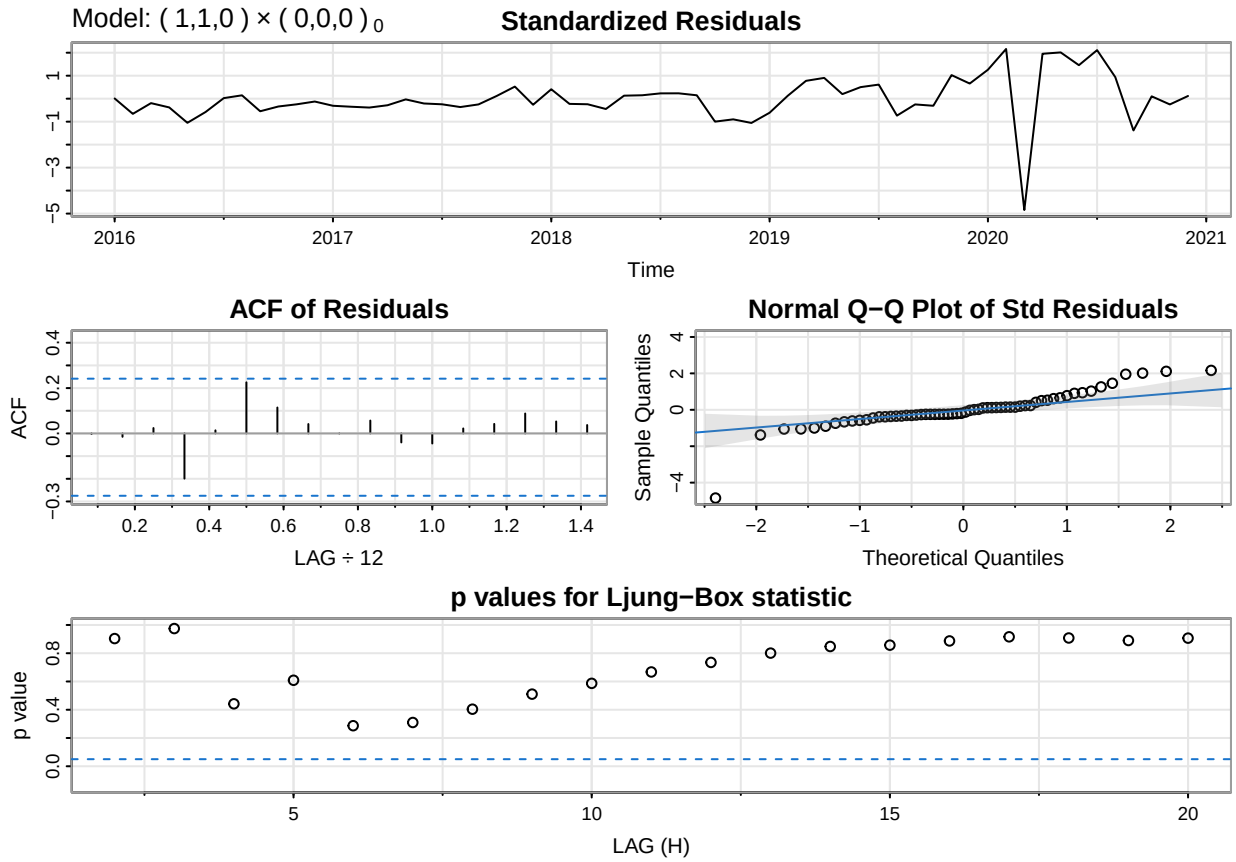
```
## <><><><><><><><><><><><><><><>
##
## Coefficients:
##          Estimate     SE t.value p.value
## constant   2.8025 0.7545  3.7142   5e-04
##
## sigma^2 estimated as 33.58969 on 58 degrees of freedom
##
## AIC = 6.419893  AICc = 6.421082  BIC = 6.490318
##
```




```
sarima(msft_ts, 1, 1, 0, 0, 0, 0, 0) # Non-seasonal ARIMA(1, 1, 0) with zero mean
```

```
## initial  value 1.761951
## iter   2 value 1.758420
## iter   3 value 1.758412
## iter   4 value 1.758371
## iter   4 value 1.758371
## iter   4 value 1.758371
## final  value 1.758371
## converged
## initial  value 1.753707
## iter   2 value 1.753671
## iter   3 value 1.753652
## iter   3 value 1.753652
```

