Time Series Analysis:
Forecasting Tesla with ARIMA

Nader Liddawi

Abstract:

Stock prices are notoriously unpredictable, with some suggesting they move in a random walk with drift. Our study focuses on Time Series Analysis since its techniques come naturally to data ordered by time. The reason to predict market prices is to guide the investor or trader when to enter or exit his or her stock position. Also, one can enter into trading strategies in other assets, like options, that rely on predictable forecast of stock prices during the contract life. We investigated the Tesla weekly stock prices using the ARIMA model of Time Series in order to forecast future values. Tesla came naturally to us since it has become a very popular stock, included in the SP500 index in short order and has a high variance with a highly uncertain future. We plotted the data; normalized it; found dependence orders and parameter estimates; diagnosed the residuals; forecasted the Tesla stock prices using the parameters; and back-tested the model on historical data. We found that if we exclude our model from fitting the forecast data and compare the actual, realized prices with the forecast data, we get a high variance. We suggest future study in modelling with GARCH that accounts for non-constant variance we saw in our original data set, the log data and its residuals.

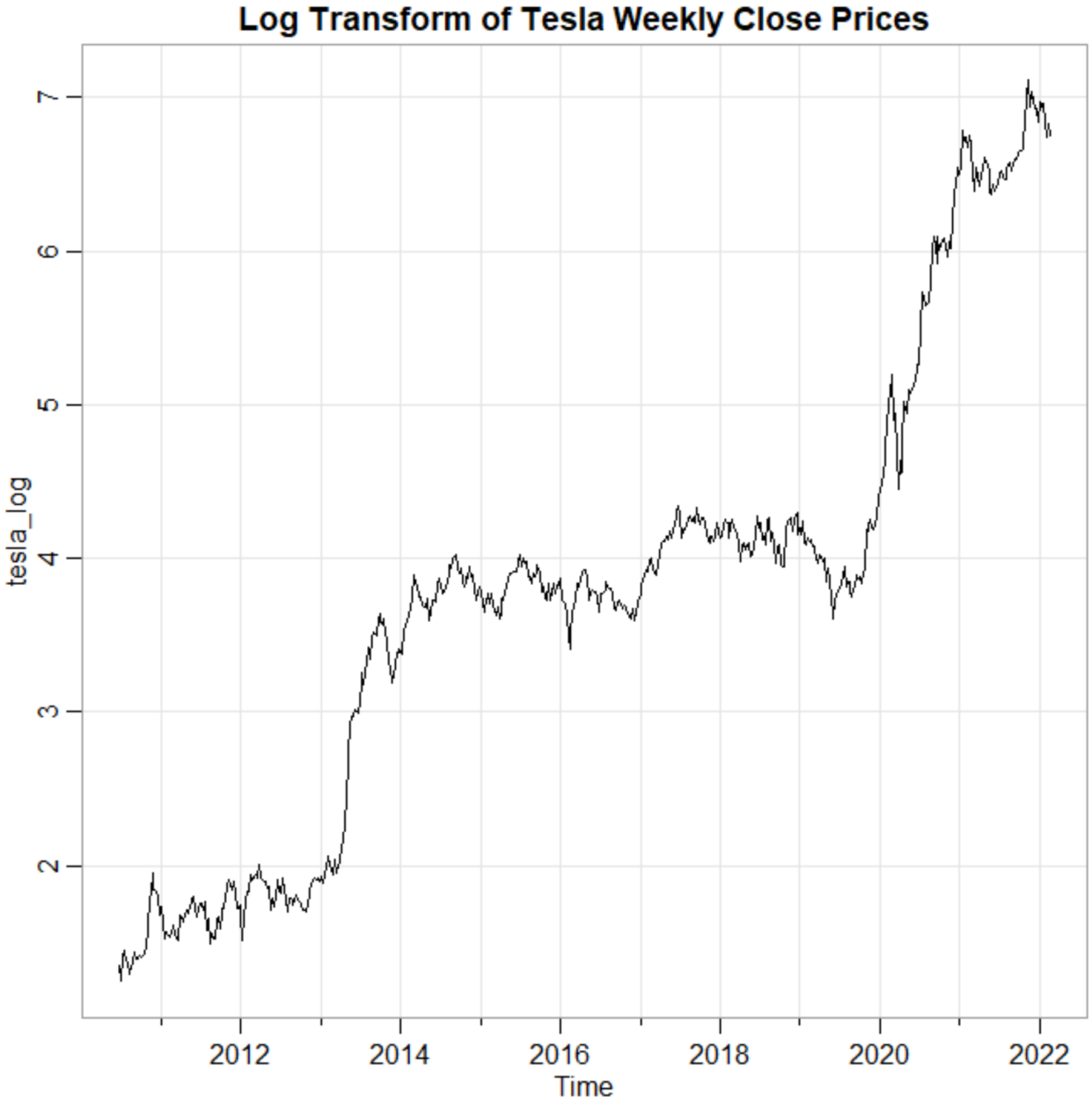
Introduction

The data represents the weekly closing (Friday) prices of Tesla common stock since its market debut in July 2010 thru Feb. 11, 2022.

