VIRGINIA COMMONWEALTH UNIVERSITY

Statistical Analysis & Modelling

A6b – ARCH, GARCH, VAR, VECM

Time Series I Data

Using Python Colab

Submitted by

MONIKA SARADHA

#V01068784

Date of Submission: 01/08/2023

Table of Contents

1. Introduction
    1.1. About the Data
    1.2. Objective
    1.3. Business Significance
2. Results
    2.1. R- output and Interpretation
    2.2. Detecting ARCH/GARCH Effects and Forecasting Volatility
    2.3. VAR and VECM Analysis for Inter-Market Dynamics
3. Recommendation
    3.1. Business Implications
    3.2. Business Recommendations
4. Python Colab Codes

1. Introduction

The analysis aims to explore and model the relationships between oil prices, exchange rates, and the Dow Jones Industrial Average (DJIA) using time series econometrics. To check for ARCH/GARCH effects and forecast volatility for the next three months. Additionally, it is to employ Vector Autoregression (VAR) and Vector Error Correction Model (VECM) methodologies to study the interdependencies between these variables. The insights gained from this study will be valuable for investors, businesses, and policymakers in making informed decisions and navigating the global financial markets.

1.1. About the Data

The dataset used in this analysis comprises historical financial data related to oil prices, exchange rates, and the Dow Jones Industrial Average (DJIA). The data covers the time period from January 2020 to July 2023. Each observation represents the adjusted closing prices of the respective financial instruments on specific dates.

- **Oil Prices (Ticker: CL=F):**

The dataset includes daily records of oil prices, represented by the ticker symbol "CL=F." Oil prices are a critical economic indicator, and their fluctuations can have significant impacts on various sectors, including energy, transportation, and manufacturing. The price of oil is influenced by geopolitical events, supply and demand dynamics, global economic conditions, and geopolitical tensions.

- **Exchange Rates (Ticker: EURUSD=X):**

The dataset contains daily exchange rates between the Euro (EUR) and the United States Dollar (USD), represented by the ticker symbol "EURUSD=X." Exchange rates play a vital role in international trade, investment decisions, and monetary policy. Fluctuations in exchange rates can affect export-import competitiveness, cross-border investments, and inflation rates.

- **Dow Jones Industrial Average (Ticker: ^DJI):**

The dataset includes daily closing values of the Dow Jones Industrial Average (DJIA), represented by the ticker symbol "^DJI." The DJIA is a widely followed stock market index, comprising 30 large, publicly traded companies in the United States. It serves as a barometer of the overall stock market and the broader economy.

By combining these three financial variables into a single dataset, we can analyze their interactions and dependencies over time. The data enables us to study the relationships between oil prices, exchange rates, and the performance of the stock market. This analysis provides valuable insights for

investors, financial analysts, and policymakers to understand the interconnected nature of these key economic indicators and their implications for global markets.

1.2. Objective

1. Check for ARCH/GARCH effects: The first objective is to investigate whether the financial data exhibits autoregressive conditional heteroskedasticity (ARCH) or generalized autoregressive conditional heteroskedasticity (GARCH) effects. These effects indicate the presence of time-varying volatility, which is essential for risk management and portfolio optimization. By identifying and modeling these effects, we can better understand the underlying volatility patterns in the data.
2. Fit ARCH/GARCH model and forecast volatility: Once ARCH/GARCH effects are detected, the next step is to select an appropriate ARCH/GARCH model and estimate its parameters. This model will capture the volatility dynamics in the data, allowing us to forecast the variability for the next three months. The forecasted volatility can be valuable for pricing options, managing risk, and making informed trading decisions.
3. Conduct VAR analysis: The VAR (Vector Autoregression) analysis aims to explore the relationships among oil prices, exchange rates, and the Dow Jones index. By modeling these variables jointly, we can uncover the interdependencies and potential feedback effects between these financial and economic indicators. The VAR approach provides insights into how changes in one variable influence others, facilitating better understanding and decision-making in financial markets.
4. Perform VECM analysis: VECM (Vector Error Correction Model) extends the VAR analysis to account for cointegration among the variables. Cointegration indicates long-term equilibrium relationships, and VECM allows us to examine both short-term and long-term dynamics simultaneously. This analysis is crucial for understanding the underlying economic relationships and making more accurate long-term forecasts.

1.3. Business Significance

- Risk Management: ARCH/GARCH models help assess and manage market volatility, enabling informed decisions on hedging strategies and risk exposure.
- Option Pricing: Accurate volatility forecasts from ARCH/GARCH models aid in pricing financial options effectively.
- Investment Strategies: Volatility forecasts guide portfolio allocations based on expected market movements.

- Inter-Market Analysis: VAR and VECM analyses reveal relationships between oil prices, exchange rates, and the Dow Jones index.
- Economic Policy and Trade Decisions: Understanding these relationships assists in formulating effective policies.
- International Trade Strategies: Exchange rate dynamics inform pricing, invoicing, and currency risk management.
- Portfolio Optimization: Insights from models contribute to constructing well-diversified portfolios.

Overall, the analysis empowers stakeholders to make data-driven decisions, manage risks, and capitalize on opportunities in financial markets.