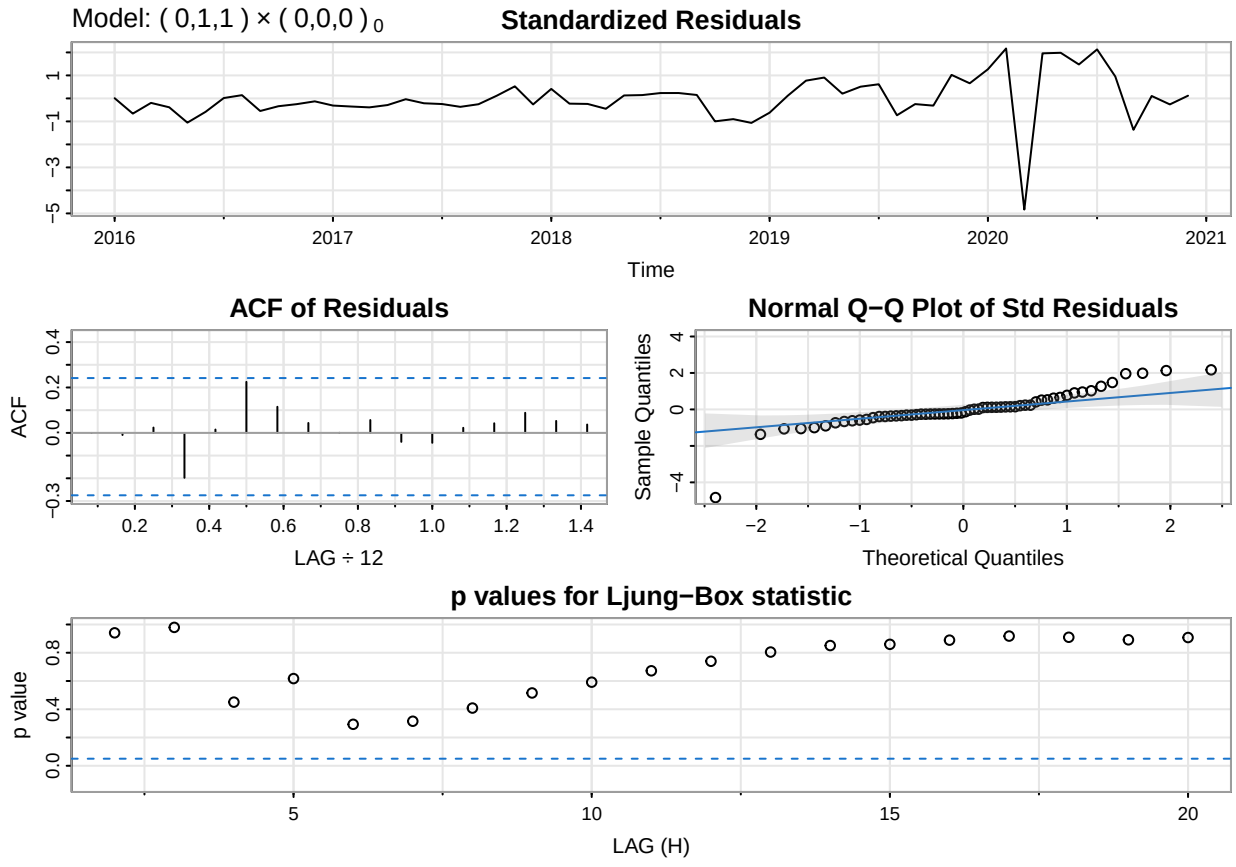
```
## iter   3 value 1.753652
## final  value 1.753652
## converged
## <><><><><><><><><><><><><><><>
##
## Coefficients:
##          Estimate     SE t.value p.value
## ar1       -0.0826 0.1291 -0.6399  0.5248
## constant   2.8064 0.6954  4.0356  0.0002
##
## sigma^2 estimated as 33.35431 on 57 degrees of freedom
##
## AIC = 6.446875  AICc = 6.450507  BIC = 6.552513
##
```




```
sarima(msft_ts, 0, 1, 1, 0, 0, 0, 0) # Non-seasonal ARIMA(0, 1, 1) with zero mean
```

```
## initial  value 1.757110
## iter   2 value 1.753530
## iter   3 value 1.753528
## iter   4 value 1.753527
## iter   5 value 1.753527
## iter   5 value 1.753527
## iter   5 value 1.753527
## final  value 1.753527
```

```
## converged
## initial  value 1.753561
## iter   2 value 1.753560
## iter   2 value 1.753560
## iter   2 value 1.753560
## final  value 1.753560
## converged
## <><><><><><><><><><><><><><>
##
## Coefficients:
##          Estimate     SE t.value p.value
## ma1       -0.0848 0.1304 -0.6498  0.5184
## constant   2.8067 0.6892  4.0722  0.0001
##
## sigma^2 estimated as 33.34803 on 57 degrees of freedom
##
## AIC = 6.446693  AICc = 6.450325  BIC = 6.55233
##
```

