RELATIONSHIP BETWEEN US/CANADA EXCHANGE RATE AND COMMODITY PRICES

ECONOMETRICS II: PROJECT I

NANA OSEI SARPONG
11371873

Contents

INTRODUCTION

Commodity currency is floating currency that exhibit co-movement with world prices of primary commodities due to a country's heavy dependence on commodity exports (Chen et al., 2010). Canada as a country boasts of a wide range of natural resources and depends on revenue from the exploitation and export of such resources. According to Statistics Canada, export commodities such as energy products, metal and non-metallic mineral products, metal ores and non-metallic minerals, forestry products, building and packaging materials, farm, fishing and intermediate food products contributed $ 443.5 billion in 2022, accounting for 56.92% of total exports. The large dependence on such well traded commodities on the world market characterises the Canadian economy making it feasible to regard the Canadian dollar as a commodity currency. In this paper, we set out to explore the empirical relationship between in Canadian exchange rate and commodity prices.

THEORETICAL FRAMEWORK

According to Chen et al. (2010), the exchange rate is forward looking and embodies information about future movement of commodity prices. Changes in commodity prices affects exports, and results in fluctuations in the terms of trade which finally translates into exchange rate movements. Expectations of commodity price shocks are thereby reflected in the current exchange rate due to expected impact on future export income and future exchange rate values.

Commodity price linkages through asset markets due to a portfolio channel serve as a plausible explanation for a relationship between the exchange rate and commodity prices. Countries that are major exporters of commodities will potentially observe a surge in foreign investment if market participants expect commodity prices will go up. This inflow affects the rate of exchange due to the increased demand for commodity currency and serves as a potential explanation for a connection between the exchange rate and commodity prices.

Macro-models also result in an equivalent relationship between the exchange rate and commodity prices. Models relating to money market equilibrium, uncovered interest parity and purchasing power parity conditions lend themselves to such a relationship. Purchasing power parity condition requires the exchange rate to equalize the prices of baskets of goods in different countries. As such, commodity price changes are reflected in exchange rate changes to restore equilibrium. Money market equilibrium and uncovered interest parity posit that investors will take advantage of discrepancies between foreign and domestic rates of return which affects the exchange rate and may affect the price of imports. All these link commodity prices and the rate of exchange.

METHODOLOGY

In this study, we will employ various techniques to investigate the relationship between the Canadian exchange rate and commodity prices. Our approach will descriptive statistics analysis, correlation analysis, unit root testing using the augmented Dickey-Fuller (ADF) test, cointegration testing with the Engel-Granger test, regression analysis, and Granger causality testing. Through these analytical techniques, we aim to comprehensively understand the empirical dynamics between the exchange rate and commodity prices in Canada.

DATA DESCRIPTION

This study makes use of data obtained from Statistics Canada. Contained in the data set are seven monthly time series variables with 544 observations from January 1972 to April 2017. The exchange rate is measured as the United States dollar to the Canadian dollar, noon spot rate average and all price indices are Fisher commodity price indices in terms of US dollars. Below is a table containing descriptive statistics of the variables in our data.

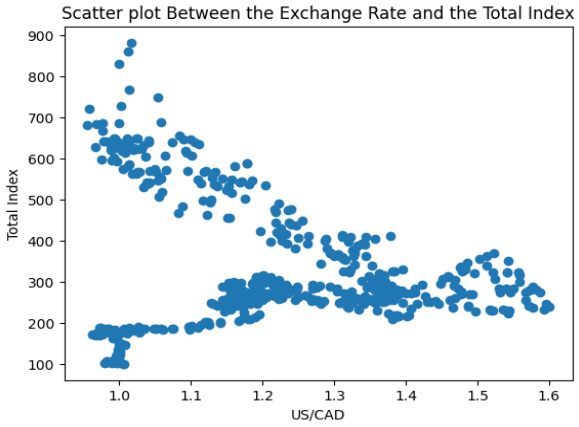
Table 1: Descriptive Statistics

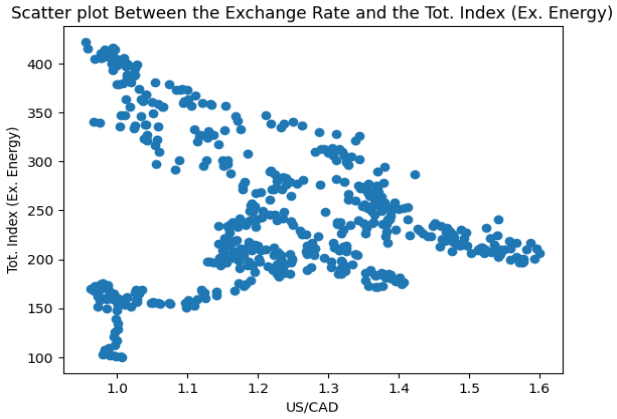
Variable	**Minimum**	**Median**	**Maximum**	**Mean**	**Standard deviation**
Exchange rate (US/CAD)	0.9553	1.2046	1.6003	1.2215	0.1667
Total Commodity Price Index	100.0000	281.5500	881.3000	333.0594	150.0565
Total Commodity Price Index (Excluding Energy)	100.0000	222.9000	422.1000	242.4548	72.4168
Energy Index	99.8000	570.3000	2755.8000	739.9851	492.4327
Metals and Minerals Index	100.0000	272.4500	779.9000	337.3149	167.2277
Agriculture Index	100.0000	176.0000	322.9000	187.2169	45.9929
Fish Index	86.3000	629.2500	1503.6000	624.0253	367.8258
Forestry Index	100.0000	269.4500	411.0000	260.8846	72.9207

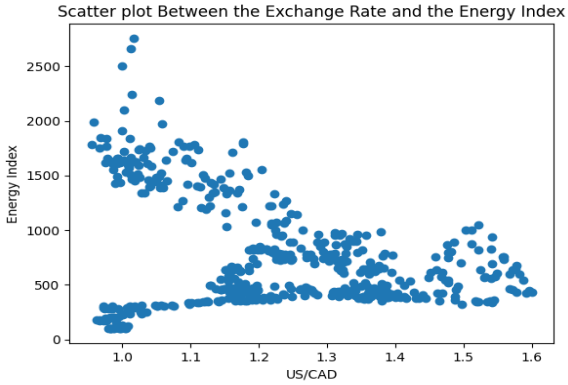
Compared to most of the other variables, the exchange rate has the least volatility with a standard deviation of a mere 0.1667 around an average of $1.2215 US/CAD. All the price indices have a starting value of 100 at 1972 but from the descriptive statistics, only the Energy and Fish indices fall below their initial value at a point in time. The indices vary vastly in terms of their averages and their volatility, but the Energy index compared to all the other indices has the largest mean (739.9851) and the largest standard deviation (492.4327), reflecting high volatility in energy prices.

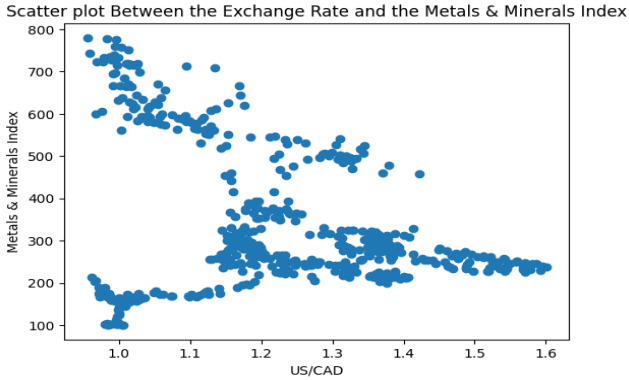
CORRELATION ANALYSIS

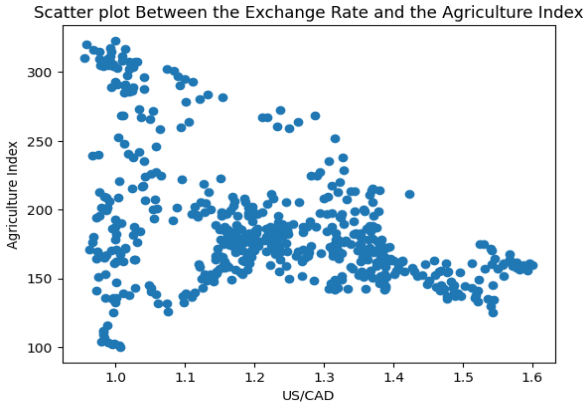
A relationship between the exchange rate can easily be explored by looking at scatter plots and the correlation between the exchange rate and the various price indices.

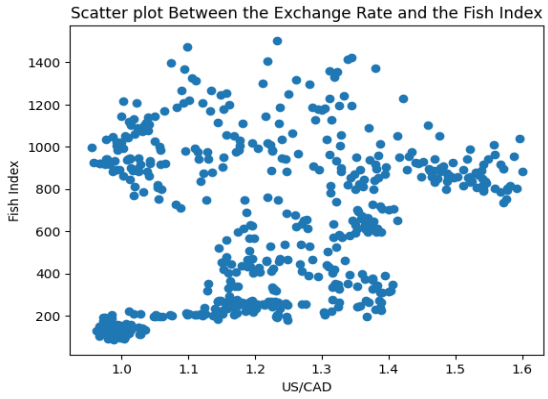


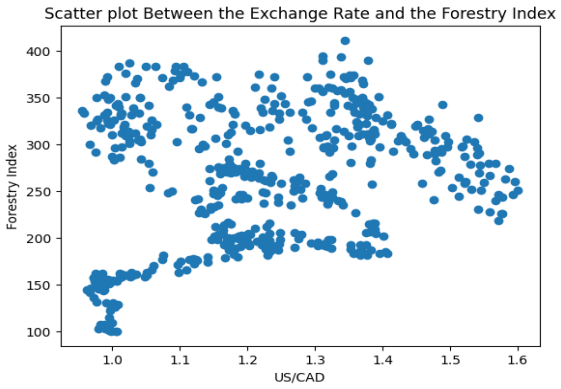












The scatter plots are mostly inconclusive. Most of the plots do not show a clear pattern to enable us to theorize about the nature of the relationship. For most of the graphs, there seems to be a negative relationship or not relationship at all.

The table below displays the Pearson correlation coefficients between the exchange rate and the various price indices.

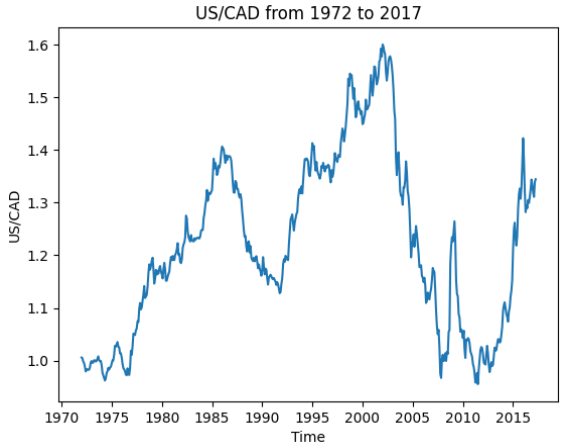
Table 2: Pearson correlation coefficients

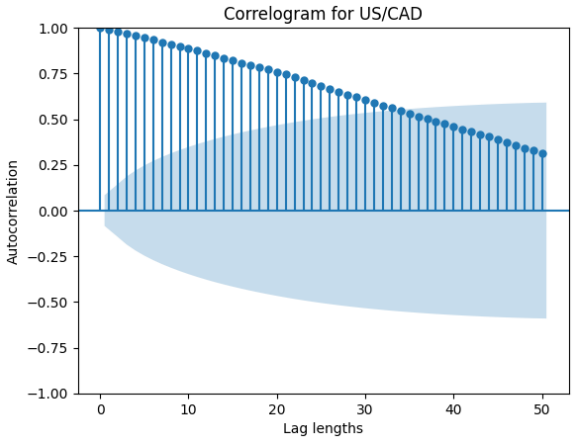
Index	**Correlation coefficient**	**P-value**
Total Commodity Price Index	-0.2714	0.0000
Total Commodity Price Index (Excluding Energy)	-0.1818	0.0000
Energy Index	-0.2806	0.0000
Metals and Minerals Index	-0.3088	0.0000
Agriculture Index	-0.4164	0.0000
Fish Index	0.2378	0.0000
Forestry Index	0.2578	0.0000

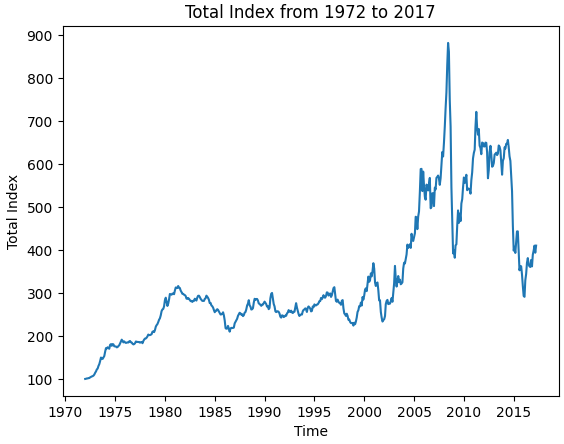
All the indices apart from the Fish and the Forest indices exhibit a negative relationship. The correlation coefficients are statistically significant at the 5% level as seen in the p-value. However, when dealing with time series data, one must be careful when interpreting such traditional measures since they can be 'non-sensical' in the presence of a unit root.

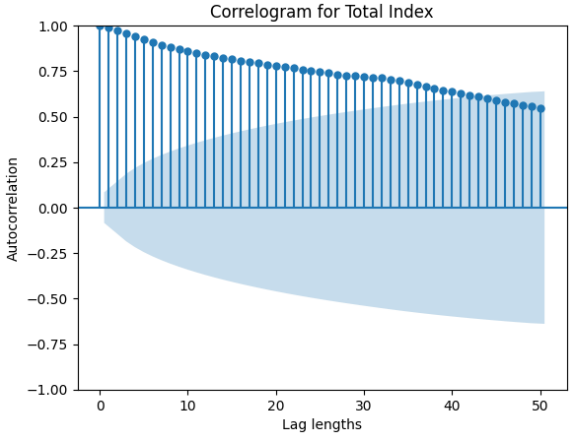
UNIT ROOT TEST

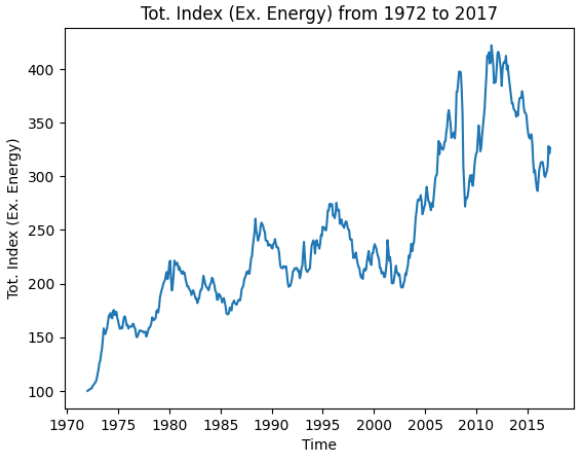
The following graphs display how the various variables evolve over time and their respective autocorrelation functions shown with a correlogram.

