# Time Series Analysis and Forecasting of Amazon Stock Prices

By Prottush Das

MSc Statistics

# 1. Introduction

Time series analysis is a powerful tool used in forecasting future values based on previously observed values. It is particularly useful in financial markets, where it helps in analyzing stock price movements and predicting future prices. This project focuses on the time series analysis and forecasting of Amazon's stock prices (AMZN) using historical data from January 1, 2015, to January 1, 2019. The analysis involves various steps, including data collection, transformation, model fitting, and evaluation.

The primary objectives of this project are:

- To fetch and pre-process historical stock price data for Amazon (AMZN).
- To perform exploratory data analysis (EDA) on the stock prices.
- To apply time series decomposition techniques to understand the components of the stock prices.
- To perform transformations such as logarithmic and square root transformations to stabilize variance.
- To develop an ARIMA model for forecasting future stock prices.
- To evaluate the accuracy of the model by comparing the forecasted values with the actual stock prices.

# 2. Data Collection

## 2.1 Data Source

The stock price data for Amazon (AMZN) was retrieved from Yahoo Finance using the `quantmod` package in R. The data was collected for the period from January 1, 2015, to January 1, 2019. The closing prices were used for the analysis.

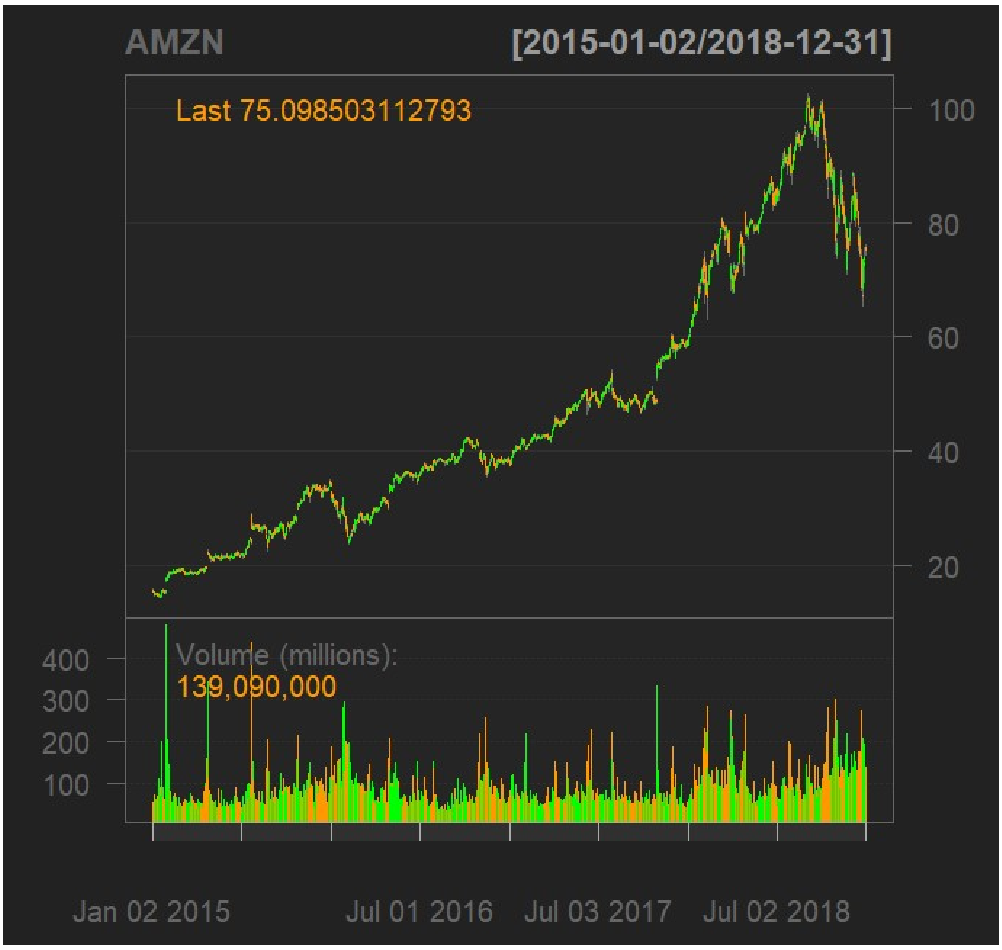
## 2.2 Data Preprocessing

The data retrieved was processed to remove any missing values using the `na.omit` function. The closing prices were extracted as the primary variable for analysis.

