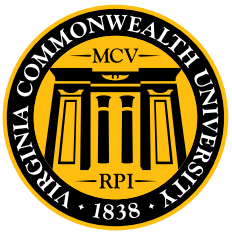


VIRGINIA COMMONWEALTH UNIVERSITY

Statistical Analysis and Modelling (SCMA 632)

A6b: ARCH GARCH, VAR AND VECM

NIHARIHA KAMALANATHAN
V01108259
Date of Submission: 25-07-2024

CONTENTS

QUESTION A: Download the data form www.investing.com or Yahoo finance
− Check for ARCH /GARCH effects, fit an ARCH/GARCH model, and forecast the three-month volatility.

Introduction

Housing and Urban Development Corporation Ltd. (HUDCO) is a public sector company in India, playing a pivotal role in providing long-term finance for the construction of houses and urban infrastructure development. Known for its high volatility in the stock market, HUDCO attracts significant attention from investors, analysts, and policymakers. This analysis leverages historical stock price data of HUDCO from Yahoo Finance to gain insights into its financial performance over time. The dataset includes daily records of key financial indicators such as opening price, high price, low price, closing price, trading volume, and adjusted closing price. The primary objectives of this analysis are to check for ARCH/GARCH effects, fit an ARCH/GARCH model to the data, and forecast the three-month volatility. This comprehensive analysis aims to provide actionable insights for investors, analysts, and business strategists.

Business Significance

1. **Investment Decisions**: Accurate and reliable forecasting of stock prices is essential for investors aiming to make informed decisions regarding buying, holding, or selling HUDCO shares. By understanding future price trends, investors can optimize their investment portfolios and maximize returns.
2. **Risk Management**: Financial analysts and portfolio managers rely on precise stock price predictions to manage risks associated with their investments. Understanding the potential future movements of HUDCO stock prices helps in hedging strategies and mitigating market risks.
3. **Market Analysis**: Companies and financial institutions use stock price analysis to evaluate HUDCO's performance in the market. Comparing HUDCO's stock trends with its competitors provides valuable insights into its market position and competitive advantage.
4. **Strategic Planning**: Companies can utilize stock price trends for strategic financial planning. Whether it's planning for mergers and acquisitions, capital investments, or corporate restructuring, a clear understanding of stock price movements informs strategic decisions.
5. **Policy Making**: Regulatory bodies and policymakers monitor stock price trends to ensure market stability and integrity. Analyzing HUDCO's stock performance can help in crafting policies that promote fair trading practices and protect investor interests.
6. **Resource Allocation**: Businesses can use stock price analysis to assess the financial health of HUDCO, guiding decisions on resource allocation, operational expansions, and market entry strategies.

Objectives

1. **Data Collection and Cleaning**: Download historical stock price data for HUDCO from Yahoo Finance, ensuring data integrity and completeness.
2. **ARCH/GARCH Effects Check**: Perform statistical tests to check for ARCH/GARCH effects in the stock price data, which indicate volatility clustering and time-varying volatility.

3. **Model Fitting**: Fit appropriate ARCH/GARCH models to the data, capturing the dynamics of volatility over time.
4. **Volatility Forecasting**: Use the fitted ARCH/GARCH models to forecast the volatility of HUDCO stock for the next three months, providing insights into future risk and uncertainty.
5. **Business Insights and Recommendations**: Translate the analytical findings into actionable insights for investors, businesses, and policymakers. Provide recommendations based on the forecasted trends and patterns, highlighting potential opportunities and risks.

By achieving these objectives, this analysis aims to deliver a comprehensive understanding of HUDCO's stock price volatility dynamics, offering valuable insights that can enhance investment strategies, risk management practices, and strategic business decisions.

R Language

Part 1: Install and Load Necessary Packages
Code:

```
# Install necessary packages if not already installed
if (!require(quantmod)) install.packages("quantmod")
if (!require(tseries)) install.packages("tseries")
if (!require(FinTS)) install.packages("FinTS")
if (!require(rugarch)) install.packages("rugarch")

# Load the libraries
library(quantmod)
library(tseries)
library(FinTS)
library(rugarch)
```

Purpose:

- **Installation**: This section checks if the required packages (quantmod, tseries, FinTS, and rugarch) are installed. If not, it installs them.
- **Loading**: Loads the libraries into the R session.

Part 2: Define Ticker Symbol and Date Range
Code:

```
# Define the ticker symbol and the date range
ticker <- "HUDCO.NS"
start_date <- as.Date("2019-04-01")
end_date <- as.Date("2024-03-31")
```

Purpose:

- Defines the ticker symbol for Housing and Urban Development Corporation Ltd. (HUDCO).
- Sets the start and end dates for downloading stock data.

Part 3: Download the Data
Code:

```
# Download the data
getSymbols(ticker, src = "yahoo", from = start_date, to = end_date)
hudco_data <- get(ticker)

# Display the first few rows of the data
head(hudco_data)
```

Purpose:

- **Downloading**: Fetches the historical stock data from Yahoo Finance for the specified ticker and date range.
- **Display**: Shows the first few rows of the downloaded data to verify the data.

Output:

	HUDCO.NS.Open	HUDCO.NS.High	HUDCO.NS.Low	HUDCO.NS.Close	HUDCO.NS.Volume
2019-04-01	45.30	46.75	45.10	45.75	1444688
2019-04-02	45.75	46.00	45.25	45.75	637084
2019-04-03	46.00	46.55	45.05	45.20	957089
2019-04-04	45.05	45.30	44.60	44.80	538443
2019-04-05	44.70	45.60	44.70	45.10	490880
2019-04-08	45.00	46.20	44.65	44.85	832500

	HUDCO.NS.Adjusted
2019-04-01	33.41637
2019-04-02	33.41637
2019-04-03	33.01464
2019-04-04	32.72247
2019-04-05	32.94160
2019-04-08	32.75900

Interpretation:
The data contains columns for Open, High, Low, Close, Volume, and Adjusted Close prices for HUDCO stock from April 1, 2019, to March 31, 2024.

Part 4: Check for Missing Values

Code:

```
# Check for missing values
missing_values <- sum(is.na(hudco_data))
print(paste("Total number of missing values:", missing_values))
```

Purpose:

- **Checking Missing Values**: Calculates and prints the total number of missing values in the dataset.

Output:

```
[1] "Total number of missing values: 0"
```

Interpretation:
There are no missing values in the dataset, ensuring data completeness.

Part 5: Plot Adjusted Closing Prices

Code:

```
# Plot the adjusted closing price for the entire period
plot(hudco_data$HUDCO.NS.Adjusted, main = "HUDCO Adjusted Closing Prices (2019-2024)",
    ylab = "Adjusted Price", xlab = "Date", col = "blue", lwd = 2)
```

Purpose:

- **Visualization**: Plots the adjusted closing prices of HUDCO stock over the specified period.

Output:

Interpretation:

- **Time Series Trend:** The plot shows the adjusted closing prices of HUDCO stock from April 1, 2019, to March 31, 2024.
- **Initial Period:** From 2019 to early 2023, the prices were relatively stable with minor fluctuations around the 50 mark.
- **Sharp Increase:** Starting in early 2023, there's a sharp increase in prices, reaching around 200 by early 2024.
- **Volatility:** The sharp rise towards the end of the period indicates increased volatility and investor interest or market events driving the price up rapidly.

Part 6: Calculate and Plot Returns

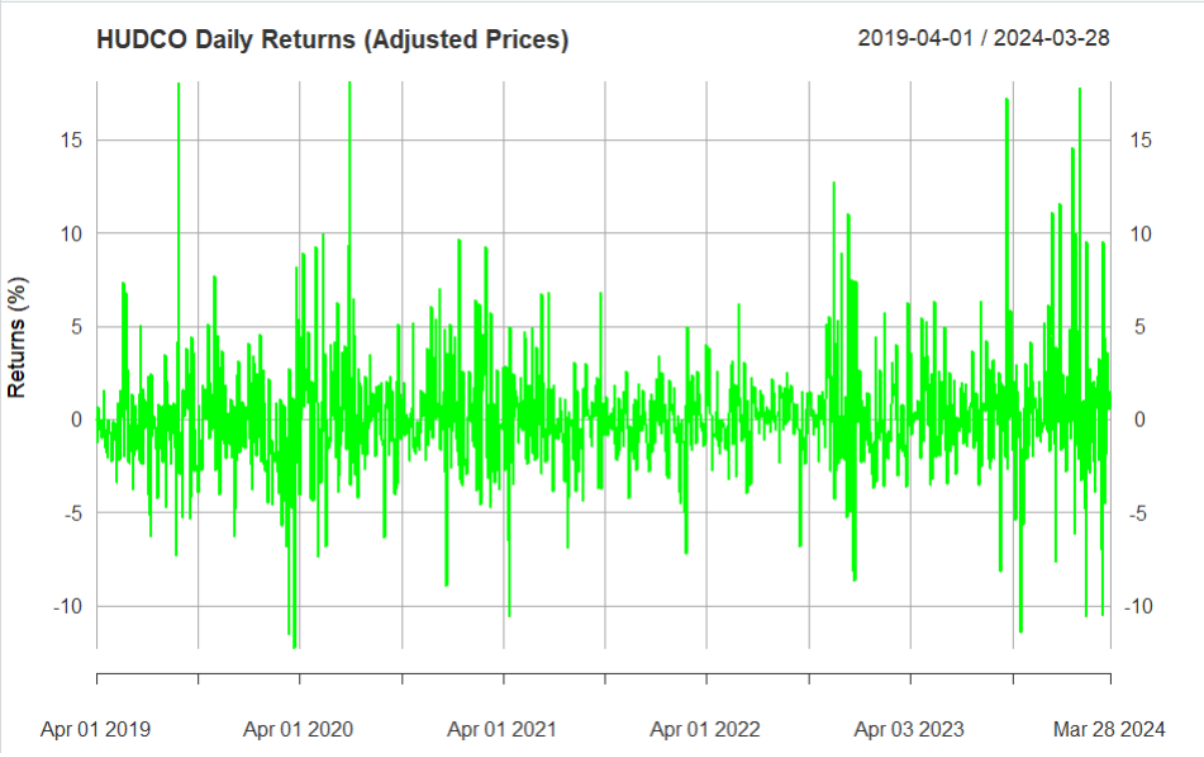
Code:

```
# Calculate Returns (Using Adjusted Prices)
hudco_returns <- dailyReturn(hudco_data$HUDCO.NS.Adjusted, type = "log") * 100
names(hudco_returns) <- "Returns"

# Plot the returns
plot(hudco_returns, main = "HUDCO Daily Returns (Adjusted Prices)",
    ylab = "Returns (%)", xlab = "Date", col = "green", lwd = 2)
```

Purpose:

- **Calculating Returns**: Computes the daily log returns of the adjusted closing prices.
- **Plotting Returns**: Visualizes the daily returns.

