Time Series Analysis of Amazon and Google Stocks From 2008 To 2024

Haowei Chen

Hong Kong University of Science and Technology

Abstract. Amazon and Google are two world-leading companies in artificial intelligence. Thanks to their extraordinary performances in technologies, both Amazon and Google tended to become members of the most popular stocks in the US stock market. In this report, we apply techniques of time series analysis to the adjusted close prices and daily returns of these two stocks from 2008 to 2024 for the purpose of analyzing their performances theoretically. Data used in this paper is obtained from Yahoo Finance.[2]

Keywords: Time Series Analysis · Google · Amazon.

1 Introduction

Amazon and Google are two world-leading companies in artificial intelligence. Thanks to their extraordinary performances in technologies, both Amazon and Google tended to become members of the most popular stocks in the US stock market. In this report, we apply techniques of time series analysis to the adjusted close prices and daily returns of these two stocks from 2008 to 2024 for the purpose of analyzing their performances theoretically. Data used in this paper is obtained from Yahoo Finance.[2]

The outline of this report is organized as below. To begin with, we visualize the daily returns of both stocks and test their stationarities and autocorrelations. Then we analyze their fractal behaviors using Hurst exponents, detrended fluctuation analysis, and multifractality. In Section 5, the causal relation between two stocks is studied by fitting a **VARMA** model to them. In Section 6, we compute the Fourier transforms of both stocks and then study their power spectral densities. Finally, we observe the behaviors of empirical mode decompositions of stocks. The report is ended with concluded remarks on our observations.

2 Data Preprocessing

Adjusted close prices of Google and Amazon from 2008 to 2024 are obtained from Yahoo!Finance. We visualize these data first:

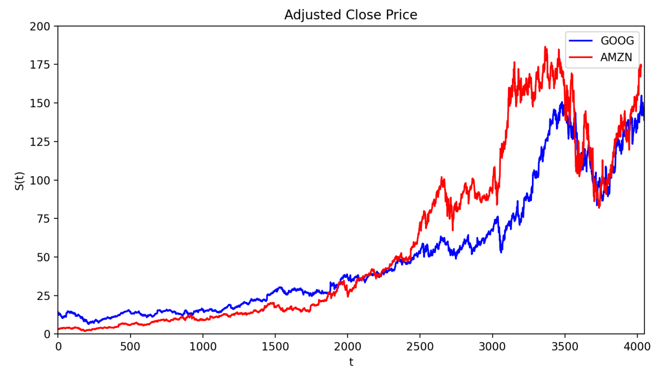

Fig. 1. Figure of Daily Adjusted Close Prices

As is observed in the figure, both stocks have performed well in the past 16 years. Their stock prices nowadays are several times of they were in 2008. By the formula of daily return,

$$X(t) = \ln[\frac{S(t)}{S(t-1)}] \tag{1}$$

we obtain the figure of daily returns after subtracting means from them as below:

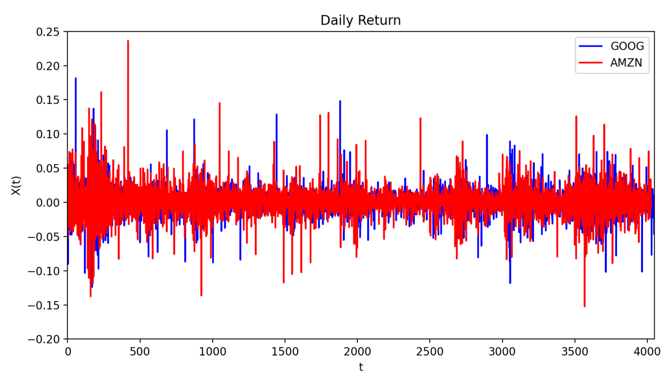

Fig. 2. Figure of Daily Returns

To make closer observations, we plot the daily returns of Google and Amazon alternatively.

For Google, the figure of daily returns is shown below:

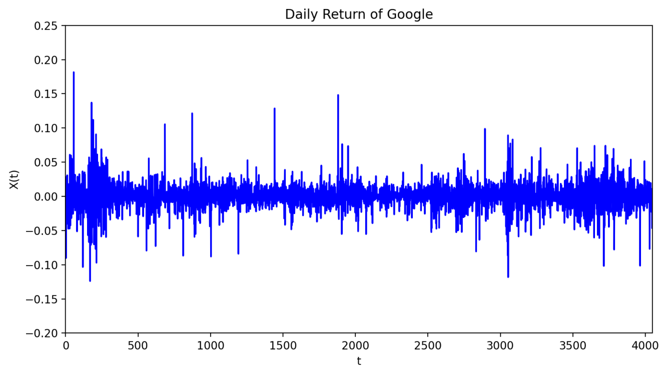

Fig. 3. Daily Returns of Google

For Amazon, the figure of daily returns is shown below:

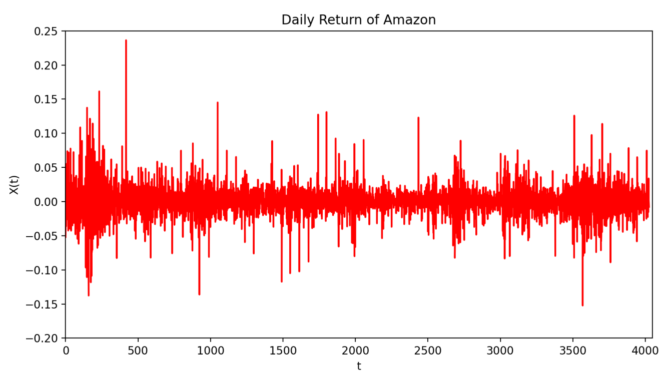

Fig. 4. Daily Returns of Amazon

For convenience, in this report, we denote the daily return series by X and the adjusted close price series by S hereafter.

3 Stationarity and Autocorrelation

In this section, we test the stationarity and autocorrelation of stocks.

3.1 Augmented Dickey-Fuller Test

In this subsection, we focus on the stationarity of daily return series using augmented Dickey-Fuller test.

Perform the augmented Dickey-Fuller test on the adjusted close price series of Google, the test result is shown below:

```
   Augmented Dickey-Fuller Results
=====================================
Test Statistic                -25.058
P-value                         0.000
Lags                                6
-------------------------------------

Trend: Constant
Critical Values: -3.43 (1%), -2.86 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

Fig. 5. Augmented Dickey-Fuller Test of Google

Since $-25.058 < -3.43$, we can refuse the null hypothesis. Hence, the series is stable.

Perform the augmented Dickey-Fuller test on the adjusted close price series of Amazon, the test result is shown below:

```
   Augmented Dickey-Fuller Results
=====================================
Test Statistic                -47.002
P-value                         0.000
Lags                                1
-------------------------------------

Trend: Constant
Critical Values: -3.43 (1%), -2.86 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

Fig. 6. Augmented Dickey-Fuller Test of Amazon

Since $-47.002 < -3.43$, we can refuse the null hypothesis. Hence, the series is stable, too.

Both p-values are zero, which indicates that our conclusion is very convincing. Hence, both daily returns are stable.

3.2 (Partial) Autocorrelation Function

In this subsection, we compute the autocorrelation functions of partial autocorrelation functions of two stocks.

The autocorrelation function of Google is plotted in the figure below:

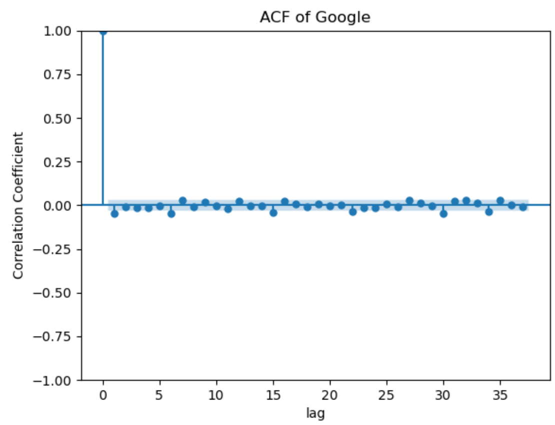

Fig. 7. ACF of Google

The partial autocorrelation function of Google is plotted in the figure below:

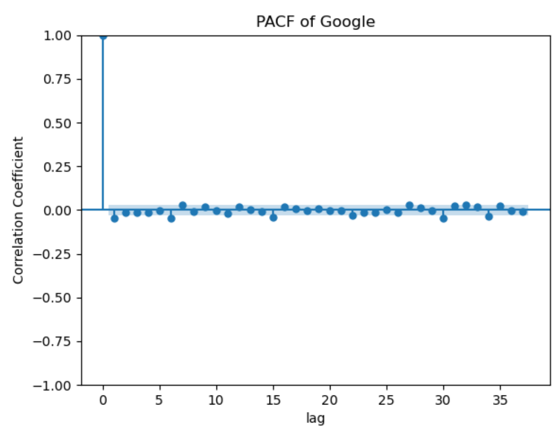

Fig. 8. PACF of Google

Since both ACF and PACF enter the blue area after the lag of 1, the best-fitting ARMA model tends to be ARMA(1, 1).