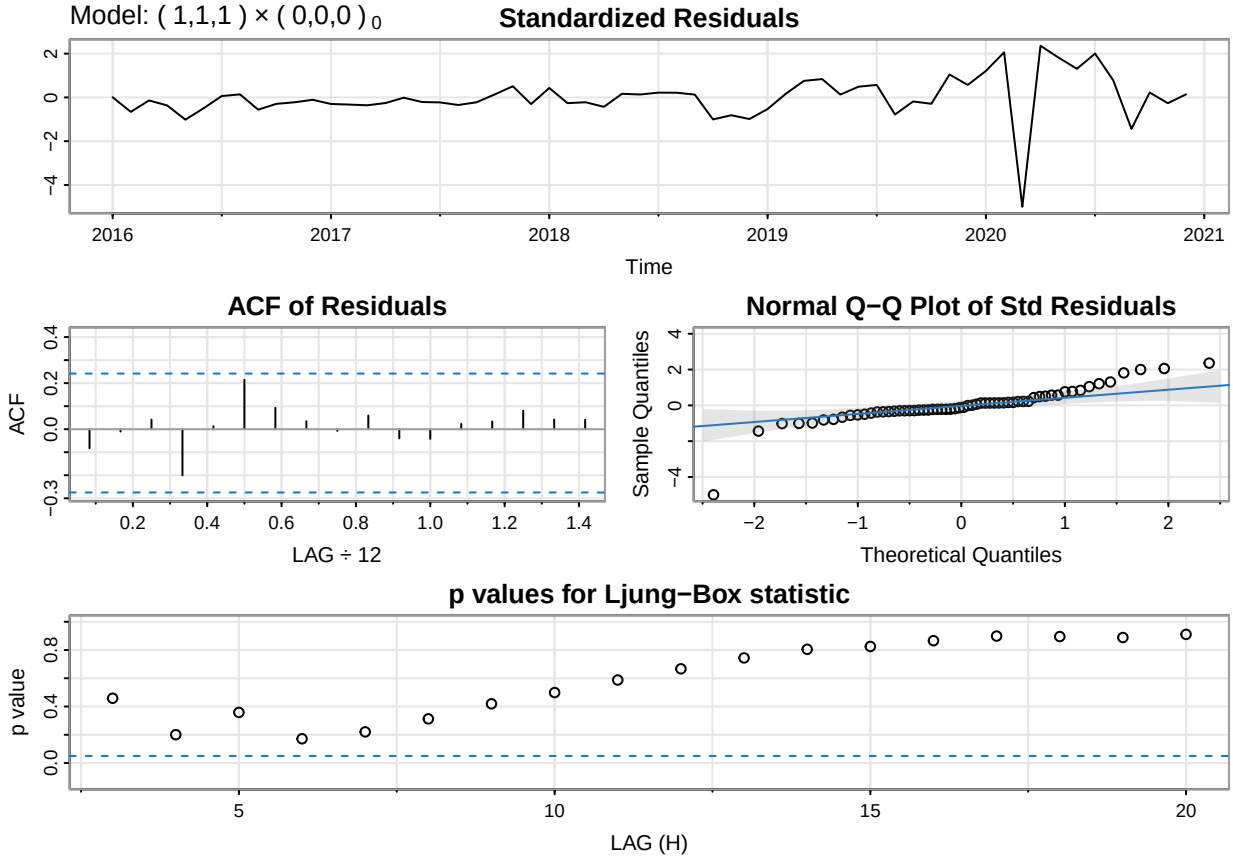



```
sarima(msft_ts, 1, 1, 1, 0, 0, 0, 0) # Non-seasonal ARIMA(1, 1, 1) with zero mean
```

```
## initial  value 1.761951
## iter   2 value 1.761672
## iter   3 value 1.758209
## iter   4 value 1.758203
```

```
## iter   5 value 1.758192
## iter   6 value 1.758103
## iter   7 value 1.757917
## iter   8 value 1.757330
## iter   9 value 1.757005
## iter  10 value 1.756828
## iter  11 value 1.756786
## iter  12 value 1.756513
## iter  13 value 1.755093
## iter  14 value 1.754207
## iter  15 value 1.753774
## iter  16 value 1.753688
## iter  17 value 1.753504
## iter  18 value 1.752771
## iter  19 value 1.746676
## iter  20 value 1.746630
## iter  21 value 1.740681
## iter  22 value 1.734251
## iter  23 value 1.729601
## iter  24 value 1.722151
## iter  25 value 1.720103
## iter  26 value 1.716853
## iter  27 value 1.715363
## iter  28 value 1.709301
## iter  29 value 1.704391
## iter  30 value 1.695759
## iter  31 value 1.692751
## iter  32 value 1.685968
## iter  33 value 1.666510
## iter  34 value 1.666318
## iter  35 value 1.665792
## iter  36 value 1.657073
## iter  37 value 1.652089
## iter  38 value 1.649562
## iter  39 value 1.649539
## iter  40 value 1.648762
## iter  41 value 1.637884
## iter  42 value 1.636293
## iter  42 value 1.636293
## iter  43 value 1.636292
## iter  43 value 1.636292
## iter  43 value 1.636292
## final  value 1.636292
## converged
## initial  value 1.762969
## iter   2 value 1.760225
## iter   3 value 1.757759
## iter   4 value 1.757261
## iter   5 value 1.757112
## iter   6 value 1.757108
## iter   7 value 1.757107
## iter   7 value 1.757107
## iter   7 value 1.757107
## final  value 1.757107
```