Model Specification & Model Fitting

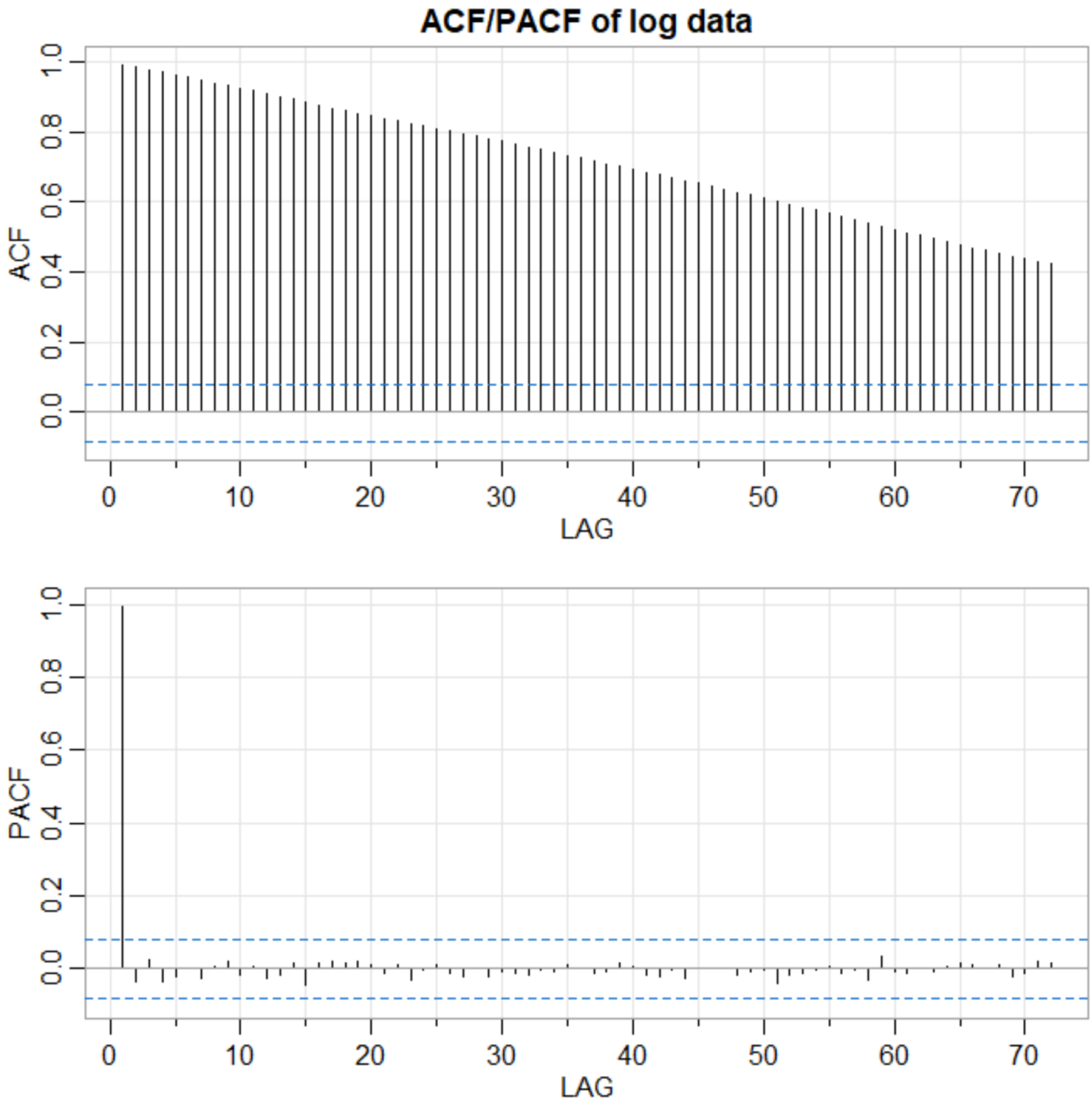
To achieve good predictive power in forecasting future prices, we must ensure that the data has uniform variance. That is, we need to log transform the data such that this normalization technique can bring the data to constant variance with respect to time.

Even after transformation, we do still see that 2019-2022 period have higher variance than the rest of the data set. However, the logarithm did clean up the data decently.



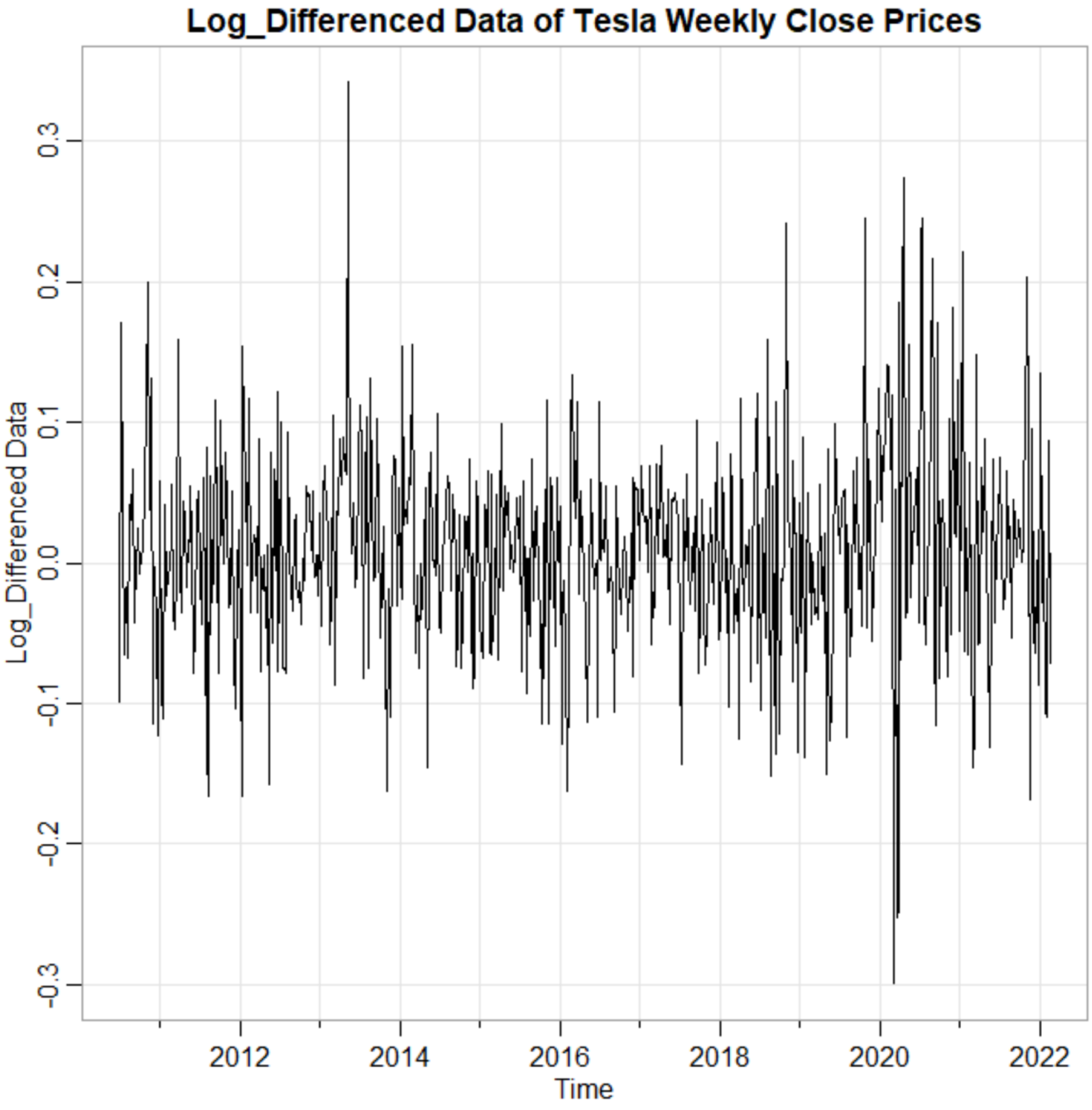
Our log data is also non-stationary since the residual terms are correlated, as shown in the ACF plot. The slowly decreasing sequence of bars need to be eliminated to satisfy stationarity.