The autocorrelation function of Amazon is plotted in the figure below:

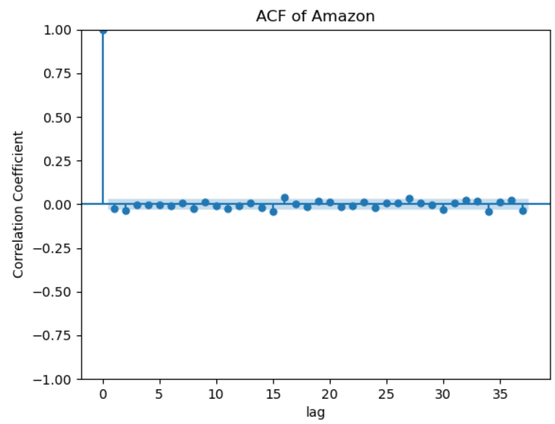

Fig. 9. ACF of Amazon

The partial autocorrelation function of Amazon is plotted in the figure below:

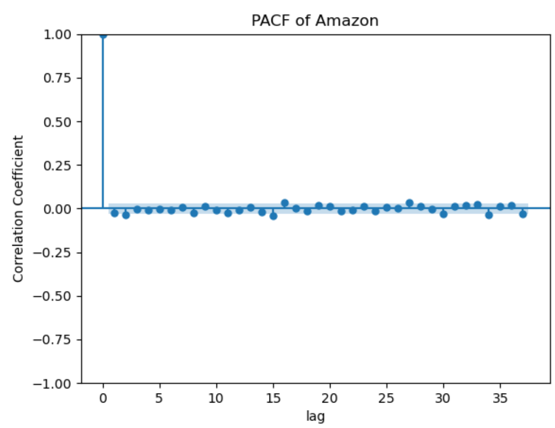

Fig. 10. PACF of Amazon

Since both ACF and PACF enter the blue area after the lag of 1, the best-fitting **ARMA** model tends to be $\mathbf{ARMA}(1,1)$.

In order to find the best-fitting **ARMA** model for these two stocks, we compute the AIC value for each possible pair of p and q. The AIC is defined in terms of the negative of the maximum value of the natural logarithm of the likelihood L of the model, given the data, adjusted for the number of adjustable parameters in the model. The equation for AIC is [3]:

$$AIC := -2\ln({\sigma_\epsilon}^2) + 2k \tag{2}$$

where n is the number of residuals, ${\sigma_\epsilon}^2$ maximum likelihood of is the residuals' variance, and k is the sum of model parameters. The best-fitting model shall have the smallest AIC value. The test outputs match our guesses. For detailed outputs of AIC value of each pair of (p, q) orders, please refer to the appendix 9.1.

3.3 ACF of Absolute Value

In this subsection, we compute the autocorrelation functions of the absolute values of daily returns.

For Google, the results can be visualized as below:

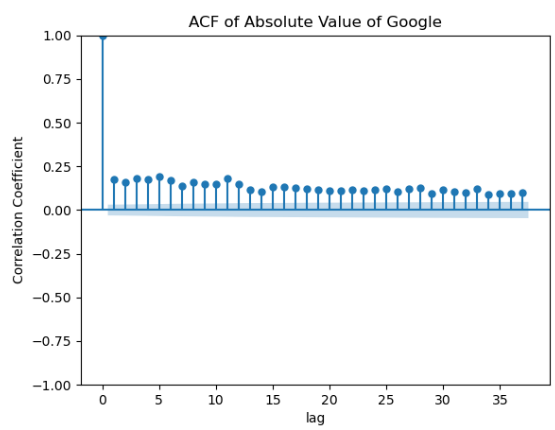

Fig. 11. ACF of Absolute Value of Google

For Amazon, the results can be visualized as follows:

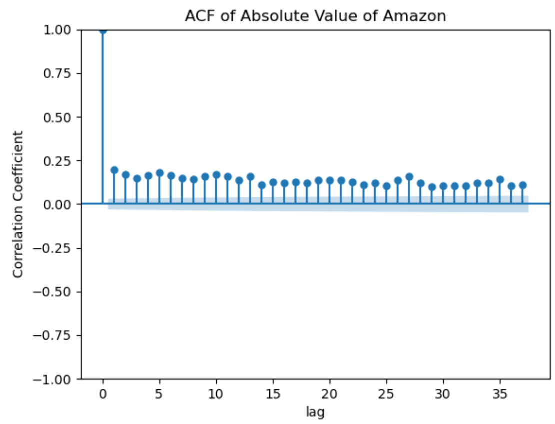

Fig. 12. ACF of Absolute Value of Amazon

Note that both ACF of absolute values of daily returns show strong correlations since they never enter the blue area, which is quite different from what ACF of daily returns behave. This phenomenon suggests that both the daily return time series are not independent.

4 Fractal Behaviour of Time Series

In this section, we examine the fractal behaviors of daily return series through different methods including Hurst exponents, detrended fluctuation analysis and multifractality testings.

4.1 Hurst Exponent

In this subsection, we focus on Hurst exponents of two stocks.

For Google, the Hurst exponent obtained is 0.4672. For Amazon, the Hurst exponent obtained is 0.4626. Both the Hurst exponents are almost 0.5, which implies that the adjusted close prices of these two stocks Google and Amazon can be considered as random walks.

The figure of rescaled range of X and window sizes for Google is shown below:

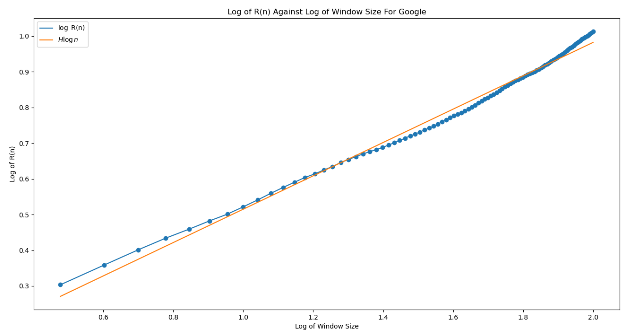

Fig. 13. Rescaled Range $R(n)$ of X Against Window Size n For Google

The figure of rescaled range of X and window sizes for Amazon is shown below:

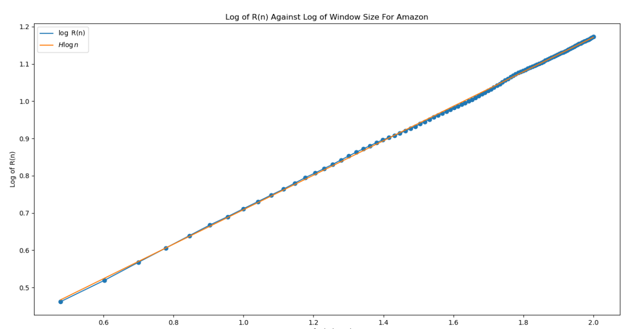

Fig. 14. Rescaled Range $R(n)$ of X Against Window Size n For Amazon

Note that in both figures above, $\log R(n)$ turns out to be proportional to $\log n$ and fits well with $H \log n$, it is proved that $R(n) \sim n^H$.

4.2 Detrended Fluctuation Analysis

In this subsection, we perform the detrended fluctuation analysis on stocks.

The scaling exponent for Google in detrended fluctuation analysis is 0.4975, while for Amazon is 0.4920. Both exponents are almost 0.5, which implies that both time series are like white noise. This conclusion is consistent with what we obtained due to the analysis of Hurst exponents.

The log-log plot of $f(n)$ against window size n for Google is shown below:

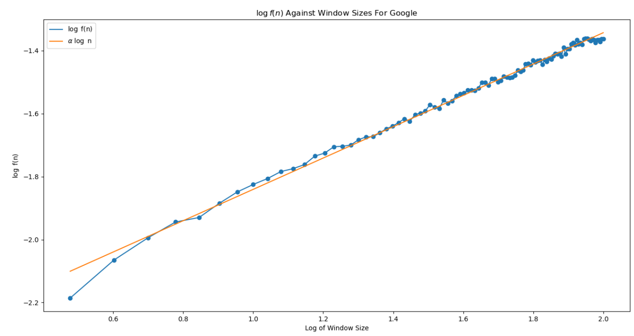

Fig. 15. Detrended Fluctuation $F(n)$ Against Window Size n For Google

The log-log plot of $f(n)$ against window size n for Amazon is shown below:

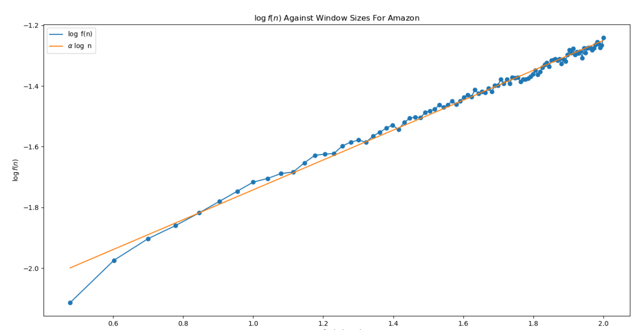

Fig. 16. Detrended Fluctuation $F(n)$ Against Window Size n For Amazon

Note that in both figures above, $\log F(n)$ turns out to be proportional to $\log n$ and fits well with $\alpha \log n$, it is proved that $F(n) \sim n^{\alpha}$.