# 3. Exploratory Data Analysis (EDA)

## Time Series Chart

A time series chart of the stock prices was plotted using the `chartSeries` function to visualize the historical price movements of Amazon stock. The chart provided insights into the trend and seasonality in the stock prices over the selected period.



For this analysis, we are isolate weekly close prices as the benchmark for future predictions, as those values are assumed to best reflect the changes in real values of the stock during the time frame.

## Observations From Examining The Graph

Upon briefly looking at the data, we can see that the stock value increased over time since 2015, but spiked during 2018. There is no apparent pattern that can be used to scale the value of the stock price because the trends are not linear, and there is no mathematical formula to describe the change in curve or the fluctuations between increasing or decreasing prices.

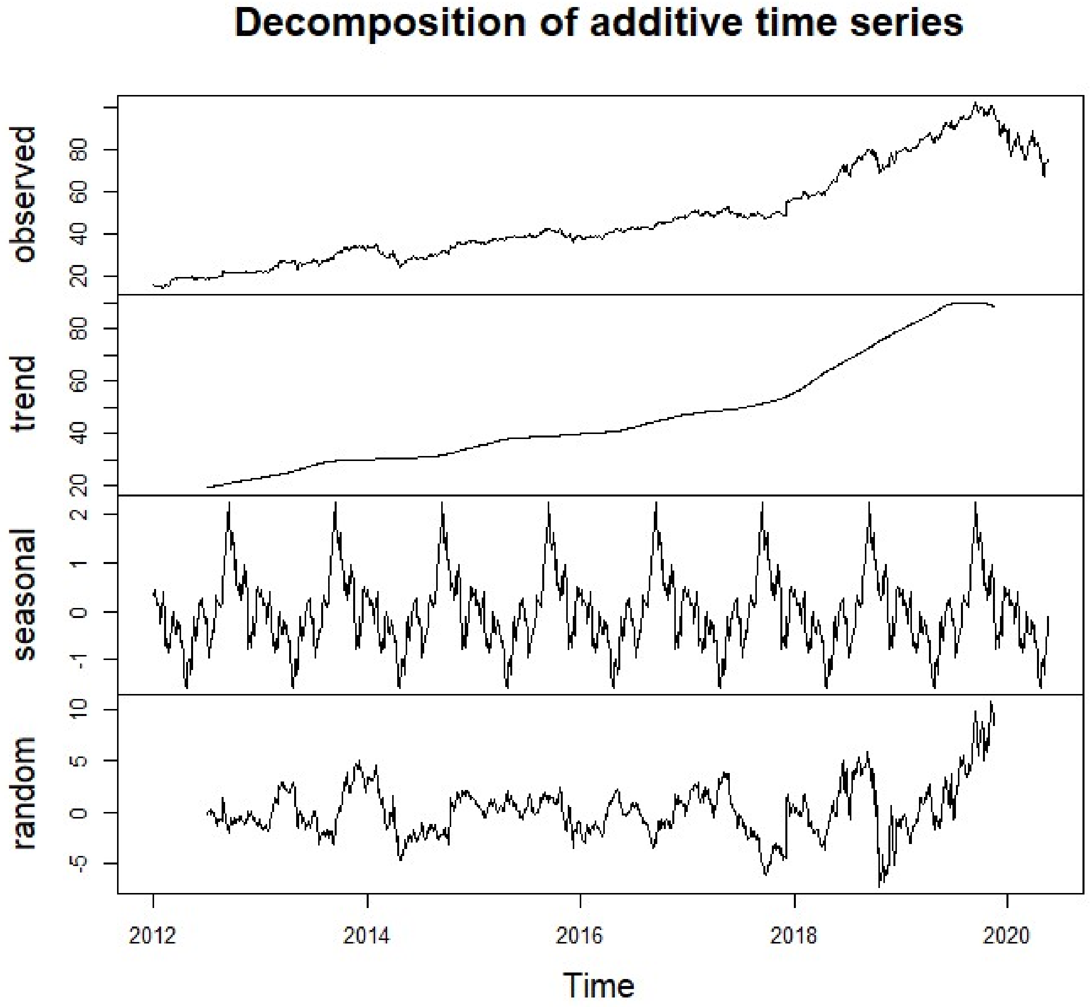
As the stock price spikes in 2018, the variance between datapoints also appears to increase, with stock prices being especially volatile at the most recent dates.