And we can eliminate this correlation via the simple Differencing method.



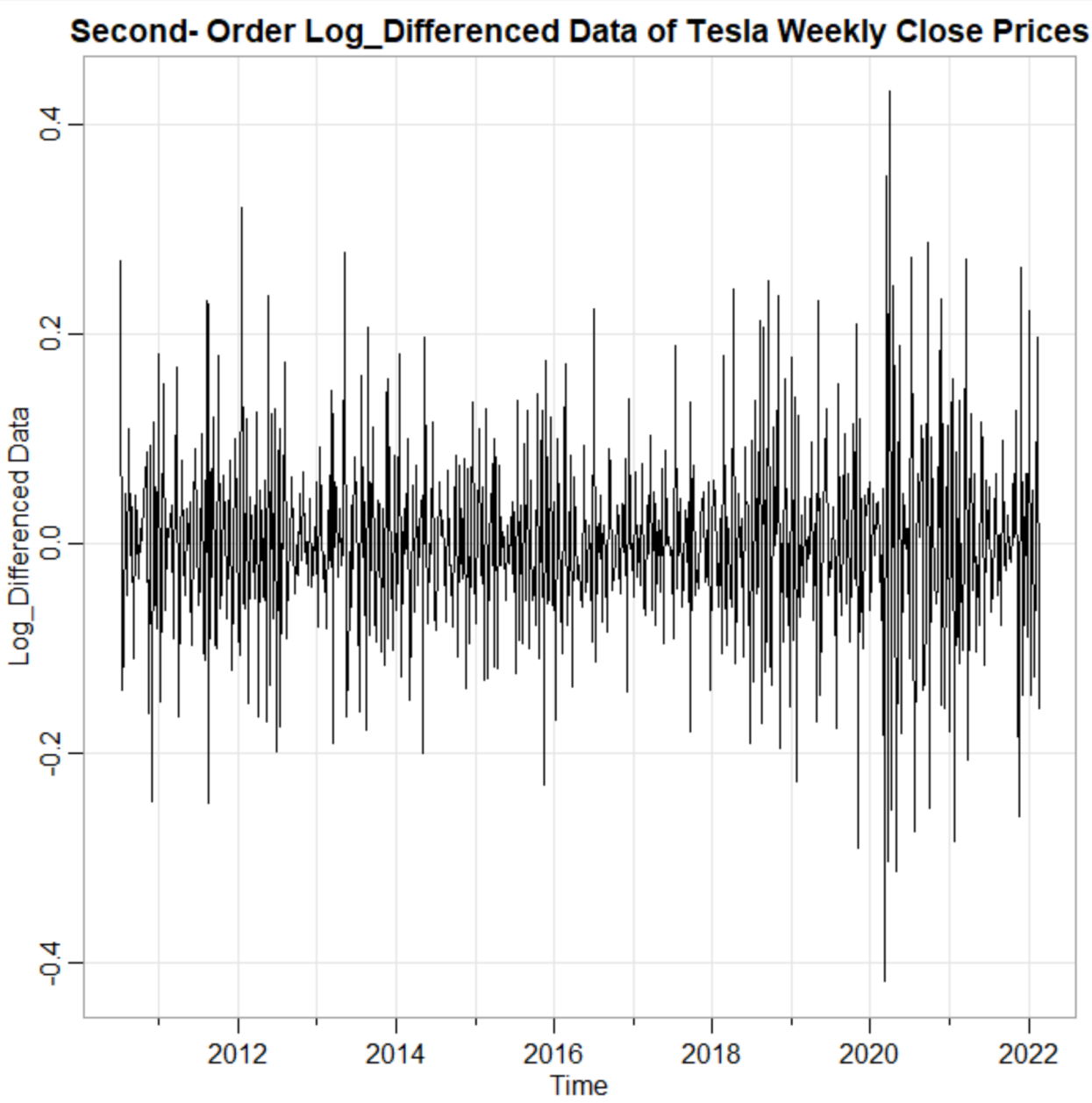
Note: No seasonal, periodic spikes exist in the plots

In order to attain stationarity, we need to eliminate the trend in the price with respect to time. So, we can take the difference of every neighboring two data points, like between x2 and x1. (Xt - Xt-1) can get us to a constant mean, which means no trend and no drift.



Take the second-order differencing to eliminate the residual trend in the first-order-differenced data.

The second-order differencing is simply the difference of the first-order-differenced data.



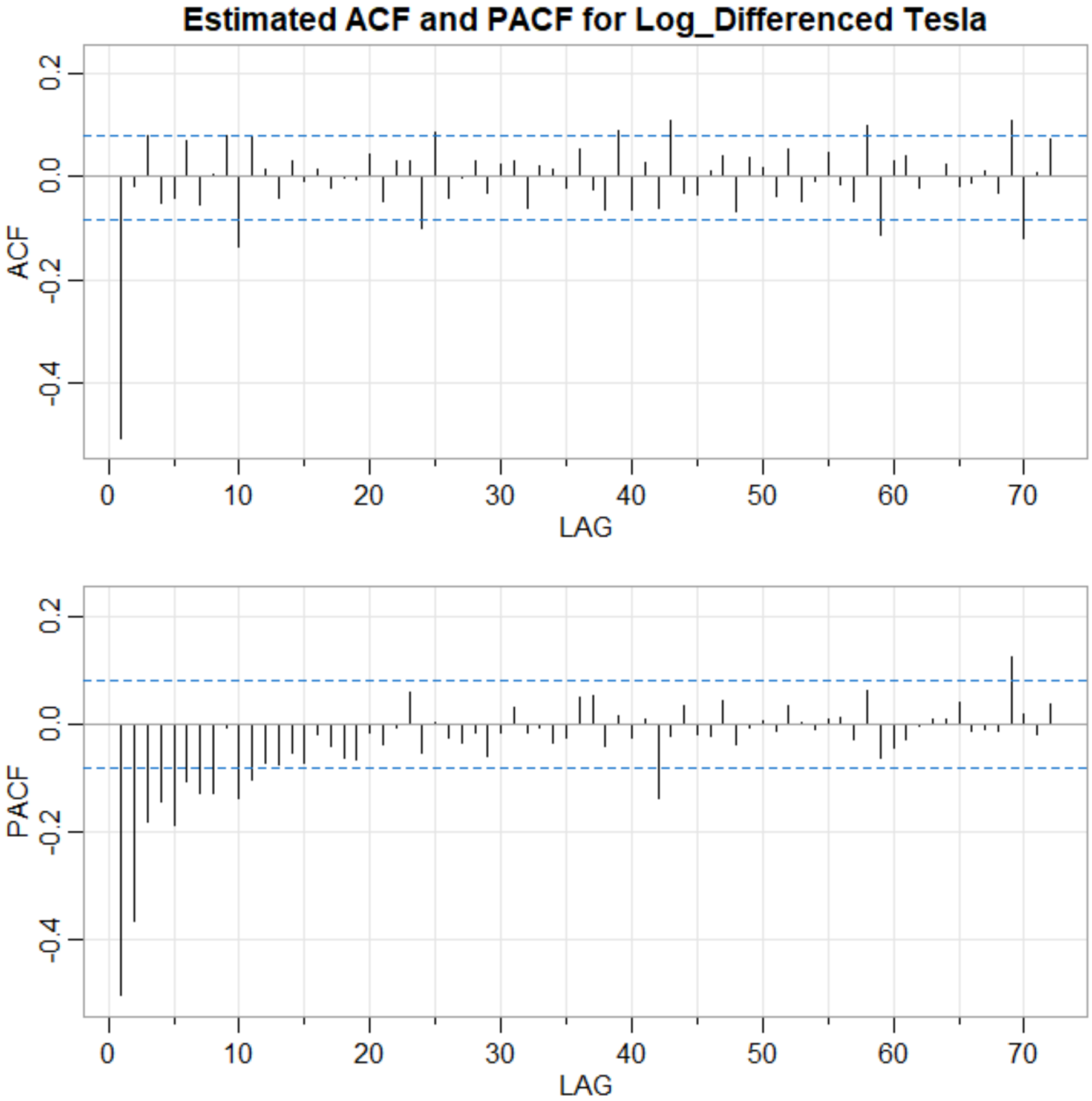
The plot now becomes more jagged, with sharper movements to the upside and downside.