4.3 Multi-fractality

In this subsection, we check the multi-fractality by computing $M(q, \tau)$ for 5 values of q: $1, 5, 10, 20$ and 40 where $M(q, \tau)$ is defined as below:

$$
\begin{aligned}
M(q,\tau) &:= \left\langle |\delta_\tau Y(t)|^q \right\rangle \\
&= \left\langle |Y(t+\tau) - Y(t)|^q \right\rangle
\end{aligned} \tag{3}
$$

The figure for $M(q,\tau)^{1/q}$ against τ for Google is shown below:

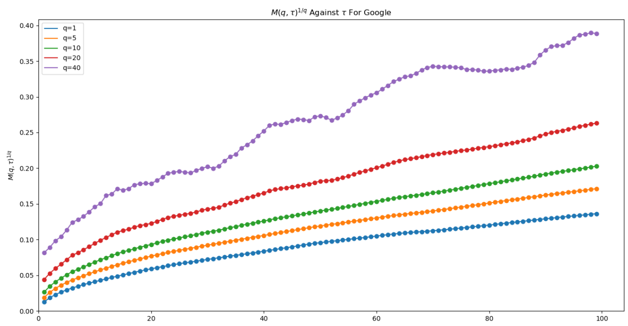

Fig. 17. Figure of $M(q,\tau)^{1/q}$ For Google

The figure for $M(q,\tau)^{1/q}$ against τ for Amazon is shown below:

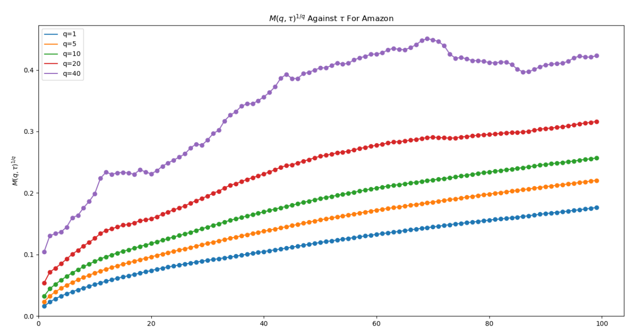

Fig. 18. Figure of $M(q,\tau)^{1/q}$ For Amazon

Now we obtain the figure for $f(q)/q$:

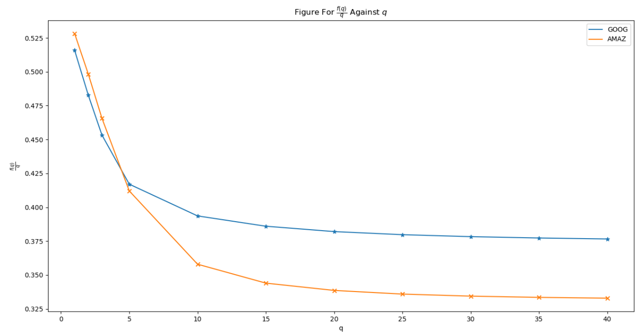

Fig. 19. Figure of $f(q)/q$

According to these plots, it can be concluded that both prices are multifractal.

By performing MDFA on daily return series, we obtain figures of scaling exponents $\alpha(q)$ against their orders q. For Google, the figure is shown below:

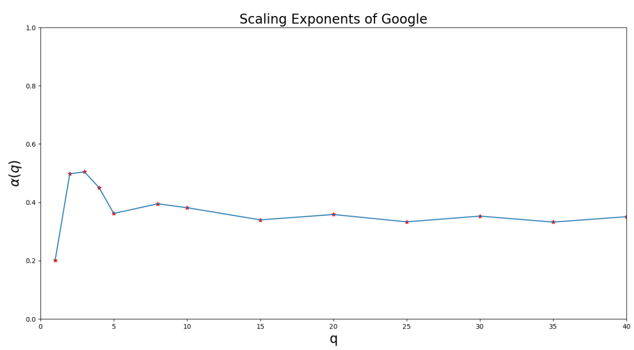

Fig. 20. MDFA Results of Google

For Amazon, the figure is shown below:

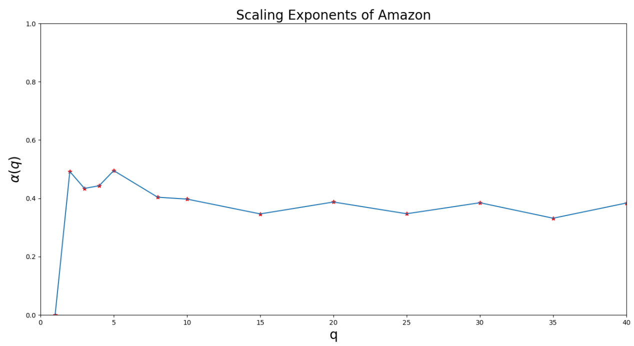

Fig. 21. MDFA Results of Amazon

Note that in both figures, when $q = 2$, $\alpha(q)$ are around 0.5, which agrees with our results obtained for Hurst exponents in DFA.

5 Granger Causality

In this section, we study the causal relation between two series of daily returns by fitting a **VARMA** model to them.

In order to determine the best order pair (p, q) of our **VARMA** model, we try all possible pairs with both values of p and q within the range from 1 to 5. The order pair $(1, 2)$ turns out to own the smallest AIC value. For detailed AIC outputs, please refer to the Appendix 9.2.

By fitting a **VARMA**$(1, 2)$ model to daily return series, we obtain the following coefficients:

For the equation of the daily return series of Google, we have:

Results for equation r1

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	2.307e-05	0.000	0.074	0.941	-0.001	0.001
L1.r1	-0.0448	3.422	-0.013	0.990	-6.751	6.662
L1.r2	-0.0034	0.586	-0.006	0.995	-1.153	1.146
L1.e(r1)	-0.0005	3.423	-0.000	1.000	-6.709	6.708
L1.e(r2)	0.0003	0.586	0.001	1.000	-1.148	1.149
L2.e(r1)	-0.0130	0.165	-0.079	0.937	-0.336	0.310
L2.e(r2)	-0.0032	0.013	-0.256	0.798	-0.028	0.021

Fig. 22. Coefficients For the Equation of Google

For the equation of the daily return series of Amazon, we have:

Results for equation r2

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-3.445e-06	0.000	-0.009	0.993	-0.001	0.001
L1.r1	0.0166	4.542	0.004	0.997	-8.885	8.918
L1.r2	-0.0234	0.744	-0.031	0.975	-1.481	1.434
L1.e(r1)	-0.0028	4.542	-0.001	1.000	-8.905	8.900
L1.e(r2)	-0.0005	0.745	-0.001	0.999	-1.460	1.459
L2.e(r1)	-0.0569	0.216	-0.263	0.792	-0.481	0.367
L2.e(r2)	-0.0341	0.014	-2.462	0.014	-0.061	-0.007

Fig. 23. Coefficients For the Equation of Amazon

As is shown by the p-values of coefficients in the results, nearly all coefficients are not significant since their p-values are almost 1. We perform F-tests to determine the Granger causality and try to figure out the reason.