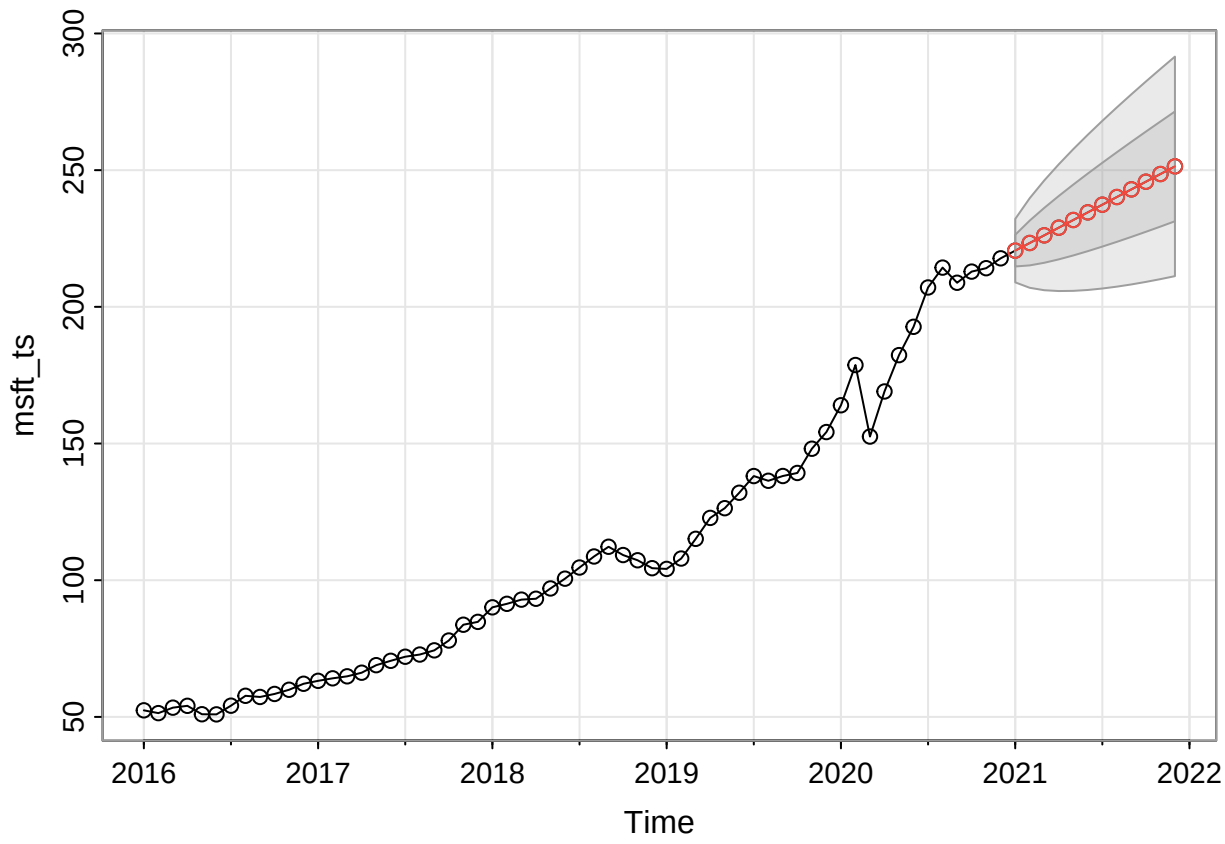
```
## converged

## Warning in sqrt(diag(fitit$var.coef)): NaNs produced

## Warning in sqrt(diag(fitit$var.coef)): NaNs produced

## <><><><><><><><><><><><><><><>
##
## Coefficients:
##          Estimate     SE t.value p.value
## ar1        0.9021    NaN     NaN     NaN
## ma1       -0.9030    NaN     NaN     NaN
## constant   2.8026 0.7488  3.7427   4e-04
##
## sigma^2 estimated as 33.58954 on 56 degrees of freedom
##
## AIC = 6.487685  AICc = 6.495081  BIC = 6.628535
##
```



Forecasting

From the analysis above, the model with the lowest AIC and BIC was a non-seasonal ARIMA(0, 0) model with a $d = 1.$, so we can forecast the next 12 data points below:

```
sarima.for(msft_ts, n.ahead = 12, 0, 1, 0)
```




```
## $pred
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2021 220.5534 223.3559 226.1584 228.9609 231.7634 234.5659 237.3684 240.1709
##           Sep      Oct      Nov      Dec
## 2021 242.9734 245.7759 248.5784 251.3809
##
## $se
##            Jan       Feb       Mar       Apr       May       Jun       Jul
## 2021  5.795661  8.196303 10.038380 11.591322 12.959492 14.196413 15.333878
##            Aug       Sep       Oct       Nov       Dec
## 2021 16.392605 17.386984 18.327490 19.222034 20.076759
```