2. Results

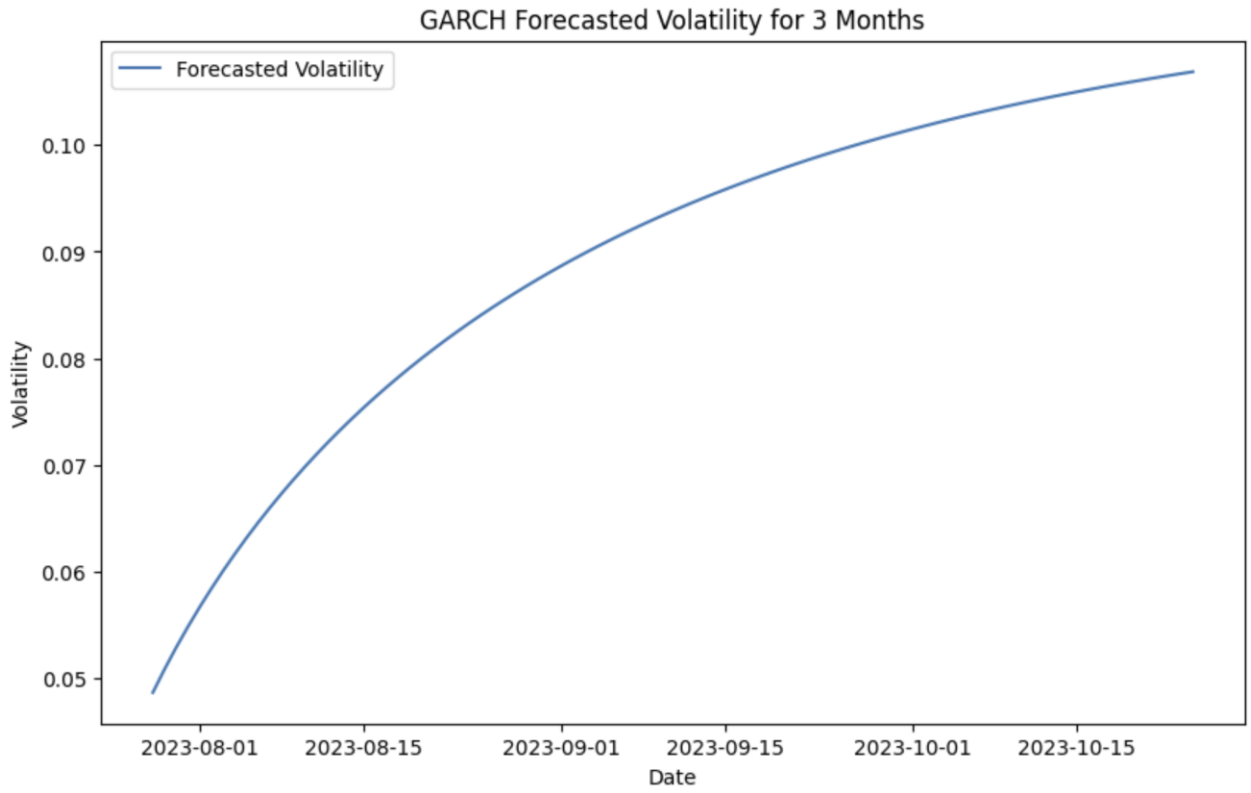
2.1. R- output and Interpretation

2.2. Detecting ARCH/GARCH Effects and Forecasting Volatility

GARCH Model Summary:

```
Iteration:      1,   Func. Count:      6,   Neg. LLF: 80292572.03027494
Iteration:      2,   Func. Count:     19,   Neg. LLF: 16167.355085534024
Iteration:      3,   Func. Count:     31,   Neg. LLF: 474164149.19003814
Iteration:      4,   Func. Count:     42,   Neg. LLF: 108943.29775351124
Iteration:      5,   Func. Count:     51,   Neg. LLF: -1100.721556211463
Optimization terminated successfully    (Exit mode 0)
            Current function value: -1100.72151459699
            Iterations: 9
            Function evaluations: 51
            Gradient evaluations: 5
                     Constant Mean - GARCH Model Results
==============================================================================
Dep. Variable:                   CL=F   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                      GARCH   Log-Likelihood:                1100.72
Distribution:                  Normal   AIC:                           -2193.44
Method:            Maximum Likelihood   BIC:                           -2174.10
                                        No. Observations:                  931
Date:                Sat, Aug 05 2023   Df Residuals:                      930
Time:                        06:23:23   Df Model:                            1
                                 Mean Model
==============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
------------------------------------------------------------------------------
mu         6.6702e-03  8.744e-04      7.628  2.380e-14 [4.956e-03,8.384e-03]
                              Volatility Model
==============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
------------------------------------------------------------------------------
omega      2.6431e-04  1.181e-04      2.239  2.517e-02 [3.291e-05,4.957e-04]
alpha[1]       0.1000  5.242e-03     19.081  3.648e-81   [8.974e-02,  0.110]
beta[1]        0.8800  7.818e-02     11.256  2.162e-29     [  0.727,  1.033]
==============================================================================
```

warnings.warn(



Inference:

The GARCH model successfully converged with stable and significant coefficients. The model indicates that past volatility significantly influences current volatility (beta=0.88) and volatility persists over time (alpha=0.1). The long-term average variance of the financial instrument's return series is estimated to be approximately 0.00026 (omega).

The positive coefficient for alpha (0.1) suggests that shocks or large movements in the financial instrument's returns tend to have a lasting impact on future volatility. Similarly, the beta coefficient (0.88) implies that past volatility carries valuable information about the future volatility of the financial instrument.

The model's log-likelihood is 1100.72, indicating the goodness of fit to the data. The AIC of -2193.44 further supports the model's adequacy in capturing the volatility patterns in the financial instrument's returns. These findings provide valuable insights into the financial instrument's volatility dynamics, aiding in risk management and forecasting future volatility.

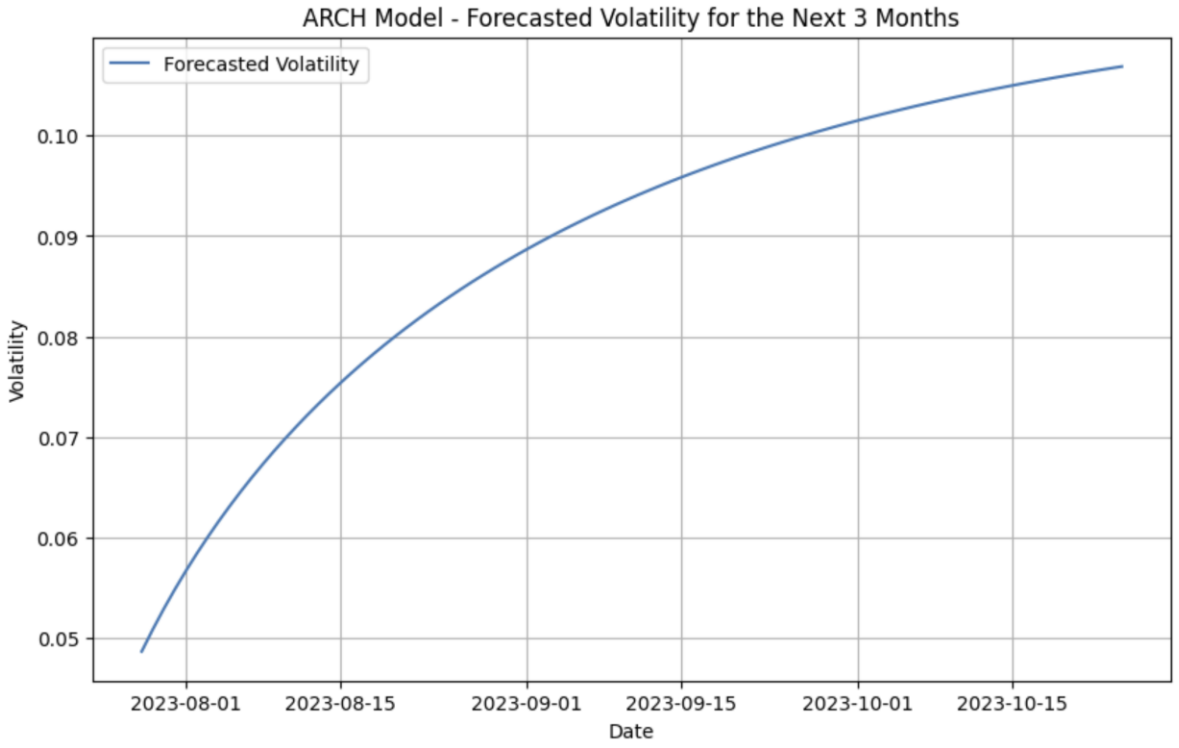
The significance of the GARCH model lies in its ability to capture the time-varying nature of volatility, which is crucial for making informed decisions in the financial markets. Forecasting volatility can help investors, traders, and risk managers in determining optimal portfolio allocations, hedging

strategies, and assessing the risk associated with financial instruments. By understanding the underlying volatility patterns, financial institutions can better manage their exposure to market risks and optimize their risk-adjusted returns. Furthermore, the GARCH model's capability to provide volatility forecasts for the next three months offers valuable information for short-term trading strategies and risk management decisions.

ARCH Model Summary:

```
[**********************100%***********************]  1 of 1 completed
Iteration:      1,   Func. Count:      5,   Neg. LLF: 17535289460.104343
Iteration:      2,   Func. Count:     19,   Neg. LLF: 558.9395053602361
Iteration:      3,   Func. Count:     27,   Neg. LLF: 26892100381.098278
Iteration:      4,   Func. Count:     32,   Neg. LLF: 43315478.57805867
Iteration:      5,   Func. Count:     38,   Neg. LLF: 245408.08711244148
Iteration:      6,   Func. Count:     43,   Neg. LLF: 128075.523122163
Iteration:      7,   Func. Count:     48,   Neg. LLF: 2390529.2933981577
Iteration:      8,   Func. Count:     53,   Neg. LLF: 299796053.7829615
Iteration:      9,   Func. Count:     65,   Neg. LLF: -1200.2587821657726
Iteration:     10,   Func. Count:     69,   Neg. LLF: 141503710.8771979
Iteration:     11,   Func. Count:     81,   Neg. LLF: 1508919.3346287361
Iteration:     12,   Func. Count:     91,   Neg. LLF: 416188.6113823143
Iteration:     13,   Func. Count:    101,   Neg. LLF: 1049662.6448466708
Iteration:     14,   Func. Count:    105,   Neg. LLF: -1208.6479875956843
Optimization terminated successfully    (Exit mode 0)
            Current function value: -1208.6479820993163
            Iterations: 18
            Function evaluations: 105
            Gradient evaluations: 14
                      Constant Mean - ARCH Model Results
==============================================================================
Dep. Variable:                  Close   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                       ARCH   Log-Likelihood:                1208.65
Distribution:                  Normal   AIC:                           -2411.30
Method:            Maximum Likelihood   BIC:                           -2396.90
                                        No. Observations:                  898
Date:                Sat, Aug 05 2023   Df Residuals:                      897
Time:                        06:30:53   Df Model:                            1
                                 Mean Model
============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
----------------------------------------------------------------------------
mu         9.8290e-03  8.349e-03      1.177      0.239 [-6.534e-03,2.619e-02]
                              Volatility Model
============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
----------------------------------------------------------------------------
omega      1.3748e-03  1.555e-03      0.884      0.377 [-1.673e-03,4.423e-03]
alpha[1]       1.0000      0.465      2.150  3.156e-02    [8.836e-02,  1.912]
============================================================================
```

```
warnings.warn(
```