The ACF plot will guide us on what order MA term will be.

We can say that MA order is 1 since bar at lag 1 falls outside of insignificance bounds.

The PACF plot will guide us on what order AR term will be.

We can say that the AR order is 3 since the bar at lag 3 falls outside of the insignificance bounds. We could also have chosen AR order of 10 or 69 but we want parsimonious model.



ARIMA(3,2,1) is our candidate model, since 3rd lag in AR and appeared significant, the fact that we used 2nd order differencing and that the 1st lag in MA appeared significant.

We can derive the parameter estimates via Maximum Likelihood Estimation in R code.

MA parameter is significant to our forecast since p-value < .05, but no AR estimates are significant since p-values > .05.

	Estimate	SE	t.value	p.value
ar1	0.0180	0.0406	0.4426	0.6582
ar2	0.0383	0.0408	0.9389	0.3482
ar3	0.0857	0.0409	2.0975	0.0364
ma1	-1.0000	0.0066	-151.9130	0.0000

Note: Even though AR(3) has p-value <.05, we cannot just eliminate AR(1) and AR(2) but keep AR(3)

Now, let's try ARIMA(2,2,1) as our candidate model and output the parameter estimates derived via Maximum Likelihood Estimation.

	Estimate	SE	t.value	p.value
ar1	0.0205	0.0406	0.5042	0.6143
ar2	0.0428	0.0407	1.0516	0.2934
ma1	-1.0000	0.0070	-143.0457	0.0000

Still, the model has insignificant parameter estimates with p-values > .05. So, we must keep searching for better model that satisfies p-value condition.

Let's try to eliminate the AR terms altogether due to the insignificance of our AR terms in forecasting.

It appears that MA(1) order alone is good with p-value<.05.

In fact, we tried numerous candidate models with different orders and found ARIMA(0,2,1) to be the only significant model.

If we would have found multiple models that satisfy the p-value and residual criteria, then we would have selected the model with the least AIC score, which penalizes for model complexity.

```
     Estimate      SE   t.value p.value
ma1        -1 0.0099 -101.1601       0
```

```
$AIC
[1] -2.344523
```

Residual Diagnostics

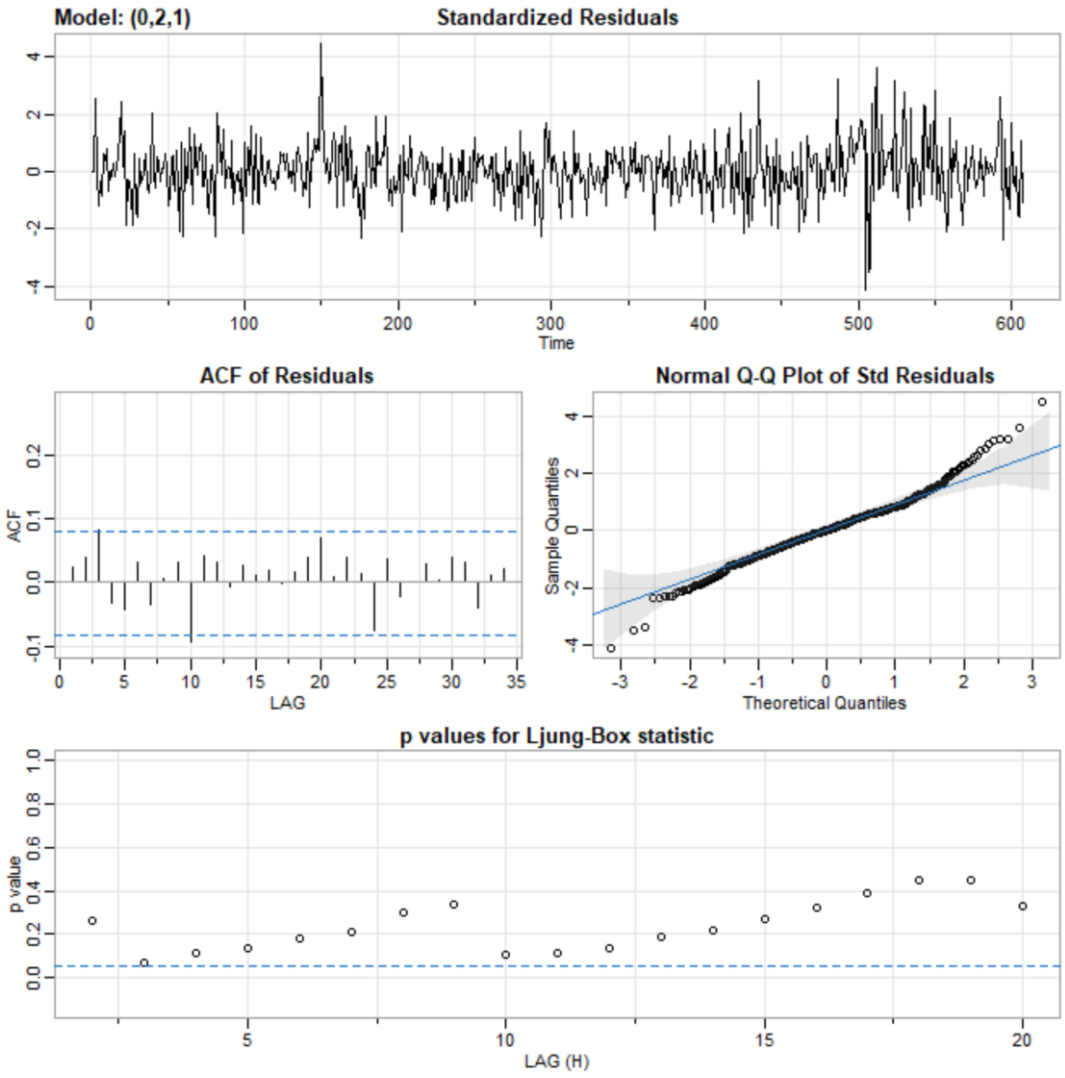
We want to ensure that time series assumptions about error terms are satisfied: Non-correlation and Normality.