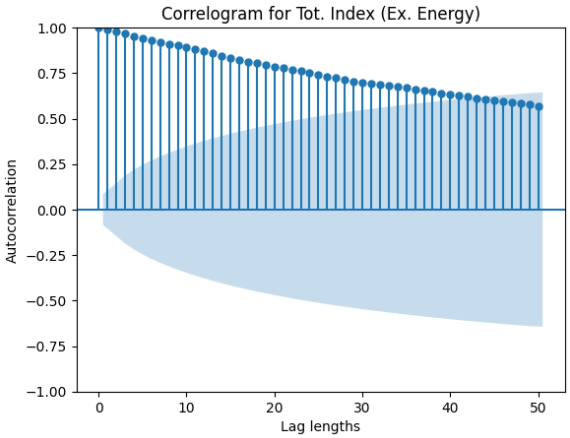

US/CAD from 1972 to 2017
US/CAD
Time

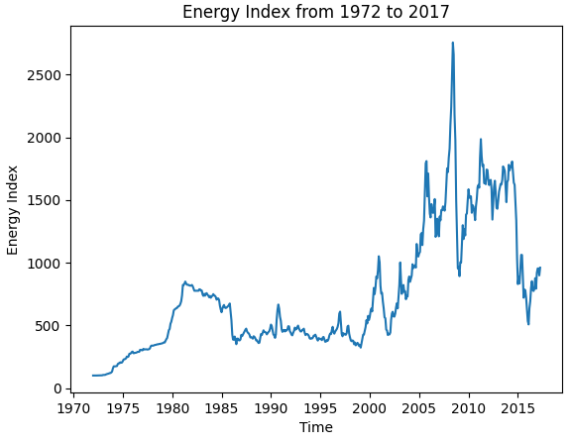

Correlogram for US/CAD
Autocorrelation
Lag lengths

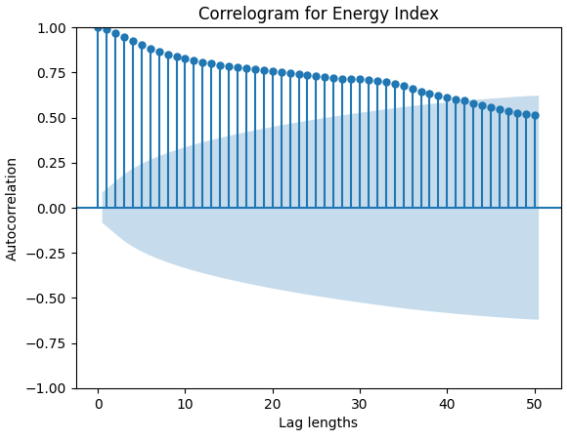

Total Index from 1972 to 2017
Total Index
Time

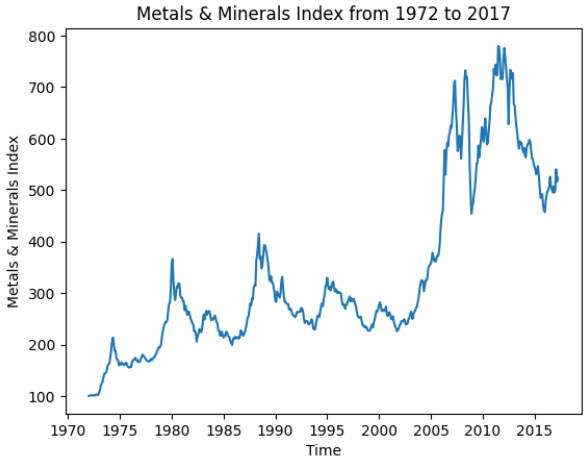

Correlogram for Total Index
Autocorrelation
Lag lengths

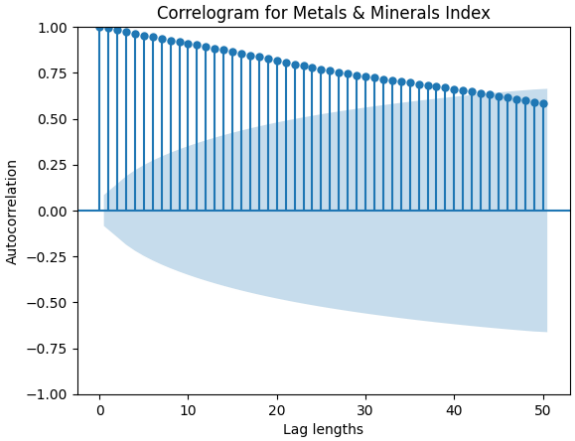

Tot. Index (Ex. Energy) from 1972 to 2017
Tot. Index (Ex. Energy)
Time

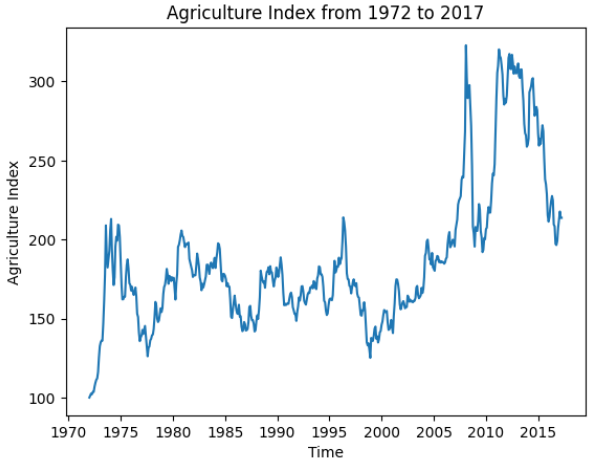

Correlogram for Tot. Index (Ex. Energy)
Autocorrelation
Lag lengths

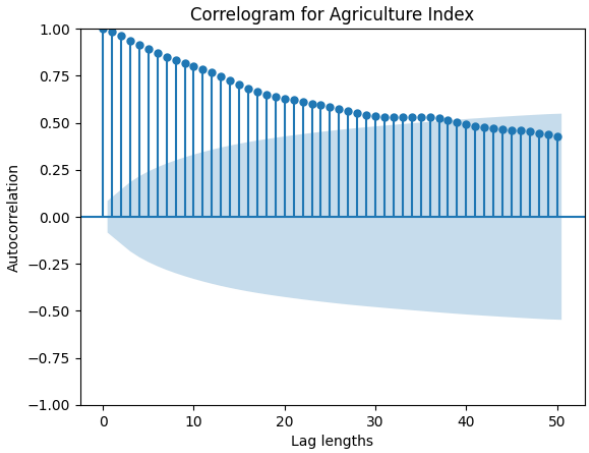

Energy Index from 1972 to 2017
Energy Index
Time

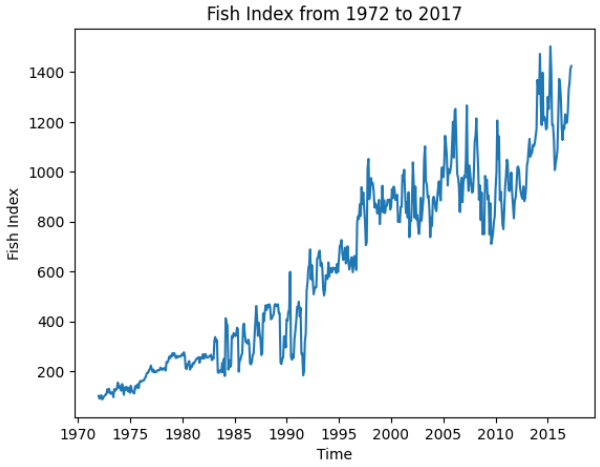

Correlogram for Energy Index
Autocorrelation
Lag lengths

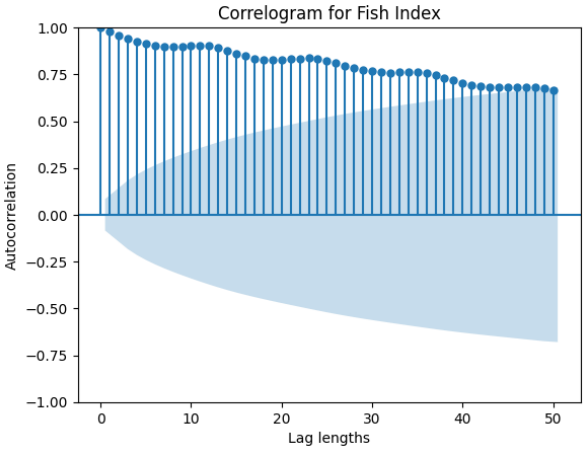

Metals & Minerals Index from 1972 to 2017
Metals & Minerals Index
Time

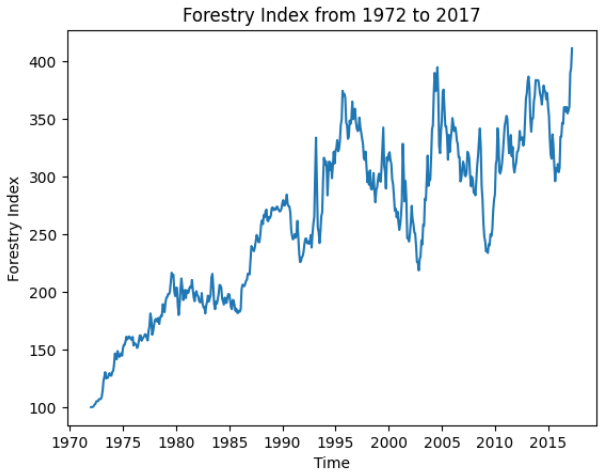

Correlogram for Metals & Minerals Index
Autocorrelation
Lag lengths

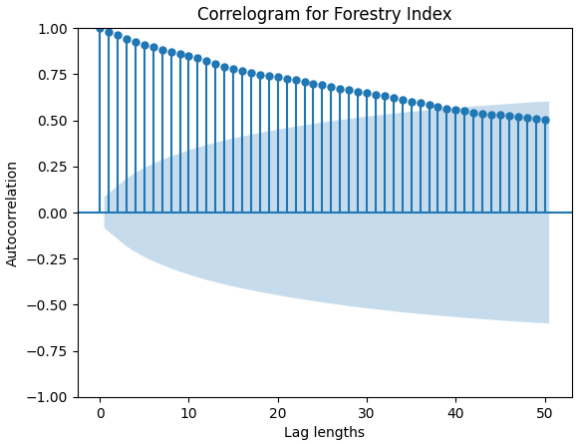

Agriculture Index from 1972 to 2017
Agriculture Index
Time


Correlogram for Agriculture Index
Autocorrelation
Lag lengths







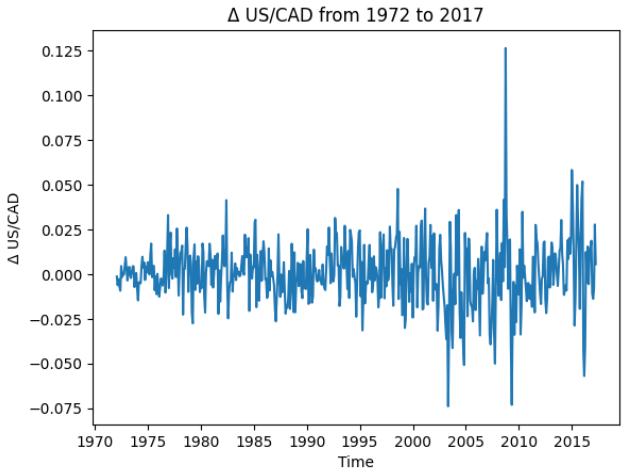


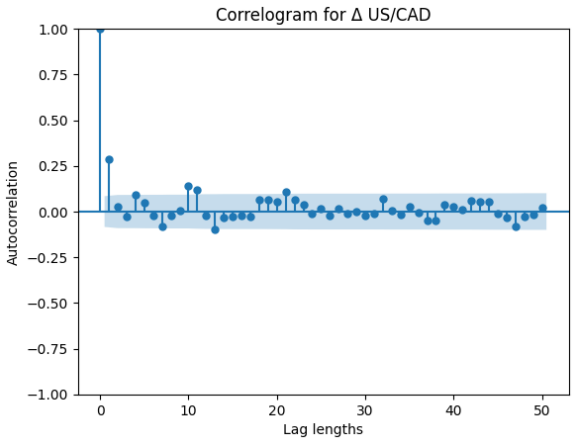
The graphs confirm suspicion of a possible unit root. Some variables show a clear trend pattern. The respective correlograms show that the variables are correlated with their past values and this correlation does not disappear even after 50 lags. We can objectively confirm this by conducting a unit root test. The test to be used is the augmented Dickey Fuller test with constant and trend. The table below shows the results of the test under the null hypothesis of the presence of a unit root. Lags are chosen for the test using the Bayesian Information Criterion (BIC).

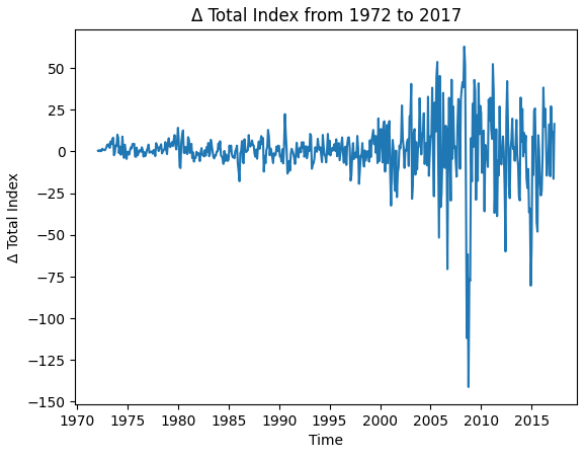
Table 3: Augmented Dickey Fuller Test at Levels

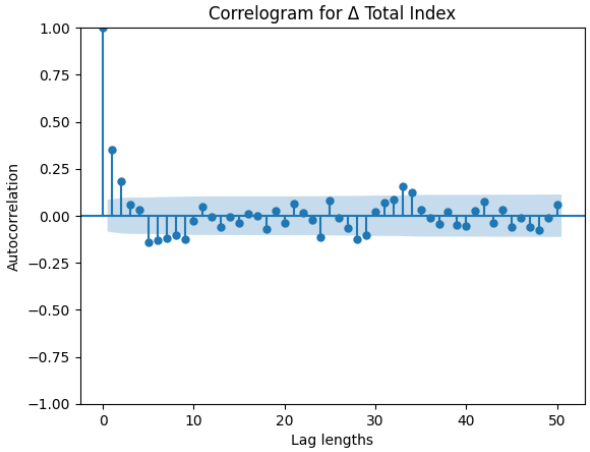
Variable	Test Statistic	P-Value	Optimal lags
Exchange rate (US/CAD)	-1.8204	0.6949	1
Total Commodity Price Index	-3.1898	0.0865	1
Total Commodity Price Index (Excluding Energy)	-3.0732	0.1128	1
Energy Index	-3.3186	0.0632	1
Metals and Minerals Index	-2.3206	0.4226	1
Agriculture Index	-3.3486	0.0587	1
Fish Index	-3.0112	0.1290	13
Forestry Index	-4.0985	0.0063	1

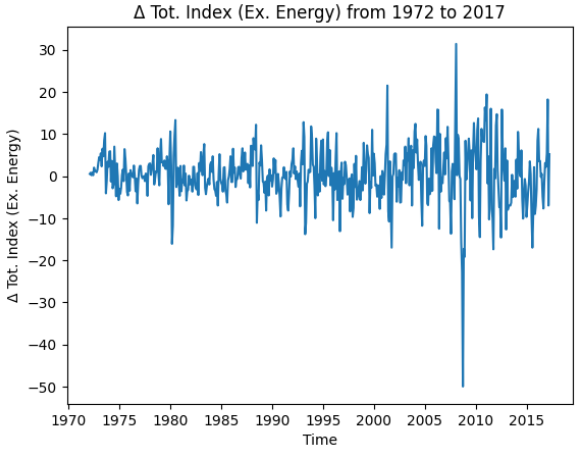
Based on the test, we fail to reject the null hypothesis at the 5% level of significance and conclude that a unit root is present in the variables. The Forestry Index however is the only variable where we reject the null hypothesis. It is thereby stationary. We can proceed with first differencing to rid the variables of the unit root issue.