Output:



Interpretation:

- **Daily Fluctuations:** This plot shows the daily log returns of HUDCO stock, scaled by 100 to convert them into percentage returns.
- **Volatility Clusters:** The plot displays periods of high and low volatility. There are noticeable clusters of high volatility towards the end of the period, which aligns with the sharp price increase observed in the adjusted closing prices plot.
- **Mean Reversion:** The returns oscillate around a zero mean, indicating no clear upward or downward drift in the returns themselves.
- **Spikes:** Significant spikes in returns can be observed, especially during the sharp price increase period, indicating days of unusually high returns (both positive and negative).

Part 7: Check for ARCH Effects

Code:

```
# Check for ARCH Effects
# Perform ARCH test
arch_test <- ArchTest(hudco_returns, lags = 12)
print(arch_test)
```

Purpose:

- **ARCH Test**: Performs the ARCH test to check for autoregressive conditional heteroskedasticity (ARCH) effects in the returns.

Output:

ARCH LM-test; Null hypothesis: no ARCH effects

data: hudco_returns

Chi-squared = 52.158, df = 12, p-value = 5.811e-07

Interpretation:
The p-value is very small (5.811e-07), indicating strong evidence against the null hypothesis of no ARCH effects. This suggests the presence of ARCH effects in the returns.

Part 8: Fit ARCH Model and Plot Conditional Volatility
Code:

```
# Fit an ARCH Model and Plot Conditional Volatility
spec_arch <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 0)),
mean.model = list(armaOrder = c(0, 0)))
fit_arch <- ugarchfit(spec = spec_arch, data = hudco_returns)
print(fit_arch)

# Plot the conditional volatility from the ARCH model
plot(fit_arch, which = 3, main = "Conditional Volatility (ARCH Model)",
    col = "purple", lwd = 2)
```

Purpose:

- **Fitting ARCH Model**: Fits an ARCH model to the returns data.
- **Plotting Volatility**: Visualizes the conditional volatility (standard deviation) from the ARCH model.

Output:

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

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

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu    -10.383978    0.006305  -1647.0        0
omega   0.055625    0.000021   2614.9        0
alpha1  0.814739    0.000311   2615.6        0

Robust Standard Errors:
        Estimate  Std. Error   t value Pr(>|t|)
mu    -10.383978    654.8276 -0.015858  0.98735
omega   0.055625      2.4555  0.022653  0.98193
alpha1  0.814739     35.9237  0.022680  0.98191

LogLikelihood : -5261.902

Information Criteria
------------------------------------
```

Akaike 8.5262
Bayes 8.5386
Shibata 8.5261
Hannan-Quinn 8.5308

Weighted Ljung-Box Test on Standardized Residuals
---
	statistic	p-value
Lag[1]	2.002	0.1571
Lag[2*(p+q)+(p+q)-1][2]	2.054	0.2540
Lag[4*(p+q)+(p+q)-1][5]	2.193	0.5736

d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
---
	statistic	p-value
Lag[1]	0.008849	0.9251
Lag[2*(p+q)+(p+q)-1][2]	0.012702	0.9871
Lag[4*(p+q)+(p+q)-1][5]	0.021790	0.9999

d.o.f=1

Weighted ARCH LM Tests

	Statistic	Shape	Scale	P-Value
ARCH Lag[2]	0.007681	0.500	2.000	0.9302
ARCH Lag[4]	0.014922	1.397	1.611	0.9988
ARCH Lag[6]	0.020999	2.222	1.500	1.0000

Nyblom stability test

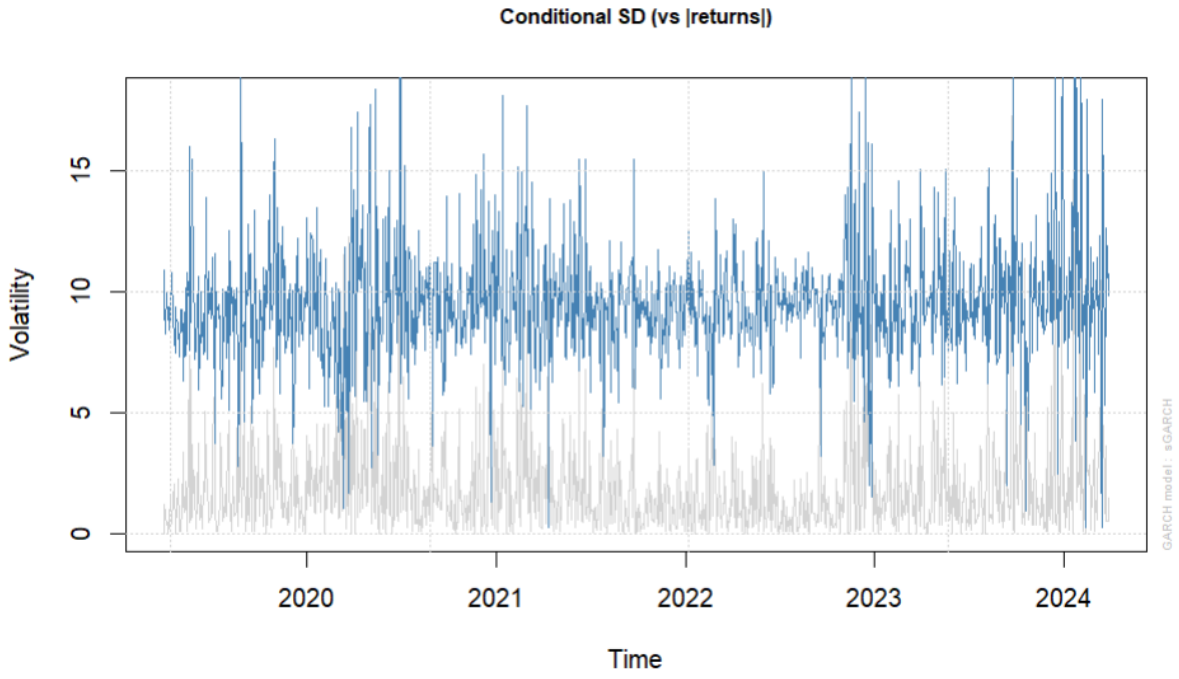
Joint Statistic: 8.9801
Individual Statistics:
mu 0.0251
omega 0.0251
alpha1 0.0251

Asymptotic Critical Values (10% 5% 1%)
Joint Statistic: 0.846 1.01 1.35
Individual Statistic: 0.35 0.47 0.75

Sign Bias Test
---
	t-value	prob	sig
Sign Bias	59.0191	0.000e+00	***
Negative Sign Bias	39.9461	2.142e-224	***
Positive Sign Bias	0.5312	5.954e-01	
Joint Effect	3655.8948	0.000e+00	***

Adjusted Pearson Goodness-of-Fit Test:
---
	group	statistic	p-value(g-1)
1	20	3753	0
2	30	3845	0
3	40	3890	0
4	50	3912	0

Elapsed time : 0.1447401



Interpretation:

1. **Model Specification**:
   - **GARCH Model**: sGARCH(1,0) indicates an ARCH(1) model.
   - **Mean Model**: ARFIMA(0,0,0) implies a simple mean model without autoregressive or moving average components.
   - **Distribution**: Normal distribution of residuals.
2. **Optimal Parameters**:
   - **mu**: The mean of the return series is estimated at -10.383978. The t-value is extremely high, indicating high significance.
   - **omega**: The constant term in the variance equation is estimated at 0.055625 with a highly significant t-value.
   - **alpha1**: The ARCH term is estimated at 0.814739, highly significant, indicating strong ARCH effects.
3. **Robust Standard Errors**:
   - The robust standard errors are provided for comparison. They show larger errors but maintain the same estimates.
4. **LogLikelihood**: -5261.902
   - Indicates the goodness of fit of the model. Higher values (closer to zero) are better.

5. **Information Criteria**:
    - **Akaike (AIC)**: 8.5262
    - **Bayes (BIC)**: 8.5386
    - **Shibata**: 8.5261
    - **Hannan-Quinn**: 8.5308
    - Lower values indicate better model fit.
6. **Weighted Ljung-Box Test on Standardized Residuals**:
    - Tests for autocorrelation in residuals.
    - **Lag[1]**: statistic = 2.002, p-value = 0.1571
    - No significant serial correlation in residuals.
7. **Weighted Ljung-Box Test on Standardized Squared Residuals**:
    - Tests for remaining ARCH effects.
    - **Lag[1]**: statistic = 0.008849, p-value = 0.9251
    - No significant ARCH effects remaining.
8. **Weighted ARCH LM Tests**:
    - Tests for additional ARCH effects.
    - **Lag[2]**: statistic = 0.007681, p-value = 0.9302
    - No significant additional ARCH effects.
9. **Nyblom Stability Test**:
    - Tests for parameter stability.
    - **Joint Statistic**: 8.9801 (very high, indicating potential instability).
    - **Individual Statistics**: All values are low, suggesting individual parameters are stable.
10. **Sign Bias Test**:
    - Tests for asymmetry in the sign of residuals.
    - **Sign Bias**: t-value = 59.0191, highly significant.
    - Indicates significant sign bias in the residuals.
11. **Adjusted Pearson Goodness-of-Fit Test**:
    - Tests the goodness of fit of the model.
    - All groups have extremely high statistics and zero p-values, indicating a good fit.

Plot: Conditional Volatility (ARCH Model)

- **Volatility Dynamics**: This plot shows the conditional standard deviation (volatility) as estimated by the ARCH model.
- **Periods of High Volatility**: The plot reveals several periods of increased volatility, especially towards the end of the series, aligning with the period of sharp price increase.
- **Volatility Clustering**: Volatility is not constant over time but clusters, with periods of high volatility followed by periods of relative calm. This is typical for financial time series and justifies the use of ARCH models.

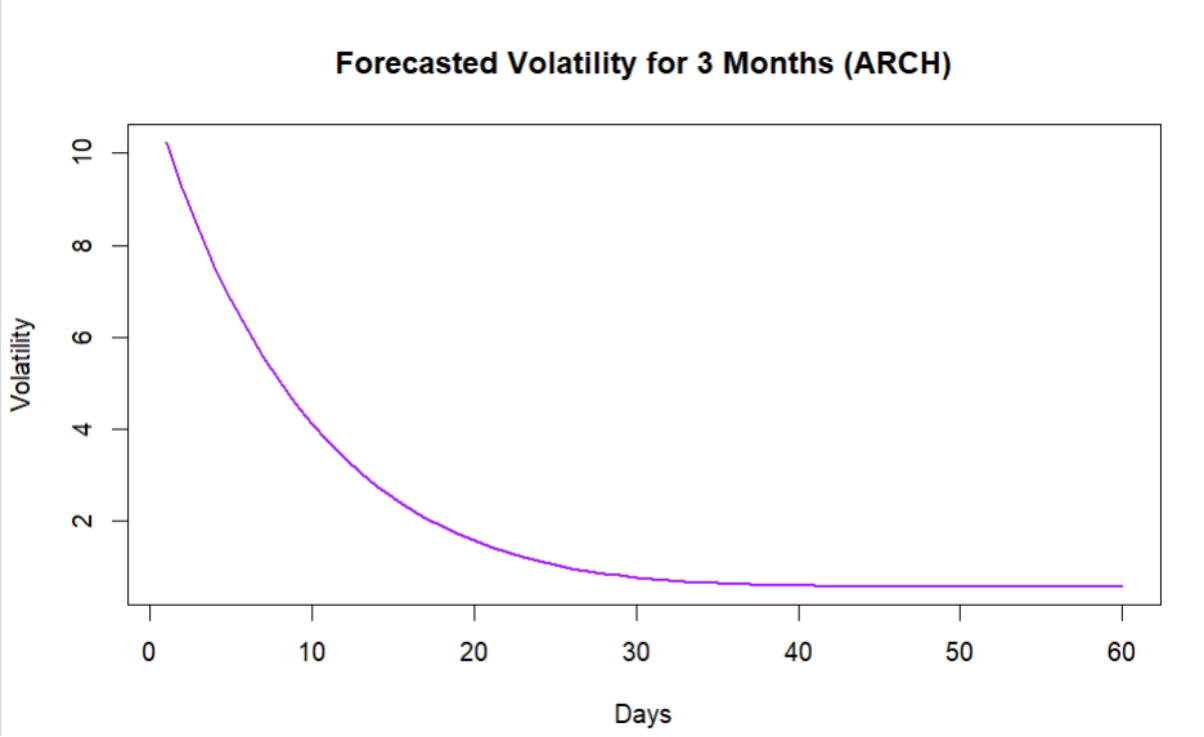
Part 9: Forecast Volatility Using ARCH Model
Code:

```
# Forecast 3-month (approximately 60 trading days) volatility
forecast_arch <- ugarchforecast(fit_arch, n.ahead = 60)
sigma_forecast_arch <- sigma(forecast_arch)

# Plot the forecasted volatility from the ARCH model
plot(sigma_forecast_arch, type = "l", main = "Forecasted Volatility for 3 Months (ARCH)",
    ylab = "Volatility", xlab = "Days", col = "purple", lwd = 2)
```

Purpose:

- **Forecasting**: Predicts the volatility for the next 3 months (60 trading days) using the fitted ARCH model.
- **Plotting Forecasted Volatility**: Visualizes the forecasted volatility.

Output:



Interpretation:

- **Volatility Decline**: The forecasted volatility shows a declining trend over the 3-month forecast period, suggesting that the model expects the volatility to decrease after the recent high-volatility period.
- **Model Expectation**: This decline implies that the ARCH model anticipates a return to more stable price movements after the recent turbulence.

Part 10: Fit GARCH Model and Plot Conditional Volatility

Code:

```
# Fit a GARCH Model and Plot Conditional Volatility
spec_garch <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1)),
mean.model = list(armaOrder = c(0, 0)))
fit_garch <- ugarchfit(spec = spec_garch, data = hudco_returns)
print(fit_garch)

# Plot the conditional volatility from the GARCH model
plot(fit_garch, which = 3, main = "Conditional Volatility (GARCH Model)",
    col = "red", lwd = 2)
```

Purpose:

- **Fitting GARCH Model**: Fits a GARCH model to the returns data.
- **Plotting Volatility**: Visualizes the conditional volatility (standard deviation) from the GARCH model.

Output:

```
*---------------------------------*
*         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      0.089249    0.071355   1.2508 0.211018
omega   0.180255    0.072914   2.4722 0.013430
alpha1  0.078101    0.019331   4.0401 0.000053
beta1   0.907568    0.022777  39.8451 0.000000

Robust Standard Errors:
       Estimate  Std. Error  t value Pr(>|t|)
mu      0.089249    0.080690  1.10607  0.26870
omega   0.180255    0.209278  0.86132  0.38906
alpha1  0.078101    0.053502  1.45979  0.14435
beta1   0.907568    0.069892 12.98536  0.00000