Residual non-correlation is satisfied since ACF of Residuals plot has hardly any bars outside the insignificance bounds.

Normality is satisfied since the residual points hug the line, more or less, in the Normal Q-Q plot.

However, there does exist large and non-constant residual variance in the Standardized Residuals plot, especially at time 150 and ~500.

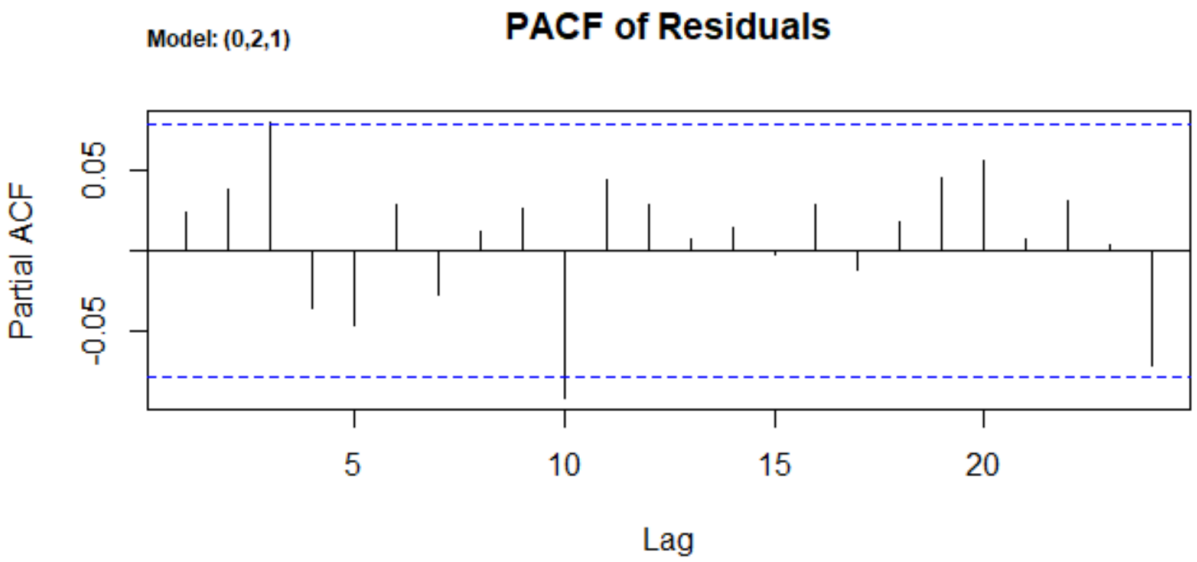
Note: GARCH model can be used in addition to ARIMA to complement and handle for such non-constant variance found in the Standardized Residuals plot.

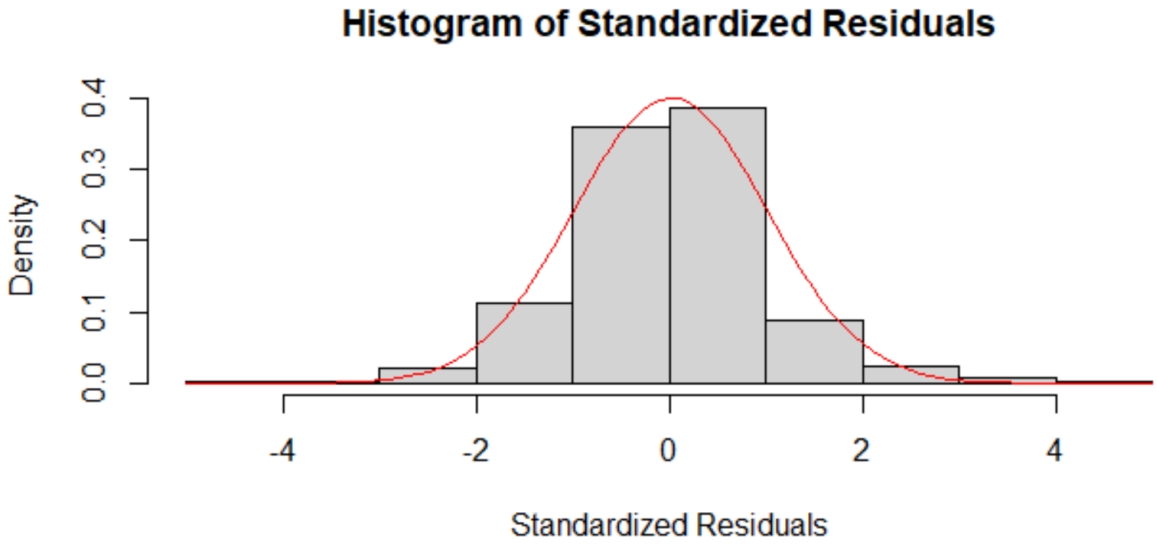


Moreover, PACF of Residuals plot shows hardly any bars outside insignificance bounds, suggesting little residual correlation.

And the Histogram plot forms a bell-shaped curve, suggesting residual normality.

Lastly, the Ljung-Box statistic in the previous page illustrates most points are above the significance level. So, we fail to reject the null hypothesis that there exists no residual correlation.





We have now shown that the ARIMA(0,2,1) model has significant parameter estimates and has no residual assumption violations.