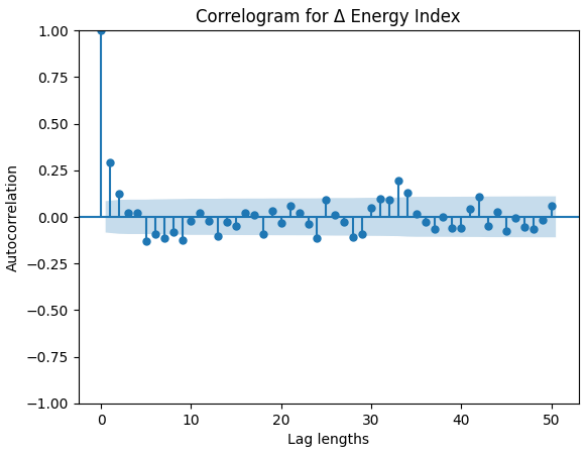


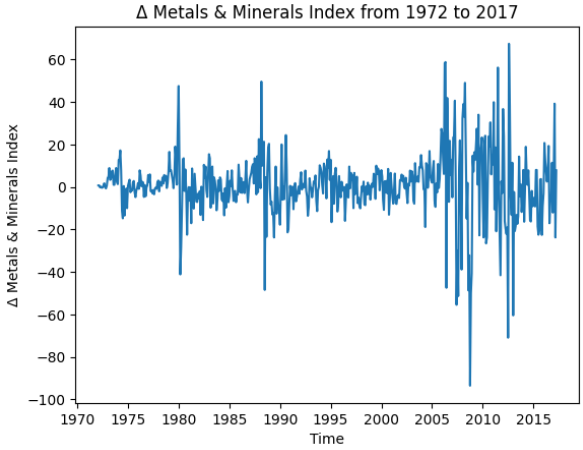


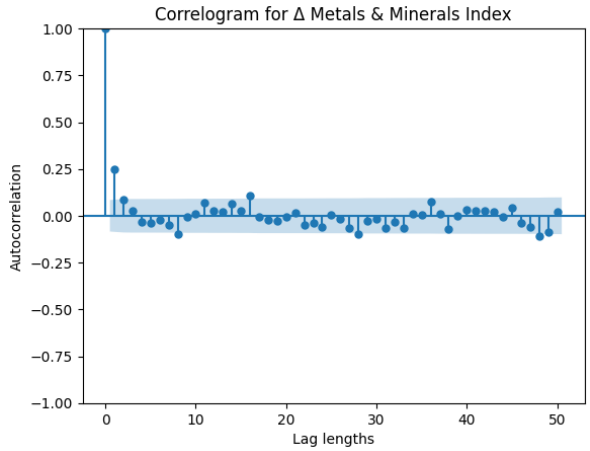

Δ Total Index from 1972 to 2017
Δ Total Index
Time

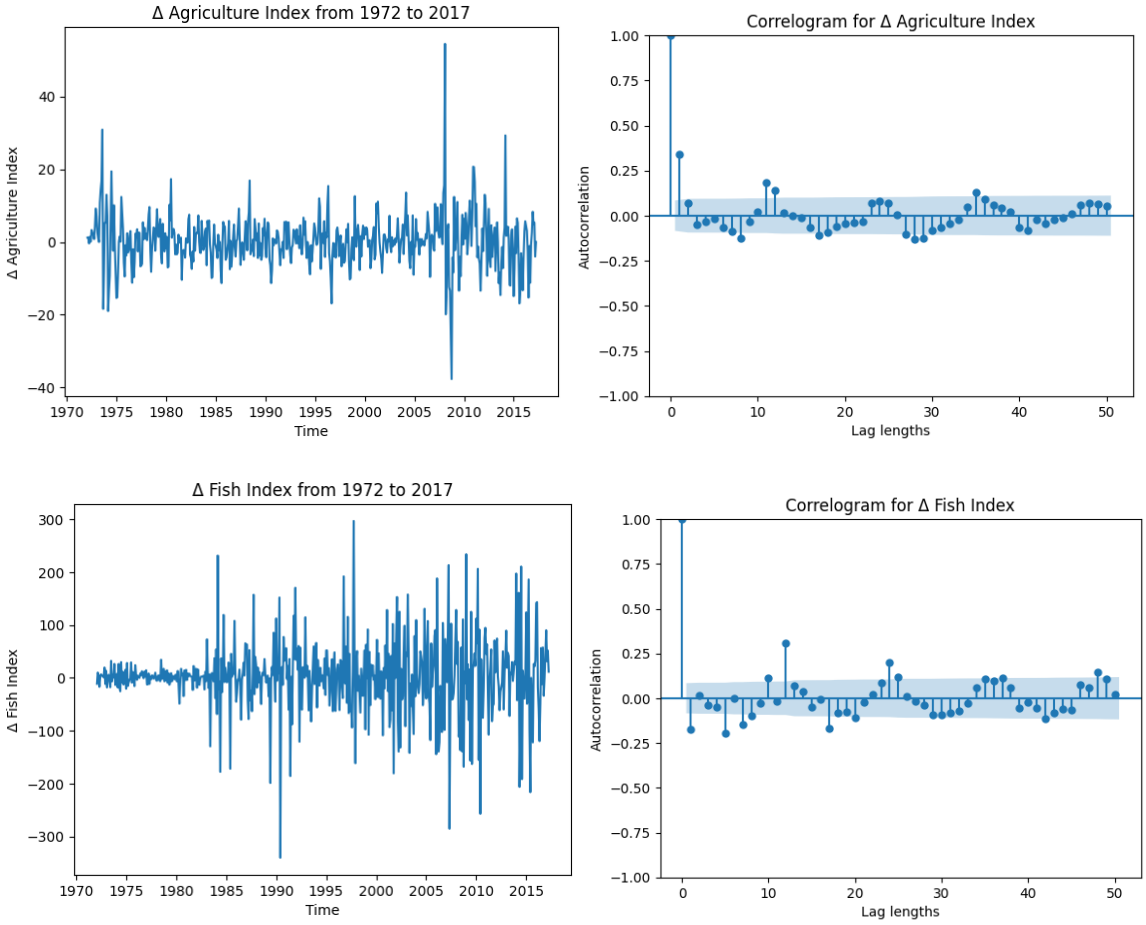

Correlogram for Δ Total Index
Autocorrelation
Lag lengths


Δ Tot. Index (Ex. Energy) from 1972 to 2017
Δ Tot. Index (Ex. Energy)
Time


Correlogram for Δ Energy Index
Autocorrelation
Lag lengths


Δ Metals & Minerals Index from 1972 to 2017
Δ Metals & Minerals Index
Time


Correlogram for Δ Metals & Minerals Index
Autocorrelation
Lag lengths



From the graphs above, it is observed that after first differencing, the discernible pattern is removed from the variables. The autocorrelation function shown by the correlogram shows that the relationship with lags of the variable disappears. We conduct the augmented Dickey Fuller test once more to confirm that the unit root problem has been tackled.

Table 4: Augmented Dickey Fuller Test at first Differences

Variable	**Test Statistic**	**P-Value**	**Optimal lags**
Exchange rate (US/CAD)	-17.2789	0.0000	0
Total Commodity Price Index	-16.0451	0.0000	0
Total Commodity Price Index (Excluding Energy)	-16.4766	0.0000	0
Energy Index	-17.1613	0.0000	0
Metals and Minerals Index	-18.0159	0.0000	0
Agriculture Index	-16.2813	0.0000	0
Fish Index	-7.5743	0.0000	12

The p-value shows that the presence of the unit root disappears after first differencing. We now turn to how well changes in commodity prices can be forecasted by using the exchange rate which according to theory and findings of Chen et al. (2010) has good forecasting ability. For simplicity of analysis, we shall focus on forecasting just the total commodity price index.

COINTEGRATION TEST

Although variables containing unit root could lead to a spurious regression when used in a model, there exists a possibility of cointegration, a phenomenon whereby a linear combination of the variables is stationary. The test used is the Engel-Granger cointegration test with the null hypothesis of no cointegration. The table below displays the results of the test.

Table 5: Engle-Granger Cointegration Test Between Total Price Index and the Exchange Rate

Test Statistic	P-Value
-2.9844	0.2706

Based on the test results, we fail to reject the null hypothesis and conclude that there exists no long run relationship between the exchange rate and the total commodity price index.

This may be true but perhaps the exchange rate may still be useful in making predictions of the total commodity price index.

REGRESSION ANALYSIS

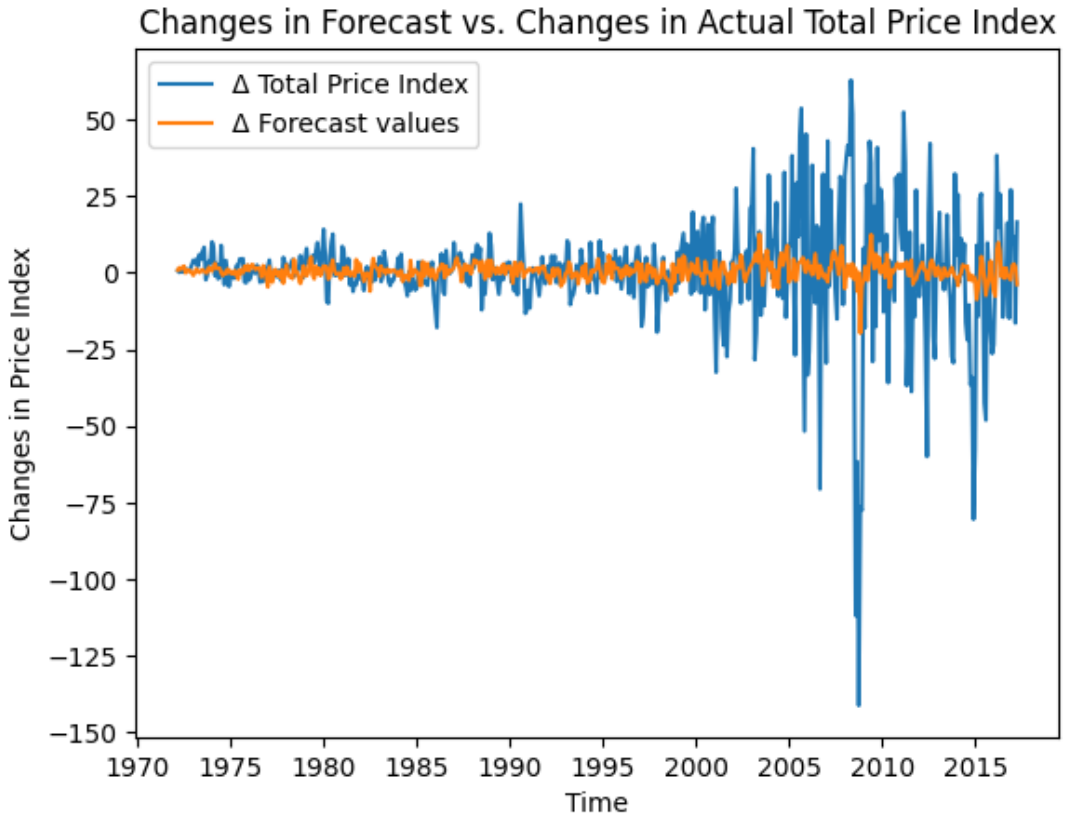
We run the following regression model:

$$\Delta tcpi_{t+1} = \beta_0 + \beta_1 \Delta ex_t + \varepsilon_t$$

The results are as follows:

Table 6: Regression of Future Changes in Total Commodity Price Index on Changes in the Exchange Rate

Dependent variable: Δ Total Commodity Price Index $_{t+1}$				
	Coefficient	**Std Error**	**T Statistic**	**P-value**
constant	0.6703	0.757	0.885	0.377
Δ Exchange rate	-161.3003	42.614	-3.785	0.000



The regression results show that there is a negative and significant relationship between changes in the exchange rate and changes in future total commodity prices. The graph above shows how close the forecast based on our model is to actual changes in commodity prices. There is considerable co-movement although somewhat tempered in volatility. The negative coefficient from our regression however contradicts the findings of Chen et al. (2010) who found positive significant relationships between world commodity prices and the exchange rates of five selected countries.

GRANGER CAUSALITY TEST

A Granger causality test is conducted to confirm the usefulness of changes in the exchange rate in predicting future changes in the total commodity price index. The table below shows the results of the test.

Table 7: Granger Causality Test

SSR Based F-test	P-value	Lag order
1.7408	0.1876	1
1.0963	0.3348	2
1.0974	0.3497	3
0.9347	0.4434	4
2.0709	0.0676	5
2.5373	0.0198	6
2.5582	0.0134	7
3.1865	0.0015	8
3.7176	0.0002	9
3.6129	0.0001	10

The test results show that changes in the exchange rate may not be useful for forecasting future changes in the total commodity price index until after the fifth lag order.

CONCLUSION

This study investigates the relationship between the Canadian exchange rate and commodity prices, given Canada's heavy dependence on commodity exports. The analysis examines the empirical link considering theoretical frameworks and methodological approaches such as correlation analysis, regression analysis, cointegration testing, and Granger causality testing.

Our findings suggest that there exists no long run relationship between the exchange rate and commodity prices. Regression analysis demonstrates a negative and significant relationship between changes in the exchange rate and future changes in the total commodity price index. However, Granger causality tests suggest that changes in the exchange rate may not be useful for forecasting future changes in the total commodity price index until after the fifth lag order.

While the study finds evidence of some relationship between the Canadian exchange rate and commodity prices, it underscores the importance of employing various analytical tools to understand the complex dynamics between the exchange rate and commodity price indices. The results suggest limited predictive power of the exchange rate on future commodity price movements, indicating the need for further research and consideration of additional factors in forecasting commodity prices.