By setting the daily return series of Amazon as the second column, we obtain the following result:

```
number of lags (no zero) 1
ssr based F test:         F=0.0723  , p=0.7881  , df_denom=4022, df_num=1
ssr based chi2 test:   chi2=0.0723  , p=0.7880  , df=1
likelihood ratio test: chi2=0.0723  , p=0.7880  , df=1
parameter F test:         F=0.0723  , p=0.7881  , df_denom=4022, df_num=1

Granger Causality
number of lags (no zero) 2
ssr based F test:         F=0.1012  , p=0.9038  , df_denom=4019, df_num=2
ssr based chi2 test:   chi2=0.2026  , p=0.9037  , df=2
likelihood ratio test: chi2=0.2026  , p=0.9037  , df=2
parameter F test:         F=0.1012  , p=0.9038  , df_denom=4019, df_num=2

Granger Causality
number of lags (no zero) 3
ssr based F test:         F=1.2071  , p=0.3055  , df_denom=4016, df_num=3
ssr based chi2 test:   chi2=3.6277  , p=0.3046  , df=3
likelihood ratio test: chi2=3.6261  , p=0.3048  , df=3
parameter F test:         F=1.2071  , p=0.3055  , df_denom=4016, df_num=3

Granger Causality
number of lags (no zero) 4
ssr based F test:         F=1.1330  , p=0.3389  , df_denom=4013, df_num=4
ssr based chi2 test:   chi2=4.5422  , p=0.3376  , df=4
likelihood ratio test: chi2=4.5396  , p=0.3379  , df=4
parameter F test:         F=1.1330  , p=0.3389  , df_denom=4013, df_num=4
```

Fig. 24. F-test Results for Granger Causality With Lag up to 4

Since p-values for all 4 F-tests here are at least 0.3, we can not reject the null hypothesis, which implies that the daily return series of Amazon does not Granger cause the daily return series of Google.

By setting the daily return series of Google as the second column, we obtain the following result:

```
number of lags (no zero) 1
ssr based F test:         F=0.7215  , p=0.3957  , df_denom=4022, df_num=1
ssr based chi2 test:   chi2=0.7220  , p=0.3955  , df=1
likelihood ratio test: chi2=0.7220  , p=0.3955  , df=1
parameter F test:         F=0.7215  , p=0.3957  , df_denom=4022, df_num=1

Granger Causality
number of lags (no zero) 2
ssr based F test:         F=4.5255  , p=0.0109  , df_denom=4019, df_num=2
ssr based chi2 test:   chi2=9.0622  , p=0.0108  , df=2
likelihood ratio test: chi2=9.0520  , p=0.0108  , df=2
parameter F test:         F=4.5255  , p=0.0109  , df_denom=4019, df_num=2

Granger Causality
number of lags (no zero) 3
ssr based F test:         F=3.3254  , p=0.0189  , df_denom=4016, df_num=3
ssr based chi2 test:   chi2=9.9937  , p=0.0186  , df=3
likelihood ratio test: chi2=9.9813  , p=0.0187  , df=3
parameter F test:         F=3.3254  , p=0.0189  , df_denom=4016, df_num=3

Granger Causality
number of lags (no zero) 4
ssr based F test:         F=2.6971  , p=0.0292  , df_denom=4013, df_num=4
ssr based chi2 test:   chi2=10.8128 , p=0.0288  , df=4
likelihood ratio test: chi2=10.7983 , p=0.0289  , df=4
parameter F test:         F=2.6971  , p=0.0292  , df_denom=4013, df_num=4
```

Fig. 25. F-test Results for Granger Causality With Lag up to 4

Since the p-values become about $0.01 - 0.02$ at lags 2 to 4, we can reject the null hypothesis, which implies that the daily return series of Google Granger causes the daily return series of Amazon.

6 Fourier Transform and Power Spectrum

In this section, we study the power spectrum of stocks.

First, we perform the Fourier transform on two series.

The figure of magnitudes of coefficients of the Fourier transform of Google is shown below:

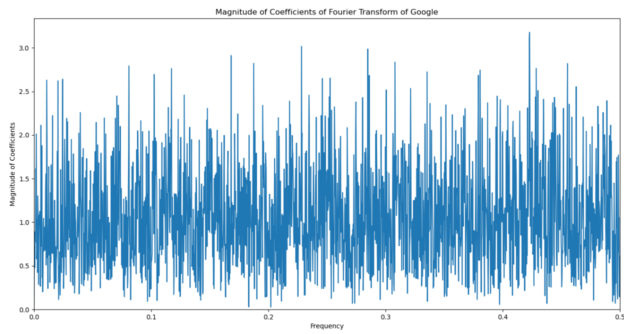

Fig. 26. Magnitudes of Fourier Transform of Google

The figure of magnitudes of coefficients of the Fourier transform of Amazon is shown below:

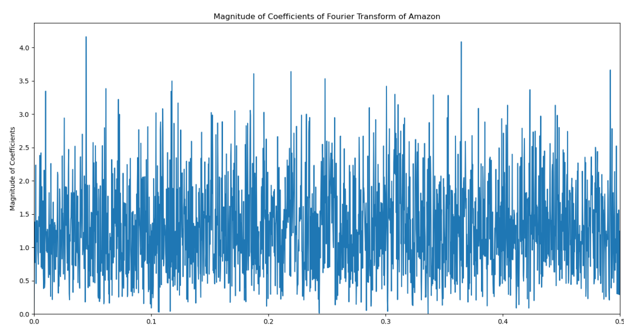

Fig. 27. Magnitudes of Fourier Transform of Amazon

Then we visualize the power spectral densities.

The power spectral density of Google is shown below:

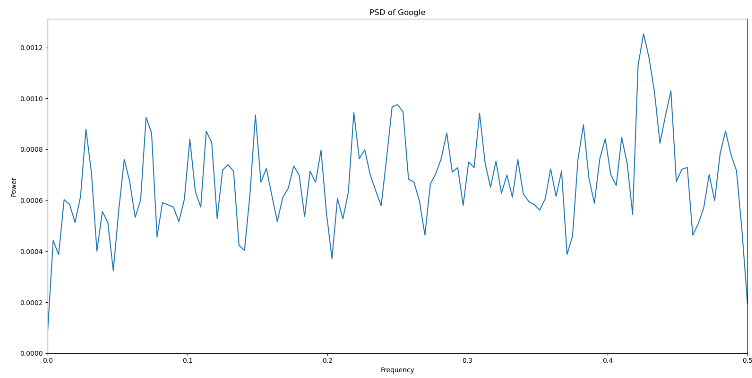

Fig. 28. PSD of Google

The power spectral density of Amazon is shown below:

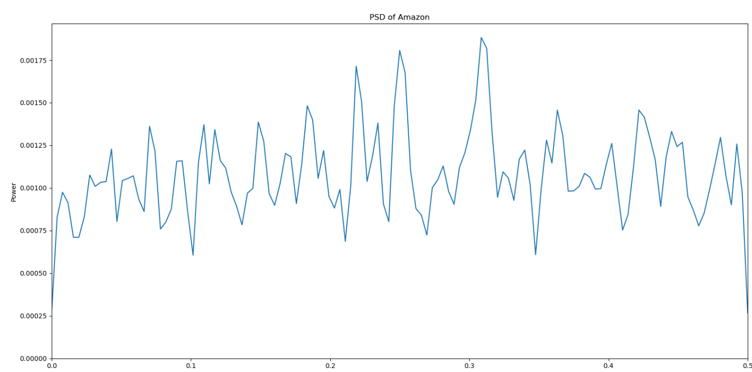

Fig. 29. PSD of Amazon

Both figures turn out to show a wide range of frequency and chaotic density patterns. To take clearer observations, we replot both figures in log-log plots and fit the curves with straight lines.

For Google, we obtain the following result:

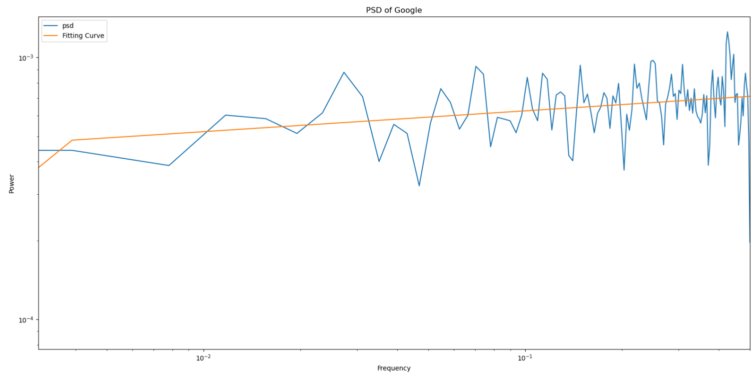

Fig. 30. PSD of Google

For Amazon, we obtain the following result:

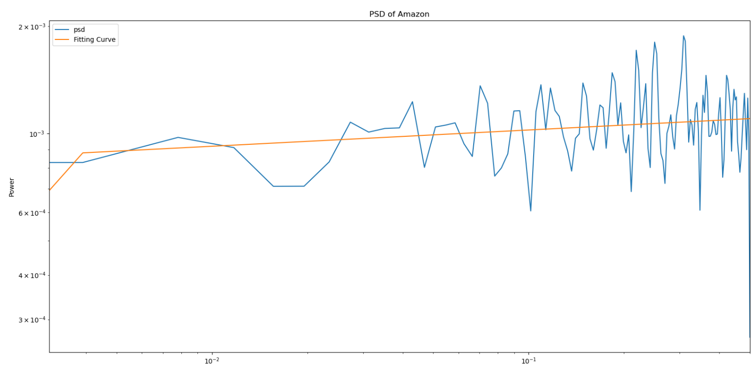

Fig. 31. PSD of Amazon

The slopes of the fitting lines are 0.05 and 0.04, both are very close to 0. Theoretically, a relation exists between the PSD exponent β and the DFA exponent α obtained from the same time series: $\beta = 2\alpha - 1$. [4] Recall that our scaling exponents computed through DFA are almost 0.5, this output matches the theory.