We diagnosed the residuals to ensure that no residual pattern exists that can be explained away by some phenomena other than irreducible white noise.

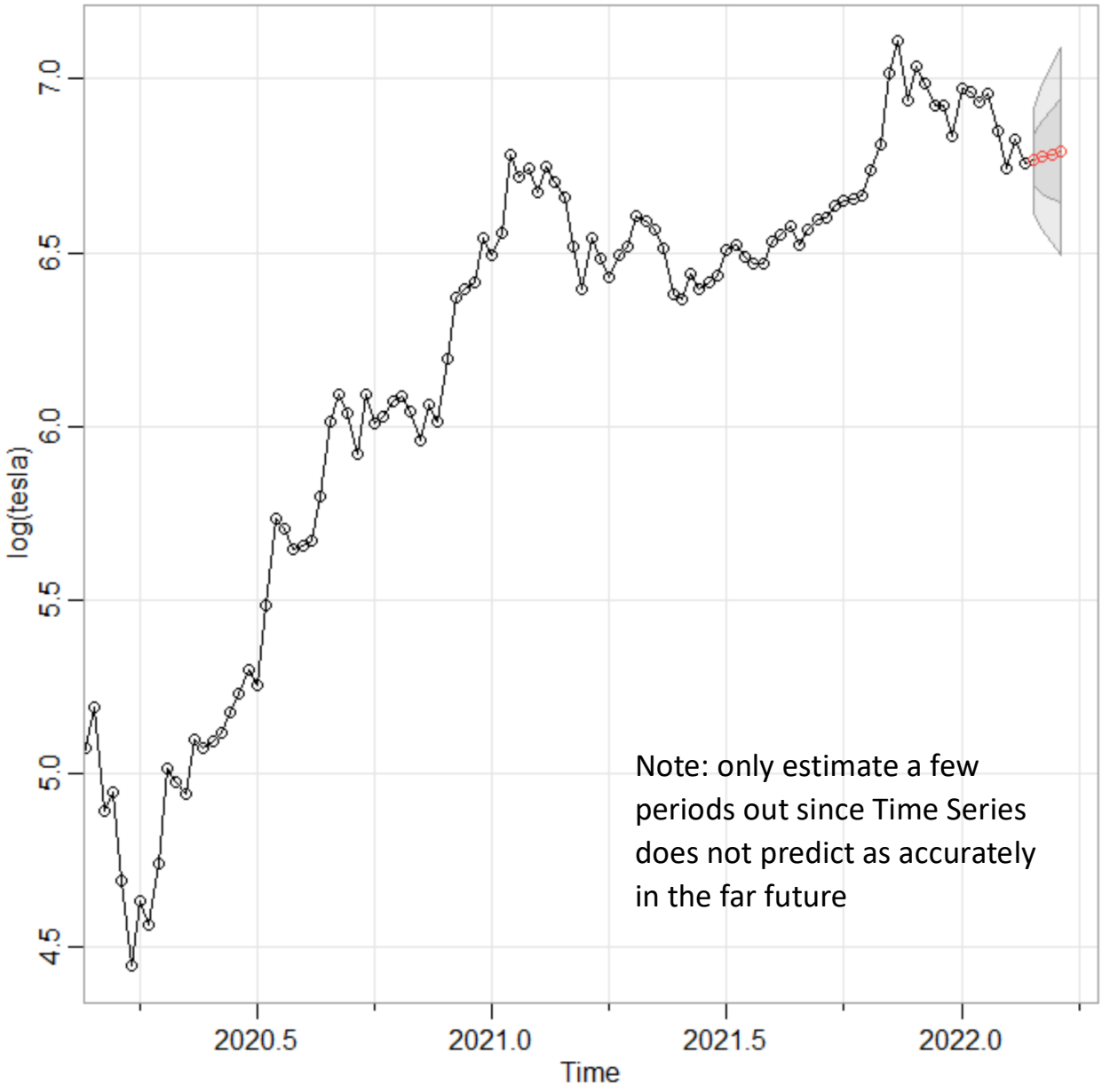
We are now ready to forecast our log data using the model parameters we found.

Forecasting

We forecast 4 weeks out the log data of weekly Tesla stock price.

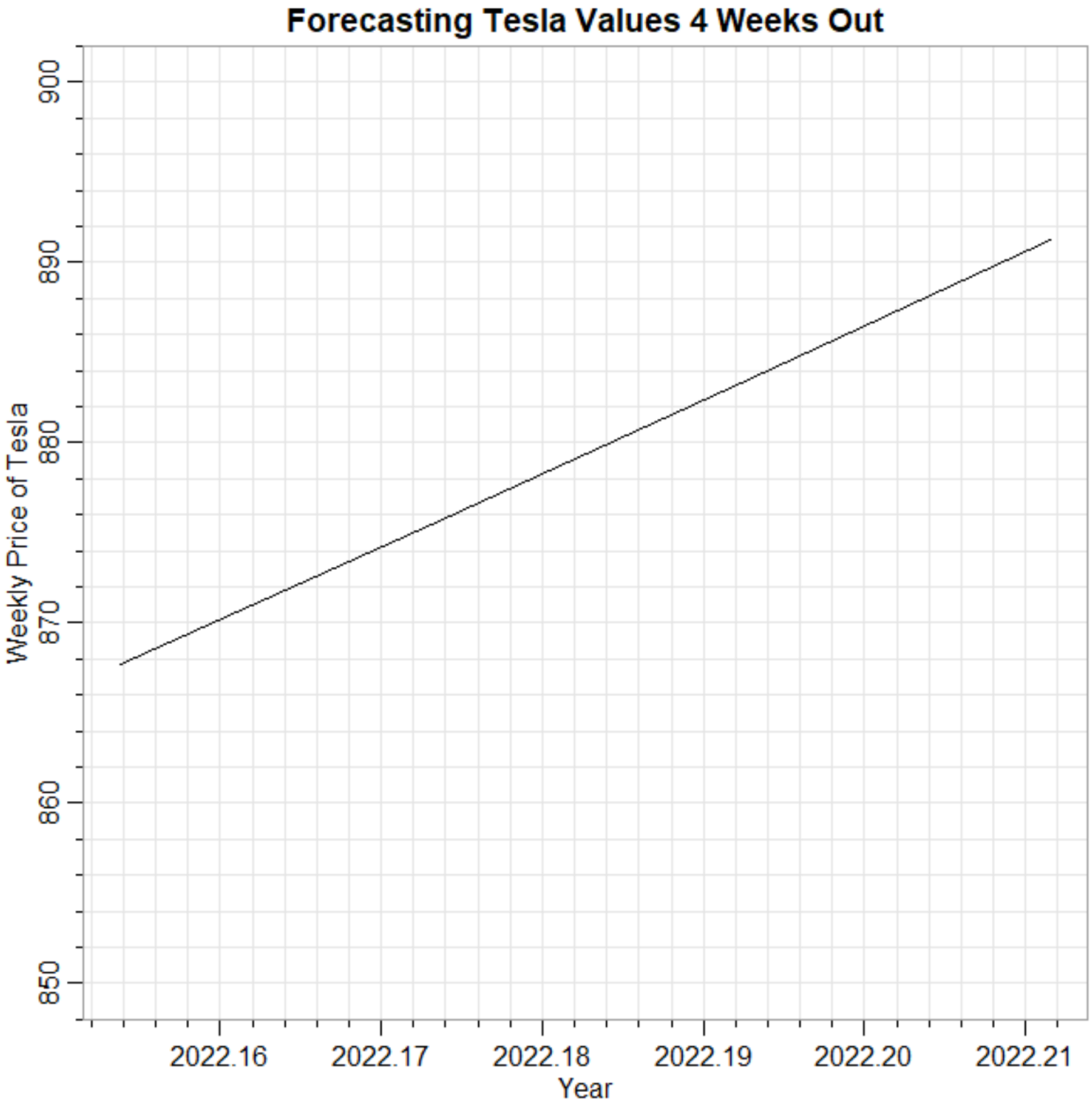
The inner, dark band represents the Confidence Interval, where the mean is expected to lie with 95% confidence.