REFERENCES

Chen, Y.-C., Rogoff, K.S., Rossi, B. (2010) “Can exchange rates forecast commodity prices” Quarterly Journal of Economics 125, pp. 1145 – 94

Statistics Canada, Table 12-10-0122-01, https://www.international.gc.ca/transparency-transparence/state-trade-commerce-international/2023.aspx?lang=eng

appendix-i

February 6, 2024

[1]:
```
# Importing relevant libraries
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from scipy.stats import pearsonr
from statsmodels.graphics.tsaplots import plot_acf
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.stattools import coint
from statsmodels.tsa.stattools import grangercausalitytests
from statsmodels.tsa.api import VAR
```

[3]:
```
# Loading the first dataset
exchange_rates = pd.read_csv('StatsCanExchangeRates.csv')
exchange_rates.head()
```

[3]:

	REF_DATE	GEO	DGUID	Type of currency	UOM	UOM_ID	SCALAR_FACTOR	SCALAR_ID	VECTOR	COORDINATE	VALUE	STATUS	SYMBOL	TERMINATED	DECIMALS
0	1950-10	Canada	NaN	United States dollar, noon spot rate, average	Dollars	81	units	0	v37426	1.10	1.053333	NaN	NaN	NaN	8
1	1950-10	Canada	NaN	United States dollar, 90-day forward noon rate	Dollars	81	units	0	v37437	1.22	1.047313	NaN	NaN	NaN	8
2	1950-10	Canada	NaN	Belgian franc, noon spot rate, average	Dollars	81	units	0	v37448	1.20	0.020928	NaN	NaN	t	8
3	1950-10	Canada	NaN	Danish krone, noon spot rate, average	Dollars	81	units	0	v37452	1.30	0.152562	NaN	NaN	NaN	8
4	1950-10	Canada	NaN	French franc, noon spot rate, average	Dollars	81	units	0	v37453	1.40	0.003014	NaN	NaN	t	8

[4]:
```
# Filtering for only US/CAD related data
exchange_rates = exchange_rates[exchange_rates['Type of currency'] == 'United␣
 ↪States dollar, noon spot rate, average']
exchange_rates.head()
```

[4]:

	REF_DATE	GEO	DGUID	Type of currency	\
0	1950-10	Canada	NaN	United States dollar, noon spot rate, average	
13	1950-11	Canada	NaN	United States dollar, noon spot rate, average	
26	1950-12	Canada	NaN	United States dollar, noon spot rate, average	
39	1951-01	Canada	NaN	United States dollar, noon spot rate, average	
55	1951-02	Canada	NaN	United States dollar, noon spot rate, average	

	UOM	UOM_ID	SCALAR_FACTOR	SCALAR_ID	VECTOR	COORDINATE	VALUE	\
0	Dollars	81	units	0	v37426	1.1	1.053333	
13	Dollars	81	units	0	v37426	1.1	1.040312	
26	Dollars	81	units	0	v37426	1.1	1.053078	
39	Dollars	81	units	0	v37426	1.1	1.051875	
55	Dollars	81	units	0	v37426	1.1	1.049125	

	STATUS	SYMBOL	TERMINATED	DECIMALS
0	NaN	NaN	NaN	8
13	NaN	NaN	NaN	8
26	NaN	NaN	NaN	8
39	NaN	NaN	NaN	8
55	NaN	NaN	NaN	8

[5]:
```
# Filtering for only relevant columns
filtered_ex_rate = exchange_rates[['REF_DATE', 'VALUE']]
filtered_ex_rate.columns = ['Date','US/CAD']

# Converting the date to a date type
filtered_ex_rate['Date'] = pd.to_datetime(filtered_ex_rate['Date'],␣
 ↪format='%Y-%m')

filtered_ex_rate.head()
```

```
C:\Programs\Anaconda3\TEMP\ipykernel_10168\872351796.py:6:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  filtered_ex_rate['Date'] = pd.to_datetime(filtered_ex_rate['Date'],
format='%Y-%m')
```

[5]:
```
          Date    US/CAD
0   1950-10-01  1.053333
13  1950-11-01  1.040312
26  1950-12-01  1.053078
39  1951-01-01  1.051875
55  1951-02-01  1.049125
```

[6]:
```
filtered_ex_rate.tail()
```

[6]:
```
             Date    US/CAD
20644 2016-12-01  1.332935
20677 2017-01-01  1.319090
20710 2017-02-01  1.310989
20743 2017-03-01  1.338752
20776 2017-04-01  1.344395
```

[8]:
```
# Loading second dataset
price_indices = pd.read_csv('StatsCanPriceIndices.csv')
price_indices.head(10)
```

[8]:
```
  REF_DATE     GEO           DGUID                Commodity              UOM  \
0  1972-01  Canada  2016A000011124  Total, all commodities  Index, 1972=100
1  1972-01  Canada  2016A000011124  Total excluding energy  Index, 1972=100
2  1972-01  Canada  2016A000011124                  Energy  Index, 1972=100
3  1972-01  Canada  2016A000011124    Metals and Minerals  Index, 1972=100
4  1972-01  Canada  2016A000011124             Agriculture  Index, 1972=100
5  1972-01  Canada  2016A000011124                    Fish  Index, 1972=100
6  1972-01  Canada  2016A000011124               Forestry  Index, 1972=100
7  1972-02  Canada  2016A000011124  Total, all commodities  Index, 1972=100
8  1972-02  Canada  2016A000011124  Total excluding energy  Index, 1972=100
9  1972-02  Canada  2016A000011124                  Energy  Index, 1972=100

   UOM_ID SCALAR_FACTOR  SCALAR_ID     VECTOR  COORDINATE  VALUE  STATUS  \
0     166        units           0  v52673496         1.1  100.0     NaN
1     166        units           0  v52673497         1.2  100.0     NaN
2     166        units           0  v52673498         1.3  100.0     NaN
3     166        units           0  v52673499         1.4  100.0     NaN
4     166        units           0  v52673500         1.5  100.0     NaN
5     166        units           0  v52673501         1.6  100.0     NaN
6     166        units           0  v52673502         1.7  100.0     NaN
7     166        units           0  v52673496         1.1  100.4     NaN
8     166        units           0  v52673497         1.2  100.5     NaN
9     166        units           0  v52673498         1.3   99.8     NaN

   SYMBOL  TERMINATED  DECIMALS
0     NaN         NaN         1
1     NaN         NaN         1
```

```
2     NaN          NaN          1
3     NaN          NaN          1
4     NaN          NaN          1
5     NaN          NaN          1
6     NaN          NaN          1
7     NaN          NaN          1
8     NaN          NaN          1
9     NaN          NaN          1
```

[9]:
```
# Filtering data for relevant variables and placing them in different columns
filtered_price_indices = pd.DataFrame()
filtered_price_indices['Date'] = price_indices.loc[price_indices['Commodity']␣
 ↪== 'Total, all commodities', 'REF_DATE'].values
filtered_price_indices['Total Index'] = price_indices.
 ↪loc[price_indices['Commodity']=='Total, all commodities','VALUE'].values
filtered_price_indices['Tot. Index (Ex. Energy)'] = price_indices.
 ↪loc[price_indices['Commodity']=='Total excluding energy','VALUE'].values
filtered_price_indices['Energy Index'] = price_indices.
 ↪loc[price_indices['Commodity']=='Energy','VALUE'].values
filtered_price_indices['Metals & Minerals Index'] = price_indices.
 ↪loc[price_indices['Commodity']=='Metals and Minerals','VALUE'].values
filtered_price_indices['Agriculture Index'] = price_indices.
 ↪loc[price_indices['Commodity']=='Agriculture','VALUE'].values
filtered_price_indices['Fish Index'] = price_indices.
 ↪loc[price_indices['Commodity']=='Fish','VALUE'].values
filtered_price_indices['Forestry Index'] = price_indices.
 ↪loc[price_indices['Commodity']=='Forestry','VALUE'].values

# Converting the date to a date type
filtered_price_indices['Date'] = pd.to_datetime(filtered_price_indices['Date'],␣
 ↪format='%Y-%m')

filtered_price_indices.head(10)
```

[9]:

	Date	Total Index	Tot. Index (Ex. Energy)	Energy Index	\
0	1972-01-01	100.0	100.0	100.0	
1	1972-02-01	100.4	100.5	99.8	
2	1972-03-01	101.1	101.3	100.1	
3	1972-04-01	101.2	101.5	99.8	
4	1972-05-01	101.9	102.3	100.0	
5	1972-06-01	102.1	102.5	100.2	
6	1972-07-01	103.7	104.5	100.3	
7	1972-08-01	104.8	105.6	101.2	
8	1972-09-01	105.8	106.8	101.2	
9	1972-10-01	106.5	107.7	101.2	

```
   Metals & Minerals Index  Agriculture Index  Fish Index  Forestry Index
0                    100.0              100.0       100.0           100.0
1                    100.7              101.2        88.9           100.1
2                    101.4              102.5        99.0           100.2
3                    101.2              102.1       103.1           100.9
4                    101.3              103.5        86.3           102.3
5                    100.9              103.6        90.7           102.9
6                    100.6              106.9        97.4           105.0
7                    101.2              109.4       101.0           105.1
8                    102.9              111.3       105.9           105.3
9                    102.7              112.1       107.2           106.9
```

[17]:
```
# Merging both data sets into one
merged_data = pd.merge(filtered_ex_rate, filtered_price_indices, on='Date',␣
 ↪how='outer')

# Dropping missing observations
merged_data.dropna(inplace=True)

# Making the date an index
merged_data['Date'] = pd.to_datetime(merged_data['Date'], format='%Y-%m')
merged_data.set_index('Date', inplace=True)

merged_data
```

[17]:
```
              US/CAD  Total Index  Tot. Index (Ex. Energy)  Energy Index  \
Date
1972-01-01  1.005922        100.0                    100.0         100.0
1972-02-01  1.004583        100.4                    100.5          99.8
1972-03-01  0.998395        101.1                    101.3         100.1
1972-04-01  0.995594        101.2                    101.5          99.8
1972-05-01  0.988665        101.9                    102.3         100.0
…                …            …                        …             …
2016-12-01  1.332935        388.8                    304.7         919.9
2017-01-01  1.319090        398.4                    310.0         953.6
2017-02-01  1.310989        409.9                    328.2         953.8
2017-03-01  1.338752        393.5                    321.2         898.4
2017-04-01  1.344395        410.0                    326.4         959.9

            Metals & Minerals Index  Agriculture Index  Fish Index  \
Date
1972-01-01                    100.0              100.0       100.0
1972-02-01                    100.7              101.2        88.9
1972-03-01                    101.4              102.5        99.0
1972-04-01                    101.2              102.1       103.1
1972-05-01                    101.3              103.5        86.3
…                               …                  …           …
```

```
2016-12-01                  494.9              207.4      1239.7
2017-01-01                  500.9              212.4      1329.9
2017-02-01                  540.1              217.7      1361.2
2017-03-01                  516.3              213.7      1413.0
2017-04-01                  524.1              213.7      1424.7

            Forestry Index
Date
1972-01-01           100.0
1972-02-01           100.1
1972-03-01           100.2
1972-04-01           100.9
1972-05-01           102.3
…                    …
2016-12-01           357.1
2017-01-01           360.5
2017-02-01           389.7
2017-03-01           393.5
2017-04-01           411.0

[544 rows x 8 columns]
```

[18]:
```
# Creating variables in first differences
columns = merged_data.columns.to_list()

for i in columns:
  merged_data[f'\u0394 {i}'] = merged_data[i].diff()

merged_data.head()
```

[18]:
```
              US/CAD  Total Index  Tot. Index (Ex. Energy)  Energy Index  \
Date
1972-01-01  1.005922        100.0                    100.0         100.0
1972-02-01  1.004583        100.4                    100.5          99.8
1972-03-01  0.998395        101.1                    101.3         100.1
1972-04-01  0.995594        101.2                    101.5          99.8
1972-05-01  0.988665        101.9                    102.3         100.0

            Metals & Minerals Index  Agriculture Index  Fish Index  \
Date
1972-01-01                    100.0              100.0       100.0
1972-02-01                    100.7              101.2        88.9
1972-03-01                    101.4              102.5        99.0
1972-04-01                    101.2              102.1       103.1
1972-05-01                    101.3              103.5        86.3

            Forestry Index  Δ US/CAD  Δ Total Index  \
```

```
Date
1972-01-01           100.0       NaN            NaN
1972-02-01           100.1 -0.001339            0.4
1972-03-01           100.2 -0.006188            0.7
1972-04-01           100.9 -0.002801            0.1
1972-05-01           102.3 -0.006929            0.7

            Δ Tot. Index (Ex. Energy)  Δ Energy Index  \
Date
1972-01-01                        NaN             NaN
1972-02-01                        0.5            -0.2
1972-03-01                        0.8             0.3
1972-04-01                        0.2            -0.3
1972-05-01                        0.8             0.2

            Δ Metals & Minerals Index  Δ Agriculture Index  Δ Fish Index  \
Date
1972-01-01                        NaN                 NaN           NaN
1972-02-01                        0.7                 1.2         -11.1
1972-03-01                        0.7                 1.3          10.1
1972-04-01                       -0.2                -0.4           4.1
1972-05-01                        0.1                 1.4         -16.8

            Δ Forestry Index
Date
1972-01-01              NaN
1972-02-01              0.1
1972-03-01              0.1
1972-04-01              0.7
1972-05-01              1.4
```

[19]:
```
merged_data.tail()
```

[19]:
```
              US/CAD  Total Index  Tot. Index (Ex. Energy)  Energy Index  \
Date
2016-12-01  1.332935        388.8                    304.7         919.9
2017-01-01  1.319090        398.4                    310.0         953.6
2017-02-01  1.310989        409.9                    328.2         953.8
2017-03-01  1.338752        393.5                    321.2         898.4
2017-04-01  1.344395        410.0                    326.4         959.9

            Metals & Minerals Index  Agriculture Index  Fish Index  \
Date
2016-12-01                    494.9              207.4      1239.7
2017-01-01                    500.9              212.4      1329.9
2017-02-01                    540.1              217.7      1361.2
2017-03-01                    516.3              213.7      1413.0
```

```
2017-04-01                   524.1              213.7      1424.7

            Forestry Index  Δ US/CAD  Δ Total Index  \
Date
2016-12-01           357.1 -0.010865           27.1
2017-01-01           360.5 -0.013845            9.6
2017-02-01           389.7 -0.008101           11.5
2017-03-01           393.5  0.027763          -16.4
2017-04-01           411.0  0.005643           16.5

            Δ Tot. Index (Ex. Energy)  Δ Energy Index  \
Date
2016-12-01                        2.2           127.5
2017-01-01                        5.3            33.7
2017-02-01                       18.2             0.2
2017-03-01                       -7.0           -55.4
2017-04-01                        5.2            61.5

            Δ Metals & Minerals Index  Δ Agriculture Index  Δ Fish Index  \
Date
2016-12-01                      -12.1                  8.3          42.6
2017-01-01                        6.0                  5.0          90.2
2017-02-01                       39.2                  5.3          31.3
2017-03-01                      -23.8                 -4.0          51.8
2017-04-01                        7.8                  0.0          11.7

            Δ Forestry Index
Date
2016-12-01               2.6
2017-01-01               3.4
2017-02-01              29.2
2017-03-01               3.8
2017-04-01              17.5
```