GARCH Fitting

Testing Various GARCH Models

As we found that our Microsoft data is best represented by an ARIMA(0, 1, 0) model, when calculating the ideal GARCH model we can use the differenced data and ARMA(0, 0). Here I'll test GARCH(1, 1), GARCH(2, 1) and GARCH(1, 2):

```r
msft_garch11 <- ugarchspec(variance.model=list(model="sGARCH",garchOrder=c(1,1)),
                mean.model=list(armaOrder=c(0,0)),distribution.model="norm")

msft_garch11_summary <- ugarchfit(data = msft_ts, spec = msft_garch11)
```

```
## Warning in .sgarchfit(spec = spec, data = data, out.sample = out.sample, :
## ugarchfit-->waring: using less than 100 data
##  points for estimation
```

```r
msft_garch11_summary
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu       63.8318    1.077830 59.222469  0.00000
## omega     1.3355    1.856315  0.719443  0.47187
## alpha1    0.9990    0.181798  5.495117  0.00000
## beta1     0.0000    0.035098  0.000002  1.00000
##
## Robust Standard Errors:
##         Estimate  Std. Error   t value Pr(>|t|)
## mu       63.8318    1.853850 34.432002  0.00000
## omega     1.3355    2.091957  0.638404  0.52321
## alpha1    0.9990    0.042892 23.290793  0.00000
## beta1     0.0000    0.005861  0.000012  0.99999
##
## LogLikelihood : -284.4862
##
## Information Criteria
## ------------------------------------
##
## Akaike       9.6162
## Bayes        9.7558
## Shibata      9.6080
## Hannan-Quinn 9.6708
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic   p-value
## Lag[1]                      55.38 9.925e-14
## Lag[2*(p+q)+(p+q)-1][2]     78.41 0.000e+00
```

```
## Lag[4*(p+q)+(p+q)-1][5]    130.10 0.000e+00
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic   p-value
## Lag[1]                      15.25 9.397e-05
## Lag[2*(p+q)+(p+q)-1][5]     16.85 1.411e-04
## Lag[4*(p+q)+(p+q)-1][9]     17.92 6.784e-04
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.1488 0.500 2.000  0.6997
## ARCH Lag[5]    0.6900 1.440 1.667  0.8265
## ARCH Lag[7]    1.1278 2.315 1.543  0.8919
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  3.2822
## Individual Statistics:
## mu     0.30036
## omega  0.03946
## alpha1 2.04239
## beta1  0.18151
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.07 1.24 1.6
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value      prob sig
## Sign Bias            4.668 1.995e-05 ***
## Negative Sign Bias   2.261 2.772e-02  **
## Positive Sign Bias   2.833 6.427e-03 ***
## Joint Effect        30.064 1.338e-06 ***
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     255.3    2.569e-43
## 2    30     309.0    1.343e-48
## 3    40     382.5    5.648e-58
## 4    50     382.1    3.845e-53
##
##
## Elapsed time : 0.02967
```

```
msft_garch21 <- ugarchspec(variance.model=list(model="sGARCH",garchOrder=c(2,1)),
                mean.model=list(armaOrder=c(0,0)),distribution.model="norm")
```

```
msft_garch21_summary <- ugarchfit(data = msft_ts, spec = msft_garch21)
```

```
## Warning in .sgarchfit(spec = spec, data = data, out.sample = out.sample, :
## ugarchfit-->waring: using less than 100 data
##  points for estimation
```

```
msft_garch21_summary
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(2,1)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu       92.7796     2.27349  40.8094 0.000000
## omega     6.3394     4.43815   1.4284 0.153181
## alpha1    0.9990     0.24281   4.1143 0.000039
## alpha2    0.0000     0.46670   0.0000 1.000000
## beta1     0.0000     0.31850   0.0000 1.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu       92.7796    4.911969  18.8885 0.000000
## omega     6.3394    2.916333   2.1738 0.029723
## alpha1    0.9990    0.078401  12.7421 0.000000
## alpha2    0.0000    0.651855   0.0000 1.000000
## beta1     0.0000    0.638053   0.0000 1.000000
##
## LogLikelihood : -286.7826
##
## Information Criteria
## ------------------------------------
##
## Akaike       9.7261
## Bayes        9.9006
## Shibata      9.7136
## Hannan-Quinn 9.7944
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic  p-value
## Lag[1]                      57.30 3.73e-14
## Lag[2*(p+q)+(p+q)-1][2]     83.95 0.00e+00
## Lag[4*(p+q)+(p+q)-1][5]    152.97 0.00e+00
```

```
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                          statistic   p-value
## Lag[1]                       21.51 3.521e-06
## Lag[2*(p+q)+(p+q)-1][8]      29.13 3.967e-07
## Lag[4*(p+q)+(p+q)-1][14]     36.91 1.320e-07
## d.o.f=3
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[4]     2.734 0.500 2.000 0.09824
## ARCH Lag[6]     5.529 1.461 1.711 0.08622
## ARCH Lag[8]     6.622 2.368 1.583 0.12017
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  4.7294
## Individual Statistics:
## mu     0.56239
## omega  0.02742
## alpha1 0.59527
## alpha2 1.62205
## beta1  1.21810
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.28 1.47 1.88
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value      prob sig
## Sign Bias            5.283 2.242e-06 ***
## Negative Sign Bias   2.734 8.397e-03 ***
## Positive Sign Bias   2.205 3.166e-02  **
## Joint Effect        35.392 1.007e-07 ***
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     169.0    4.459e-26
## 2    30     180.0    1.004e-23
## 3    40     191.7    4.866e-22
## 4    50     247.6    2.531e-28
##
##
## Elapsed time : 0.02643394
```

```
msft_garch12 <- ugarchspec(variance.model=list(model="sGARCH",garchOrder=c(1,2)),
                mean.model=list(armaOrder=c(0,0)),distribution.model="norm")
```

```
msft_garch12_summary <- ugarchfit(data = msft_ts, spec = msft_garch12)
```

```
## Warning in .sgarchfit(spec = spec, data = data, out.sample = out.sample, :
## ugarchfit-->waring: using less than 100 data
##  points for estimation
```

```
msft_garch12_summary
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,2)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu       92.7796     2.33418  39.7483 0.000000
## omega     6.3394     5.09963   1.2431 0.213830
## alpha1    0.9990     0.36027   2.7729 0.005555
## beta1     0.0000     0.21950   0.0000 1.000000
## beta2     0.0000     0.15374   0.0000 1.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu       92.7796     5.38771  17.2206 0.000000
## omega     6.3394     2.50048   2.5353 0.011237
## alpha1    0.9990     0.44974   2.2213 0.026331
## beta1     0.0000     0.63477   0.0000 1.000000
## beta2     0.0000     0.10965   0.0000 1.000000
##
## LogLikelihood : -286.7826
##
## Information Criteria
## ------------------------------------
##
## Akaike       9.7261
## Bayes        9.9006
## Shibata      9.7136
## Hannan-Quinn 9.7944
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic  p-value
## Lag[1]                      57.30 3.73e-14
## Lag[2*(p+q)+(p+q)-1][2]     83.95 0.00e+00
## Lag[4*(p+q)+(p+q)-1][5]    152.97 0.00e+00
```