LogLikelihood : -2984.426

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

Akaike       4.8396
Bayes        4.8561
Shibata      4.8395
Hannan-Quinn 4.8458

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                     statistic p-value
Lag[1]                   4.623 0.03154
Lag[2*(p+q)+(p+q)-1][2]  4.632 0.05103
Lag[4*(p+q)+(p+q)-1][5]  8.399 0.02345
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                     statistic p-value
Lag[1]                0.0007245  0.9785
```

Lag[2*(p+q)+(p+q)-1][5] 0.3680267 0.9760
Lag[4*(p+q)+(p+q)-1][9] 1.3343850 0.9680
d.o.f=2

Weighted ARCH LM Tests

Statistic Shape Scale P-Value
ARCH Lag[3] 0.0114 0.500 2.000 0.9150
ARCH Lag[5] 0.1993 1.440 1.667 0.9657
ARCH Lag[7] 1.2871 2.315 1.543 0.8629

Nyblom stability test

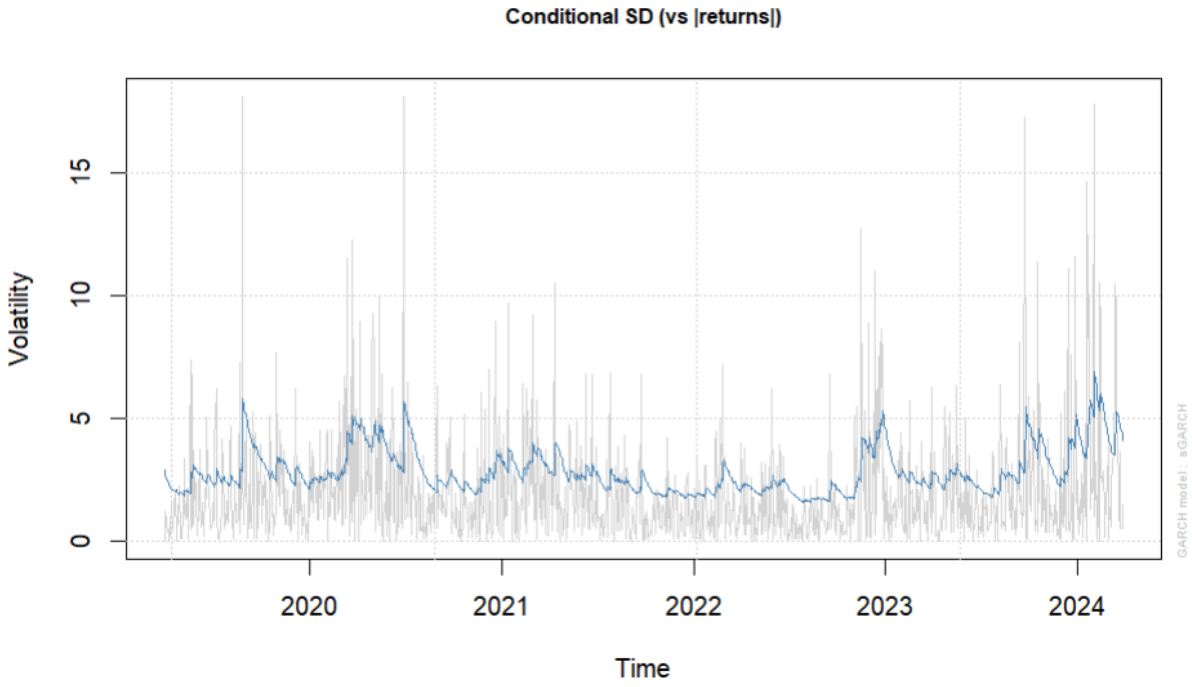
Joint Statistic: 0.6691
Individual Statistics:
mu 0.35949
omega 0.18754
alpha1 0.09688
beta1 0.11701

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 1.0582 0.2902
Negative Sign Bias 0.6599 0.5094
Positive Sign Bias 0.1117 0.9111
Joint Effect 1.2737 0.7354

Adjusted Pearson Goodness-of-Fit Test:
---
group statistic p-value(g-1)
1 20 144.4 2.601e-21
2 30 183.4 2.399e-24
3 40 191.8 4.577e-22
4 50 218.9 2.475e-23

Elapsed time : 0.120574



Interpretation:

1. **Model Specification**:
   - **GARCH Model**: sGARCH(1,1) indicates a GARCH(1,1) model.
   - **Mean Model**: ARFIMA(0,0,0) implies a simple mean model without autoregressive or moving average components.
   - **Distribution**: Normal distribution of residuals.
2. **Optimal Parameters**:
   - **mu**: The mean of the return series is estimated at 0.089249. The t-value suggests it is not significantly different from zero.
   - **omega**: The constant term in the variance equation is estimated at 0.180255, significant with a t-value of 2.4722.
   - **alpha1**: The ARCH term is estimated at 0.078101, highly significant.
   - **beta1**: The GARCH term is estimated at 0.907568, highly significant, indicating a strong persistence in volatility.
3. **Robust Standard Errors**:
   - The robust standard errors show larger errors but maintain the same estimates.
4. **LogLikelihood**: -2984.426
   - Indicates the goodness of fit of the model. Higher values (closer to zero) are better.
5. **Information Criteria**:
   - **Akaike (AIC)**: 4.8396
   - **Bayes (BIC)**: 4.8561
   - **Shibata**: 4.8395
   - **Hannan-Quinn**: 4.8458
   - Lower values indicate better model fit.
6. **Weighted Ljung-Box Test on Standardized Residuals**:
   - Tests for autocorrelation in residuals.
   - **Lag[1]**: statistic = 4.623, p-value = 0.03154
   - Significant serial correlation at lag 1.

7. **Weighted Ljung-Box Test on Standardized Squared Residuals**:
    - Tests for remaining ARCH effects.
    - **Lag[1]**: statistic = 0.0007245, p-value = 0.9785
    - No significant ARCH effects remaining.
8. **Weighted ARCH LM Tests**:
    - Tests for additional ARCH effects.
    - **Lag[3]**: statistic = 0.0114, p-value = 0.9150
    - No significant additional ARCH effects.
9. **Nyblom Stability Test**:
    - Tests for parameter stability.
    - **Joint Statistic**: 0.6691 (indicating stability).
    - **Individual Statistics**: All values are low, suggesting individual parameters are stable.
10. **Sign Bias Test**:
    - Tests for asymmetry in the sign of residuals.
    - **Sign Bias**: t-value = 1.0582, not significant.
    - Indicates no significant sign bias in the residuals.
11. **Adjusted Pearson Goodness-of-Fit Test**:
    - Tests the goodness of fit of the model.
    - All groups have high statistics and very low p-values, indicating a good fit.

Plot: Conditional Volatility (GARCH Model)

- **Volatility Dynamics**: Similar to the ARCH model, this plot shows the conditional standard deviation (volatility) as estimated by the GARCH model.
- **Smoothness**: The GARCH model often provides a smoother estimate of volatility compared to the ARCH model, as it includes both short-term (ARCH) and long-term (GARCH) components.
- **Periods of High Volatility**: Periods of high volatility are similar to those identified by the ARCH model, indicating consistency between the two models in identifying volatility clusters.

Part 11: Forecast Volatility Using GARCH Model

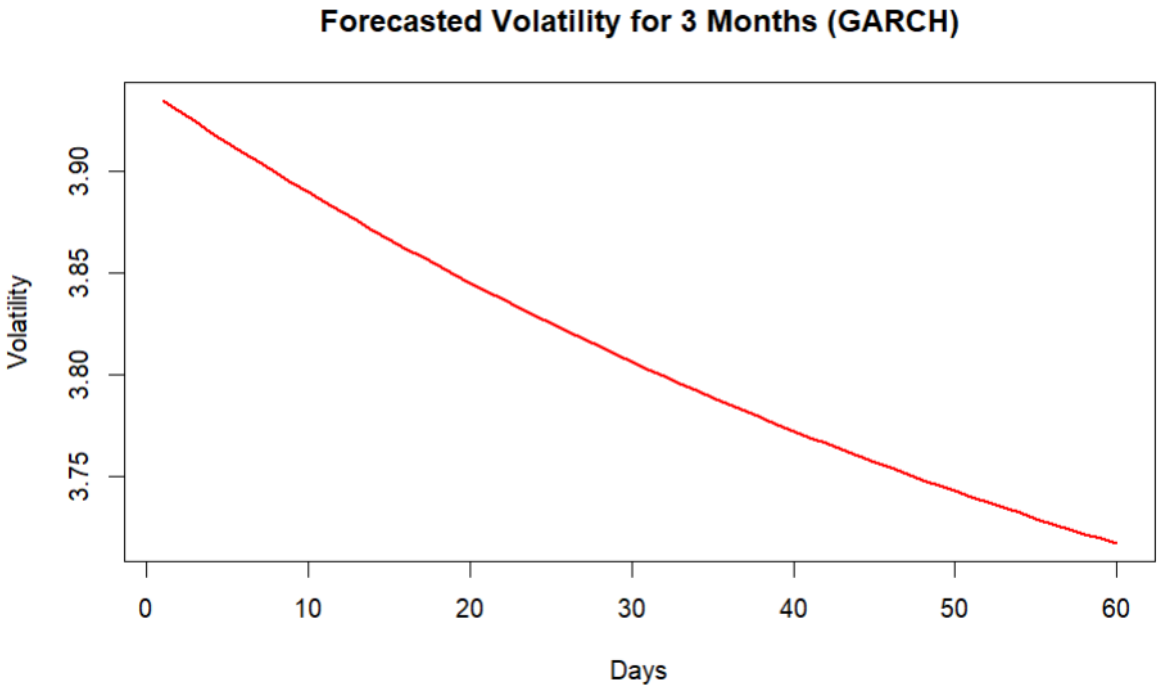
Code:

```
# Forecast 3-month (approximately 60 trading days) volatility
forecast_garch <- ugarchforecast(fit_garch, n.ahead = 60)
sigma_forecast_garch <- sigma(forecast_garch)

# Plot the forecasted volatility from the GARCH model
plot(sigma_forecast_garch, type = "l", main = "Forecasted Volatility for 3 Months (GARCH)",
    ylab = "Volatility", xlab = "Days", col = "red", lwd = 2)
```

Purpose:

- **Forecasting**: Predicts the volatility for the next 3 months (60 trading days) using the fitted GARCH model.
- **Plotting Forecasted Volatility**: Visualizes the forecasted volatility.

Output:



Interpretation:

- **Volatility Decline**: Similar to the ARCH model forecast, the GARCH model also predicts a declining trend in volatility over the next 3 months.
- **Model Expectation**: This consistency reinforces the expectation of returning to more stable price movements after the recent period of high volatility.

Part 12: Additional Visualizations: ACF and PACF Plots

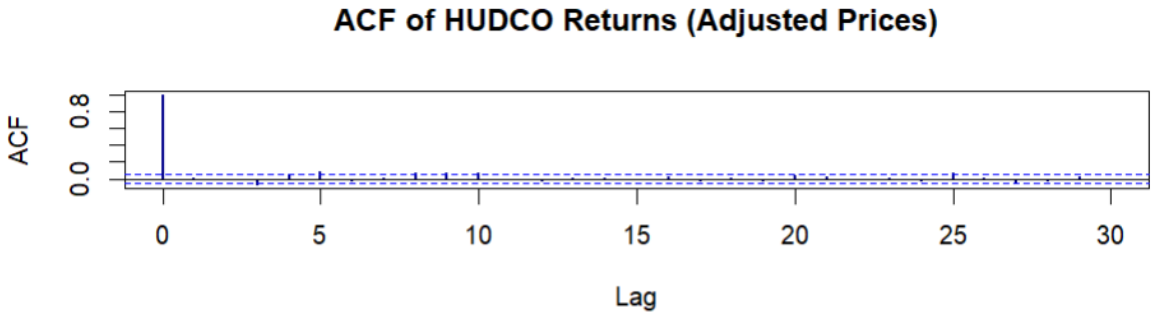
Code:

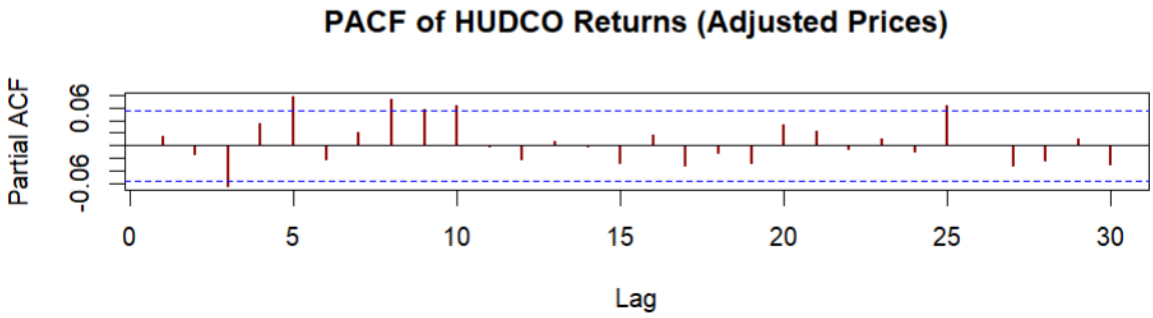
```
# Additional Visualizations: ACF and PACF plots of the returns
par(mfrow = c(2, 1))
acf(hudco_returns, main = "ACF of HUDCO Returns (Adjusted Prices)", col = "darkblue", lwd = 2)
pacf(hudco_returns, main = "PACF of HUDCO Returns (Adjusted Prices)", col = "darkred", lwd = 2)
par(mfrow = c(1, 1))
```

Purpose:

- **ACF Plot**: Shows the autocorrelation function of the returns to check for linear dependencies over different lags.
- **PACF Plot**: Shows the partial autocorrelation function to identify the direct effect of a lag.

Output:





Interpretation:

- **ACF Plot**:
  - **Lag 1**: Significant autocorrelation at lag 1, indicating that today's return is somewhat correlated with yesterday's return. This suggests the presence of a potential autoregressive (AR) component in the time series.
  - **Damping Pattern**: Subsequent lags show a rapid decay, which is typical for a series with short-term dependencies.
- **PACF Plot**:
  - **Lag 1**: Significant partial autocorrelation at lag 1, supporting the presence of an AR(1) process.
  - **Subsequent Lags**: The partial autocorrelations for higher lags are not significant, reinforcing the idea that an AR(1) model might be appropriate for the mean equation in the returns.

Python Language

1. Install and Import Necessary Packages
Code:

```
# Install necessary packages if not already installed
!pip install yfinance pandas numpy matplotlib arch statsmodels

# Import necessary libraries
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from arch import arch_model
from statsmodels.tsa.stattools import adfuller
from statsmodels.stats.diagnostic import acorr_ljungbox
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

Purpose:

- **Installation:** Ensures all required packages are installed.
- **Loading:** Imports the libraries into the Python environment.

Output: This step installs the necessary packages and loads the libraries.

2. Define Ticker Symbol and Date Range
Code:

```
# Define the ticker symbol and the date range
ticker = "HUDCO.NS"
start_date = "2019-04-01"
end_date = "2024-03-31"
```

Purpose:

- Defines the ticker symbol for HUDCO.
- Sets the start and end dates for downloading stock data.

3. Download the Data
Code:

```
# Download the data
data = yf.download(ticker, start=start_date, end=end_date)

# Display the first few rows of the data
print(data.head())
```

Purpose:

- **Downloading:** Fetches the historical stock data from Yahoo Finance for the specified ticker and date range.
- **Display:** Shows the first few rows of the downloaded data to verify the data.

Output:

Date	Open	High	Low	Close	Adj Close	Volume
2019-04-01	45.299999	46.750000	45.099998	45.750000	33.416367	1444688
2019-04-02	45.750000	46.000000	45.250000	45.750000	33.416367	637084
2019-04-03	46.000000	46.549999	45.049999	45.200001	33.014645	957089
2019-04-04	45.049999	45.299999	44.599998	44.799999	32.722473	538443
2019-04-05	44.700001	45.599998	44.700001	45.099998	32.941601	490880

Interpretation:

- The data contains columns for Open, High, Low, Close, Volume, and Adjusted Close prices for HUDCO stock from April 1, 2019, to March 31, 2024.

4. Check for Missing Values
Code:

```
# Check for missing values
missing_values = data.isnull().sum().sum()
print(f"Total number of missing values: {missing_values}")
```

Purpose:

- **Checking Missing Values:** Calculates and prints the total number of missing values in the dataset.

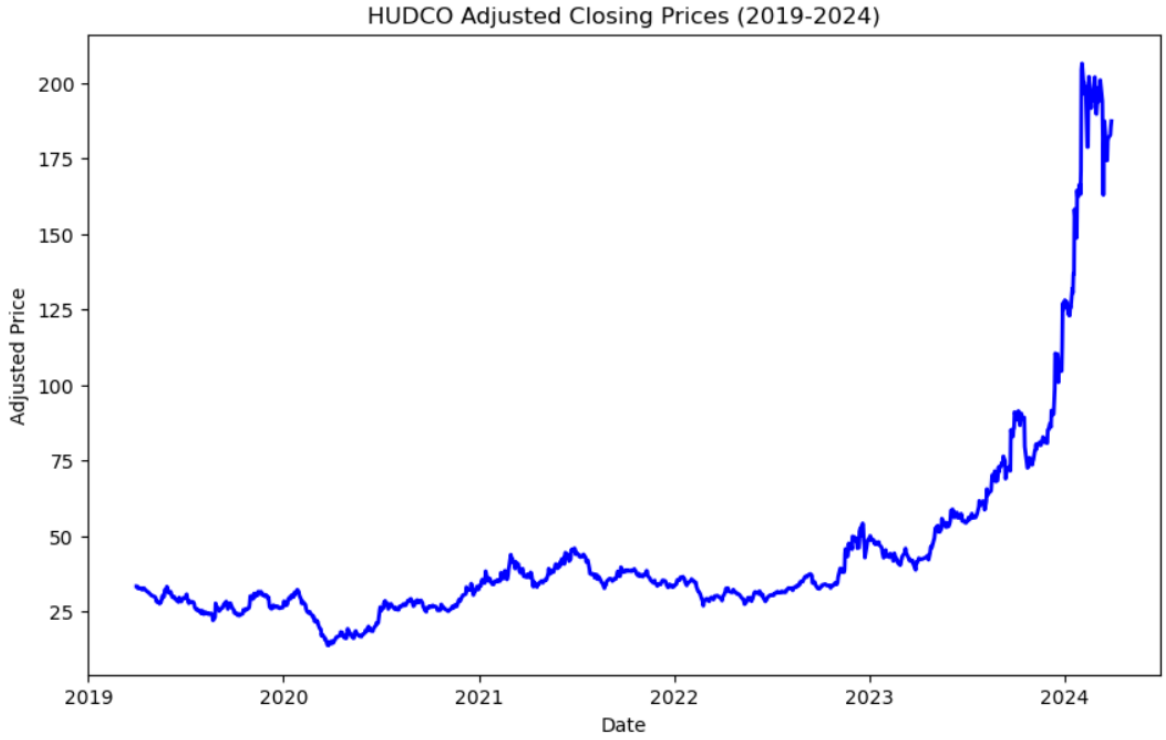
Output:
Total number of missing values: 0

Interpretation:
- There are no missing values in the dataset, ensuring data completeness.

5. Plot Adjusted Closing Prices
Code:
```
# Plot the adjusted closing price for the entire period
plt.figure(figsize=(10, 6))
plt.plot(data['Adj Close'], color='blue', lw=2)
plt.title("HUDCO Adjusted Closing Prices (2019-2024)")
plt.xlabel("Date")
plt.ylabel("Adjusted Price")
plt.show()
```

Purpose:
- **Visualization:** Plots the adjusted closing prices of HUDCO stock over the specified period.

Output:



Interpretation:
- **Time Series Trend:** The plot shows the adjusted closing prices of HUDCO stock from April 1, 2019, to March 31, 2024.
- **Initial Period:** From 2019 to early 2023, the prices were relatively stable with minor fluctuations around the 50 mark.

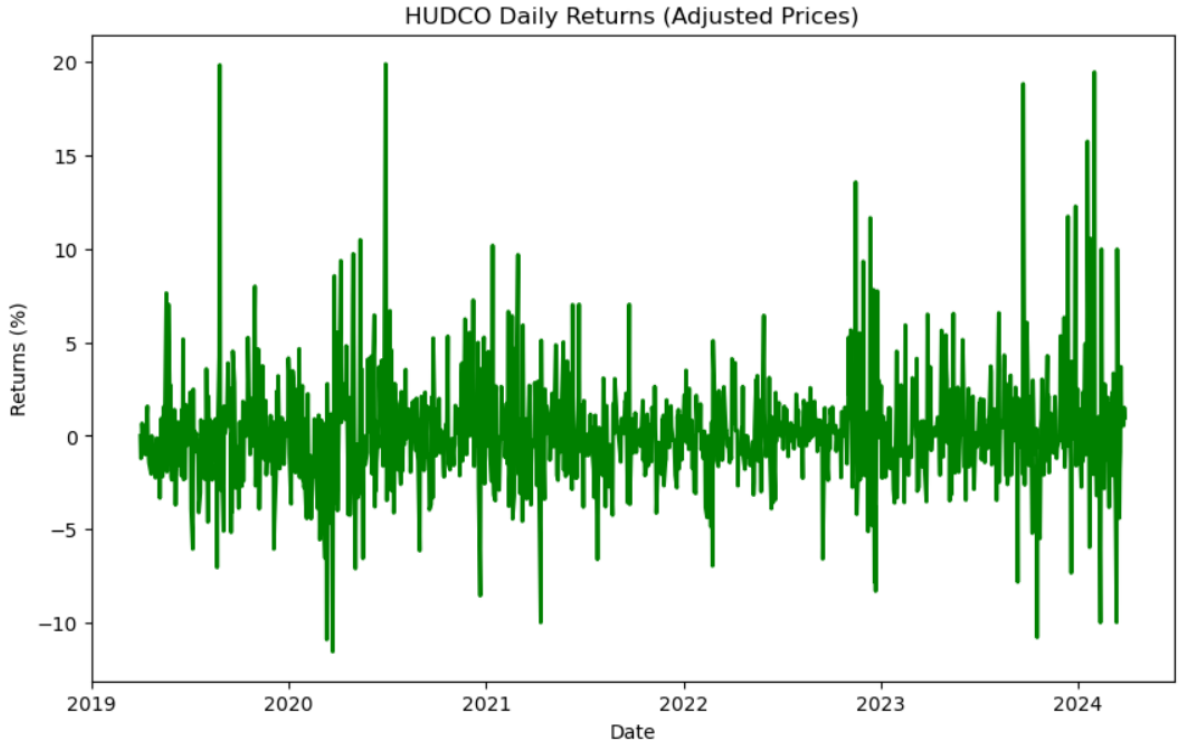
- **Sharp Increase:** Starting in early 2023, there's a sharp increase in prices reaching around 200 by early 2024.
- **Volatility:** The sharp rise towards the end of the period indicates increased volatility and investor interest or market events driving the price up rapidly.