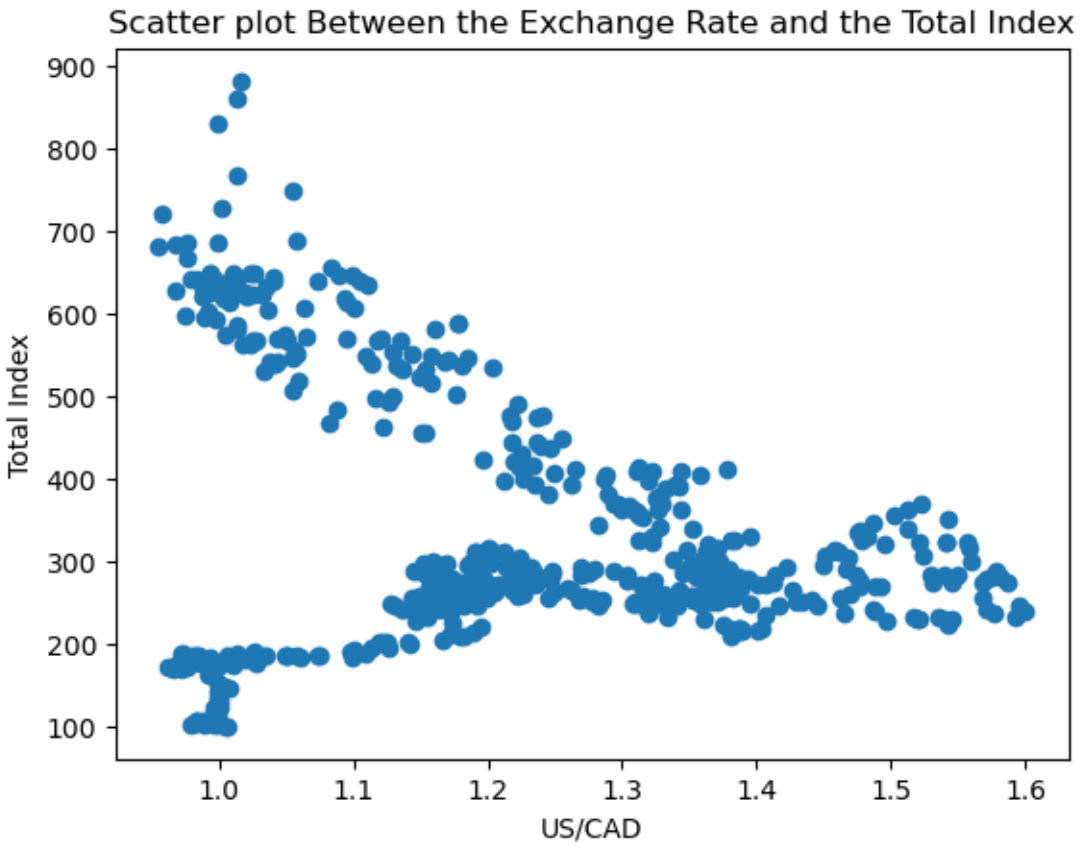
[20]:
```
# Summary statistics
merged_data.describe()
```

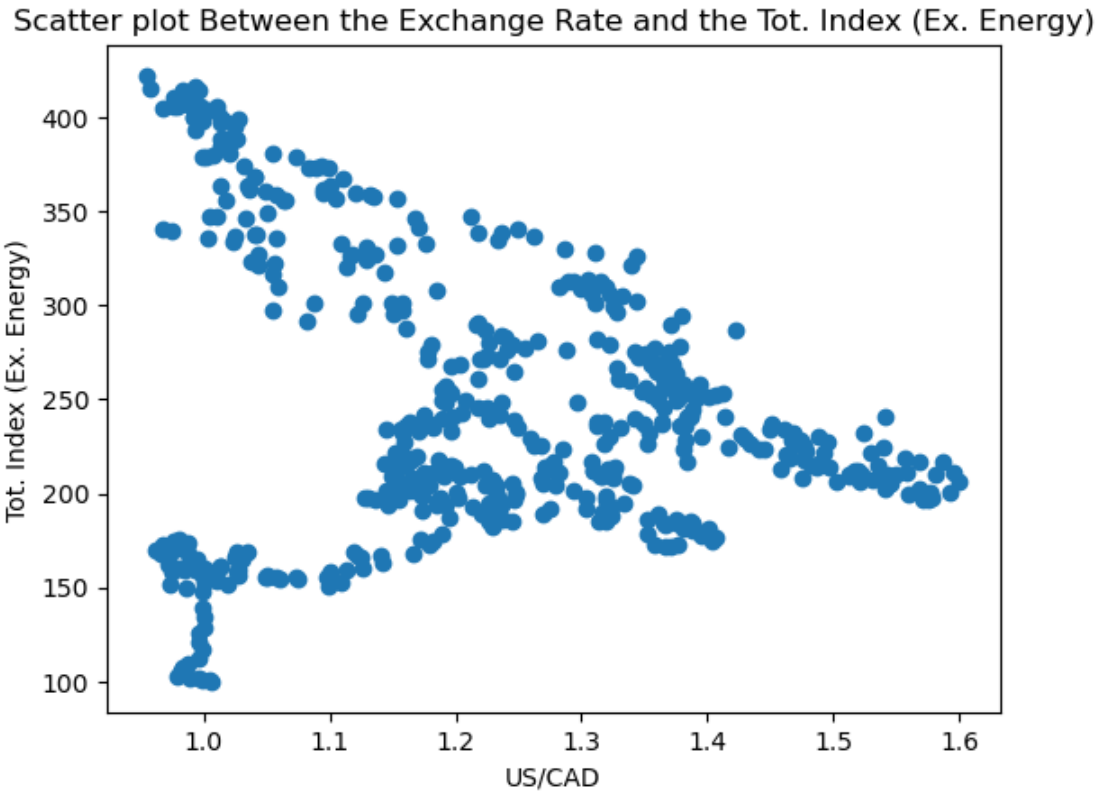
[20]:
```
           US/CAD  Total Index  Tot. Index (Ex. Energy)  Energy Index  \
count  544.000000   544.000000               544.000000     544.00000
mean     1.221514   333.059375               242.454779     739.98511
std      0.166675   150.056518                72.416773     492.43273
min      0.955300   100.000000               100.000000      99.80000
25%      1.074648   247.800000               195.025000     397.30000
50%      1.204636   281.550000               222.900000     570.30000
75%      1.352579   393.125000               282.550000     942.90000
max      1.600286   881.300000               422.100000    2755.80000
```

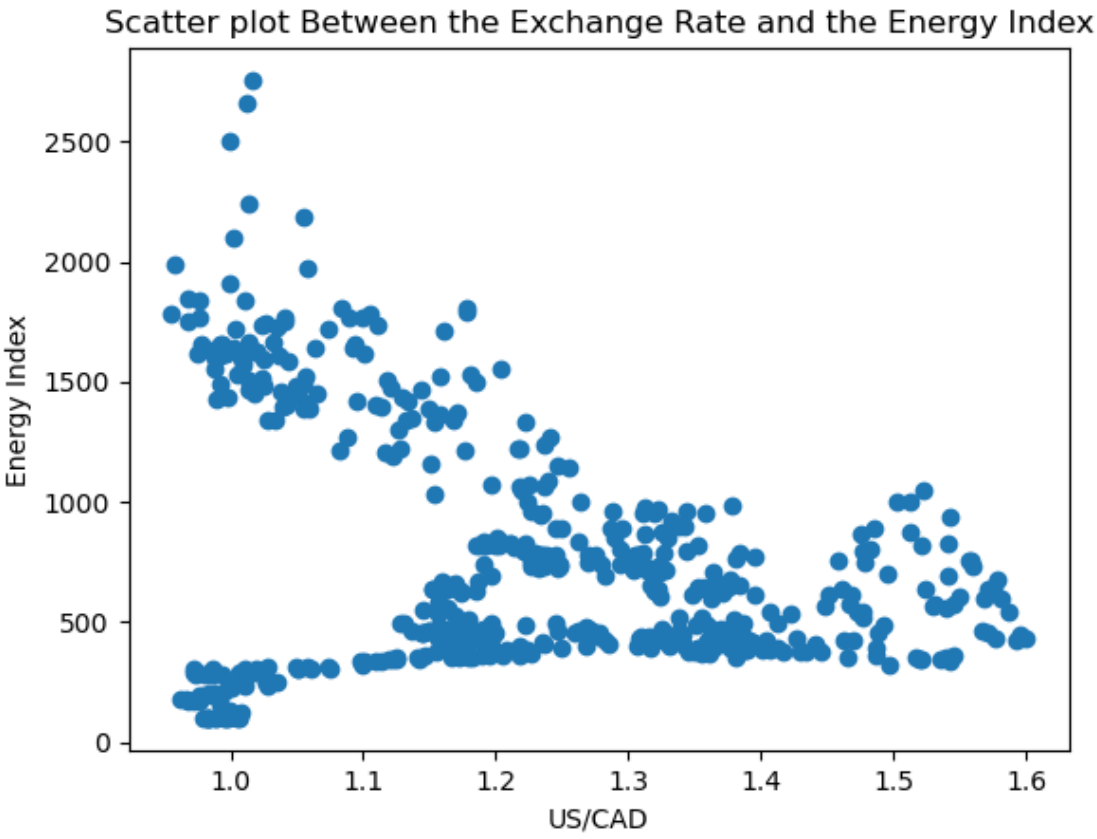
```
       Metals & Minerals Index  Agriculture Index   Fish Index  \
count               544.000000         544.000000   544.000000
mean                337.314890         187.216912   624.025368
std                 167.227726          45.992942   367.825837
min                 100.000000         100.000000    86.300000
25%                 229.850000         159.250000   255.525000
50%                 272.450000         176.000000   629.250000
75%                 422.650000         199.350000   923.375000
max                 779.900000         322.900000  1503.600000

       Forestry Index    Δ US/CAD  Δ Total Index  Δ Tot. Index (Ex. Energy)  \
count      544.000000  543.000000     543.000000                 543.000000
mean       260.884559    0.000623       0.570902                   0.416943
std         72.920674    0.017763      17.820562                   6.532706
min        100.000000   -0.073986    -141.300000                 -50.000000
25%        197.550000   -0.009104      -3.300000                  -2.800000
50%        269.450000    0.000390       0.700000                   0.400000
75%        320.750000    0.010876       5.950000                   3.700000
max        411.000000    0.126455      62.900000                  31.400000

       Δ Energy Index  Δ Metals & Minerals Index  Δ Agriculture Index  \
count      543.000000                 543.000000           543.000000
mean         1.583610                   0.781031             0.209392
std         72.124869                  15.303400             7.013472
min       -476.900000                 -93.600000           -37.700000
25%        -13.000000                  -4.900000            -3.400000
50%          1.400000                   0.600000             0.200000
75%         17.050000                   7.000000             3.700000
max        264.600000                  67.400000            54.500000

       Δ Fish Index  Δ Forestry Index
count    543.000000        543.000000
mean       2.439595          0.572744
std       68.660809         10.389353
min     -340.300000        -45.200000
25%      -18.500000         -4.150000
50%        4.100000          0.700000
75%       29.300000          5.450000
max      296.900000         48.700000
```

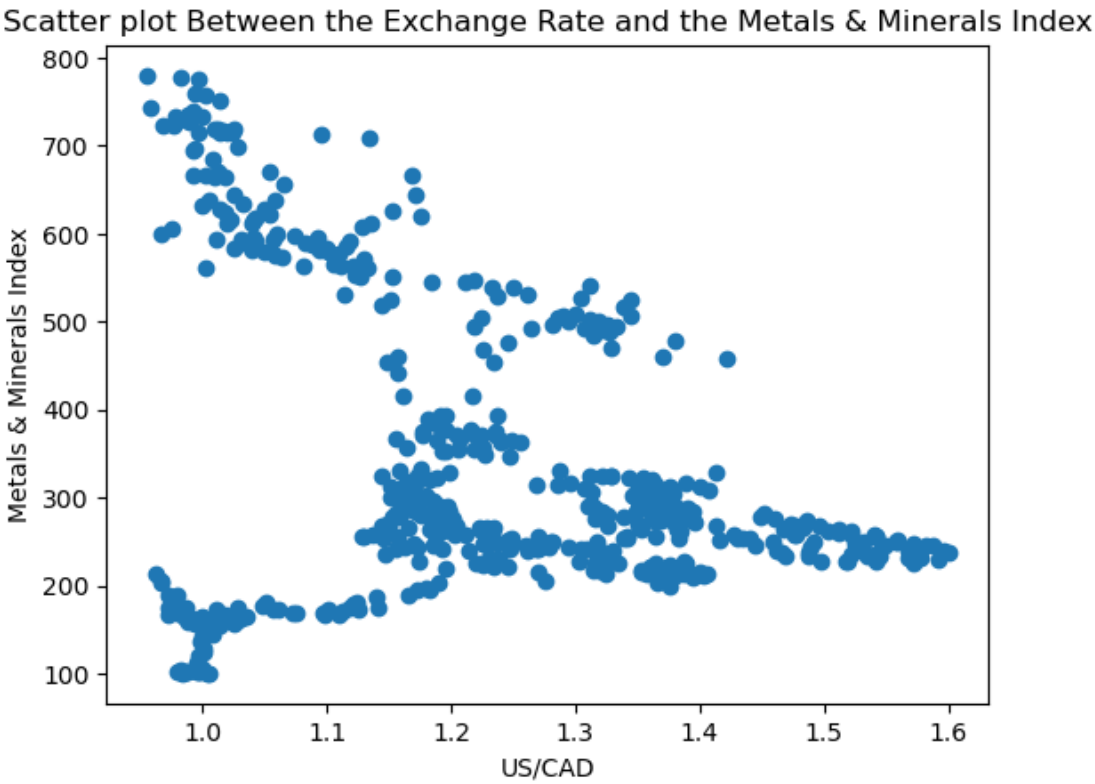
[21]:
```
levels = ['Total Index', 'Tot. Index (Ex. Energy)', 'Energy Index', 'Metals &␣
 ↪Minerals Index', 'Agriculture Index', 'Fish Index', 'Forestry Index']
for index in levels:
  plt.scatter(merged_data['US/CAD'], merged_data[index])
  plt.xlabel('US/CAD')
  plt.ylabel(index)
  plt.title(f'Scatter plot Between the Exchange Rate and the {index}')
```