7 Empirical Mode Decomposition

In this section, we perform empirical mode decompositions on stocks and analyze the IMFs we obtain. For convenience, in this section, we use c_j to denote the IMF of order j. In this case, c_1 and c_2 are the first two IMFs.

For Google, we can decompose its daily return series into 9 IMFs. we plot the first, second, fourth, sixth, and ninth IMFs as below:

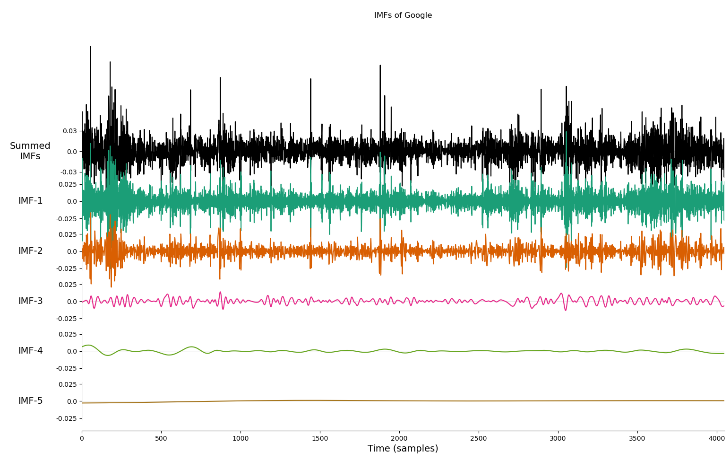

Fig. 32. IMFs of Google

Please note that the numbers that IMFs are labeled in this figure are the order of them in the five IMFs we mentioned above instead of their exact orders in the empirical mode decomposition. Labels in the figure for Amazon are assigned following the same rule.

For Amazon, we can decompose its daily return series into 8 IMFs. we plot the first, second, fourth, sixth, and eighth IMF as below:

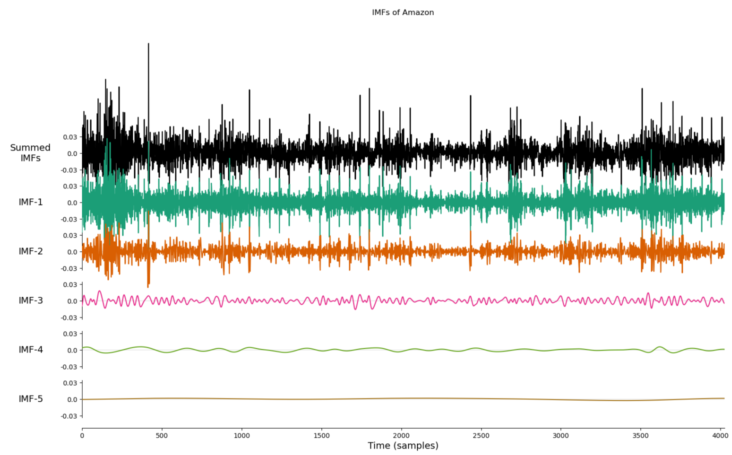

Fig. 33. IMFs of Amazon

For each IMF we obtain, we compute its Hurst exponent. The result can be visualized as below:

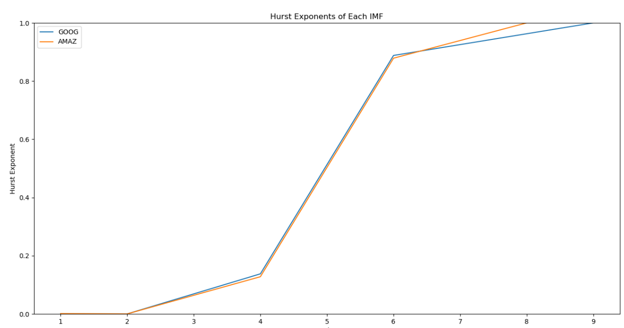

Fig. 34. Hurst Exponents of IMFs

As is shown in the figure above, the curve of Hurst exponent tends to own a trend to increase from 0 to 1 as the order of the IMF increases. This trend implies that high-frequency parts in the series are like pink noises while low-frequency parts own long-term memories.

For the first two IMFs of Google, their power spectral densities are shown below:

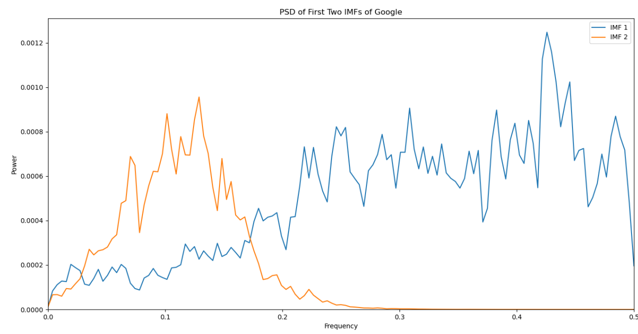

Fig. 35. PSD of First Two IMFs of Google

For the first two IMFs of Amazon, their power spectral densities are shown below:

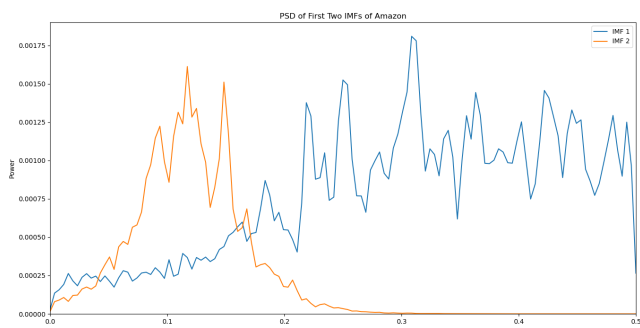

Fig. 36. PSD of First Two IMFs of Amazon

Note that not only for Google but also for Amazon, the first IMFs have their peaks on the right of the second ones, which indicates that the first IMFs have higher frequencies than the second ones. This matches what the theory predicts.

For further comparison, we plot the PSDs of $X - c_1$, $X - c_1 - c_2$ together with X. For Google, the result is shown below:

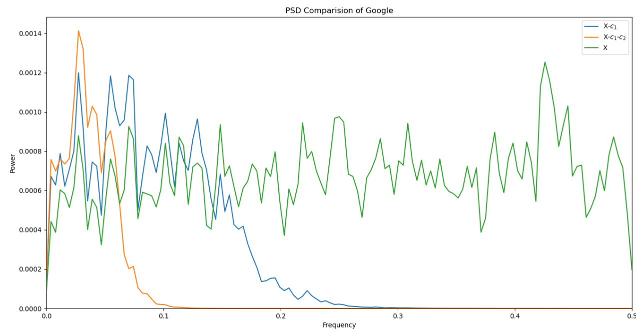

Fig. 37. PSD Comparison Figure of Google

For Amazon, the result is shown below:

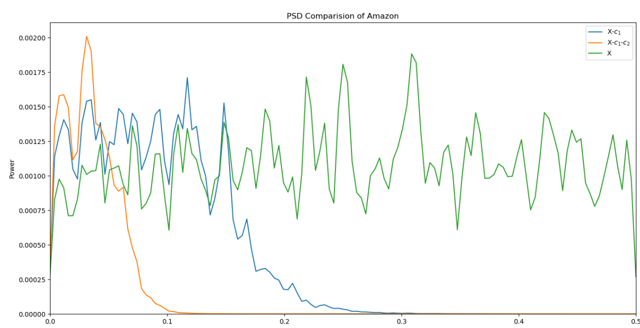

Fig. 38. PSD Comparison Figure of Amazon

It can be obviously observed that $X - c_1$ lacks high-frequency parts compared with the original time series X, while $X - c_1 - c_2$ lacks even more high-frequency parts than $X - c_1$. $X - c_1 - c_2$ concentrates in the low-frequency area. This can be explained by the facts that IMFs are obtained by decomposing the original series X and the frequency of c_j decreases as its order increases. While subtracting c_1 and c_2 from X, we in fact keep extracting its high-frequency parts.

8 Concluded Remarks

In this report, we apply time series analysis techniques to the adjusted close price series and the daily return series of two stocks: Google and Amazon, to observe the behaviors and features they showed from 2008 to 2024. The results we obtained indicate that the two stocks turn out to behave similarly. The daily return series of both stocks fit $\mathbf{ARMA}(1,1)$ models, show dependency in time and multifractality, and behave like white noises. Though it is examined that daily returns of Google Granger cause ones of Amazon at lags 2 to 4, coefficients in the $\mathbf{VARMA}(1,2)$ model to fit them are not significant.