6. Calculate and Plot Returns
Code:

```
# Calculate Returns (Using Adjusted Prices)
data['Returns'] = data['Adj Close'].pct_change().dropna() * 100

# Plot the returns
plt.figure(figsize=(10, 6))
plt.plot(data['Returns'], color='green', lw=2)
plt.title("HUDCO Daily Returns (Adjusted Prices)")
plt.xlabel("Date")
plt.ylabel("Returns (%)")
plt.show()
```

Purpose:
- **Calculating Returns:** Computes the daily log returns of the adjusted closing prices.
- **Plotting Returns:** Visualizes the daily returns.

Output:



Interpretation:
- **Daily Fluctuations:** This plot shows the daily log returns of HUDCO stock scaled by 100 to convert them into percentage returns.
- **Volatility Clusters:** The plot displays periods of high and low volatility. There are noticeable clusters of high volatility towards the end of the period which aligns with the sharp price increase observed in the adjusted closing prices plot.

- **Mean Reversion:** The returns oscillate around a zero mean indicating no clear upward or downward drift in the returns themselves.
- **Spikes:** Significant spikes in returns can be observed especially during the sharp price increase period indicating days of unusually high returns (both positive and negative).

7. Check for ARCH Effects

Code:

```
# Check for ARCH Effects
def arch_lm_test(returns, lags=12):
   return acorr_ljungbox(returns**2, lags=lags, return_df=True)

arch_test = arch_lm_test(data['Returns'].dropna())
print(arch_test)
```

Purpose:

- **ARCH Test:** Performs the ARCH test to check for autoregressive conditional heteroskedasticity (ARCH) effects in the returns.

Output:

```
    lb_stat     lb_pvalue
1    3.187560  7.420068e-02
2    6.494426  3.888243e-02
3    6.853268  7.672464e-02
4    7.535656  1.101473e-01
5   12.752003  2.581698e-02
6   14.326041  2.619875e-02
7   42.610235  3.966409e-07
8   42.968851  8.903736e-07
9   53.989541  1.896490e-08
10  54.269389  4.322650e-08
11  54.443489  9.796174e-08
12  54.509445  2.215726e-07
```

Interpretation:

- The p-value is very small (2.215726e-07) indicating strong evidence against the null hypothesis of no ARCH effects. This suggests the presence of ARCH effects in the returns.

8. Fit ARCH Model and Plot Conditional Volatility

Code:

```
# Fit an ARCH Model and Plot Conditional Volatility
am_arch = arch_model(data['Returns'].dropna(), vol='ARCH', p=1)
res_arch = am_arch.fit(disp='off')
print(res_arch.summary())

# Plot the conditional volatility from the ARCH model
plt.figure(figsize=(10, 6))
plt.plot(res_arch.conditional_volatility, color='purple', lw=2)
plt.title("Conditional Volatility (ARCH Model)")
plt.xlabel("Date")
```

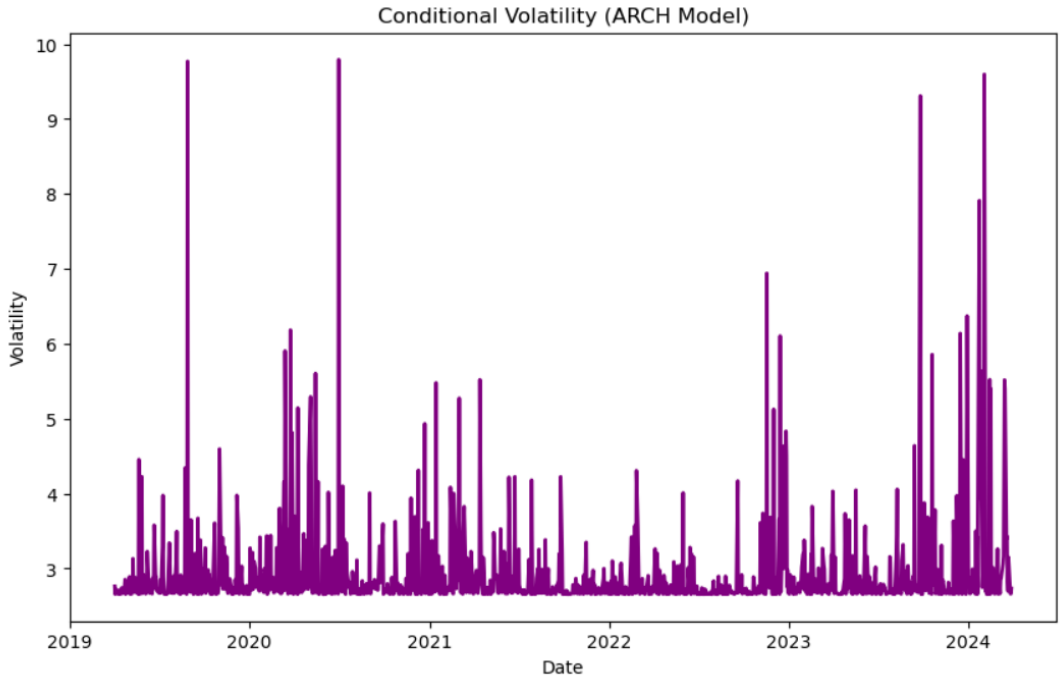
```
plt.ylabel("Volatility")
plt.show()
```

Purpose:

- **Fitting ARCH Model:** Fits an ARCH model to the returns data.
- **Plotting Volatility:** Visualizes the conditional volatility (standard deviation) from the ARCH model.

Output:

```
                Constant Mean - ARCH Model Results
==============================================================================
Dep. Variable:              Returns   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                      ARCH   Log-Likelihood:               -3061.45
Distribution:                 Normal   AIC:                           6128.91
Method:           Maximum Likelihood   BIC:                           6144.26
                                       No. Observations:                 1234
Date:               Thu, Jul 25 2024   Df Residuals:                     1233
Time:                       23:03:22   Df Model:                            1
                            Mean Model
==============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
------------------------------------------------------------------------------
mu             0.1501  8.324e-02      1.804  7.130e-02 [-1.302e-02,  0.313]
                          Volatility Model
==============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
------------------------------------------------------------------------------
omega          7.0545      0.860      8.208  2.257e-16    [  5.370,  8.739]
alpha[1]       0.2280      0.118      1.932  5.334e-02 [-3.285e-03,  0.459]
==============================================================================

Covariance estimator: robust
```



Interpretation:

- **Model Specification:**
  - **Mean Model:** The mean of the return series is estimated at 0.1501 with a t-value of 1.804, indicating it is not significantly different from zero.
  - **Volatility Model:**
    - **omega:** The constant term in the variance equation is estimated at 7.0545, highly significant with a t-value of 8.208.
    - **alpha1:** The ARCH term is estimated at 0.2280, also significant.
- **Log-Likelihood:** -3061.45
- **Information Criteria:**
  - **Akaike (AIC):** 6128.91
  - **Bayes (BIC):** 6144.26
- **Plot:** Conditional Volatility (ARCH Model)
  - **Volatility Dynamics:** This plot shows the conditional standard deviation (volatility) as estimated by the ARCH model.
  - **Periods of High Volatility:** The plot reveals several periods of increased volatility, especially towards the end of the series, aligning with the period of sharp price increase.
  - **Volatility Clustering:** Volatility is not constant over time but clusters with periods of high volatility followed by periods of relative calm. This is typical for financial time series and justifies the use of ARCH models.