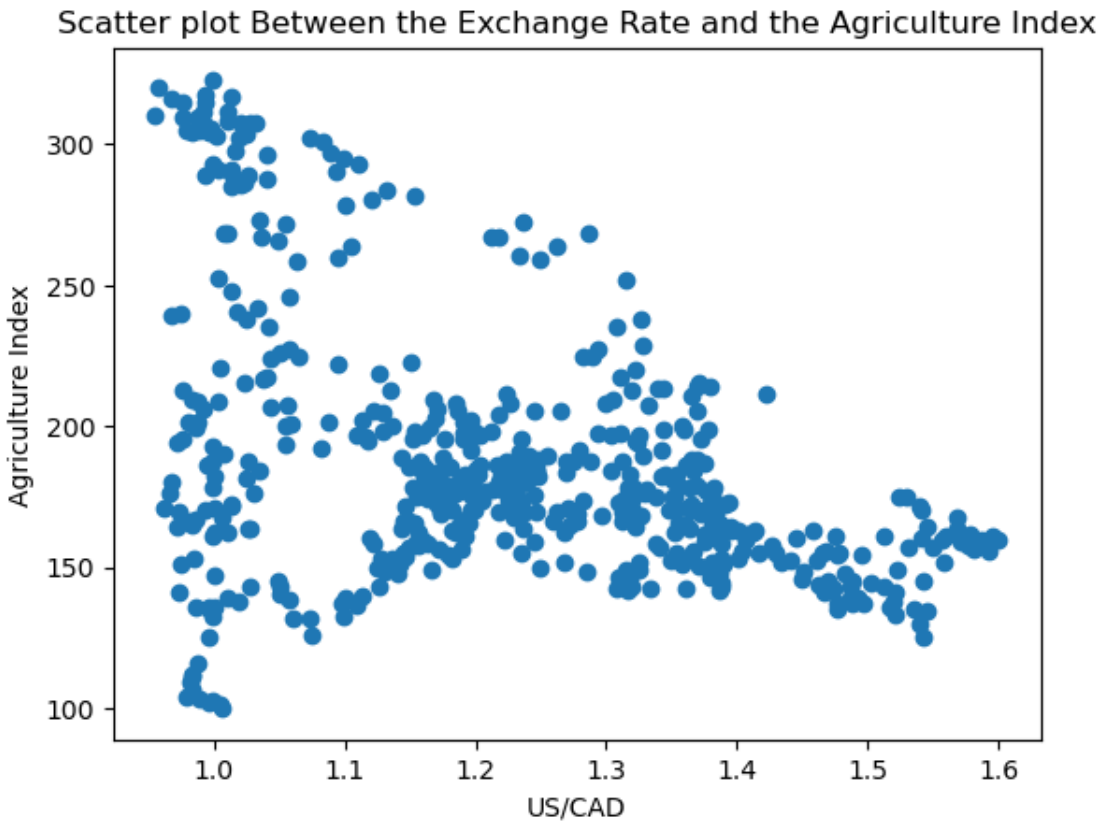
```
plt.show()
```

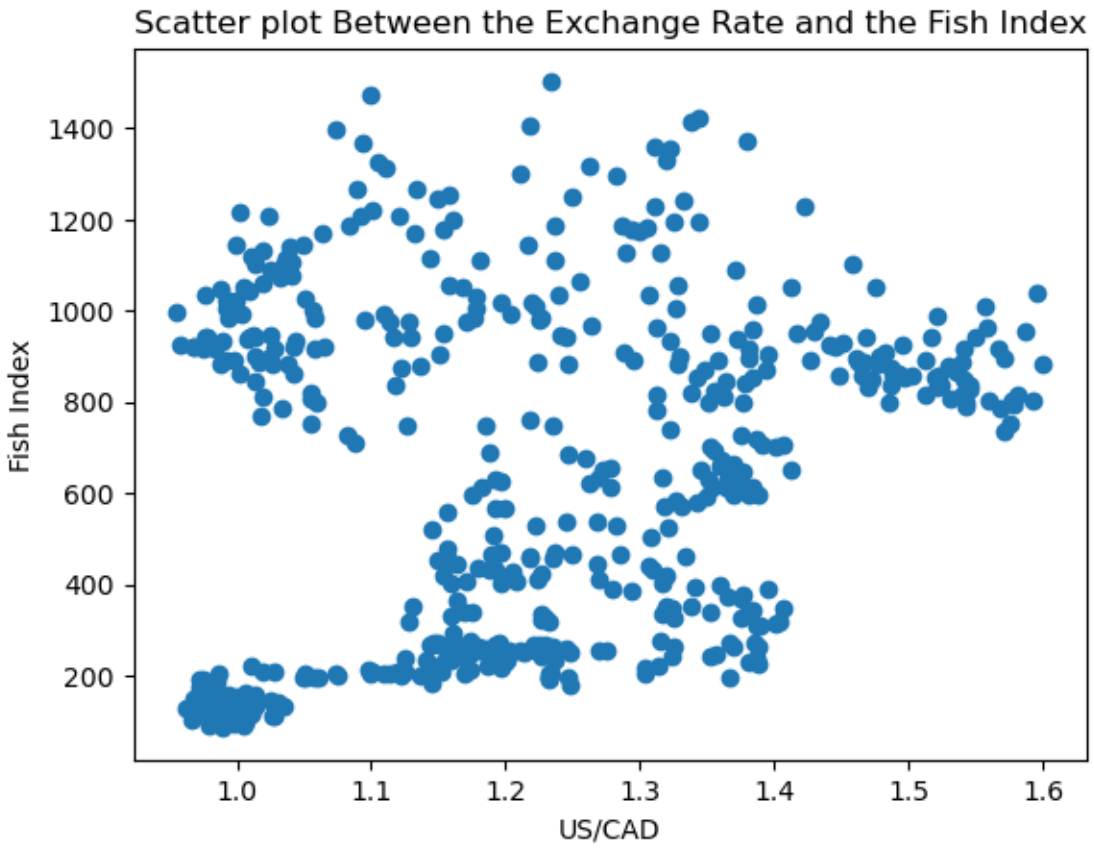

Scatter plot Between the Exchange Rate and the Total Index
Total Index
900
800
700
600
500
400
300
200
100
1.0
1.1
1.2
1.3
1.4
1.5
1.6
US/CAD

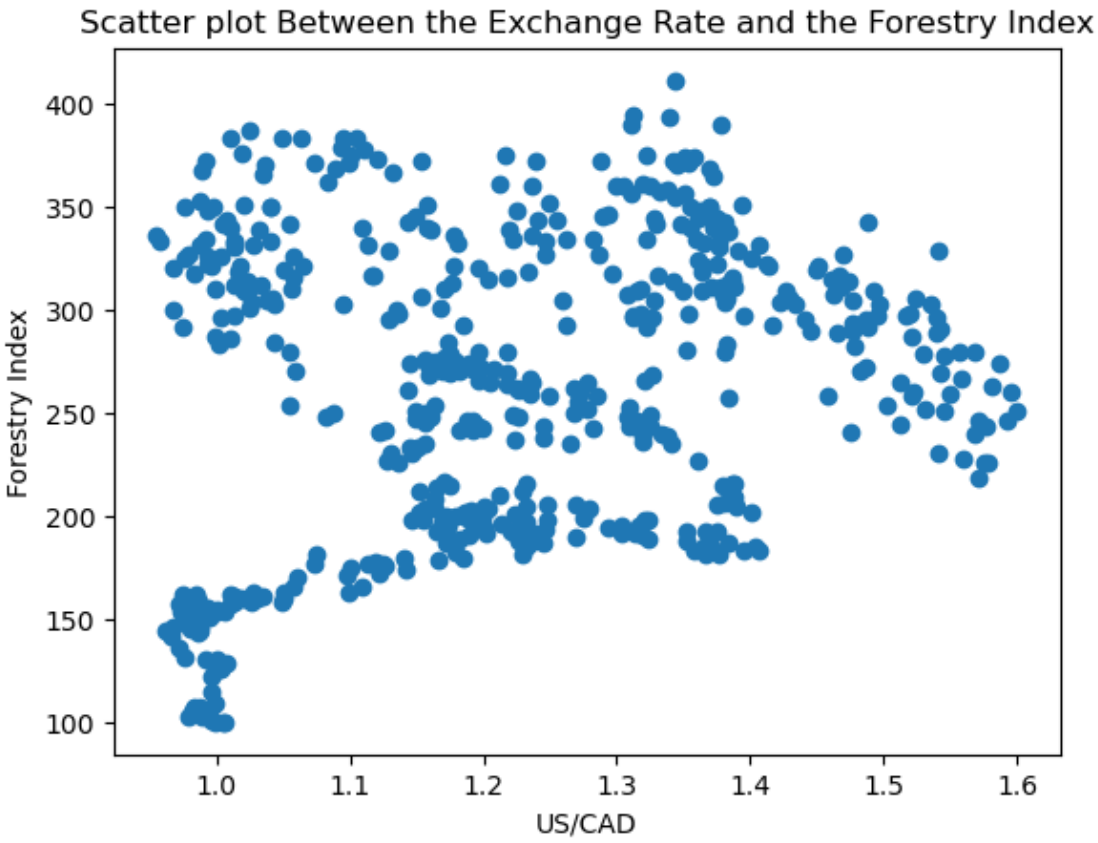

Scatter plot Between the Exchange Rate and the Tot. Index (Ex. Energy)
Tot. Index (Ex. Energy)
400
350
300
250
200
150
100
1.0
1.1
1.2
1.3
1.4
1.5
1.6
US/CAD


Scatter plot Between the Exchange Rate and the Energy Index
Energy Index
2500
2000
1500
1000
500
0
1.0
1.1
1.2
1.3
1.4
1.5
1.6
US/CAD

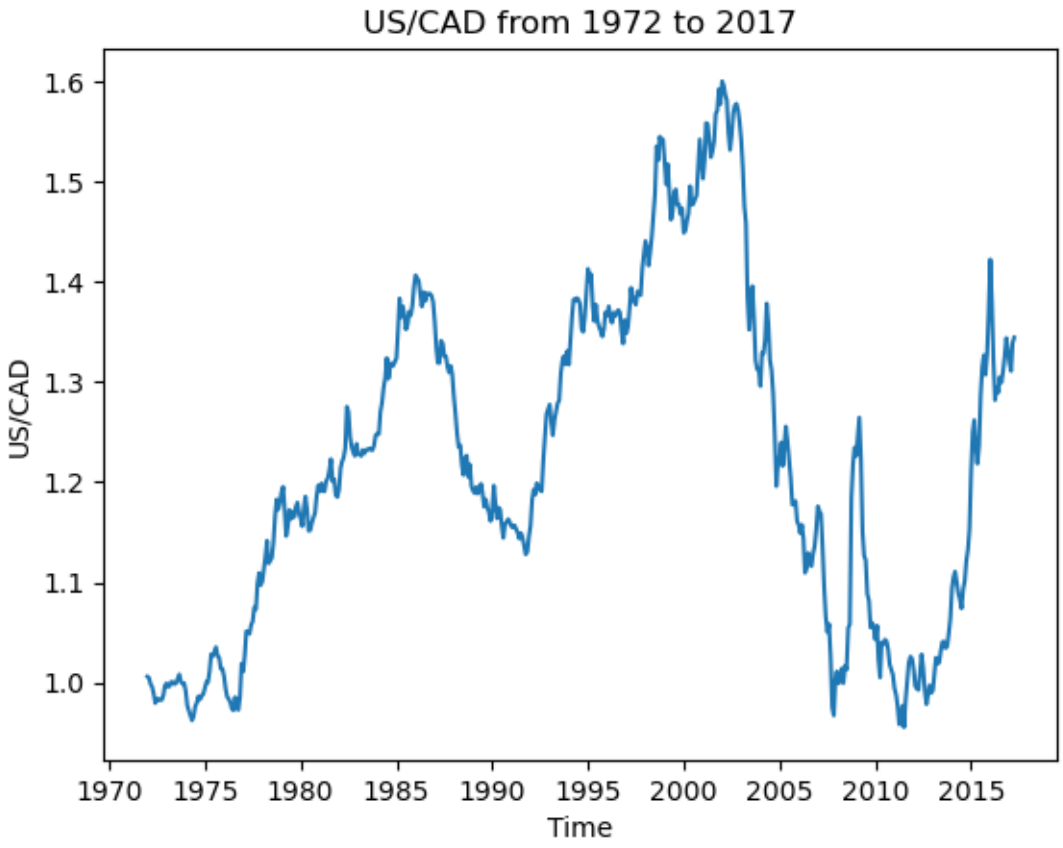

Scatter plot Between the Exchange Rate and the Metals & Minerals Index
Metals & Minerals Index
800
700
600
500
400
300
200
100
1.0
1.1
1.2
1.3
1.4
1.5
1.6
US/CAD

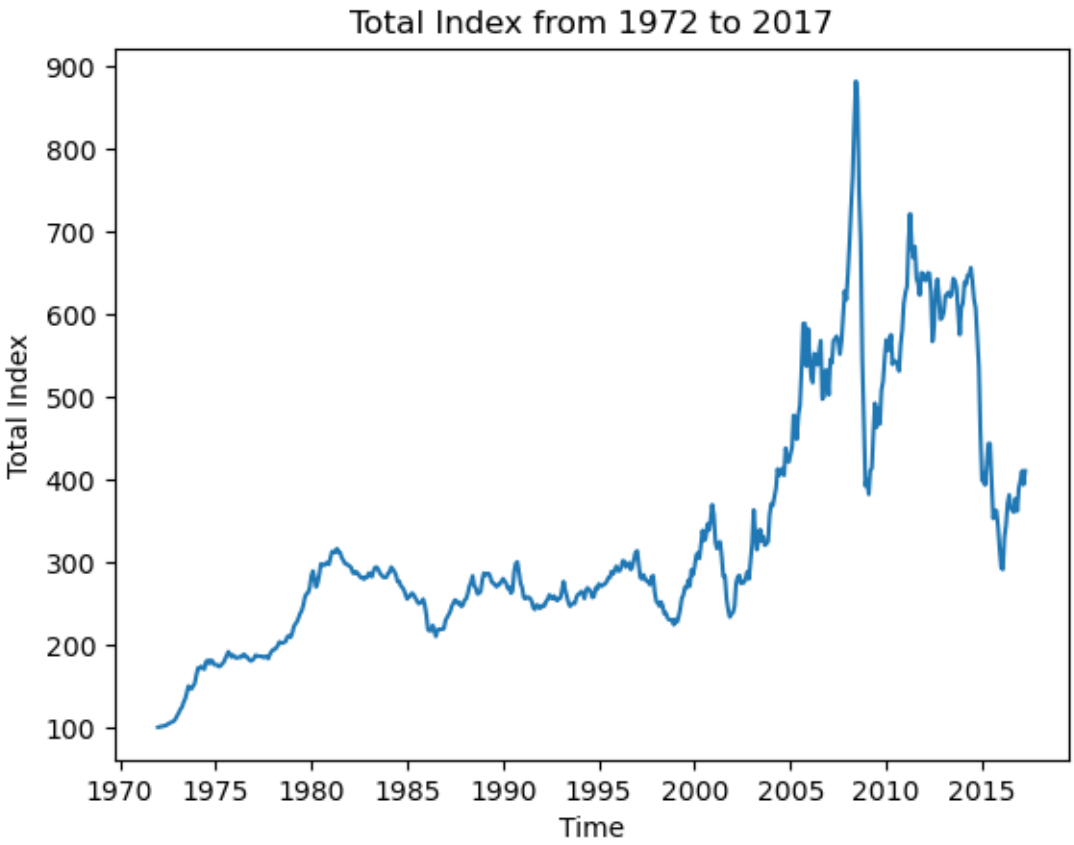

Scatter plot Between the Exchange Rate and the Agriculture Index
Agriculture Index
300
250
200
150
100
1.0
1.1
1.2
1.3
1.4
1.5
1.6
US/CAD