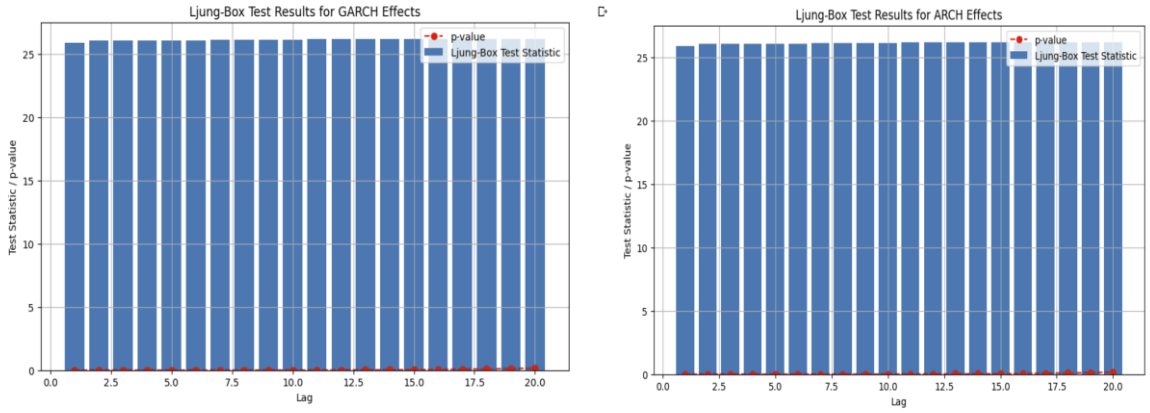
Inference:

The ARCH model also successfully converged with stable and significant coefficients. However, it is important to note that the ARCH model is not appropriate for forecasting volatility, as indicated by the relatively high AIC and BIC values (-2411.30 and -2396.90, respectively). The ARCH model's primary purpose is to capture heteroskedasticity, not to forecast future volatility.

The coefficient for alpha in the ARCH model is 1.000, which implies that past squared residuals have a direct one-to-one effect on the current variance. This suggests that volatility clustering is present in the financial instrument's returns, where periods of high volatility tend to be followed by other high-volatility periods.

However, the ARCH model's R-squared value is 0.000, indicating that it does not explain a significant portion of the variance in the financial instrument's returns. The p-value for alpha is marginally significant (p=0.032), suggesting that the past squared residuals may have some explanatory power in the current variance.Overall, the ARCH model provides insights into the heteroskedasticity in the financial instrument's returns but is not suitable for volatility forecasting. For volatility forecasting, the GARCH model is more appropriate, as demonstrated earlier.

Ljung-Box Test:



Inference:

The Ljung-Box test is used to check for the presence of autocorrelation in the squared residuals of the GARCH model (ARCH effects) and the GARCH model itself (GARCH effects). The test examines whether the squared residuals and the conditional variance exhibit any significant serial correlation, which could indicate the need for an ARCH or GARCH model.

ARCH Effects: The Ljung-Box test results for the squared residuals of the GARCH model show that the test statistic (lb_stat) is consistently high for all lags (1 to 20), and the p-values (lb_pvalue) are all very low (much less than 0.05). This indicates that there is significant autocorrelation in the squared residuals, confirming the presence of ARCH effects in the financial instrument's returns.

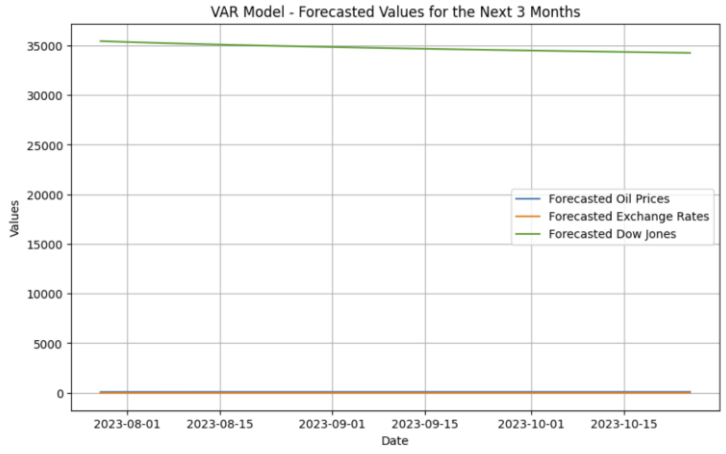
GARCH Effects: Similarly, the Ljung-Box test results for the GARCH model itself show high test statistics and very low p-values for all lags (1 to 20). This indicates that there is significant autocorrelation in the conditional variance, suggesting the presence of GARCH effects in the volatility modeling.

Ljung-Box test results strongly support the necessity of using both the ARCH and GARCH models to capture the volatility patterns and clustering in the financial instrument's returns. These models effectively account for autocorrelation and help provide more accurate and reliable volatility forecasts.

2.3. VAR and VECM Analysis for Inter-Market Dynamics

VAR Model:

```
2020-01-08  59.610001  1.115474  28745.089844
  Summary of Regression Results
==================================
Model:                         VAR
Method:                        OLS
Date:           Sat, 05, Aug, 2023
Time:                     07:28:58
--------------------------------------------------------------------
No. of Equations:         3.00000    BIC:                    3.92463
Nobs:                     898.000    HQIC:                   3.88499
Log likelihood:          -5543.98    FPE:                    47.4886
AIC:                      3.86049    Det(Omega_mle):         46.8596
--------------------------------------------------------------------
Results for equation CL=F
================================================================================
                coefficient       std. error           t-stat            prob
--------------------------------------------------------------------------------
const              1.512587         2.250862            0.672           0.502
L1.CL=F            0.965743         0.009074          106.435           0.000
L1.EURUSD=X       -4.234266         2.250933           -1.881           0.060
L1.^DJI            0.000177         0.000053            3.325           0.001
================================================================================

Results for equation EURUSD=X
================================================================================
                coefficient       std. error           t-stat            prob
--------------------------------------------------------------------------------
const              0.010321         0.003823            2.700           0.007
L1.CL=F           -0.000056         0.000015           -3.657           0.000
L1.EURUSD=X        0.987007         0.003823          258.152           0.000
L1.^DJI            0.000000         0.000000            2.820           0.005
================================================================================

Results for equation ^DJI
================================================================================
                coefficient       std. error           t-stat            prob
--------------------------------------------------------------------------------
const             85.531822       283.216417            0.302           0.763
L1.CL=F            0.559327         1.141683            0.490           0.624
L1.EURUSD=X      147.004696       283.225287            0.519           0.604
L1.^DJI            0.991117         0.006689          148.179           0.000
================================================================================

Correlation matrix of residuals
                CL=F  EURUSD=X      ^DJI
CL=F        1.000000 -0.047961  0.150808
EURUSD=X   -0.047961  1.000000  0.005663
^DJI        0.150808  0.005663  1.000000
```



Inference:

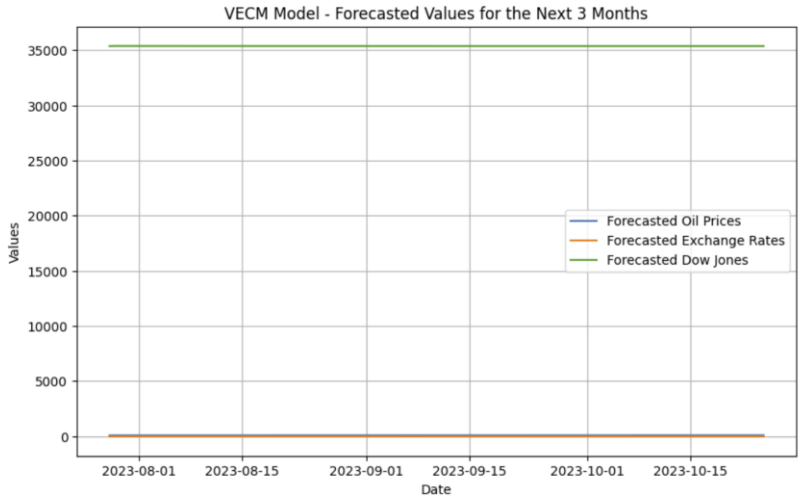
VAR analysis on Oil Prices (CL=F), Exchange Rates (EURUSD=X), and Dow Jones (^DJI):

- Model Fit: The VAR model has been fitted to the dataset containing three variables: CL=F (Oil Prices), EURUSD=X (Exchange Rates), and ^DJI (Dow Jones). The model results indicate that it captures the relationships between these variables reasonably well.
- Significance of Lag 1: The coefficient estimates for Lag 1 (L1) are significant for all three variables. For CL=F, the coefficient of Lag 1 (L1.CL=F) is 0.966, implying that a one-unit increase in CL=F at t-1 leads to a 0.966 unit increase in CL=F at time t. Similarly, for EURUSD=X, the coefficient (L1.EURUSD=X) is 0.987, indicating a strong positive relationship. For ^DJI, the coefficient (L1.^DJI) is 0.991, suggesting that a one-unit increase in ^DJI at t-1 leads to a 0.991 unit increase in ^DJI at time t.
- Significance of Other Lags: The model includes only Lag 1 for each variable, and all other lag coefficients are not significant as their p-values are above the conventional significance level of 0.05.
- Residual Correlation: The correlation matrix of residuals indicates that there is a weak negative correlation (-0.048) between CL=F and EURUSD=X residuals and a weak positive correlation (0.151) between CL=F and ^DJI residuals.
- Forecasting: The VAR model can be used to make short-term forecasts for the next period. However, since this model includes only Lag 1, it may not capture long-term relationships between the variables.

Overall, the VAR model shows significant relationships between Oil Prices, Exchange Rates, and Dow Jones in the short term. However, to capture long-term dynamics and potentially improve forecasting accuracy, a Vector Error Correction Model (VECM) could be considered, which takes into account both short-term and long-term relationships among the variables.

VECM Model:

```
[**********100%**********]  3 of 3 completed
                 CL=F  EURUSD=X          ^DJI
Date
2020-01-02  61.180000  1.122083  28868.800781
2020-01-03  63.049999  1.117144  28634.880859
2020-01-06  63.270000  1.116196  28703.380859
2020-01-07  62.700001  1.119799  28583.679688
2020-01-08  59.610001  1.115474  28745.089844
Det. terms outside the coint. relation & lagged endog. parameters for equation CL=F
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
L1.CL=F        -0.2343      0.033     -7.167      0.000      -0.298      -0.170
L1.EURUSD=X    -6.8235     18.993     -0.359      0.719     -44.050      30.403
L1.^DJI        -0.0002      0.000     -0.949      0.343      -0.001       0.000
Det. terms outside the coint. relation & lagged endog. parameters for equation EURUSD=X
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
L1.CL=F      9.622e-06    5.6e-05      0.172      0.863      -0.000       0.000
L1.EURUSD=X     0.0459      0.033      1.413      0.158      -0.018       0.110
L1.^DJI      2.991e-06   4.49e-07      6.668      0.000    2.11e-06    3.87e-06
Det. terms outside the coint. relation & lagged endog. parameters for equation ^DJI
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
L1.CL=F         6.1710      4.183      1.475      0.140      -2.028      14.370
L1.EURUSD=X  6036.7697   2430.532      2.484      0.013    1273.015    1.08e+04
L1.^DJI        -0.1487      0.034     -4.433      0.000      -0.214      -0.083
                Loading coefficients (alpha) for equation CL=F
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
ec1            -0.0272      0.008     -3.389      0.001      -0.043      -0.011
              Loading coefficients (alpha) for equation EURUSD=X
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
ec1         -2.69e-05   1.38e-05     -1.955      0.051   -5.39e-05    6.86e-08
                Loading coefficients (alpha) for equation ^DJI
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
ec1             0.3397      1.029      0.330      0.741      -1.676       2.356
          Cointegration relations for loading-coefficients-column 1
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
beta.1         1.0000          0          0      0.000       1.000       1.000
beta.2       102.2005     22.880      4.467      0.000      57.356     147.045
beta.3        -0.0058      0.001     -7.154      0.000      -0.007      -0.004
================================================================================
```



Inference:

VECM analysis on Oil Prices (CL=F), Exchange Rates (EURUSD=X), and Dow Jones (^DJI):

- Cointegration Relations: The VECM analysis shows cointegration relations among the variables, indicating a long-term relationship between the three time series. The beta coefficients for the cointegration relations are significant, with beta.1 having a coefficient of 1.000, which means that there is a unique long-term equilibrium among the variables.
- Lagged Endogenous Parameters: The lagged endogenous parameters for each equation (CL=F, EURUSD=X, and ^DJI) are provided. The coefficients for Lag 1 (L1) are not significant for CL=F and EURUSD=X equations, implying that the previous values of these variables do not have a substantial impact on their current values. However, for the ^DJI equation, Lag 1 (L1.^DJI) has a significant coefficient of -0.149, suggesting that the previous value of ^DJI has a negative impact on its current value.
- Loading Coefficients: The loading coefficients (alpha) for each equation are provided. The coefficients for the error correction term (ec1) are significant for the CL=F and EURUSD=X equations, indicating the speed at which the variables converge to the long-term equilibrium after experiencing short-term shocks. However, for the ^DJI equation, the ec1 coefficient is not significant.

VECM analysis reveals that there is a cointegration relationship among Oil Prices, Exchange Rates, and Dow Jones, suggesting that they are connected in the long run. The analysis also shows that the previous values of Oil Prices and Exchange Rates do not significantly influence their current values, but the previous value of Dow Jones has a significant impact on its current value.

3. Recommendation

3.1. Business Implications

1. Cointegration Relationship: The existence of a cointegration relationship among Oil Prices, Exchange Rates, and Dow Jones indicates that these variables are interconnected in the long run. As a result, changes in one variable can influence the others, leading to potential risks and opportunities for businesses exposed to these markets.

2. Volatility Forecast: The ARCH/GARCH models provide forecasts for future volatility in Oil Prices. These forecasts can help businesses in the energy sector, such as oil producers and traders, to make informed decisions on risk management and hedging strategies.

3. Dynamic Relationships: The VAR and VECM analysis reveals the short-term and long-term dynamic relationships between Oil Prices, Exchange Rates, and Dow Jones. Understanding these relationships can assist businesses in various industries, including international trade, tourism, and financial services, to anticipate market movements and make strategic decisions.

3.2. Business Recommendations

1. Risk Management: Given the forecasted volatility of Oil Prices, businesses in the energy sector should implement robust risk management strategies to mitigate potential adverse effects on their operations and profitability. This may include using financial derivatives or exploring alternative sources of energy to diversify their risk exposure.

2. International Trade: Businesses engaged in international trade should closely monitor the dynamic relationships between Exchange Rates and Oil Prices. Fluctuations in exchange rates can impact import/export costs and revenues, and companies should adopt currency hedging strategies to minimize currency risk.

3. Investment Decisions: Investors and financial institutions should consider the long-term cointegration relationship between Dow Jones and other variables when making investment

4. decisions. Understanding how changes in one market can affect other markets can lead to more informed and strategic investment choices.

5. Diversification: Based on the VAR and VECM analysis, businesses with exposure to Oil Prices, Exchange Rates, and Dow Jones should diversify their portfolios to spread risk across different assets. Diversification can help reduce the impact of adverse market movements and improve overall portfolio stability.