## Time Series Decomposition

The time series data was decomposed into its three components:

- **Trend**: The overall direction in which the stock prices are moving.
- **Seasonality**: The repeating short-term cycle in the stock prices.
- **Residual**: The remaining fluctuations in the stock prices after removing trend and seasonality.

The decomposition was performed using the `decompose` function in R, and the components were plotted for visualization.



- The **trend component** describes the overall pattern of the series over the entire range of time, taking into account increases and decreases in prices together. From the plot above, the trend is overall increasing.

- The **seasonal component** describes the fluctuations in stock price based on the calendar or fiscal year. From the plot above, the peak in stock price occurs every year at Q4 (July, August, and September) and the trough in stock price occurs every year at Q2 (January, February, and March), with clear oscillating fluctuations in between.

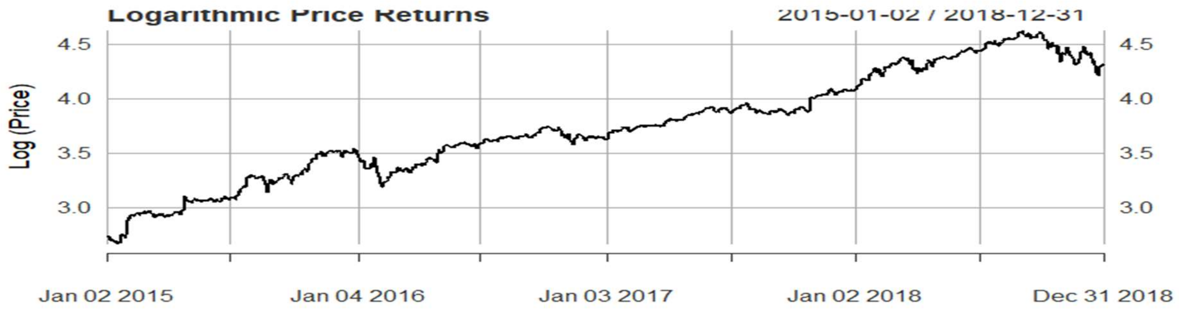
- The **random (residual) error, or noise** section describes the trends that cannot be explained by trend or seasonal components. Statistically, these errors are the difference between the observed price and the estimated price. Random error is particularly important for this project because a statistical model can only be fit if the residuals are independent and independently distributed. One notable observation is that from mid-2018 to 2020, the plot shows more fluctuation, meaning there is greater variance and greater statistical error. This means more recent and future points become more unpredictable, as shown in the spike in stock price from plots.

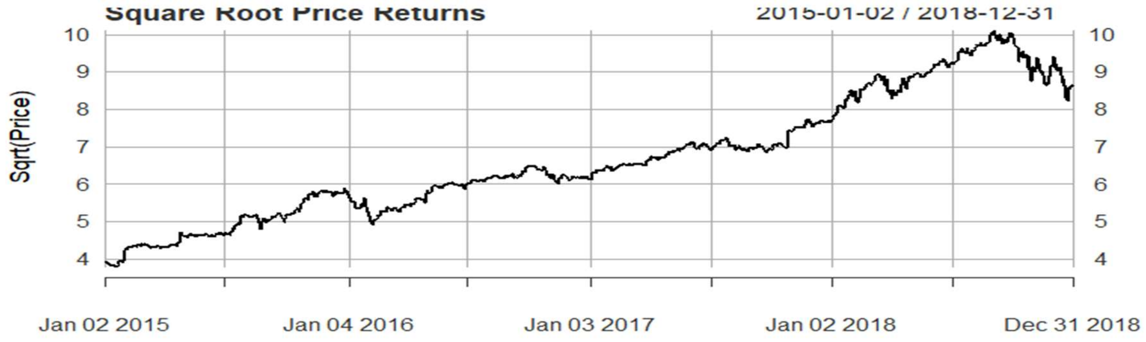
# 4. Smoothing the Data

## Logarithmic and Square Root Transformations

To stabilize the variance and make the data more suitable for modelling, two types of transformations were applied:

- **Logarithmic Transformation**: The natural logarithm of the stock prices was taken to stabilize the variance over time. There are many reasons to use **logarithmic returns**, but in short, the fluctuations in prices transformed into returns can be better compared over time and used to describe trends. The result is a smoothed curve with reduced variation in the time series, so a forecasting model can fit more accurately.
- **Square Root Transformation**: The square root of the stock prices was calculated as an alternative variance-stabilizing transformation. **Square root values** instead of raw prices are used to scale the volatility between points to manage the time horizon of the stock. This is especially important because the longer a position is held, the greater a potential loss can be found.