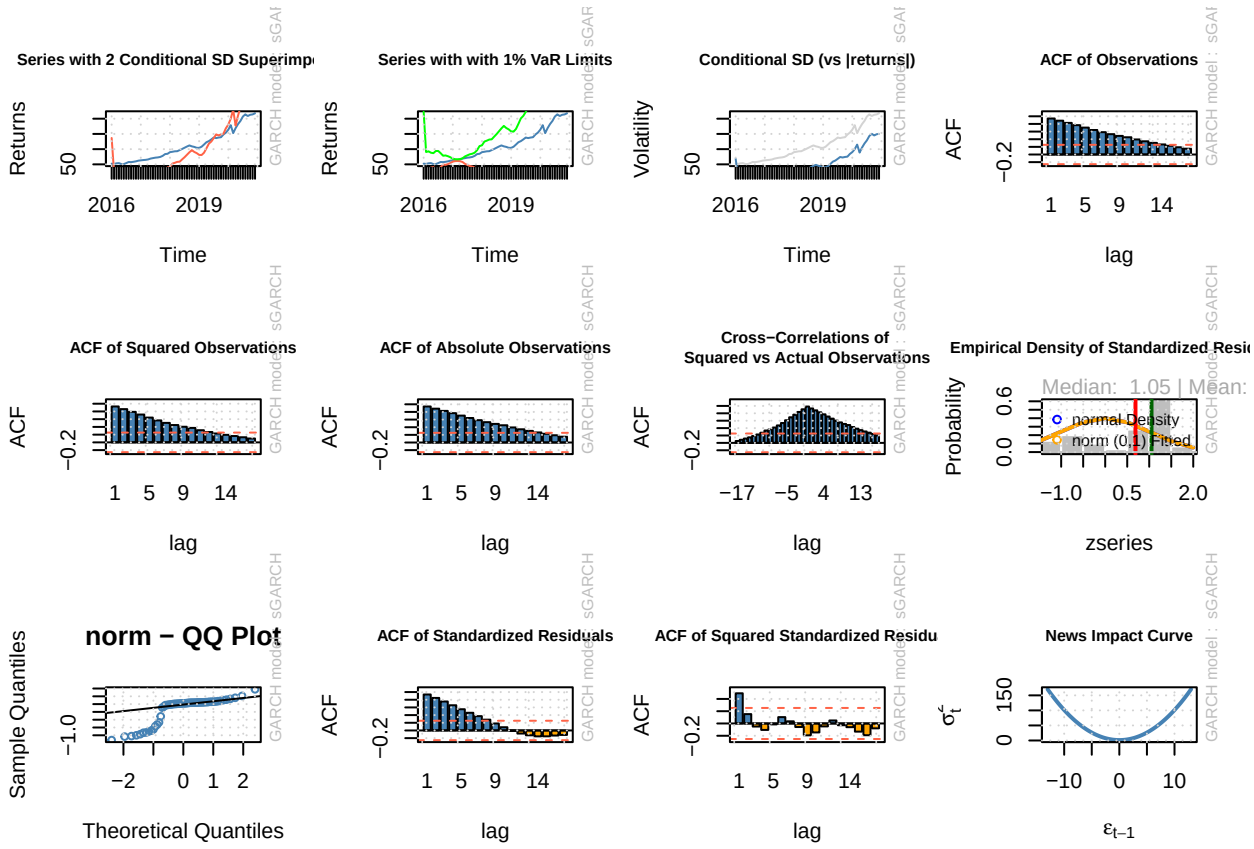
```
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                          statistic   p-value
## Lag[1]                       21.51 3.522e-06
## Lag[2*(p+q)+(p+q)-1][8]      29.13 3.968e-07
## Lag[4*(p+q)+(p+q)-1][14]     36.90 1.321e-07
## d.o.f=3
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[4]     2.734 0.500 2.000 0.09825
## ARCH Lag[6]     5.529 1.461 1.711 0.08622
## ARCH Lag[8]     6.622 2.368 1.583 0.12018
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  4.6905
## Individual Statistics:
## mu     0.56239
## omega  0.02742
## alpha1 0.59527
## beta1  1.21807
## beta2  1.73526
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.28 1.47 1.88
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value      prob sig
## Sign Bias            5.283 2.242e-06 ***
## Negative Sign Bias   2.734 8.397e-03 ***
## Positive Sign Bias   2.205 3.166e-02  **
## Joint Effect        35.392 1.007e-07 ***
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     169.0    4.459e-26
## 2    30     180.0    1.004e-23
## 3    40     191.7    4.866e-22
## 4    50     247.6    2.531e-28
##
##
## Elapsed time : 0.02152395
```