Acknowledgements

This report is written in the LaTeXtemplate provided by LCNS [1]. All codes I used for this report can be found in Appendix 9.3.

References

1. Lncs homepage, `http://www.springer.com/lncs`, last accessed 4 Oct 2017
2. Yahoo finance, `https://finance.yahoo.com/`, last accessed 24 Apr 2024
3. Bonakdari, H., Zeynoddin, M.: Chapter 5 - goodness-of-fit precision criteria. In: Bonakdari, H., Zeynoddin, M. (eds.) Stochastic Modeling, pp. 187–264. Elsevier (2022). `https://doi.org/https://doi.org/10.1016/B978-0-323-91748-3.00003-3`
4. Heneghan, C., McDarby, G.: Establishing the relation between detrended fluctuation analysis and power spectral density analysis for stochastic processes. Physical review E **62**(5), 6103 (2000)

9 Appendices

9.1 AIC Values For ARMA Models

Test results for AIC values of each pair of orders (p, q) for Google are shown below:

```
[[1, 1, -20692.94582065474], [1, 2, -20692.426925191518], [1, 3, -20688.289804431904], [1, 4,
-20688.904162035884], [2, 1, -20689.942735954486], [2, 2, -20687.994921141377], [2, 3,
-20689.77217788076], [2, 4, -20688.994873778145], [3, 1, -20689.08084838853], [3, 2,
-20687.09141742791], [3, 3, -20687.823645807024], [3, 4, -20687.183075014145], [4, 1,
-20687.976082293884], [4, 2, -20686.02915299382], [4, 3, -20684.05414659548], [4, 4,
-20686.612015527993]]
```

Fig. 39. AIC Values(Google)

It can be observed that $(1, 1)$ has the smallest AIC value here, which proves our guess.

Test results for AIC values of each pair of orders (p, q) for Amazon are shown below:

```
[[1, 1, -18855.742861296985], [1, 2, -18854.94131521583], [1, 3, -18852.873250205113], [1, 4,
-18851.6550766054], [2, 1, -18854.88509391444], [2, 2, -18853.177231757487], [2, 3,
-18850.898393134335], [2, 4, -18849.034373914452], [3, 1, -18852.855736441517], [3, 2,
-18851.175985866408], [3, 3, -18848.894572692974], [3, 4, -18848.090357030633], [4, 1,
-18850.99808689137], [4, 2, -18849.009817600025], [4, 3, -18847.00977456736], [4, 4,
-18846.19290119736]]
```

Fig. 40. AIC Values(Amazon)

It can be observed that $(1, 1)$ has the smallest AIC value here, which proves our guess.

9.2 AIC Values For VARMA Model

The outputs of AIC values of all order pairs for the **VARMA** model are shown below:

```
[[1, 1, -39432.108937709854], [1, 2, -39438.64139848195], [1, 3, -39435.56160399335], [1, 4,
-39430.43402547401], [1, 5, -39427.46918178101], [2, 1, -39438.05640965728], [2, 2, -39430.02210255016],
[2, 3, -39427.53691657762], [2, 4, -39422.65176381811], [2, 5, -39419.16718623552], [3, 1,
-39435.692073847036], [3, 2, -39427.715770882394], [3, 3, -39419.8476598258], [3, 4,
-39414.95872287122], [3, 5, -39411.37887583941], [4, 1, -39430.91673026615], [4, 2, -39422.92413396393],
[4, 3, -39414.98067445784], [4, 4, -39407.024548771944], [4, 5, -39403.15324008887], [5, 1,
-39426.47409345233], [5, 2, -39418.493022179144], [5, 3, -39410.548435960176], [5, 4,
-39402.56548239899], [5, 5, -39394.622525269675]]
```