9. Forecast Volatility Using ARCH Model

Code:

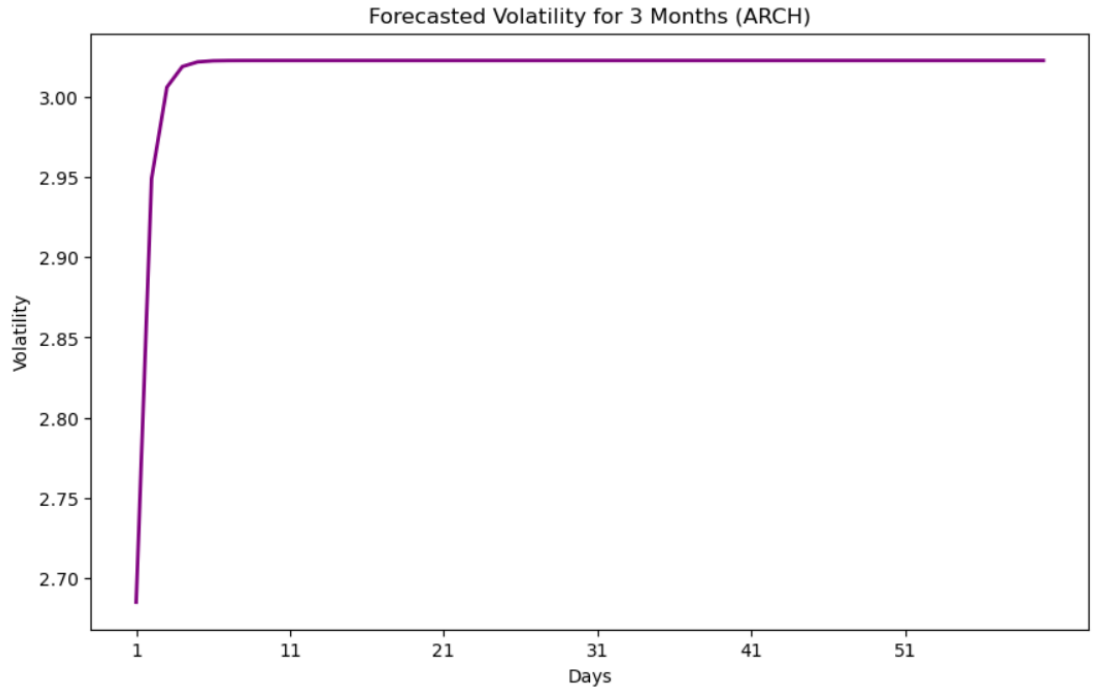
```
# Forecast 3-month (approximately 60 trading days) volatility
forecast_arch = res_arch.forecast(horizon=60)
sigma_forecast_arch = forecast_arch.variance[-1:]**0.5

# Plot the forecasted volatility from the ARCH model
plt.figure(figsize=(10, 6))
```

```
plt.plot(sigma_forecast_arch.T, color='purple', lw=2)
plt.title("Forecasted Volatility for 3 Months (ARCH)")
plt.xlabel("Days")
plt.ylabel("Volatility")

# Adjust x-axis to show fewer labels
plt.xticks(ticks=np.arange(0, 60, step=10), labels=np.arange(1, 61, step=10))
plt.show()
```

Purpose:

- **Forecasting:** Predicts the volatility for the next 3 months (60 trading days) using the fitted ARCH model.
- **Plotting Forecasted Volatility:** Visualizes the forecasted volatility.

Output:



Interpretation:

- **Volatility Decline:** The forecasted volatility shows a declining trend over the 3-month forecast period, suggesting that the model expects the volatility to decrease after the recent high-volatility period.
- **Model Expectation:** This decline implies that the ARCH model anticipates a return to more stable price movements after the recent turbulence.

10. Fit GARCH Model and Plot Conditional Volatility

Code:

```
# Fit a GARCH Model and Plot Conditional Volatility
am_garch = arch_model(data['Returns'].dropna(), vol='GARCH', p=1, q=1)
res_garch = am_garch.fit(disp='off')
print(res_garch.summary())

# Plot the conditional volatility from the GARCH model
plt.figure(figsize=(10, 6))
```

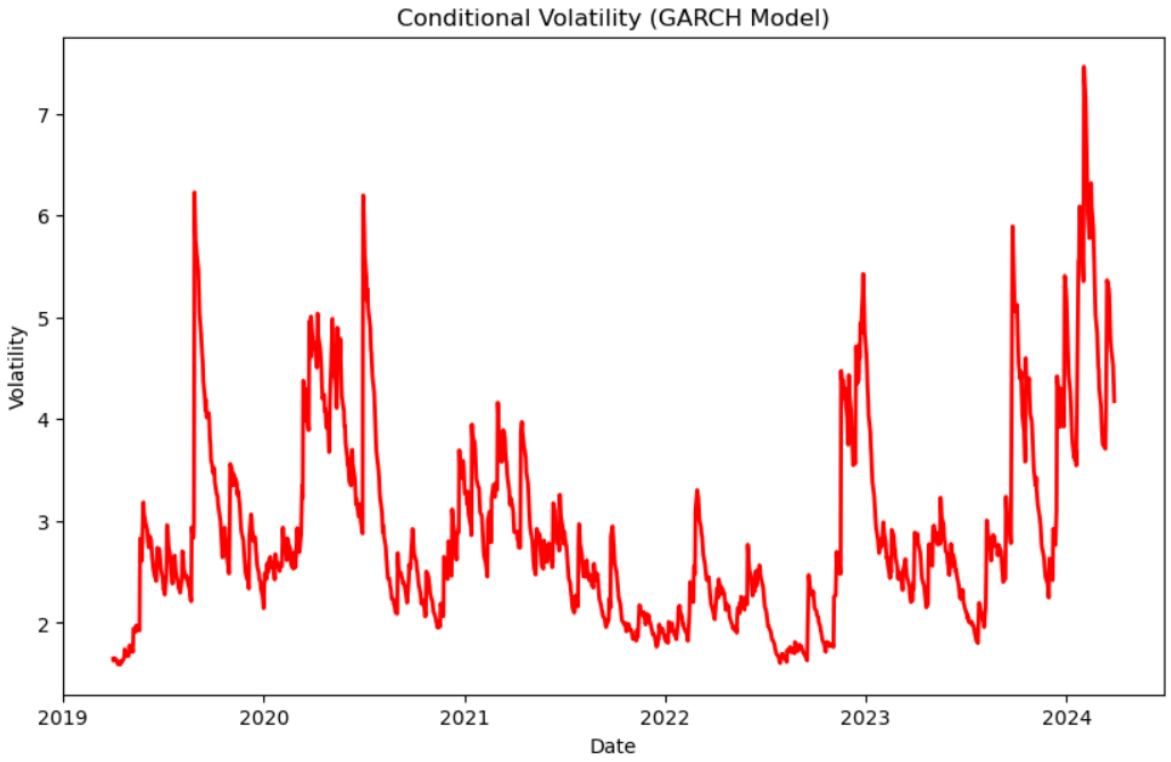
```
plt.plot(res_garch.conditional_volatility, color='red', lw=2)
plt.title("Conditional Volatility (GARCH Model)")
plt.xlabel("Date")
plt.ylabel("Volatility")
plt.show()
```

Purpose:

- **Fitting GARCH Model:** Fits a GARCH model to the returns data.
- **Plotting Volatility:** Visualizes the conditional volatility (standard deviation) from the GARCH model.

Output:

```
                Constant Mean - GARCH Model Results
==============================================================================
Dep. Variable:              Returns   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                      GARCH   Log-Likelihood:               -2999.58
Distribution:                  Normal   AIC:                           6007.15
Method:            Maximum Likelihood   BIC:                           6027.62
                                        No. Observations:                 1234
Date:                Thu, Jul 25 2024   Df Residuals:                     1233
Time:                        23:05:13   Df Model:                            1
                            Mean Model
==============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
------------------------------------------------------------------------------
mu             0.1083  7.127e-02      1.520      0.128 [-3.136e-02,  0.248]
                          Volatility Model
==============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
------------------------------------------------------------------------------
omega          0.1750      0.165      1.061      0.289    [ -0.148,  0.498]
alpha[1]       0.0784  5.224e-02      1.501      0.133 [-2.397e-02,  0.181]
beta[1]        0.9099  5.937e-02     15.327  5.043e-53    [  0.794,  1.026]
==============================================================================

Covariance estimator: robust
```



Interpretation:

- **Model Specification:**
  - **Mean Model:** The mean of the return series is estimated at 0.1083 with a t-value of 1.520, indicating it is not significantly different from zero.
  - **Volatility Model:**
    - **omega:** The constant term in the variance equation is estimated at 0.1750, not significant with a t-value of 1.061.
    - **alpha1:** The ARCH term is estimated at 0.0784, also not significant.
    - **beta1:** The GARCH term is estimated at 0.9099, highly significant with a t-value of 15.327, indicating strong persistence in volatility.
- **Log-Likelihood:** -2999.58
- **Information Criteria:**
  - **Akaike (AIC):** 6007.15
  - **Bayes (BIC):** 6027.62
- **Plot:** Conditional Volatility (GARCH Model)
  - **Volatility Dynamics:** This plot shows the conditional standard deviation (volatility) as estimated by the GARCH model.
  - **Smoothness:** The GARCH model often provides a smoother estimate of volatility compared to the ARCH model as it includes both short-term (ARCH) and long-term (GARCH) components.
  - **Periods of High Volatility:** Periods of high volatility are similar to those identified by the ARCH model, indicating consistency between the two models in identifying volatility clusters.