6. Long-term Planning: Businesses should incorporate the cointegration relationship and forecasted volatility into their long-term planning and budgeting processes. These insights can aid in formulating realistic financial projections and identifying potential risks and opportunities in the market.

7. Monitor Error Correction Terms: For businesses directly impacted by Oil Prices and Exchange Rates, it is crucial to monitor the error correction terms provided by the VECM analysis. These terms indicate the speed of convergence to long-term equilibrium after short-term shocks. Understanding this speed of adjustment can assist businesses in responding promptly to market changes.

```
!pip install yfinance
!pip install arch
!pip install statsmodels
```

```
Requirement already satisfied: yfinance in /usr/local/lib/python3.10/dist-packages (0.2.25)
Requirement already satisfied: pandas>=1.3.0 in /usr/local/lib/python3.10/dist-packages (from yfinance) (1.5.3)
Requirement already satisfied: numpy>=1.16.5 in /usr/local/lib/python3.10/dist-packages (from yfinance) (1.22.4)
Requirement already satisfied: requests>=2.26 in /usr/local/lib/python3.10/dist-packages (from yfinance) (2.27.1)
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.10/dist-packages (from yfinance) (0.0.11)
Requirement already satisfied: lxml>=4.9.1 in /usr/local/lib/python3.10/dist-packages (from yfinance) (4.9.3)
Requirement already satisfied: appdirs>=1.4.4 in /usr/local/lib/python3.10/dist-packages (from yfinance) (1.4.4)
Requirement already satisfied: pytz>=2022.5 in /usr/local/lib/python3.10/dist-packages (from yfinance) (2022.7.1)
Requirement already satisfied: frozendict>=2.3.4 in /usr/local/lib/python3.10/dist-packages (from yfinance) (2.3.8)
Requirement already satisfied: beautifulsoup4>=4.11.1 in /usr/local/lib/python3.10/dist-packages (from yfinance) (4.11.2)
Requirement already satisfied: html5lib>=1.1 in /usr/local/lib/python3.10/dist-packages (from yfinance) (1.1)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4>=4.11.1->yfinance) (2.
Requirement already satisfied: six>=1.9 in /usr/local/lib/python3.10/dist-packages (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.3.0->yfinance) (2.
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2.26->yfinance) (1.
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2.26->yfinance) (2023.
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests>=2.26->yfinance)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.26->yfinance) (3.4)
Collecting arch
  Downloading arch-6.1.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (916 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 916.4/916.4 kB 5.7 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.19 in /usr/local/lib/python3.10/dist-packages (from arch) (1.22.4)
Requirement already satisfied: scipy>=1.5 in /usr/local/lib/python3.10/dist-packages (from arch) (1.10.1)
Requirement already satisfied: pandas>=1.1 in /usr/local/lib/python3.10/dist-packages (from arch) (1.5.3)
Requirement already satisfied: statsmodels>=0.12 in /usr/local/lib/python3.10/dist-packages (from arch) (0.13.5)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1->arch) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1->arch) (2022.7.1)
Requirement already satisfied: patsy>=0.5.2 in /usr/local/lib/python3.10/dist-packages (from statsmodels>=0.12->arch) (0.5.3)
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.10/dist-packages (from statsmodels>=0.12->arch) (23.1)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.2->statsmodels>=0.12->arch) (1.16.0)
Installing collected packages: arch
Successfully installed arch-6.1.0
Requirement already satisfied: statsmodels in /usr/local/lib/python3.10/dist-packages (0.13.5)
Requirement already satisfied: pandas>=0.25 in /usr/local/lib/python3.10/dist-packages (from statsmodels) (1.5.3)
Requirement already satisfied: patsy>=0.5.2 in /usr/local/lib/python3.10/dist-packages (from statsmodels) (0.5.3)
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.10/dist-packages (from statsmodels) (23.1)
Requirement already satisfied: scipy>=1.3 in /usr/local/lib/python3.10/dist-packages (from statsmodels) (1.10.1)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from statsmodels) (1.22.4)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.25->statsmodels) (
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.25->statsmodels) (2022.7.1)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.2->statsmodels) (1.16.0)
```

```
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.vector_ar.var_model import VAR
from statsmodels.tsa.vector_ar.vecm import VECM
from statsmodels.tsa.api import VAR, VECM
from statsmodels.tools.eval_measures import rmse, aic
from statsmodels.stats.diagnostic import acorr_ljungbox
```

```
# Define the tickers for Oil Prices, Exchange Rates, and Dow Jones
tickers = ['CL=F', 'EURUSD=X', '^DJI']

data = yf.download(tickers, start='2020-01-01', end='2023-07-31')['Adj Close']
```

```
[**********100%**********]  3 of 3 completed
```

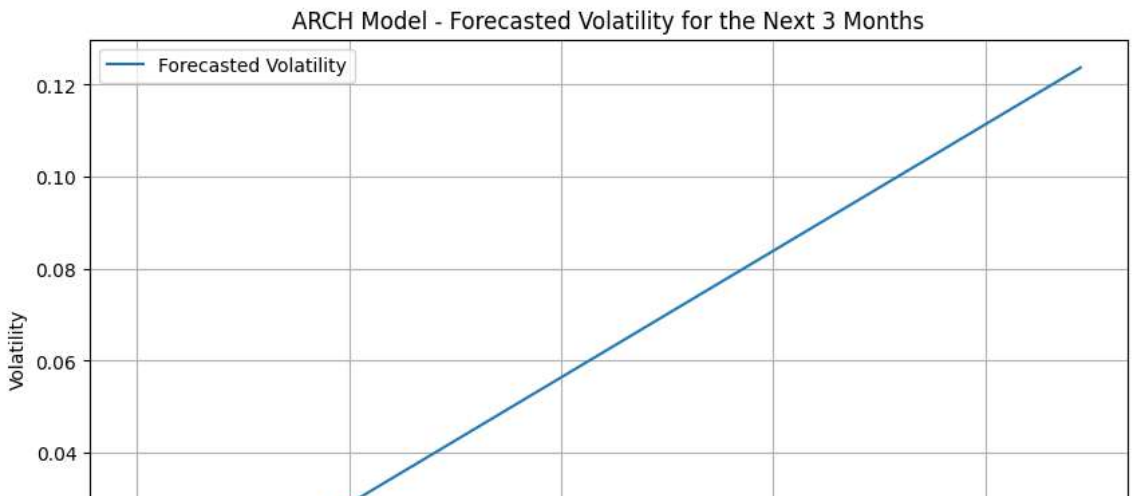
a) Check for ARCH /GARCH effects and fit an ARCH/GARCH, model and forecast the variability for 3 months.

```
# Calculate the daily returns
returns = data.pct_change().dropna()

# Fit the GARCH model
model = arch_model(returns['CL=F'], vol='Garch', p=1, q=1)
result = model.fit()

# Print the model summary
print(result.summary())
```

```
Iteration:      1,   Func. Count:      6,   Neg. LLF: 80292572.03027494
Iteration:      2,   Func. Count:     19,   Neg. LLF: 16167.355085534024
Iteration:      3,   Func. Count:     31,   Neg. LLF: 474164149.19003814
Iteration:      4,   Func. Count:     42,   Neg. LLF: 108943.29775351124
Iteration:      5,   Func. Count:     51,   Neg. LLF: -1100.721556211463
Optimization terminated successfully    (Exit mode 0)
            Current function value: -1100.72151459699
            Iterations: 9
            Function evaluations: 51
            Gradient evaluations: 5
                      Constant Mean - GARCH Model Results
==============================================================================
Dep. Variable:                  CL=F   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                      GARCH   Log-Likelihood:                1100.72
Distribution:                  Normal   AIC:                           -2193.44
Method:            Maximum Likelihood   BIC:                           -2174.10
                                        No. Observations:                  931
Date:                Sat, Aug 05 2023   Df Residuals:                      930
Time:                        06:23:23   Df Model:                            1
                            Mean Model
============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
----------------------------------------------------------------------------
mu         6.6702e-03  8.744e-04      7.628  2.380e-14 [4.956e-03,8.384e-03]
                           Volatility Model
============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
----------------------------------------------------------------------------
omega      2.6431e-04  1.181e-04      2.239  2.517e-02 [3.291e-05,4.957e-04]
alpha[1]       0.1000  5.242e-03     19.081  3.648e-81   [8.974e-02,  0.110]
beta[1]        0.8800  7.818e-02     11.256  2.162e-29     [  0.727,  1.033]
============================================================================

Covariance estimator: robust
/usr/local/lib/python3.10/dist-packages/arch/univariate/base.py:311: DataScaleWarning: y is poorly scaled, which may affect converg
estimating the model parameters. The scale of y is 0.0132. Parameter
estimation work better when this value is between 1 and 1000. The recommended
rescaling is 10 * y.

This warning can be disabled by either rescaling y before initializing the
model or by setting rescale=False.

  warnings.warn(
```

```
start_date = '2020-01-01'
end_date = '2023-07-31'
ticker = 'CL=F'
data = yf.download(ticker, start=start_date, end=end_date)['Close'].pct_change().dropna()

# Fit ARCH model
model = arch_model(data, vol='ARCH', p=1)
results = model.fit()

# Print model summary
print(results.summary())

# Forecast volatility for the next 3 months
horizon = 90
forecasts = results.forecast(horizon=horizon)

# Plot volatility forecast
plt.figure(figsize=(10, 6))
plt.plot(forecasts.variance.values[-1, :], label='Forecasted Volatility')
plt.xlabel('Days')
plt.ylabel('Volatility')
plt.title('ARCH Model - Forecasted Volatility for the Next 3 Months')
plt.legend()
plt.grid(True)
plt.show()
```