Fig. 41. AIC Values For **VARMA** Model

9.3 Codes

```python
# -*- coding: utf-8 -*-
"""
5058 Proj1

@author: 17100
"""

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df2=pd.read_csv('C:/Users/17100/.spyder-py3/5058/data/GOOG.
    csv')
df4=pd.read_csv('C:/Users/17100/.spyder-py3/5058/data/AMZN.
    csv')
print(df2.columns)
returns=[]
returns2=[]
clos=np.array(list(df2['Adj Close']))
clos2=np.array(list(df4['Adj Close']))
for i in range(1,len(clos)):
    returns.append(np.log(clos[i]/clos[i-1]))
for i in range(1, len(clos2)):
    returns2.append(np.log(clos2[i]/clos2[i-1]))

x1=np.mean(returns)
x2=np.mean(returns2)
r1=np.array(returns)-x1
r2=np.array(returns2)-x2

fig=plt.figure(figsize=(16,8), dpi=200)
n=len(r1)
m=len(r2)
tline=np.linspace(0, n, n)
tline2=np.linspace(0, m, m)
plt.plot(tline, r1, color='blue', label='GOOG')
plt.plot(tline2, r2, color='r', label='AMZN')
plt.title('Daily Return')
plt.xlim(0, n)
plt.ylim(-0.2, 0.25)
plt.legend()
plt.xlabel('t')
plt.ylabel('X(t)')
plt.show()

fig=plt.figure(figsize=(16,8), dpi=200)
n=len(r1)
m=len(r2)
tline=np.linspace(0, n, n)
tline2=np.linspace(0, m, m)
```

```
plt.plot(tline, r1, color='blue')
plt.title('Daily Return of Google')
plt.xlim(0, n)
plt.ylim(-0.2, 0.25)
plt.xlabel('t')
plt.ylabel('X(t)')
plt.show()

fig=plt.figure(figsize=(16,8), dpi=200)
n=len(r1)
m=len(r2)
tline=np.linspace(0, n, n)
tline2=np.linspace(0, m, m)
plt.plot(tline2, r2, color='r')
plt.title('Daily Return of Amazon')
plt.xlim(0, n)
plt.ylim(-0.2, 0.25)
plt.xlabel('t')
plt.ylabel('X(t)')
plt.show()

fig2=plt.figure(figsize=(16,8), dpi=200)
tline3=np.linspace(0, n+1, n+1)
tline4=np.linspace(0, m+1, m+1)
plt.plot(tline3, clos, color='blue', label='GOOG')
plt.plot(tline4, clos2, color='r', label='AMZN')
plt.title('Adjusted Close Price')
plt.xlim(0, n)
plt.ylim(0, 200)
plt.legend()
plt.xlabel('t')
plt.ylabel('S(t)')
plt.show()

from arch.unitroot import ADF
adf1 = ADF(r1)
# print(adf.pvalue)
print(adf1.summary().as_text())

adf1 = ADF(clos)
# adf1.trend = 'ct'
print(adf1.summary().as_text())

adf2 = ADF(r2)
# print(adf.pvalue)
print(adf2.summary().as_text())

adf2 = ADF(clos2)
# adf2.trend = 'ct'
print(adf2.summary().as_text())
```

```

from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

fig=plt.figure(figsize=(16,8), dpi=100)
plot_acf(r1, title='ACF of Google')
plt.xlabel('lag')
plt.ylabel('Correlation Coefficient')
plt.show()

fig2=plt.figure(figsize=(16,8), dpi=100)
plot_pacf(r1, title='PACF of Google')
plt.xlabel('lag')
plt.ylabel('Correlation Coefficient')
plt.show()

plot_acf(r2, title='ACF of Amazon')
plt.xlabel('lag')
plt.ylabel('Correlation Coefficient')
plt.show()


plot_pacf(r2, title='PACF of Amazon')
plt.xlabel('lag')
plt.ylabel('Correlation Coefficient')
plt.show()

clos3=abs(r1)
clos4=abs(r2)

plot_acf(clos3, title='ACF of Absolute Value of Google')
plt.xlabel('lag')
plt.ylabel('Correlation Coefficient')
plt.show()

plot_acf(clos4, title='ACF of Absolute Value of Amazon')
plt.xlabel('lag')
plt.ylabel('Correlation Coefficient')
plt.show()

# import statsmodels as sm
# from statsmodels.tsa.api import ARIMA
# data=r1.copy()
# data2=r2.copy()
# aic_val=[]
# for ari in range(1,5):
#     for mij in range(1,5):
#         try:
#             arma_obj=ARIMA(data, order=(ari,0,mij)).fit()
#             aic_val.append([ari,mij,arma_obj.aic])
#         except Exception as e:
```

```
#               print(e)

# print(aic_val)

# aic_val2=[]
# for ari in range(1,5):
#     for mij in range(1,5):
#         try:
#             arma_obj2=ARIMA(data2, order=(ari,0,mij)).fit()
#             aic_val2.append([ari,mij,arma_obj2.aic])
#         except Exception as e:
#             print(e)

# print(aic_val2)


def hurst(price, min_lag=3, max_lag=100):
  lags = np.arange(min_lag, max_lag + 1)
  tau = [np.std(np.subtract(price[lag:], price[:-lag]))
    for lag in lags]
  m = np.polyfit(np.log10(lags), np.log10(tau), 1)
  return m, lags, tau

m, lag, rs=hurst(clos)
print(m[0])

m2, lag2, rs2=hurst(clos2)
print(m2[0])

fig=plt.figure(figsize=(16,8), dpi=100)
plt.plot(np.log10(lag), np.log10(rs), label='$\log$ R(n)')
plt.scatter(np.log10(lag), np.log10(rs))
plt.plot(np.log10(lag), np.log10(lag)*m[0]+m[1], label='$H \
    log n$')
plt.xlabel('Log of Window Size')
plt.ylabel('Log of R(n)')
plt.legend()
plt.title('Log of R(n) Against Log of Window Size For Google'
    )

fig2=plt.figure(figsize=(16,8), dpi=100)
plt.plot(np.log10(lag2), np.log10(rs2), label='$\log$ R(n)')
plt.scatter(np.log10(lag2), np.log10(rs2))
plt.plot(np.log10(lag2), np.log10(lag2)*m2[0]+m2[1], label='
    $H \log n$')
plt.xlabel('Log of Window Size')
plt.ylabel('Log of R(n)')
plt.legend()
plt.title('Log of R(n) Against Log of Window Size For Amazon'
    )
```

```


def DFA(data,ni,fittime):
    n = len(data)//ni
    nf = int(n*ni)

    n_mean =np.mean(data[:nf])
    y = []
    y_hat = []
    for i in range(nf):
        y.append(np.sum(data[:i+1]-n_mean))
    for i in range(int(n)):
        x = np.arange(1,ni+1,1)
        y_temp = y[int(i*ni+1)-1:int((i+1)*ni)]
        coef = np.polyfit(x,y_temp,deg=fittime)
        y_hat.append(np.polyval(coef,x))
    fn = np.sqrt(sum((np.asarray(y)-np.asarray(y_hat).reshape
    (-1))**2)/nf)
    return fn

lags = np.arange(3,101)


f = []
for i in range(len(lags)):
    f.append(DFA(r1,lags[i],1))

m3 = np.polyfit(np.log10(lags), np.log10(f), 1)

f2 = []
for i in range(len(lags)):
    f2.append(DFA(r2,lags[i],1))

m4 = np.polyfit(np.log10(lags), np.log10(f2), 1)


fig3=plt.figure(figsize=(16,8),dpi=100)
plt.plot(np.log10(lags),np.log10(f), label=r'$\log$ f(n)')
plt.scatter(np.log10(lags),np.log10(f))
plt.plot(np.log10(lags), np.log10(lags)*m3[0]+m3[1], label=r'
    $\alpha$ $\log$ n')
plt.legend()
plt.title('$\log f(n)$ Against Window Sizes For Google')
plt.xlabel('Log of Window Size')
plt.ylabel('$\log$ f(n)')
plt.show()

fig4=plt.figure(figsize=(16,8), dpi=100)
plt.plot(np.log10(lags), np.log10(f2), label=r'$\log$ f(n)')
plt.scatter(np.log10(lags), np.log10(f2))
```

```
plt.plot(np.log10(lags), np.log10(lags)*m4[0]+m4[1], label=r'
    $\alpha$ $\log$ n')
plt.legend()
plt.title('$\log f(n)$ Against Window Sizes For Amazon')
plt.xlabel('Log of Window Size')
plt.ylabel('$\log f(n)$')
plt.show()

print(m3[0])
print(m4[0])

y1=np.log(clos)
y2=np.log(clos2)
qarray=np.array([1,2,3, 5, 10, 15, 20, 25, 30, 35, 40])
M=[]
M2=[]
tau=np.arange(1, 100)
for q in qarray:
    Mt=np.array([np.mean([abs(y1[i+t]-y1[i])**q for i in
    range(0, len(y1)-t)]) for t in tau])
    Mt2=np.array([np.mean([abs(y2[i+t]-y2[i])**q for i in
    range(0, len(y2)-t)]) for t in tau])
    M.append(Mt)
    M2.append(Mt2)
fig5=plt.figure(figsize=(16,8), dpi=100)
plt.plot(tau, pow(M[0], 1/qarray[0]), label='q=1')
plt.plot(tau, pow(M[1], 1/qarray[1]), label='q=5')
plt.plot(tau, pow(M[2], 1/qarray[2]), label='q=10')
plt.plot(tau, pow(M[3], 1/qarray[3]), label='q=20')
plt.plot(tau, pow(M[4], 1/qarray[4]), label='q=40')
plt.scatter(tau, pow(M[0], 1/qarray[0]))
plt.scatter(tau, pow(M[1], 1/qarray[1]))
plt.scatter(tau, pow(M[2], 1/qarray[2]))
plt.scatter(tau, pow(M[3], 1/qarray[3]))
plt.scatter(tau, pow(M[4], 1/qarray[4]))
plt.legend()
plt.xlabel(r'$\tau$')
plt.ylabel(r'$M(q, \tau)^{1/q}$')
plt.xlim(0,)
plt.ylim(0,)
plt.title(r'$M(q, \tau)^{1/q}$ Against $\tau$ For Google')
plt.show()


fig5=plt.figure(figsize=(16,8), dpi=100)
plt.plot(tau, pow(M2[0], 1/qarray[0]), label='q=1')
plt.plot(tau, pow(M2[1], 1/qarray[1]), label='q=5')
plt.plot(tau, pow(M2[2], 1/qarray[2]), label='q=10')
plt.plot(tau, pow(M2[3], 1/qarray[3]), label='q=20')
plt.plot(tau, pow(M2[4], 1/qarray[4]), label='q=40')
```

```
plt.scatter(tau, pow(M2[0], 1/qarray[0]))
plt.scatter(tau, pow(M2[1], 1/qarray[1]))
plt.scatter(tau, pow(M2[2], 1/qarray[2]))
plt.scatter(tau, pow(M2[3], 1/qarray[3]))
plt.scatter(tau, pow(M2[4], 1/qarray[4]))
plt.legend()
plt.xlabel(r'$\tau$')
plt.ylabel(r'$M(q, \tau)^{1/q}$')
plt.ylim(0,)
plt.xlim(0,)
plt.title(r'$M(q, \tau)^{1/q}$ Against $\tau$ For Amazon')
plt.show()

FB=[]
FB2=[]

for i in range(11):
    m5 = np.polyfit(np.log10(tau), np.log10(pow(M[i], 1/qarray[i])), 1)
    m6 = np.polyfit(np.log10(tau), np.log10(pow(M2[i], 1/qarray[i])), 1)
    FB.append(m5[0])
    FB2.append(m6[0])

fig6=plt.figure(figsize=(16,8), dpi=100)
plt.plot(qarray, np.array(FB), label='GOOG')
plt.plot(qarray, np.array(FB2), label='AMAZ')
plt.scatter(qarray, np.array(FB), marker='*')
plt.scatter(qarray, np.array(FB2), marker='x')
plt.title(r'Figure For $\frac{f(q)}{q}$ Against $q$')
plt.xlabel('q')
plt.ylabel(r'$\frac{f(q)}{q}$')
plt.legend()
plt.show()

def MDFA(data,ni,fittime, q):
    n = len(data)//ni
    nf = int(n*ni)

    n_mean =np.mean(data[:nf])
    y = []
    y_hat = []
    for i in range(nf):
        y.append(np.sum(data[:i+1]-n_mean))
    for i in range(int(n)):
        x = np.arange(1,ni+1,1)
        y_temp = y[int(i*ni+1)-1:int((i+1)*ni)]
        coef = np.polyfit(x,y_temp,deg=fittime)
        y_hat.append(np.polyval(coef,x))
```

```
    fn = pow(sum((np.asarray(y)-np.asarray(y_hat).reshape(-1)
    )**q)/nf, 1/q)
    return fn

qarray2=[1, 2,3, 4,5,8, 10, 15, 20, 25,30, 35, 40]
M4=[]
lag=[]
for q in qarray2:
    f3 = []
    for i in range(len(lags)):
        f3.append(MDFA(r1,lags[i],1, q))
    b=pd.DataFrame([], columns=['lags', 'f3'])
    b['lags']=np.log(lags)
    b['f3']=np.log(np.array(f3))
    b=b.dropna()

    m6 = np.polyfit(b['lags'], b['f3'], 1)
    M4.append(m6[0])

fig7=plt.figure(figsize=(16,8), dpi=100)
plt.plot(qarray2, M4)
plt.scatter(qarray2, M4, marker='*', color='r')
plt.xlabel('q', fontsize=20)
plt.ylabel(r'$\alpha(q)$', fontsize=20)
plt.title('Scaling Exponents of Google', fontsize=20)
plt.xlim(0,40)
plt.ylim(0, 1)
plt.show()

M5=[]
lag2=[]
for q in qarray2:
    f4 = []
    for i in range(len(lags)):
        f4.append(MDFA(r2,lags[i],1, q))
    b=pd.DataFrame([], columns=['lags', 'f3'])
    b['lags']=np.log(lags)
    b['f3']=np.log(np.array(f4))
    b=b.dropna()

    m7 = np.polyfit(b['lags'], b['f3'], 1)
    if m7[0]<0:
        m7[0]=0
    M5.append(m7[0])

fig8=plt.figure(figsize=(16,8), dpi=100)
plt.plot(qarray2, M5)
plt.scatter(qarray2, M5, marker='*', color='r')
plt.xlabel('q', fontsize=20)
plt.ylabel(r'$\alpha(q)$', fontsize=20)
```

```
plt.title('Scaling Exponents of Amazon', fontsize=20)
plt.xlim(0,40)
plt.ylim(0, 1)
plt.show()

from pmdarima import auto_arima
from statsmodels.tsa.stattools import adfuller
from statsmodels.tools.eval_measures import mse,rmse
from statsmodels.tsa.statespace.varmax import VARMAX,
    VARMAXResults
df2=pd.DataFrame([], columns=['r1', 'r2'])
df2['r1']=r1[:4026]
df2['r2']=r2


df3=pd.DataFrame([], columns=['r2', 'r1'])
df3['r1']=r1[:4026]
df3['r2']=r2
AIC2=[]
for i in range(1,6):
    for j in range(1,6):
        model = VARMAX(df2, order=(i,j), trend='c') # c
    indicates a constant trend
        results = model.fit(maxiter=1000, disp=False)
        AIC2.append([i,j,results.aic])
print(AIC2)

model2 = VARMAX(df2, order=(1,2), trend='c') # c indicates a
    constant trend
results2 = model2.fit(maxiter=1000, disp=False)

from statsmodels.tsa.stattools import grangercausalitytests
grangercausalitytests(df2, maxlag=4)
grangercausalitytests(df3, maxlag=4)

ffr1=np.fft.fft(r1)
ffr2=np.fft.fft(r2)
ffrr1=np.fft.fftfreq(len(r1))
ffrr2=np.fft.fftfreq(len(r2))
n1=len(r1)
n2=len(r2)
fig8=plt.figure(figsize=(16,8),dpi=100)
plt.plot(ffrr1[:n1//2], abs(ffr1)[:n1//2])
plt.xlabel('Frequency')
plt.ylabel('Magnitude of Coefficients')
plt.xlim(0,0.5)
plt.ylim(0)
plt.title('Magnitude of Coefficients of Fourier Transform of
    Google')