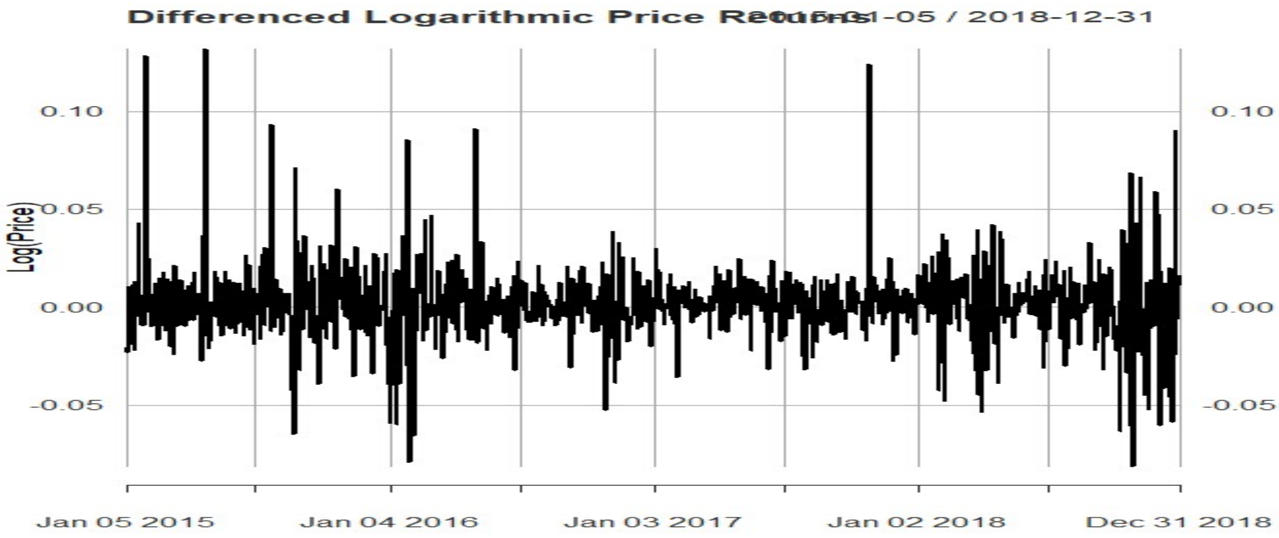
## Differencing

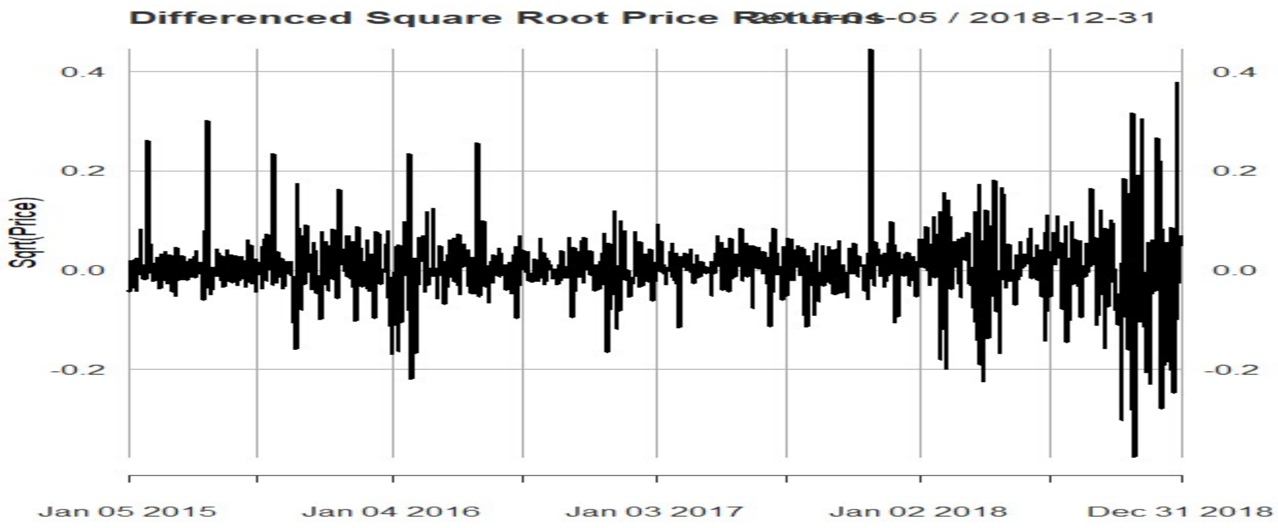
To remove non-stationarity from the transformed data, the first difference of the logarithmic and square root transformations was computed. The differenced series was plotted to visualize the stationary behaviour of the transformed data. In this project, we only difference once. Differencing to the first order does the following:

- Statistical properties, such as mean, variance and autocorrelation, are constant over time.
- Linear properties, such as y-intercept and slope, are constant over time.

Again, viewing the y-axis of the plots, the transformed data is now oscillating around 0. The deviations from 0 are much smaller than in the previous plots, meaning the data has been smoothed more. The fluctuations from these plots should be best for describing trends in the value of the stock. Confirming the observations in Step 2, most major fluctuations are found in 2018, which was when the stock returns (and price) changed the most.

If there were still fluctuations in the data, differencing a second time, or to the second order, would smooth the data further by accounting for the quadratic (curve) differences between datapoints.



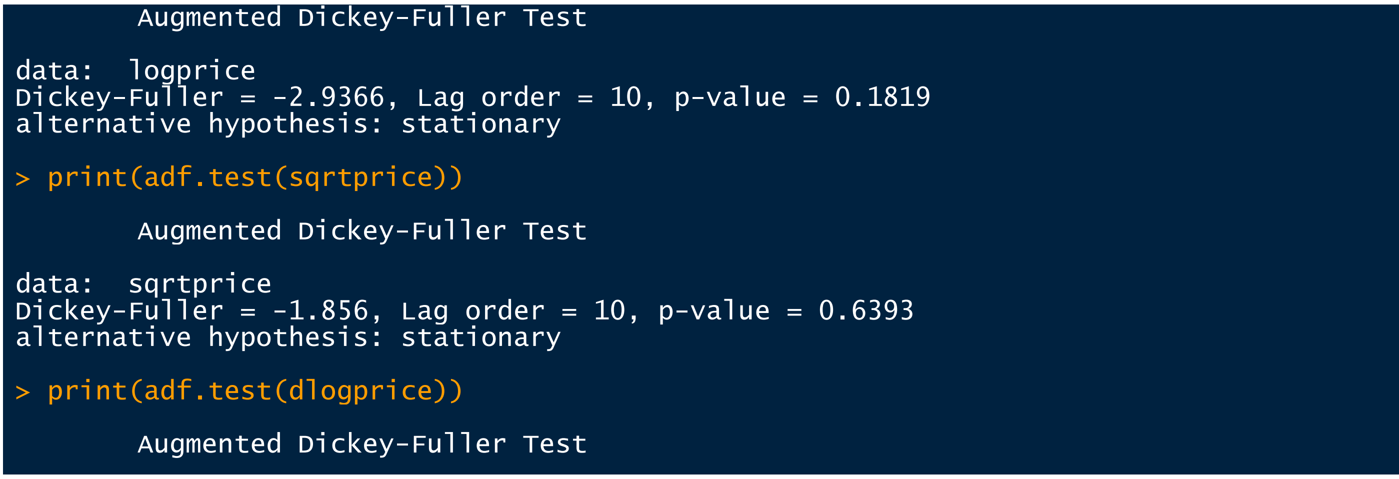


# 5. Stationarity Testing

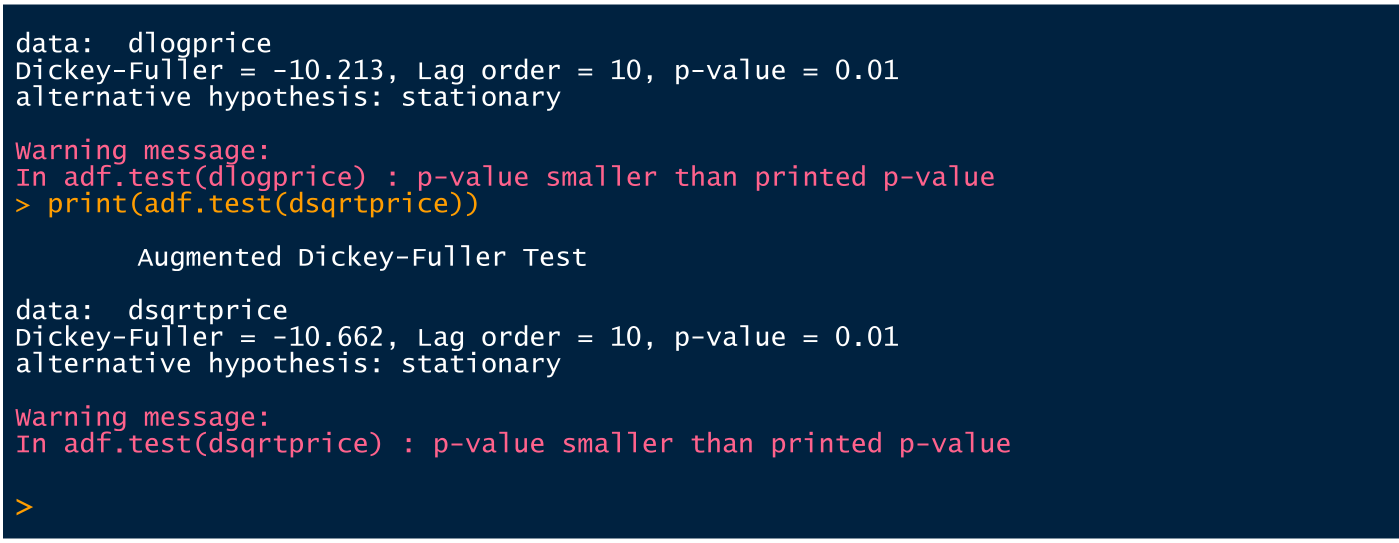
## Augmented Dickey-Fuller (ADF) Test

The Augmented Dickey-Fuller (ADF) test was conducted on the original, transformed, and differenced series to check for stationarity. Stationarity is a crucial requirement for time series forecasting models like ARIMA. The ADF test results indicated whether the series was stationary or required further differencing.

- **Ho:** The time-series data includes a unit root, and is non-stationary. The mean of the data will change over time.