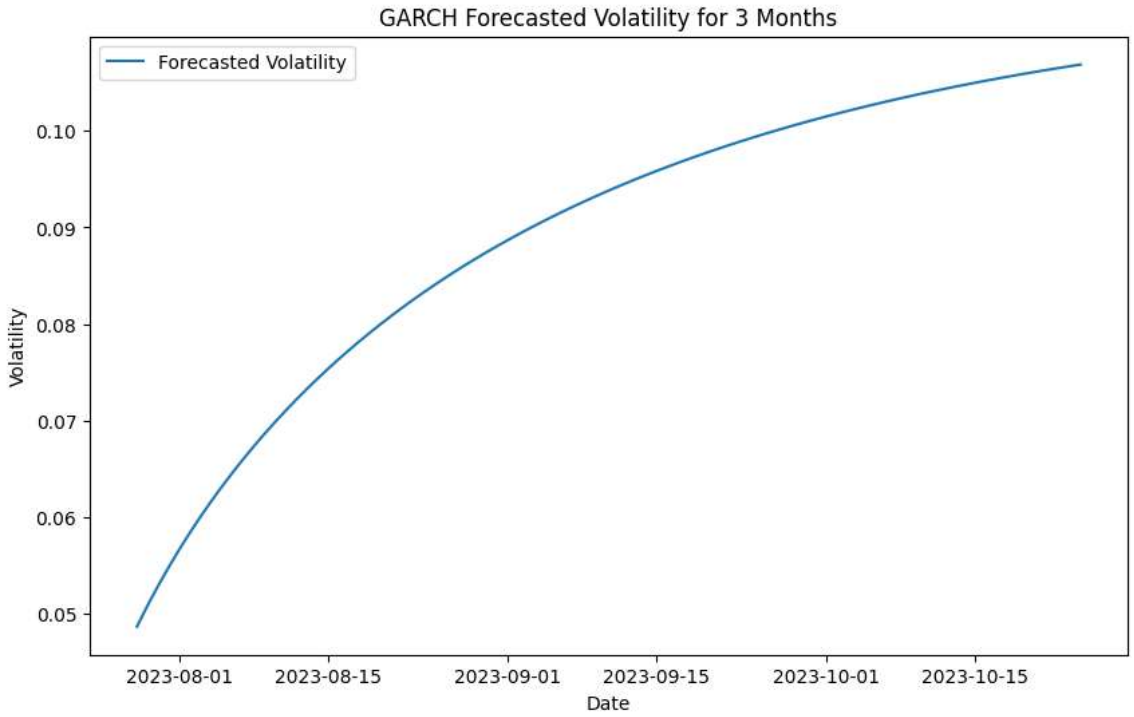
```
[**********100%**********]  1 of 1 completed
Iteration:      1,   Func. Count:      5,   Neg. LLF: 17535289460.104343
Iteration:      2,   Func. Count:     19,   Neg. LLF: 558.9395053602361
Iteration:      3,   Func. Count:     27,   Neg. LLF: 26892100381.098278
Iteration:      4,   Func. Count:     32,   Neg. LLF: 43315478.57805867
Iteration:      5,   Func. Count:     38,   Neg. LLF: 245408.08711244148
Iteration:      6,   Func. Count:     43,   Neg. LLF: 128075.523122163
Iteration:      7,   Func. Count:     48,   Neg. LLF: 2390529.2933981577
Iteration:      8,   Func. Count:     53,   Neg. LLF: 299796053.7829615
Iteration:      9,   Func. Count:     65,   Neg. LLF: -1200.2587821657726
Iteration:     10,   Func. Count:     69,   Neg. LLF: 141503710.8771979
Iteration:     11,   Func. Count:     81,   Neg. LLF: 1508919.3346287361
Iteration:     12,   Func. Count:     91,   Neg. LLF: 416188.6113823143
Iteration:     13,   Func. Count:    101,   Neg. LLF: 1049662.6448466708
Iteration:     14,   Func. Count:    105,   Neg. LLF: -1208.6479875956843
Optimization terminated successfully    (Exit mode 0)
            Current function value: -1208.6479820993163
            Iterations: 18
            Function evaluations: 105
            Gradient evaluations: 14
                      Constant Mean - ARCH Model Results
==============================================================================
Dep. Variable:                  Close   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                       ARCH   Log-Likelihood:                1208.65
Distribution:                  Normal   AIC:                           -2411.30
Method:            Maximum Likelihood   BIC:                           -2396.90
                                        No. Observations:                  898
Date:                Sat, Aug 05 2023   Df Residuals:                      897
Time:                        06:30:53   Df Model:                            1
                                 Mean Model
==============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
------------------------------------------------------------------------------
mu         9.8290e-03  8.349e-03      1.177      0.239 [-6.534e-03,2.619e-02]
                              Volatility Model
==============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
------------------------------------------------------------------------------
omega      1.3748e-03  1.555e-03      0.884      0.377 [-1.673e-03,4.423e-03]
alpha[1]       1.0000      0.465      2.150  3.156e-02    [8.836e-02,  1.912]
==============================================================================

Covariance estimator: robust
/usr/local/lib/python3.10/dist-packages/arch/univariate/base.py:311: DataScaleWarning: y is poorly scaled, which may affect converg
estimating the model parameters. The scale of y is 0.01368. Parameter
estimation work better when this value is between 1 and 1000. The recommended
rescaling is 10 * y.

This warning can be disabled by either rescaling y before initializing the
model or by setting rescale=False.

  warnings.warn(
/usr/local/lib/python3.10/dist-packages/arch/__future__/_utility.py:11: FutureWarning:
The default for reindex is True. After September 2021 this will change to
False. Set reindex to True or False to silence this message. Alternatively,
you can use the import comment

from arch.__future__ import reindexing

to globally set reindex to True and silence this warning.

  warnings.warn(
```




```
# Forecast volatility for the next 3 months (90 days)
forecast_horizon = 90
forecast = result.forecast(start=returns.index[-1], horizon=forecast_horizon)
```