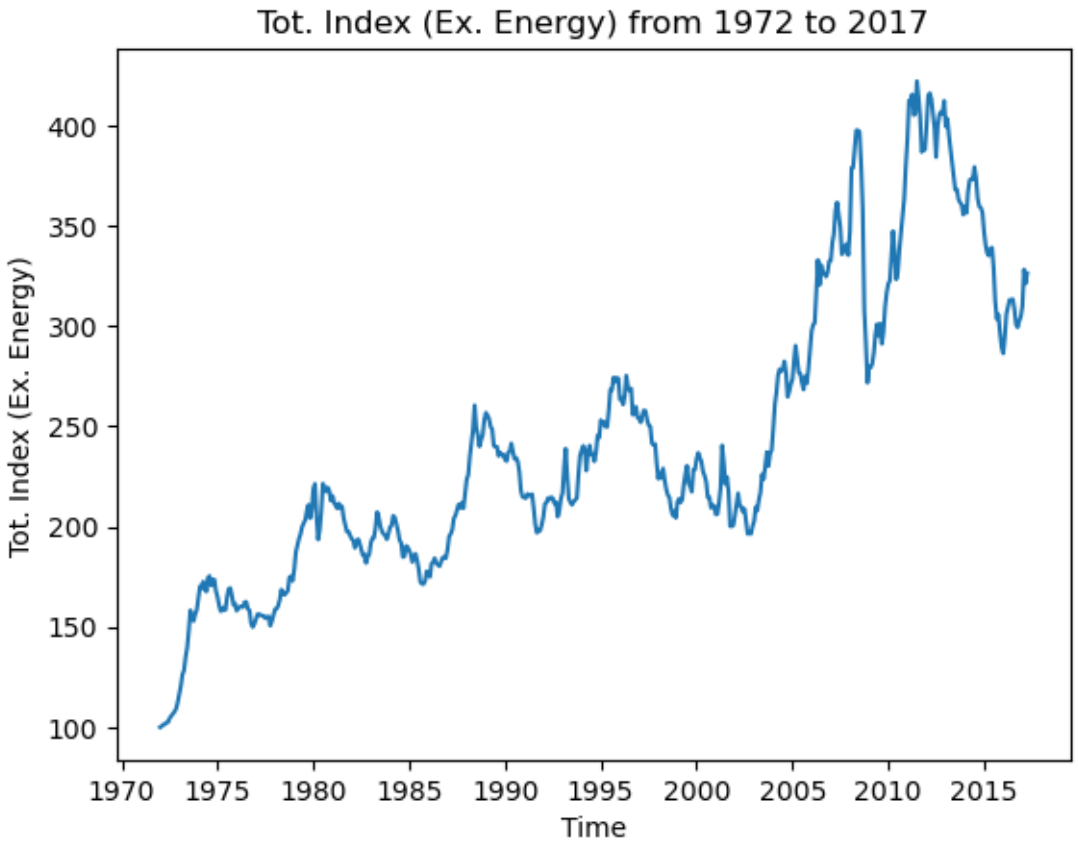

Scatter plot Between the Exchange Rate and the Fish Index
Fish Index
1400
1200
1000
800
600
400
200
1.0
1.1
1.2
1.3
1.4
1.5
1.6
US/CAD

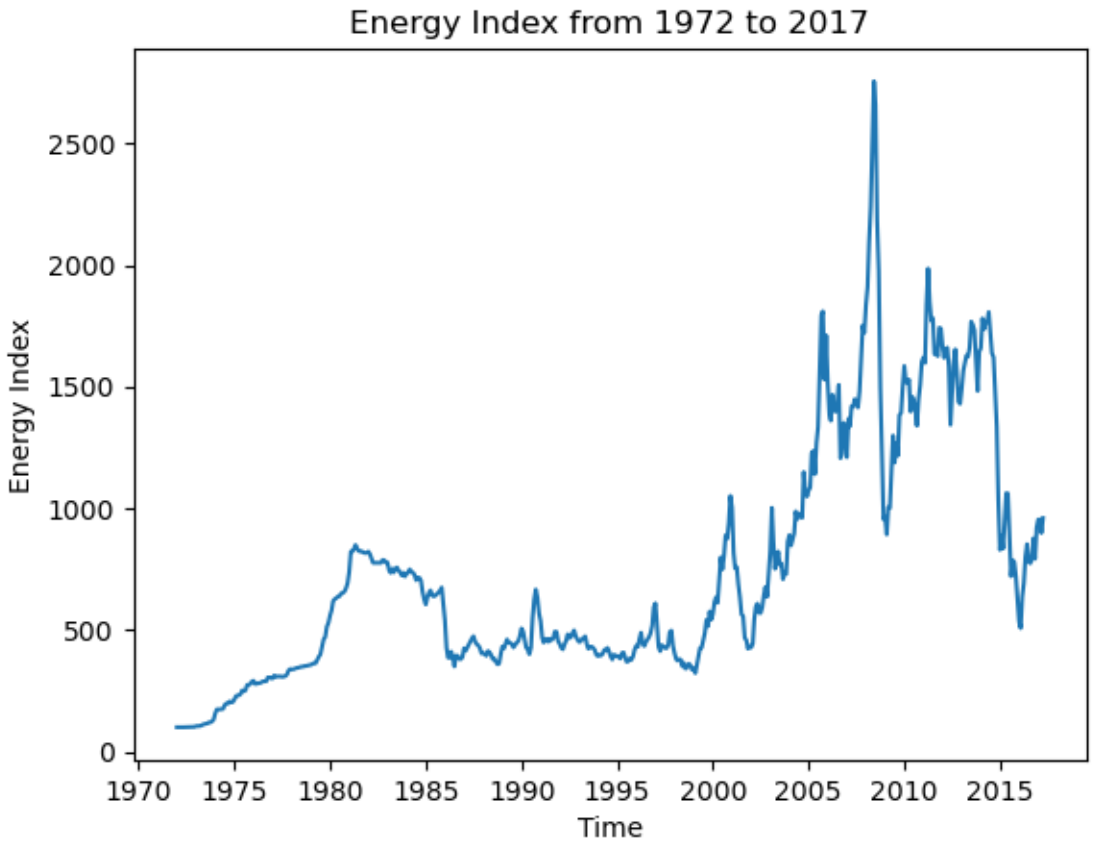


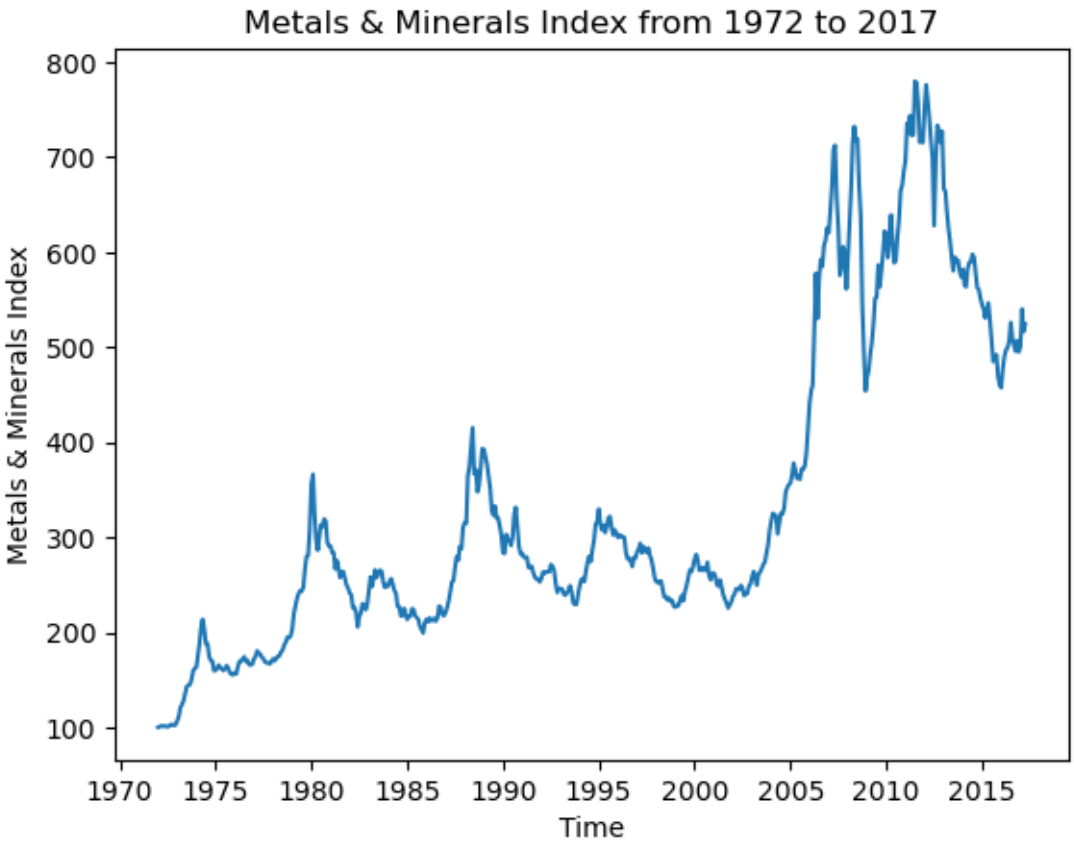
[22]:
```
# Creating line plots for all variables
for col in merged_data.columns:
  plt.plot(merged_data[col])
  plt.xlabel('Time')
  plt.ylabel(col)

  plt.title(f'{col} from 1972 to 2017')
  plt.show()
```

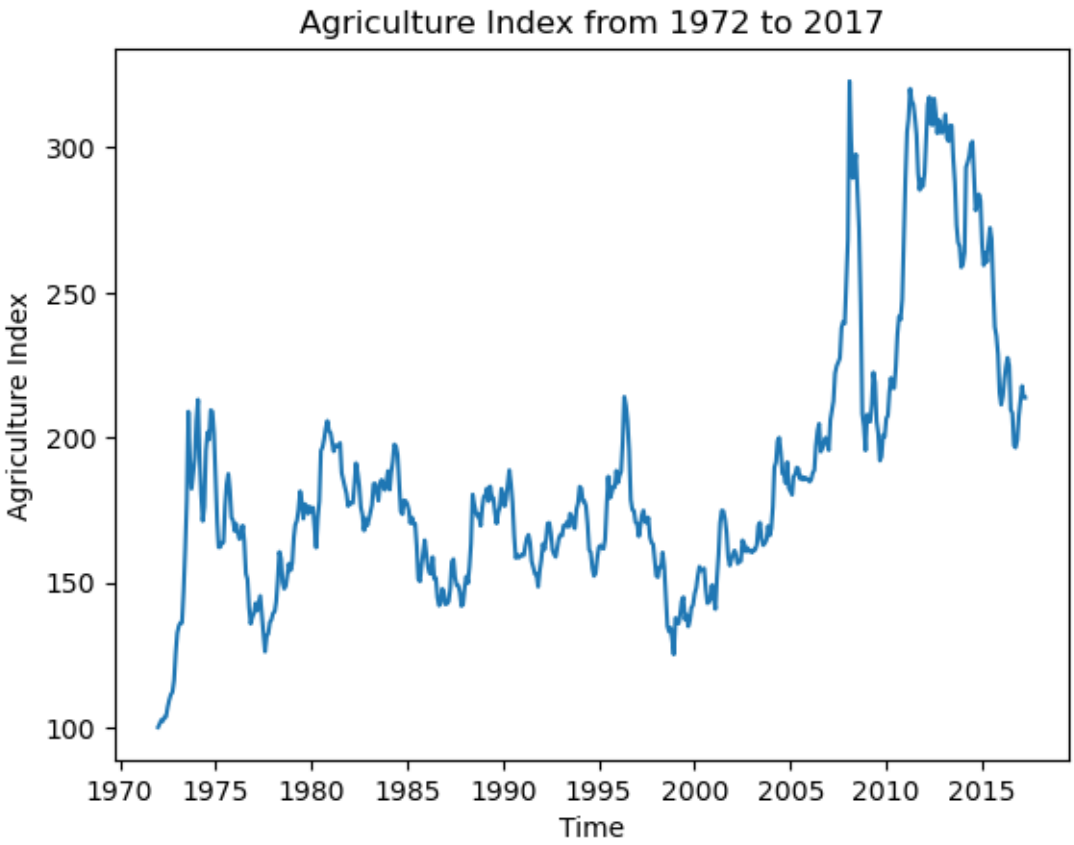

US/CAD from 1972 to 2017
US/CAD
1.6
1.5
1.4
1.3
1.2
1.1
1.0
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

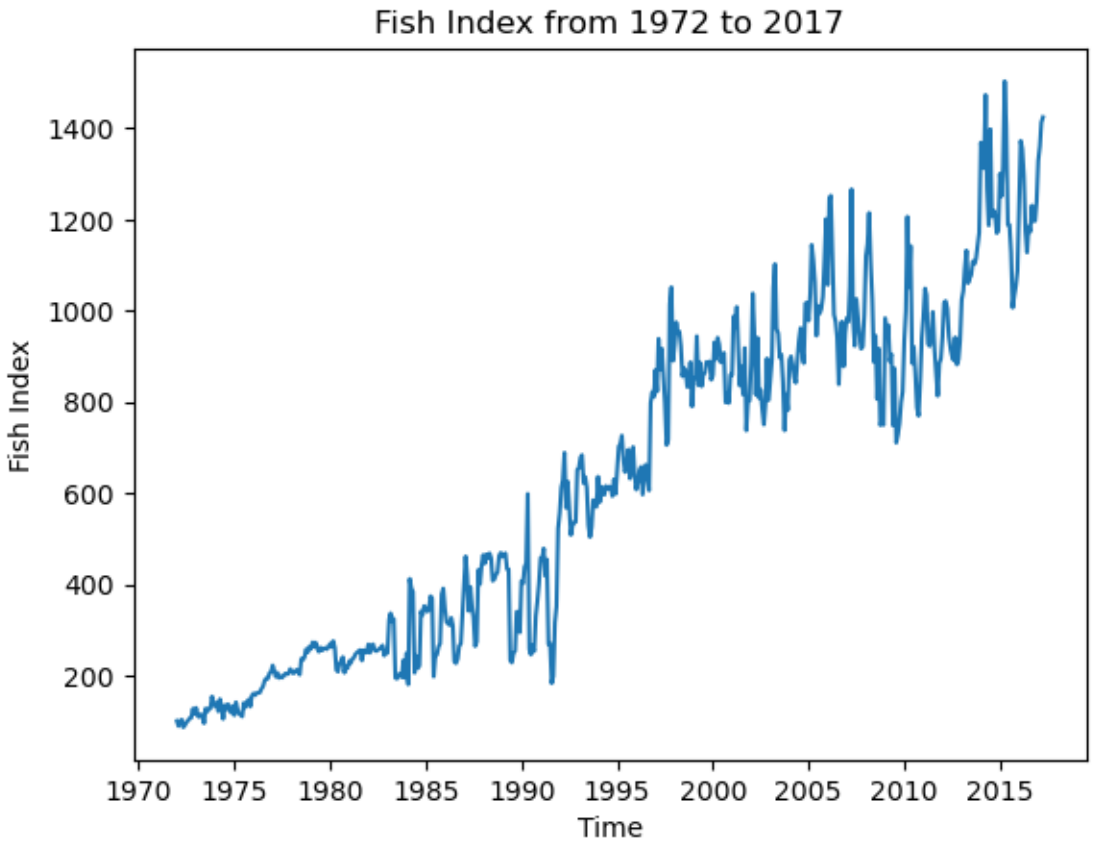

Total Index from 1972 to 2017
Total Index
900
800
700
600
500
400
300
200
100
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