- **Ha:** The time-series data does not include a unit root, and is stationary. The mean of the data will not change over time.



```
        Augmented Dickey-Fuller Test

data:  logprice
Dickey-Fuller = -2.9366, Lag order = 10, p-value = 0.1819
alternative hypothesis: stationary

> print(adf.test(sqrtprice))

        Augmented Dickey-Fuller Test

data:  sqrtprice
Dickey-Fuller = -1.856, Lag order = 10, p-value = 0.6393
alternative hypothesis: stationary

> print(adf.test(dlogprice))

        Augmented Dickey-Fuller Test
```



```
data:  dlogprice
Dickey-Fuller = -10.213, Lag order = 10, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(dlogprice) : p-value smaller than printed p-value
> print(adf.test(dsqrtprice))

        Augmented Dickey-Fuller Test

data:  dsqrtprice
Dickey-Fuller = -10.662, Lag order = 10, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(dsqrtprice) : p-value smaller than printed p-value

>
```

The results here show what types of data would be appropriate for an ARIMA model. The **Dickey-Fuller statistic** shows that the more negative the number, the stronger the null hypothesis rejection, meaning there is no unit root. Similarly with any hypothesis test, the small **p-value** means there is strong evidence against the null hypothesis, meaning there is no unit root.

As such, to properly use an ARIMA model, we should use the differenced logarithmic returns and square root values only.