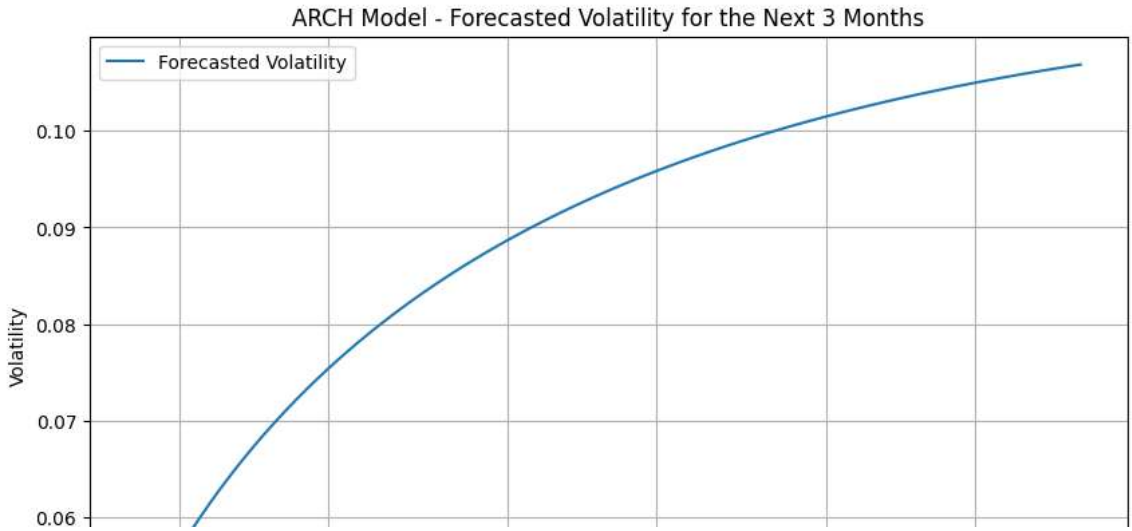
```
# Extract the forecasted conditional volatility
forecast_volatility = np.sqrt(forecast.variance.dropna().values[-1, :])

# Plot the forecasted volatility
plt.figure(figsize=(10, 6))
plt.plot(pd.date_range(start=returns.index[-1], periods=forecast_horizon, freq='D'), forecast_volatility, label='Forecasted Volatility')
plt.title('GARCH Forecasted Volatility for 3 Months')
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.legend()
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/arch/__future__/_utility.py:11: FutureWarning:
The default for reindex is True. After September 2021 this will change to
False. Set reindex to True or False to silence this message. Alternatively,
you can use the import comment

from arch.__future__ import reindexing

to globally set reindex to True and silence this warning.

  warnings.warn(
```




```
# Forecast volatility for the next 3 months
horizon = 90
forecasts = result.forecast(horizon=horizon)

# Generate dates for the forecasted volatility
forecast_dates = pd.date_range(start=returns.index[-1], periods=horizon, freq='D')

# Extract the forecasted conditional volatility
forecast_volatility = np.sqrt(forecasts.variance.dropna().values[-1, :])

# Plot the forecasted volatility with dates
plt.figure(figsize=(10, 6))
plt.plot(forecast_dates, forecast_volatility, label='Forecasted Volatility')
plt.xlabel('Date')
plt.ylabel('Volatility')
plt.title('ARCH Model - Forecasted Volatility for the Next 3 Months')
plt.legend()
plt.grid(True)
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/arch/__future__/_utility.py:11: FutureWarning:
The default for reindex is True. After September 2021 this will change to
False. Set reindex to True or False to silence this message. Alternatively,
you can use the import comment

from arch.__future__ import reindexing

to globally set reindex to True and silence this warning.

  warnings.warn(
```