And the outer, light band represents the Prediction Interval, where any data point (which is more volatile than the mean) is expected to lie with 95% confidence.



We need to transform back the data to get our real, raw weekly price forecast.

We can simply take the opposite or inverse of the logarithm: the exponential.



Our weekly forecasted prices four weeks into the future from Feb 11, 2022:

Forecast	Feb. 18	Feb. 25	Mar. 4	Mar. 11
Upper Confidence Interval	869.83	877.67	885.58	893.55
Price Estimate	867.71	875.50	883.35	891.27
Lower Confidence Interval	865.60	873.32	881.12	889.00

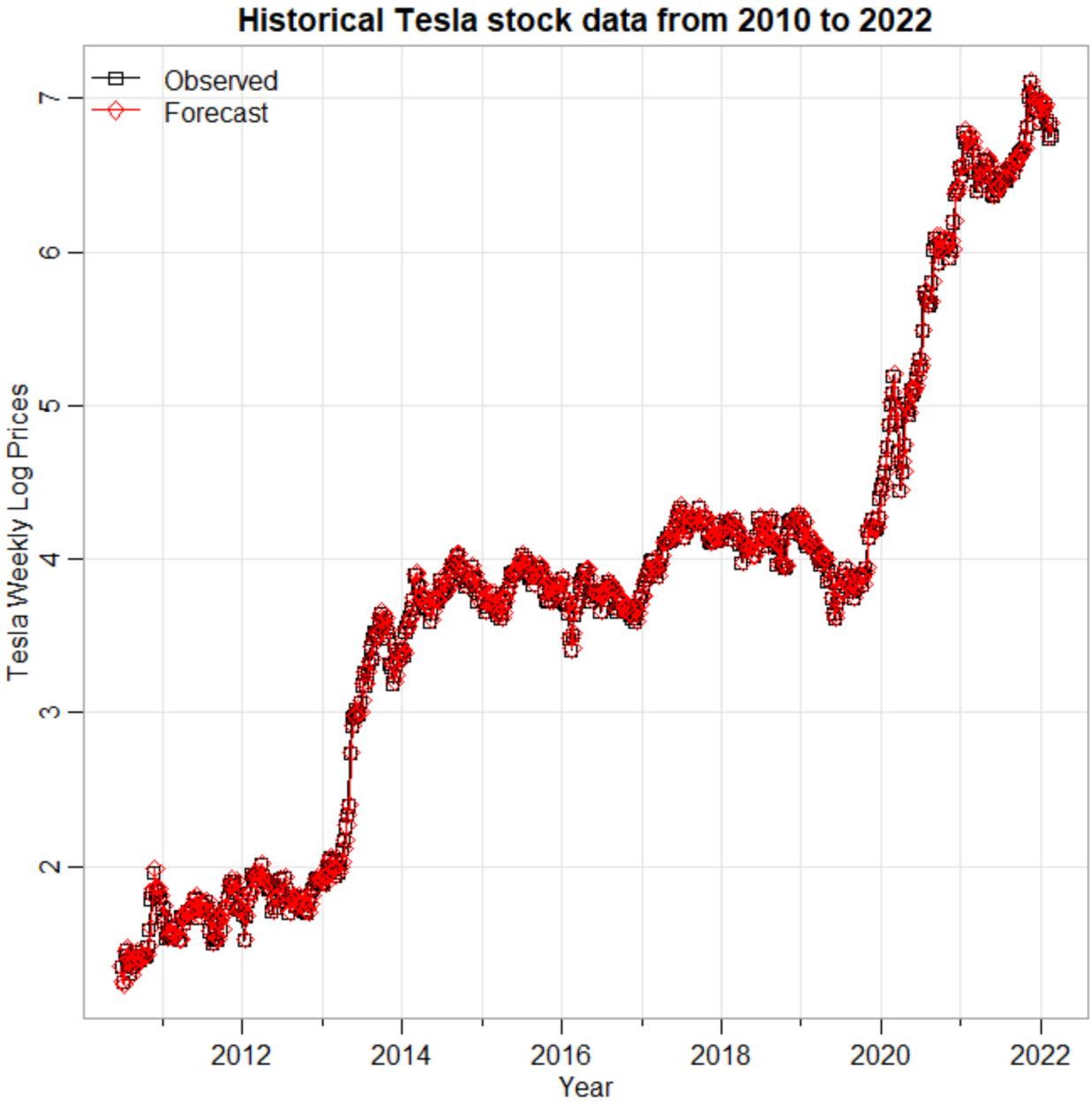
Above CI intervals are calculated thusly: estimated value +/- (standard error * z-score)

We want to see how well our model would have performed were it to have forecasted our historical observations.

This plot shows the error rate of the prediction our model estimates relative to the true historical weekly values of the Tesla stock price.

All in all, the model did a pretty good job as the forecast prices are superimposed neatly on the observed prices.

Our model's prediction has a Root Mean Squared Error of 0.07.