```

```
fig9=plt.figure(figsize=(16,8), dpi=100)
plt.plot(ffrr2[:n2//2], abs(ffr2)[:n2//2])
plt.xlabel('Frequency')
plt.ylabel('Magnitude of Coefficients')
plt.xlim(0,0.5)
plt.ylim(0,)
plt.title('Magnitude of Coefficients of Fourier Transform of
    Amazon')

from scipy import signal

freqs, psd = signal.welch(r1)
plt.figure(figsize=(16, 8))
plt.plot(freqs, psd)
plt.title("PSD of Google")
plt.xlabel("Frequency")
plt.xlim(0,0.5)
plt.ylim(0,)
plt.ylabel("Power")
plt.tight_layout()
plt.show()

freqs2, psd02 = signal.welch(r2)
plt.figure(figsize=(16, 8))
plt.plot(freqs2, psd02)
plt.title("PSD of Amazon")
plt.xlabel("Frequency")
plt.xlim(0,0.5)
plt.ylim(0,)
plt.ylabel("Power")
plt.tight_layout()
plt.show()

freqs, psd = signal.welch(r1)
m6 = np.polyfit(np.log10(freqs[1:]), np.log10(psd[1:]), 1)
plt.figure(figsize=(16, 8))
plt.loglog(freqs, psd, label='psd')
plt.loglog(freqs, (10**m6[1])*freqs**m6[0], label='Fitting
    Curve')
plt.title("PSD of Google")
plt.xlabel("Frequency")
plt.xlim(0,0.5)
plt.ylim(0,)
plt.ylabel("Power")
plt.legend()
plt.tight_layout()
plt.show()
```

```
freqs2, psd02 = signal.welch(r2)
m7 = np.polyfit(np.log10(freqs2[1:]), np.log10(psd02[1:]), 1)
plt.figure(figsize=(16, 8))
plt.loglog(freqs2, psd02, label='psd')
plt.loglog(freqs2, (10**m7[1])*freqs2**m7[0], label='Fitting
    Curve')
plt.title("PSD of Amazon")
plt.xlabel("Frequency")
plt.xlim(0,0.5)
plt.ylim(0,)
plt.legend()
plt.ylabel("Power")
plt.tight_layout()
plt.show()

import emd
t=np.arange(len(r1))
imf = emd.sift.sift(r1)
print(imf.shape)
ind=np.array([0,1,3, 5,8])
imfs=imf.T[ind].T
emd.plotting.plot_imfs(imfs)
plt.title('IMFs of Google')

imf2 = emd.sift.sift(r2)
print(imf2.shape)
ind2=np.array([0,1,3, 5,7])
imfs2=imf2.T[ind2].T
emd.plotting.plot_imfs(imfs2)
plt.title('IMFs of Amazon')

Hur1=[]
for i in range(5):
    m, lag, rs=hurst(imfs.T[i])
    if m[0]<0:
        m[0]=0
    elif m[0]>1:
        m[0]=1
    Hur1.append(m[0])
Hur2=[]
for i in range(5):
    m, lag, rs=hurst(imfs2.T[i])
    if m[0]<0:
        m[0]=0
    elif m[0]>1:
        m[0]=1
    Hur2.append(m[0])

fig2=plt.figure(figsize=(16,8), dpi=100)
plt.plot(ind+1,Hur1, label='GOOG')
```

```
plt.plot(ind2+1, Hur2, label='AMAZ')
plt.title('Hurst Exponents of Each IMF')
plt.legend()
plt.ylim(0,1)
plt.xlabel('Order')
plt.ylabel('Hurst Exponent')


fr1, psd1=signal.welch(imfs.T[0])
fr2, psd2=signal.welch(imfs.T[1])
fig3=plt.figure(figsize=(16,8), dpi=100)
plt.plot(fr1, psd1, label='IMF 1')
plt.plot(fr2, psd2, label='IMF 2')
plt.legend()
plt.ylabel('Power')
plt.xlabel('Frequency')
plt.xlim(0,0.5)
plt.ylim(0,)
plt.title('PSD of First Two IMFs of Google')

fr3, psd3=signal.welch(imfs2.T[0])
fr4, psd4=signal.welch(imfs2.T[1])
fig4=plt.figure(figsize=(16,8), dpi=100)
plt.plot(fr3, psd3, label='IMF 1')
plt.plot(fr4, psd4, label='IMF 2')
plt.legend()
plt.ylabel('Power')
plt.xlabel('Frequency')
plt.xlim(0,0.5)
plt.ylim(0,)
plt.title('PSD of First Two IMFs of Amazon')


r3=r1-imfs.T[0]
r4=r1-imfs.T[1]-imfs.T[0]
r5=r2-imfs2.T[0]
r6=r2-imfs2.T[1]-imfs2.T[0]

fr1, psd1=signal.welch(r3)
fr2, psd2=signal.welch(r4)
fig3=plt.figure(figsize=(16,8), dpi=100)
plt.plot(fr1, psd1, label=r'X-$c_1$')
plt.plot(fr2, psd2, label=r'X-$c_1$-$c_2$')
plt.plot(freqs, psd, label='X')
plt.legend()
plt.ylabel('Power')
plt.xlabel('Frequency')
plt.xlim(0,0.5)
plt.ylim(0,)
plt.title('PSD Comparision of Google')
```

```

fr3, psd3=signal.welch(r5)
fr4, psd4=signal.welch(r6)
fig4=plt.figure(figsize=(16,8), dpi=100)
plt.plot(fr3, psd3, label=r'X-$c_1$')
plt.plot(fr4, psd4, label=r'X-$c_1$-$c_2$')
plt.plot(freqs2, psd02, label='X')
plt.legend()
plt.ylabel('Power')
plt.xlabel('Frequency')
plt.xlim(0,0.5)
plt.ylim(0,)
plt.title('PSD Comparision of Amazon')
```