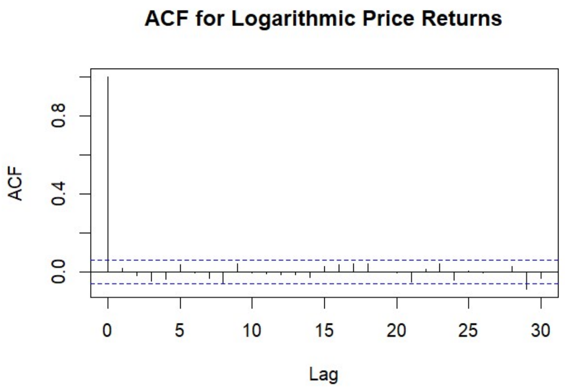
## 6. Creating Correlograms

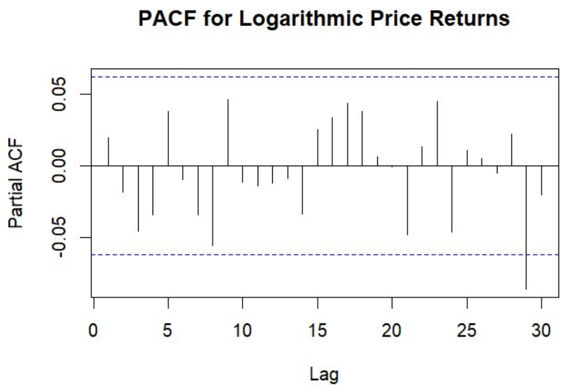
Now we can move into creating the ARIMA model with the proper data prepared. In other words, we want to find the **ARIMA(p,d,q) notation**.

**Correlograms**, or autocorrelation plots, indicate how the data is related to itself over time based on the number of periods apart, or lags.

ARIMA models integrate two types of correlograms, each allowing us to determine parts of the ARIMA notation:

- the **AutoCorrelation Function (ACF)** displays the correlation between series and lags for the Moving Average (q) of the ARIMA model
- the **Partial AutoCorrelation Function (PACF)** displays the correlation between returns and lags for the Auto-Regression (p) of the ARIMA model

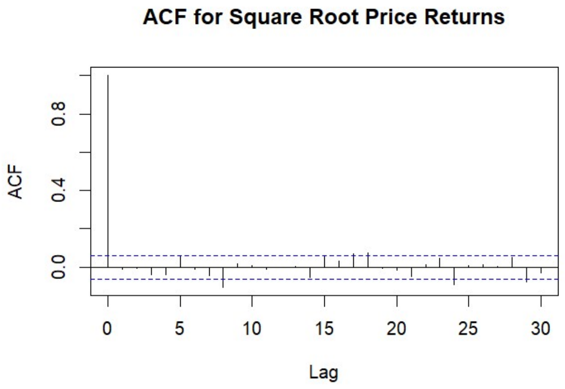


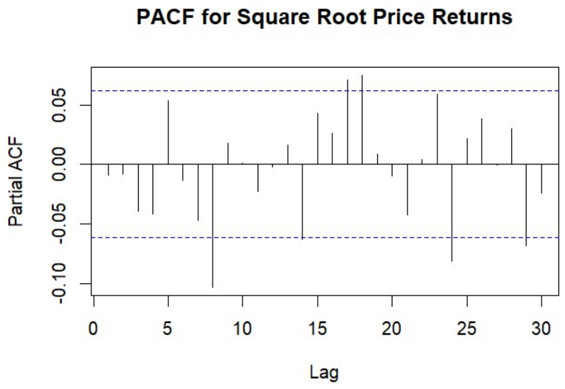


Upon viewing the ACF plot for the logarithmic returns, there is no cut-off for strong correlations (where values no longer cross the blue dotted line) since Lag 1 does not have a strong correlation, which means the p-notation is 0.

Upon viewing the PACF plot for the logarithmic returns, the cut-off for strong correlations (where values no longer cross the blue dotted line) since Lag 1 does not have a strong correlation, which means the p-notation is 0.

Alongside the results, where the d-notation is 0, the model to fit onto the logarithmic price returns is **ARIMA(0,0,0)**. This particular ARIMA model represents **white noise**, which means no model will be able to fit the square root values for this stock.





Upon viewing the ACF plot for the logarithmic returns, there is no cutoff for strong correlations (where values no longer cross the blue dotted line) since Lag 1 does not have a strong correlation, which means the p-notation is 0.

Upon viewing the PACF plot for the logarithmic returns, the cutoff for strong correlations (where values no longer cross the blue dotted line) since Lag 1 does not have a strong correlation, which means the p-notation is 0.

Alongside the results, where the d-notation is 0, the model to fit onto the logarithmic price returns is **ARIMA(0,0,0)**. This particular ARIMA model represents **white noise**, which means no model will be able to fit the square root values for this stock.

# 7. ARIMA Model Development & Programming a fitted forecast

## Model Specification

An ARIMA model was specified for the differenced logarithmic series with the following order: ARIMA(2, 0, 2). The model order was chosen based on the ACF and PACF plots and the underlying characteristics of the data.