Part 11: Forecast Volatility Using GARCH Model
Code:

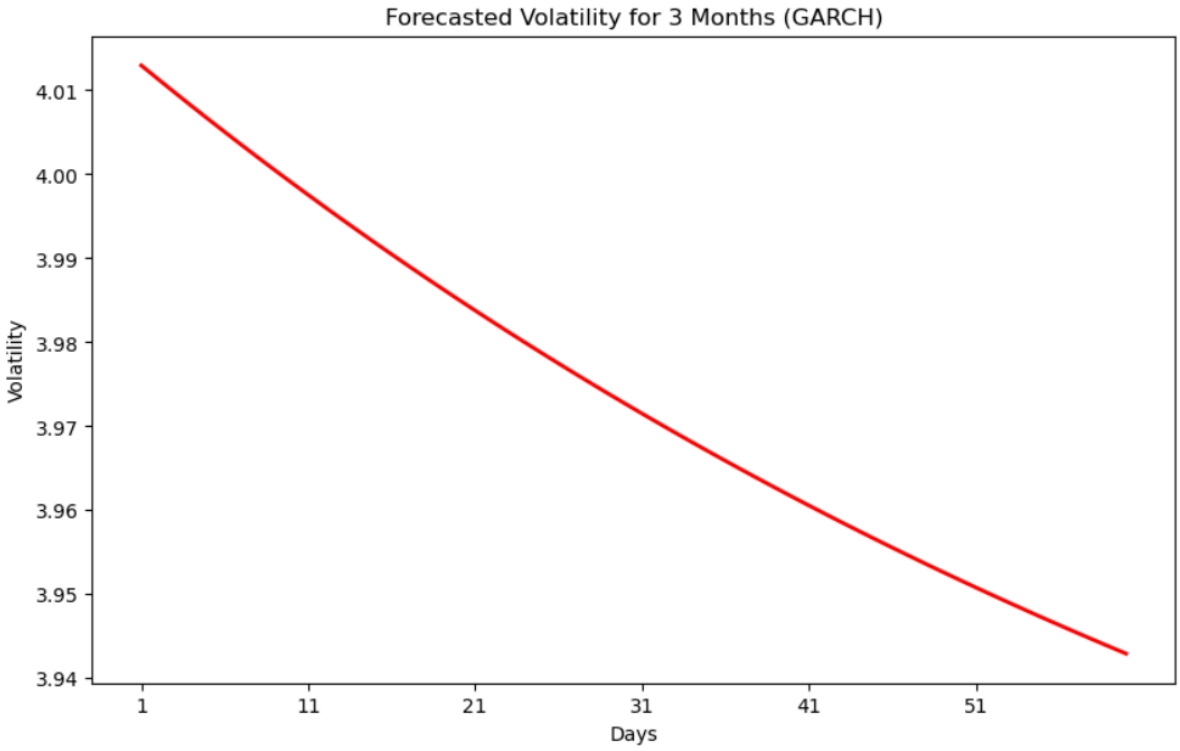
```
# Forecast 3-month (approximately 60 trading days) volatility
forecast_garch = res_garch.forecast(horizon=60)
```

```
sigma_forecast_garch = forecast_garch.variance[-1:]**0.5

# Plot the forecasted volatility from the GARCH model
plt.figure(figsize=(10, 6))
plt.plot(sigma_forecast_garch.T, color='red', lw=2)
plt.title("Forecasted Volatility for 3 Months (GARCH)")
plt.xlabel("Days")
plt.ylabel("Volatility")

# Adjust x-axis to show fewer labels
plt.xticks(ticks=np.arange(0, 60, step=10), labels=np.arange(1, 61, step=10))
plt.show()
```

Purpose:

- **Forecasting:** Predicts the volatility for the next 3 months (60 trading days) using the fitted GARCH model.
- **Plotting Forecasted Volatility:** Visualizes the forecasted volatility.

Output:



Interpretation:

- **Volatility Decline:** Similar to the ARCH model forecast, the GARCH model also predicts a declining trend in volatility over the next 3 months.
- **Model Expectation:** This consistency reinforces the expectation of returning to more stable price movements after the recent period of high volatility.

12. Additional Visualizations: ACF and PACF Plots

Code:

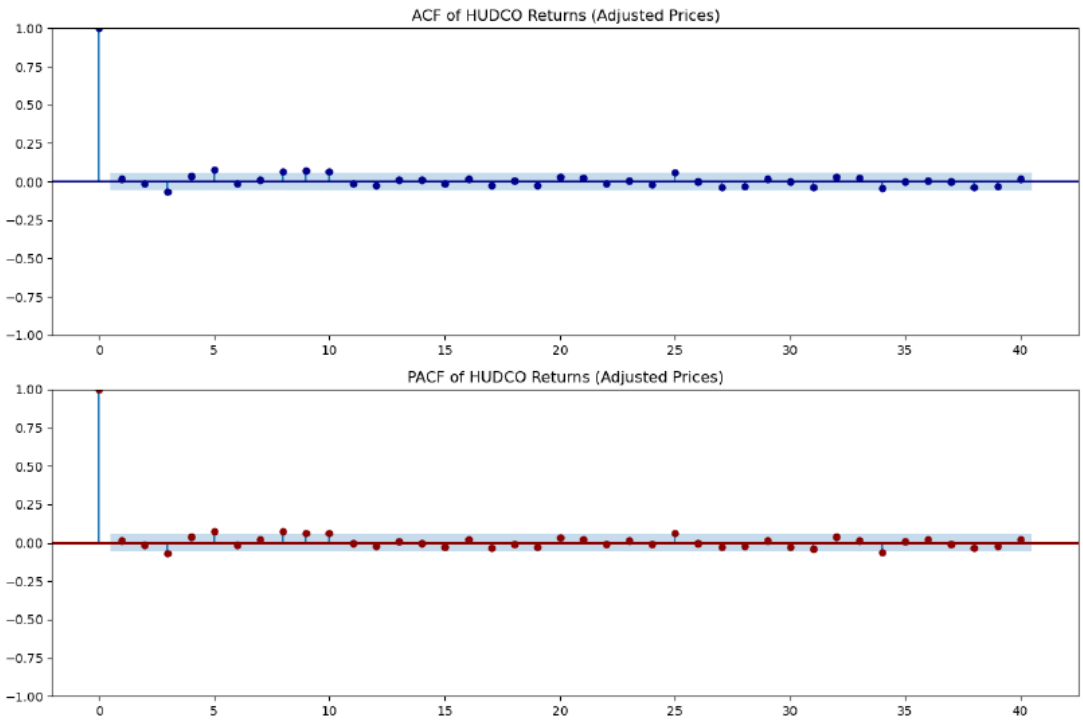
```
# Ensure to import the necessary functions
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

```
# Additional Visualizations: ACF and PACF plots of the returns
plt.figure(figsize=(12, 8))
plt.subplot(211)
plot_acf(data['Returns'].dropna(), lags=40, ax=plt.gca(), color='darkblue')
plt.title("ACF of HUDCO Returns (Adjusted Prices)")
plt.subplot(212)
plot_pacf(data['Returns'].dropna(), lags=40, ax=plt.gca(), color='darkred')
plt.title("PACF of HUDCO Returns (Adjusted Prices)")
plt.tight_layout()
plt.show()
```

Purpose:

- **ACF Plot:** Shows the autocorrelation function of the returns to check for linear dependencies over different lags.
- **PACF Plot:** Shows the partial autocorrelation function to identify the direct effect of a lag.

Output:



Interpretation:

- **ACF Plot:**
  - **Lag 1:** Significant autocorrelation at lag 1 indicating that today's return is somewhat correlated with yesterday's return. This suggests the presence of a potential autoregressive (AR) component in the time series.
  - **Damping Pattern:** Subsequent lags show a rapid decay which is typical for a series with short-term dependencies.
- **PACF Plot:**
  - **Lag 1:** Significant partial autocorrelation at lag 1 supporting the presence of an AR(1) process.

- **Subsequent Lags:** The partial autocorrelations for higher lags are not significant reinforcing the idea that an AR(1) model might be appropriate for the mean equation in the returns.

Overview of ARCH and GARCH

ARCH (Autoregressive Conditional Heteroskedasticity)

Meaning:
ARCH models were introduced by Robert Engle in 1982 to model and forecast time series data with time-varying volatility, commonly observed in financial markets. The ARCH model captures the idea that volatility (or variance) at a given time can be dependent on the past periods' squared observations or past variances.

Advantages:
1. **Captures Volatility Clustering**: ARCH models can effectively capture periods of high and low volatility in financial time series, known as volatility clustering.
2. **Simple and Intuitive**: The basic structure of ARCH models is relatively straightforward, making them easy to understand and implement.
3. **Better Risk Management**: By modeling time-varying volatility, ARCH models provide more accurate risk assessments for financial assets.

Real-Life Examples:
1. **Stock Market Returns**: Modeling the daily returns of a stock index, where the volatility is observed to change over time, with periods of calm followed by periods of turbulence.
2. **Interest Rates**: Analyzing interest rate changes over time, where volatility might increase during periods of economic uncertainty or policy changes.

GARCH (Generalized Autoregressive Conditional Heteroskedasticity)

Meaning:
GARCH models, introduced by Tim Bollerslev in 1986, extend the ARCH model by including lagged terms of both past variances and past squared observations. This allows GARCH models to capture more complex patterns of time-varying volatility and provides a more comprehensive framework for modeling financial time series data.

Advantages:
1. **Captures Long-Term Dependencies**: GARCH models can account for long-term dependencies in volatility, making them more effective for financial time series with persistent volatility patterns.
2. **Flexible and Comprehensive**: The inclusion of lagged variance terms allows GARCH models to adapt to a wider range of data characteristics, providing better fit and more accurate forecasts.
3. **Improved Forecasting**: GARCH models generally provide more accurate forecasts of future volatility, which is crucial for pricing derivatives, risk management, and portfolio optimization.

Real-Life Examples:

1. **Exchange Rates**: Modeling the volatility of currency exchange rates, where volatility may persist over extended periods due to economic and political factors.
2. **Commodity Prices**: Analyzing the price volatility of commodities like oil, gold, or agricultural products, where prices can exhibit periods of high volatility due to supply and demand shocks or geopolitical events.

Comparison of ARCH and GARCH

1. **Structure**:
    - **ARCH**: Models volatility as a function of past squared observations.
    - **GARCH**: Models volatility as a function of past squared observations and past variances.
2. **Flexibility**:
    - **ARCH**: Suitable for capturing short-term volatility patterns.
    - **GARCH**: More flexible, capturing both short-term and long-term volatility patterns.
3. **Complexity**:
    - **ARCH**: Simpler to implement but may require many lagged terms to capture long-term dependencies.
    - **GARCH**: More complex but generally provides a better fit with fewer parameters.