Discussion

To recap our Times Series Analysis, we plotted the weekly Tesla stock price data; normalized it via log transformation; took the second-order differencing to detrend the data; identified the dependence orders of the model by looking at the bars on the ACF and PACF plots; estimated the parameters (phi and theta) via Maximum Likelihood Estimation; diagnosed the residuals to meet our normality and non-correlation assumptions; chose the model that passed residual assumptions and whose parameter estimates had p-value < 0.05; forecasted the Tesla stock prices four weeks out and back-tested to see how our model would have done on the historical data.

Even though our back-testing Root Mean Squared Error was only 0.07, our RMSE for future predictions (that our model did not train on) relative to actual prices was larger, at 62.56. The larger RMSE can be owed to the fact that the markets have entered into greater correction territory. However, there have been multiple downward corrections during our historical data set. So, this leads us to believe that our prediction via back-testing has been over-fit to the data.

Forecast	Feb. 18	Feb. 25	Mar. 4	Mar. 11
Upper Confidence Interval	869.83	877.67	885.58	893.55
Price Estimate	867.71	875.50	883.35	891.27
Lower Confidence Interval	865.60	873.32	881.12	889.00

Actual	Feb. 18	Feb. 25	Mar. 4	Mar. 11
Real Price	856.98	809.87	838.29	795.35

Forecast - Actual	10.73	65.63	45.06	95.92

Root Mean Squared Error
62.56

Note: Feb. 11 was the last data point we used to fit the time series model. Feb. 18 to March 11 was the predicted four-week period. We simply compared the predicted values with the actual, realized values once they became available (i.e., once the future dates became the past)

When we train the Time Series model on these forecast values and back-test, we still get 0.07. But importantly, we need to compare our forecast from a model that did not train on the forecast values to see how well our model is accurate against these forecast values (i.e., to avoid over-fitting). Even when we take into consideration the confidence intervals, our actual prices (see in the previous page) did not fall within the confidence interval boundaries of our price estimates.

Because of these less-than-perfect results, we can use ARIMA in a collection of different models (such as linear regression model) to get a better sense of future values. Moreover, we can use the GARCH model that accounts for higher volatility and non-constant volatility that we saw in our original data set (page 2), log data (page 3) and residual diagnostics (page 10). With future attempts at GARCH, we hope to find a more predictive model to meet our forecasting needs.

Bibliography

https://finance.yahoo.com/quote/TSLA/history?p=TSLA

Shumway, Robert. Time Series Analysis and Its Applications: With R Examples. Springer, 2017.

Appendix: R Code

```r
library(astsa)
library(Metrics)

#read files and standardize frequency
data <- read.csv(file.choose(), header=TRUE)

# use (2010,26) because June 28 is 26th week of 2010
# use (2022, 8) since Feb 7 is 8th week of 2022
# these closing prices are not June 28 or Feb 7, but that week's Friday close; Yahoo F

tesla <- ts(data$Adj.Close, frequency = 52, start=c(2010,26), end=c(2022,8))
x <- as.numeric(tesla)

#plot original weekly data
dev.new(10,6)
tsplot(tesla, main='Original Tesla Weekly Close Prices', ylab='Weekly Closing Prices')


#log transform the data to eliminate non-constant variance in the original data
dev.new(10,6)
tesla_log <- log(tesla)
tsplot(tesla_log, main='Log Transform of Tesla Weekly Close Prices')
```

```r
#seasonality does NOT exist since there are no periodic spikes
# and we still have a lot of exponentially declining bars because we still have correlated terms (non-stationary)
dev.new(10,6)
acf2(log(x), max.lag=72, main='ACF/PACF of log data')


#take difference to reduce trend
dev.new(10,6)
tsplot(diff((tesla_log)), main='Log_Differenced Data of Tesla Weekly Close Prices', ylab='Log_Differenced Data')



#take double difference to further reduce trend/variance of log data (and achieve stationarity)
dev.new(10,6)
tsplot(diff(tesla_log, differences=2), main='Second- Order Log_Differenced Data of Tesla Weekly Close Prices', ylab='Log_Differenced


#the exponential decline in ACF/PACF has been eliminated
dev.new(15,15)
acf2(diff(log(x), differences=2), max.lag=72, main='Estimated ACF and PACF for Log_Differenced Tesla')
```

```r
# check p-value and residual assumptions
source("examine.mod.R")
dev.new(10,6)
bad.model <- sarima(log(x), p=3, d=2, q=1)
bad.model
examine.mod(candidate.model, 3, 2, 1)
# bad model


source("examine.mod.R")
dev.new(10,6)
candidate.model <- sarima(log(x), p=0, d=2, q=1)
candidate.model
examine.mod(candidate.model, 0, 2, 1)
# good model
#note: I got no good model while I used just first-order differencing


# you need to forecast with log data
dev.new(10,6)
forecast <- sarima.for(log(tesla), n.ahead=4, p=0, d=2, q=1, plot.all=FALSE)
forecast
```

```r
# take exponential to get real data back
exp(forecast$pred)
exp(forecast$se)


# plot the real forecast telsa prices four weeks into the future
dev.new(10,6)
tsplot(exp(forecast$pred), ylab='Weekly Price of Tesla', ylim = c(850,900),
       xlab='Year', main="Forecasting Tesla Values 4 Weeks Out")

#buy put option at 883 (green) strike and sell put option at 888 (red) strike
abline(h=c(883,888), col=c("green", "red"), lty="solid")

#buy call option at 837 (green) strike and sell call option at 832 (red) strike
abline(h=c(899,894), col=c("green", "red"), lty="solid")

#plot the 95% confidence intervals
abline(h=c(889.00, 893.55), col="darkorchid", lty="dashed")

# add legend to understand the lines
legend("bottomleft", legend=c("Buy Line", "Sell Line", "Confidence Band"), col=c("green", "red", "darkorchid"),
       lty=c("solid", "solid", "dashed"),bty=0)
```

```r

# see how model predicts previous values
prediction.model <- ts(log(x) - candidate.model$fit$residuals, frequency=52, start=c(2010,26), end=c(2022,8))


# plot how well our model did compared to true historical values
tesla <- ts(data$Adj.Close, frequency = 52, start=c(2010,26), end=c(2022,8))
dev.new(width=8, height=6)
tsplot(log(tesla), col='black', ylab="Tesla Weekly Log Prices", xlab="Year", type="o", pch=0,
       main="Historical Tesla stock data from 2010 to 2022", lty='solid')

lines((prediction.model), col="red", type="o", pch=5)
legend("topleft", legend=c("Observed", "Forecast"), lty=c("solid", "solid"), col=c("black", "red"), pch=c(0, 5), bty="n")

rmse(log(x), prediction.model)
```