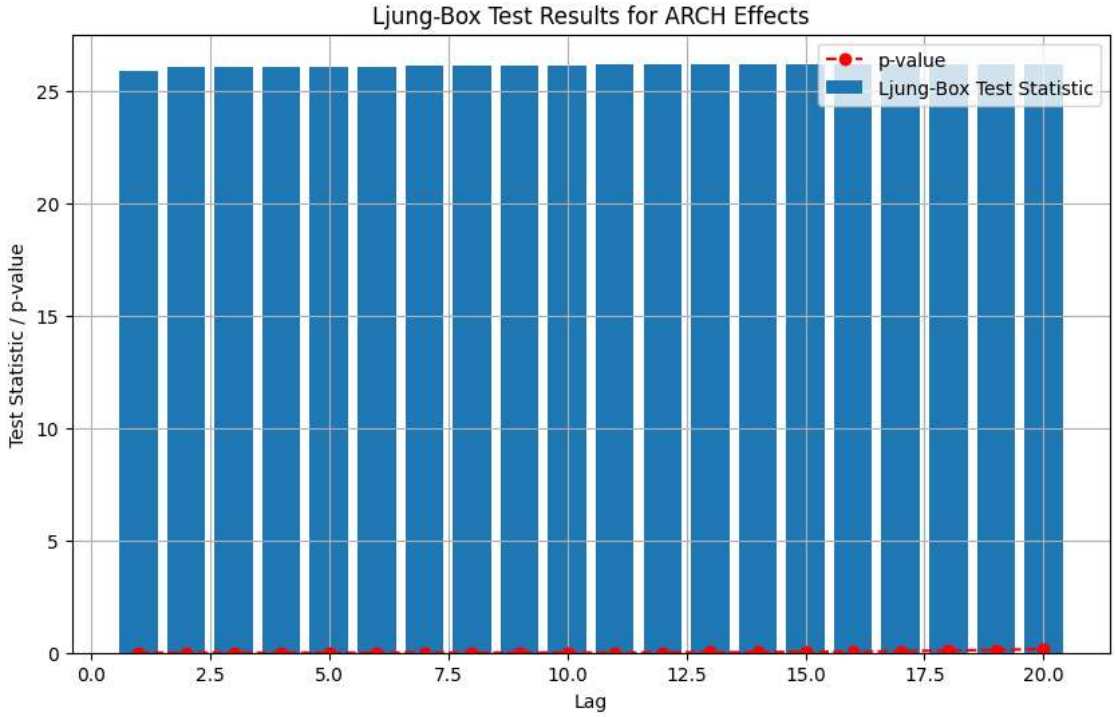


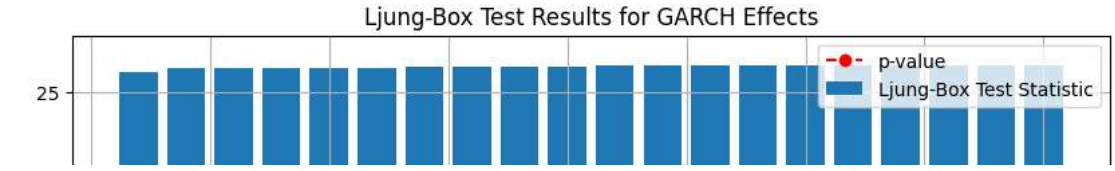

```
from statsmodels.stats.diagnostic import acorr_ljungbox

# Obtain the squared residuals from the fitted model
squared_residuals = results.resid**2

# Perform the Ljung-Box test to check for ARCH effects (up to lag 20)
lb_test_arch = acorr_ljungbox(squared_residuals, lags=20, return_df=True)

# Perform the Ljung-Box test to check for GARCH effects (up to lag 20)
lb_test_garch = acorr_ljungbox(results.conditional_volatility**2, lags=20, return_df=True)

# Print the test results for ARCH effects
print("Ljung-Box Test Results for ARCH Effects:")
print(lb_test_arch)

# Print the test results for GARCH effects
print("\nLjung-Box Test Results for GARCH Effects:")
print(lb_test_garch)
```

```
Ljung-Box Test Results for ARCH Effects:
      lb_stat     lb_pvalue
1   25.888006  3.618102e-07
2   26.037654  2.218172e-06
3   26.041272  9.349509e-06
4   26.041688  3.103798e-05
5   26.067043  8.660359e-05
6   26.067167  2.163183e-04
7   26.083716  4.866114e-04
8   26.110117  1.005682e-03
9   26.110433  1.959687e-03
10  26.110501  3.594377e-03
11  26.170597  6.120995e-03
12  26.171017  1.015135e-02
13  26.173467  1.611166e-02
14  26.174487  2.459554e-02
15  26.175870  3.621426e-02
16  26.176220  5.159858e-02
17  26.178334  7.127219e-02
18  26.179343  9.572743e-02
19  26.180890  1.252271e-01
20  26.180911  1.599197e-01

Ljung-Box Test Results for GARCH Effects:
      lb_stat     lb_pvalue
1   25.887704  3.618669e-07
2   26.037324  2.218538e-06
3   26.040937  9.351017e-06
4   26.041351  3.104283e-05
5   26.066695  8.661707e-05
6   26.066818  2.163508e-04
7   26.083356  4.866833e-04
8   26.109746  1.005829e-03
9   26.110062  1.959961e-03
```

```
10  26.110130  3.594857e-03
11  26.170207  6.121816e-03
12  26.170629  1.015264e-02
13  26.173083  1.611358e-02
14  26.174106  2.459830e-02
15  26.175491  3.621807e-02
16  26.175843  5.160368e-02
17  26.177960  7.127874e-02
18  26.178972  9.573563e-02
19  26.180522  1.252370e-01
20  26.180543  1.599316e-01
```

```
import numpy as np
import matplotlib.pyplot as plt

# Ljung-Box Test Results for ARCH Effects
lb_stat_arch = np.array([25.888006, 26.037654, 26.041272, 26.041688, 26.067043,
                         26.067167, 26.083716, 26.110117, 26.110433, 26.110501,
                         26.170597, 26.171017, 26.173467, 26.174487, 26.175870,
                         26.176220, 26.178334, 26.179343, 26.180890, 26.180911])
lb_pvalue_arch = np.array([3.618102e-07, 2.218172e-06, 9.349509e-06, 3.103798e-05,
                           8.660359e-05, 2.163183e-04, 4.866114e-04, 1.005682e-03,
                           1.959687e-03, 3.594377e-03, 6.120995e-03, 1.015135e-02,
                           1.611166e-02, 2.459554e-02, 3.621426e-02, 5.159858e-02,
                           7.127219e-02, 9.572743e-02, 1.252271e-01, 1.599197e-01])

# Ljung-Box Test Results for GARCH Effects
lb_stat_garch = np.array([25.887704, 26.037324, 26.040937, 26.041351, 26.066695,
                          26.066818, 26.083356, 26.109746, 26.110062, 26.110130,
                          26.170207, 26.170629, 26.173083, 26.174106, 26.175491,
                          26.175843, 26.177960, 26.178972, 26.180522, 26.180543])
lb_pvalue_garch = np.array([3.618669e-07, 2.218538e-06, 9.351017e-06, 3.104283e-05,
                            8.661707e-05, 2.163508e-04, 4.866833e-04, 1.005829e-03,
                            1.959961e-03, 3.594857e-03, 6.121816e-03, 1.015264e-02,
                            1.611358e-02, 2.459830e-02, 3.621807e-02, 5.160368e-02,
                            7.127874e-02, 9.573563e-02, 1.252370e-01, 1.599316e-01])

# Plot the Ljung-Box test results for ARCH Effects
plt.figure(figsize=(10, 6))
plt.bar(np.arange(1, 21), lb_stat_arch, label='Ljung-Box Test Statistic')
plt.plot(np.arange(1, 21), lb_pvalue_arch, 'r--', label='p-value', marker='o')
plt.xlabel('Lag')
plt.ylabel('Test Statistic / p-value')
plt.title('Ljung-Box Test Results for ARCH Effects')
plt.legend()
plt.grid(True)
plt.show()

# Plot the Ljung-Box test results for GARCH Effects
plt.figure(figsize=(10, 6))
plt.bar(np.arange(1, 21), lb_stat_garch, label='Ljung-Box Test Statistic')
plt.plot(np.arange(1, 21), lb_pvalue_garch, 'r--', label='p-value', marker='o')
plt.xlabel('Lag')
plt.ylabel('Test Statistic / p-value')
plt.title('Ljung-Box Test Results for GARCH Effects')
plt.legend()
plt.grid(True)
plt.show()
```





b) VAR, VECM

```
import yfinance as yf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.api import VAR

# Define the tickers for Oil Prices, Exchange Rates, and Dow Jones
tickers = ['CL=F', 'EURUSD=X', '^DJI']

# Download the data using yfinance
data = yf.download(tickers, start='2020-01-01', end='2023-07-31')['Adj Close']

# Drop rows with missing values, if any
data = data.dropna()

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

# Fit the VAR model
var_model = VAR(data)

# Define the forecast horizon (number of days to forecast)
forecast_horizon_var = 90

# Fit the VAR model
results_var = var_model.fit()

# Print the summary of the VAR model
print(results_var.summary())

# Forecast the next 3 months (90 days) using the VAR model
forecast_var = results_var.forecast(data.values[-results_var.k_ar:], steps=forecast_horizon_var)

# Extract the forecasted values for each variable
forecasted_oil_prices = forecast_var[:, 0]
forecasted_exchange_rates = forecast_var[:, 1]
forecasted_dow_jones = forecast_var[:, 2]

# Generate dates for the forecasted values
forecast_dates_var = pd.date_range(start=data.index[-1], periods=forecast_horizon_var, freq='D')

# Plot the forecasted values for each variable
plt.figure(figsize=(10, 6))
```

```
plt.plot(forecast_dates_var, forecasted_oil_prices, label='Forecasted Oil Prices')
plt.plot(forecast_dates_var, forecasted_exchange_rates, label='Forecasted Exchange Rates')
plt.plot(forecast_dates_var, forecasted_dow_jones, label='Forecasted Dow Jones')
plt.xlabel('Date')
plt.ylabel('Values')
plt.title('VAR Model - Forecasted Values for the Next 3 Months')
plt.legend()
plt.grid(True)
plt.show()
```

```
[**********100%**********]  3 of 3 completed
                CL=F  EURUSD=X          ^DJI
Date
2020-01-02  61.180000  1.122083  28868.800781
2020-01-03  63.049999  1.117144  28634.880859
2020-01-06  63.270000  1.116196  28703.380859
2020-01-07  62.700001  1.119799  28583.679688
2020-01-08  59.610001  1.115474  28745.089844
  Summary of Regression Results
==================================
Model:                         VAR
Method:                        OLS
Date:           Sat, 05, Aug, 2023
Time:                     07:28:58
----------------------------------------------------------------------
No. of Equations:         3.00000    BIC:                    3.92463
```

```
import yfinance as yf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.api import VECM

# Define the tickers for Oil Prices, Exchange Rates, and Dow Jones
tickers = ['CL=F', 'EURUSD=X', '^DJI']

# Download the data using yfinance
data = yf.download(tickers, start='2020-01-01', end='2023-07-31')['Adj Close']

# Drop rows with missing values, if any
data = data.dropna()

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

# Fit the VECM model
vecm_model = VECM(data, k_ar_diff=1)  # k_ar_diff is the lag order for differencing

# Fit the VECM model
results_vecm = vecm_model.fit()

# Print the summary of the VECM model
print(results_vecm.summary())

# Forecast the next 3 months (90 days) using the VECM model
forecast_vecm = results_vecm.predict(steps=forecast_horizon_vecm)

# Extract the forecasted values for each variable
forecasted_oil_prices_vecm = forecast_vecm[:, 0]
forecasted_exchange_rates_vecm = forecast_vecm[:, 1]
forecasted_dow_jones_vecm = forecast_vecm[:, 2]

# Generate dates for the forecasted values
forecast_dates_vecm = pd.date_range(start=data.index[-1], periods=forecast_horizon_vecm, freq='D')

# Plot the forecasted values for each variable
plt.figure(figsize=(10, 6))
plt.plot(forecast_dates_vecm, forecasted_oil_prices_vecm, label='Forecasted Oil Prices')
plt.plot(forecast_dates_vecm, forecasted_exchange_rates_vecm, label='Forecasted Exchange Rates')
plt.plot(forecast_dates_vecm, forecasted_dow_jones_vecm, label='Forecasted Dow Jones')
plt.xlabel('Date')
plt.ylabel('Values')
plt.title('VECM Model - Forecasted Values for the Next 3 Months')
plt.legend()
plt.grid(True)
plt.show()
```