## Model Fitting

The ARIMA model was fitted to the training data using the `arima` function in R. The model was then used to forecast future stock prices. The Ljung-Box test was applied to the residuals of the fitted model to check for autocorrelation, ensuring that the residuals behaved like white noise.

## Forecasting

The fitted ARIMA model was used to forecast future stock prices for one step ahead. The forecasted values were compared with the actual stock prices, and the performance of the model was evaluated.

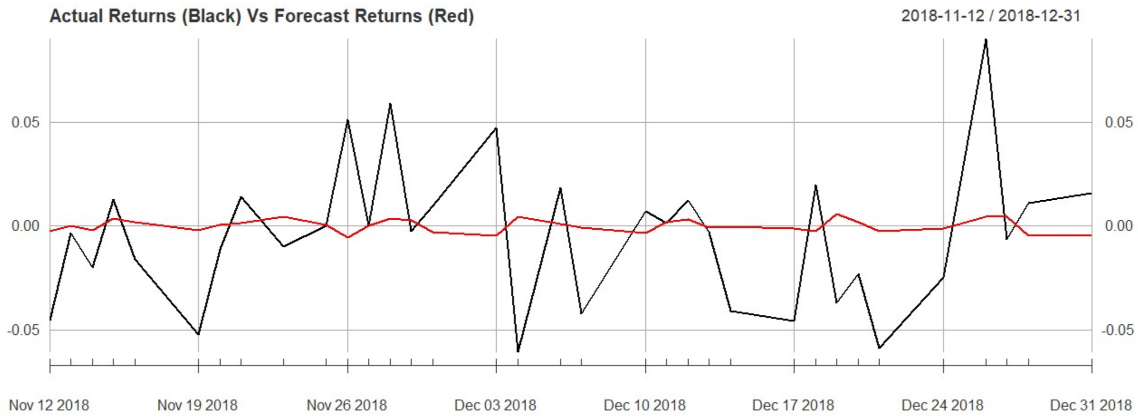
# 8. Model Evaluation

## Forecast Accuracy

The forecasted returns were compared with the actual returns by calculating the percentage of correct sign predictions. This accuracy measure indicates how often the model correctly predicted the direction of stock price movements.

### Visual Comparison

A visual comparison between the actual and forecasted returns was made by plotting the two series on the same graph. This comparison provided insights into how well the model captured the actual stock price movements.



### Accuracy Percentage

The overall accuracy of the forecast was computed as the percentage of correct sign predictions. This metric was used to assess the effectiveness of the ARIMA model in predicting stock price movements.

We now know that the model is 45.06% accurate when making a forecast of an increase or decrease in the logarithmic return of a stock.

## 9. Conclusion

The AMZN stock value has grown over time, and if the trend continues, it should continue to grow. Based on the model, the values are likely to increase, as the forecast returns are more often above 0 than below.

### Summary of Findings

The ARIMA model developed in this project provided a systematic approach to forecasting Amazon's stock prices. The model was able to capture the key features of the time series data, including trend and seasonality. The accuracy of the model was evaluated, and the results indicated that the model had a reasonable predictive ability.

### Limitations and Future Work

While the ARIMA model performed well, it has limitations, such as assuming linearity and relying on past values for prediction. Future work could involve exploring other models, such as GARCH or machine learning-based approaches, to improve the forecasting accuracy.

Additionally, extending the analysis to include external factors like market sentiment and macroeconomic indicators could enhance the model's predictive power.

### Practical Implications

The findings of this project have practical implications for traders and investors interested in Amazon stock. The time series analysis and forecasting techniques used can be applied to other stocks or financial instruments, providing valuable insights for making informed investment decisions.

## 10. References

- Hyndman, R. J., & Athanasopoulos, G. (2018). Forecasting: principles and practice. OTexts.
- Tsay, R. S. (2005). Analysis of Financial Time Series. Wiley-Interscience.
- Chatfield, C. (2003). The Analysis of Time Series: An Introduction. Chapman and Hall/CRC.
- R Documentation for `forecast`, `quantmod`, `tseries`, `timeSeries`, and `xts` packages.
- https://www.datascience.com/blog/introduction-to-forecasting-with-arima-in-r-learn-data-science-tutorials
- http://www.forecastingsolutions.com/arima.html
- https://www.quantinsti.com/blog/forecasting-stock-returns-using-arima-model]