Diagnostics

Now that the models are defined we can evaluate their diagnostics as follows:

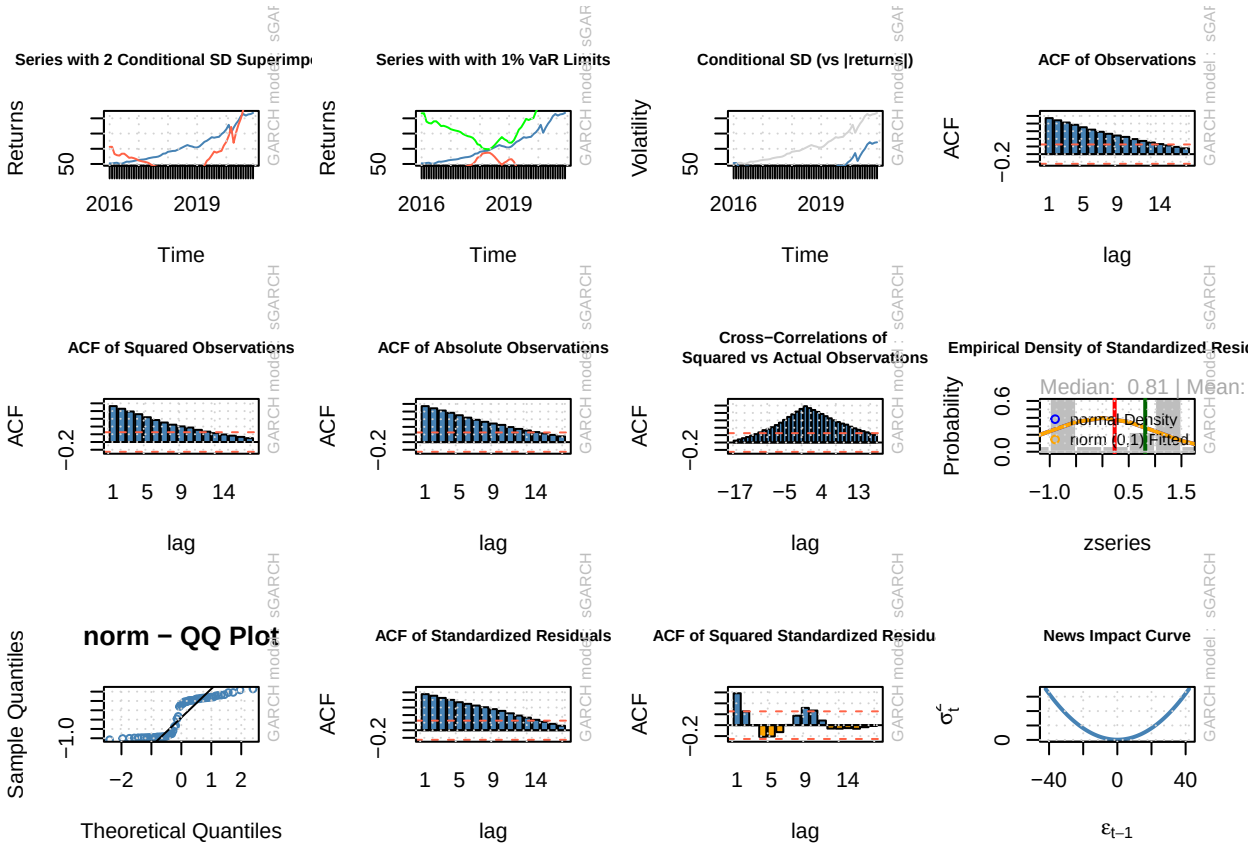
```
plot(msft_garch11_summary, which = 'all')
```

```
##
## please wait...calculating quantiles...
```



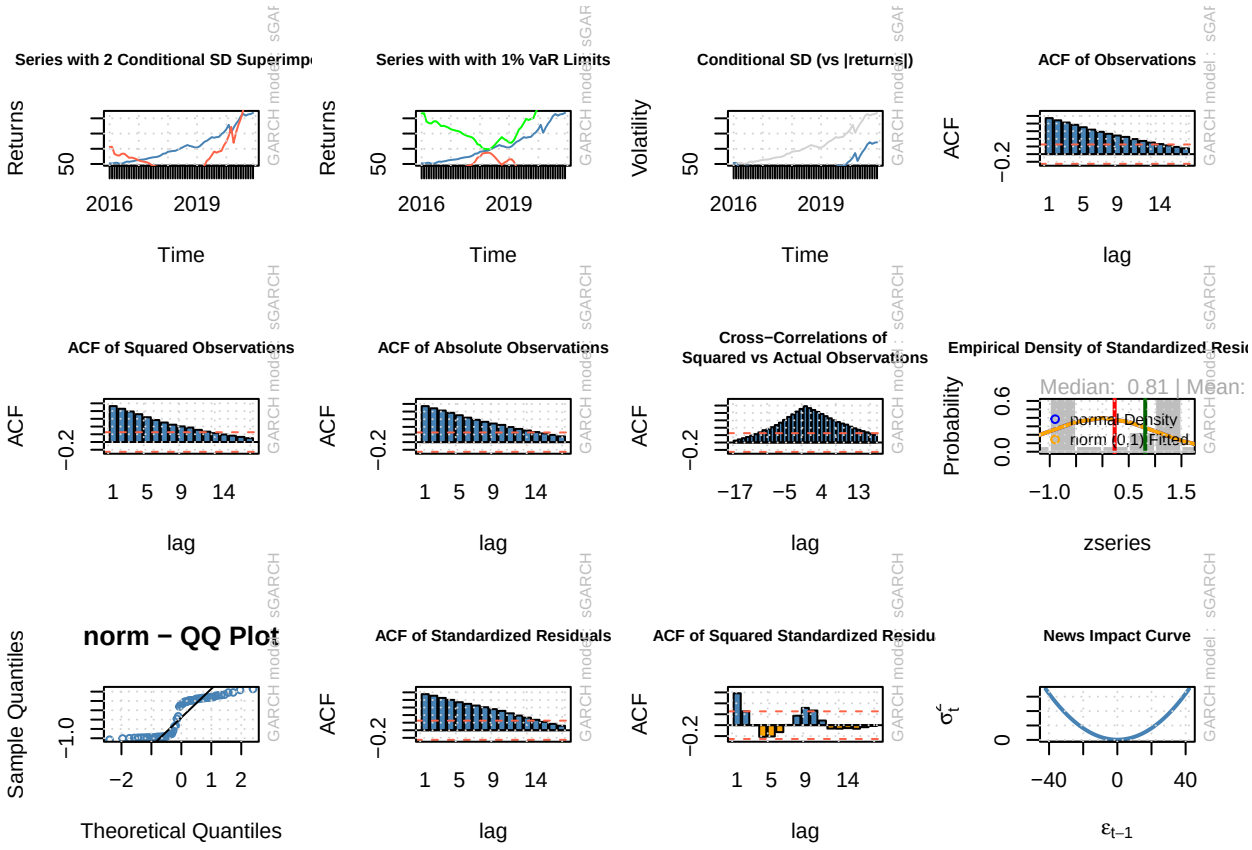

```
plot(msft_garch21_summary, which = 'all')
```

```
##
## please wait...calculating quantiles...
```




```
plot(msft_garch12_summary, which = 'all')
```

```
##
## please wait...calculating quantiles...
```



Model Selection

Looking at our normality (QQ) plots and the Akaline (AIC) values, GARCH(1, 1) appears to be the best fit for our differenced data.

5. Conclusion and Future Study

Conclusion

From the monthly data, we determined that Microsoft's stock prices most closely follow an ARIMA(0, 1, 0) model, combined with a GARCH(1, 1) model to capture volatility. The ARIMA model helps us understand the trend and differencing required to make the series stationary, while the GARCH model captures the volatility clustering commonly observed in financial time series. Our results suggest that this combination is effective in modeling and forecasting Microsoft stock prices on a monthly basis, providing valuable insights into future price movements.

Future Study

For future improvements, several approaches can be taken to enhance the robustness and accuracy of the model:

- Daily Data Usage:

Incorporating all available daily data, rather than aggregating to monthly data, would allow for a more granular analysis. Although this approach would involve more complex data cleaning and setup, it could significantly improve model accuracy by capturing finer details and more frequent seasonal patterns.

- Seasonality Consideration:

With the full set of 1511 daily records, we could better account for seasonality in the data. This would allow for the potential use of a Seasonal ARIMA (SARIMA) model, which could capture seasonal trends and provide a more nuanced understanding of stock price movements.

- Alternative Models:

Exploring alternative modeling techniques could provide new insights and potentially improve forecasting accuracy. For instance, spectral analysis methods could be employed to identify and model periodicities in the data that are not captured by ARIMA and GARCH models. Additionally, other advanced time series models, such as the State Space Model or Long Short-Term Memory (LSTM) neural networks, could be considered for capturing complex patterns in stock prices.

- External Factors:

Incorporating external variables, such as macroeconomic indicators, company financials, or industry trends, could enhance the model by providing additional context and improving its predictive power.

By implementing these improvements, we can achieve a more detailed and accurate understanding of stock price behavior, leading to better forecasting and investment strategies.

6. Appendix and Source Code

All plots used throughout the project have the associated code in line with it.