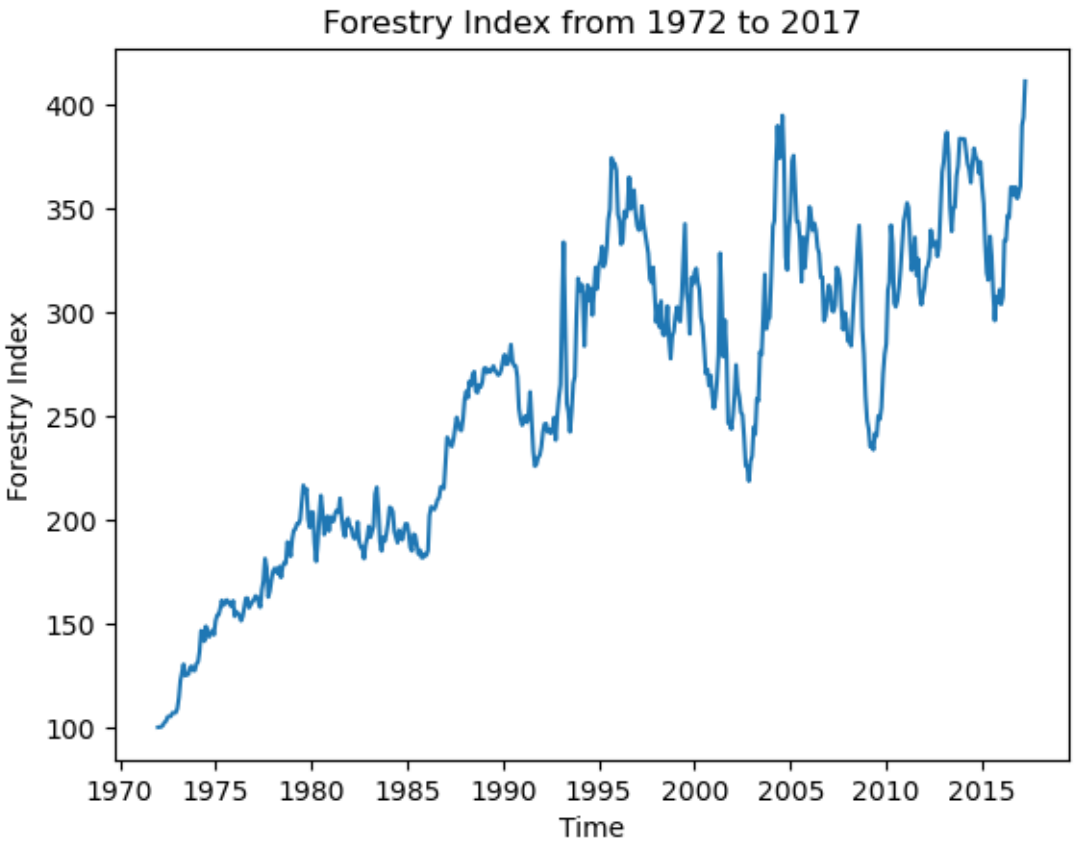

Tot. Index (Ex. Energy) from 1972 to 2017
Tot. Index (Ex. Energy)
100
150
200
250
300
350
400
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

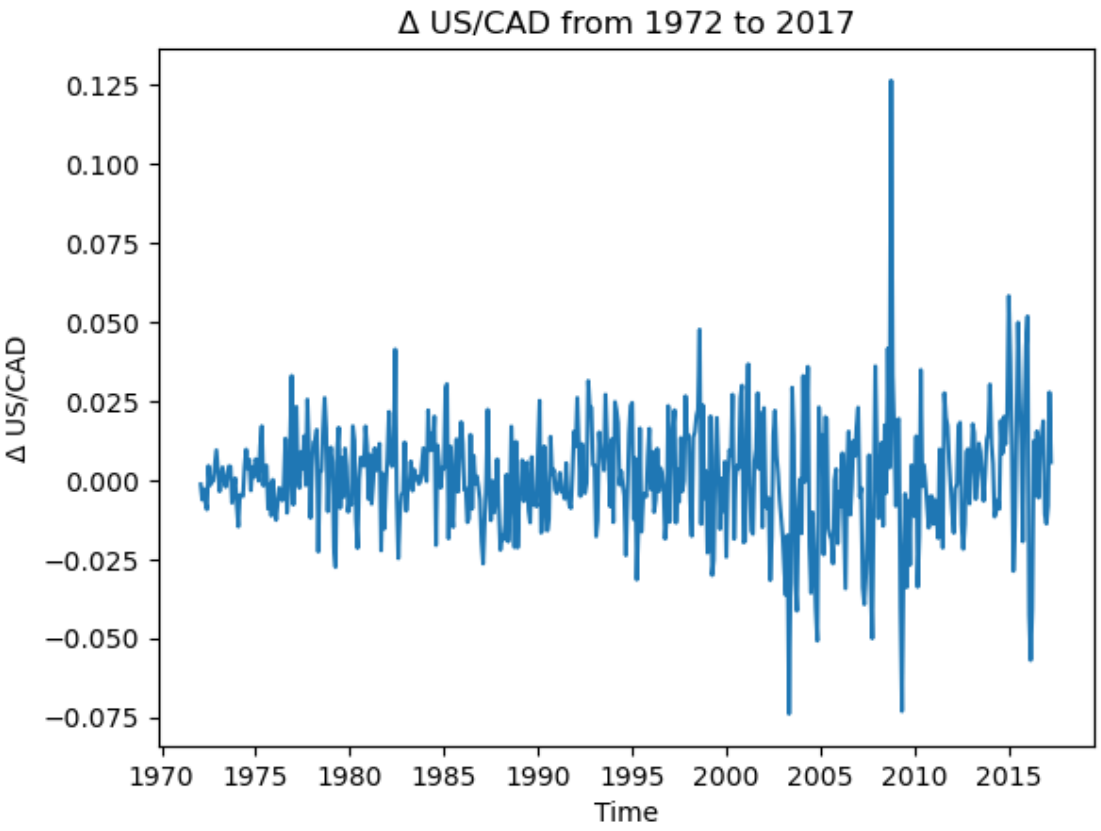

Energy Index from 1972 to 2017
Energy Index
2500
2000
1500
1000
500
0
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

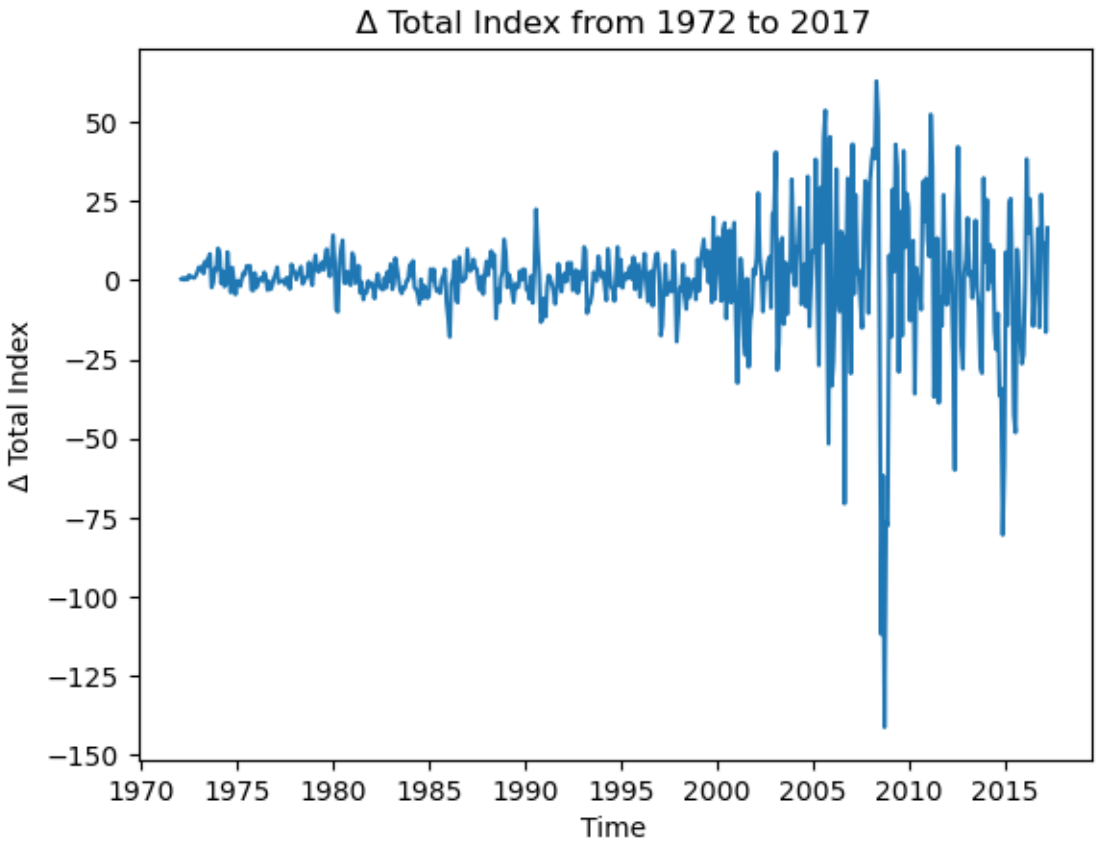

Metals & Minerals Index from 1972 to 2017
Metals & Minerals Index
800
700
600
500
400
300
200
100
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

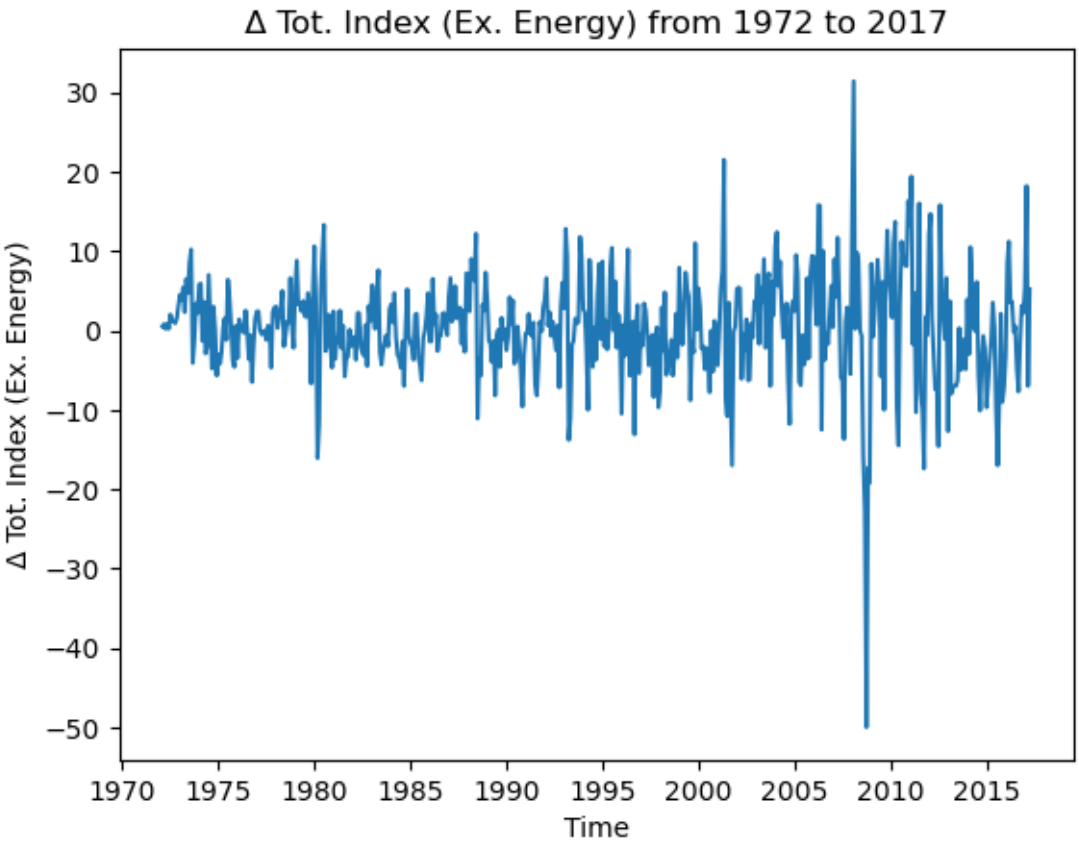

Agriculture Index from 1972 to 2017
Agriculture Index
300
250
200
150
100
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

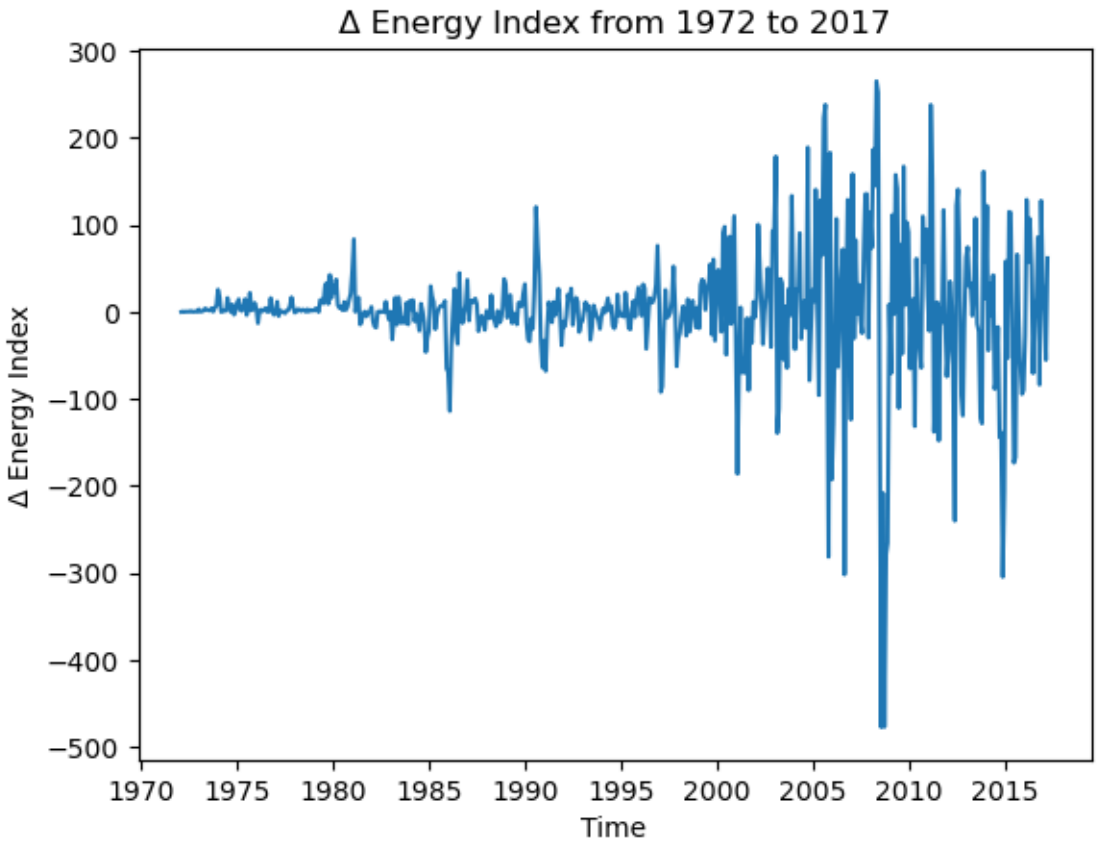

Fish Index from 1972 to 2017
Fish Index
1400
1200
1000
800
600
400
200
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

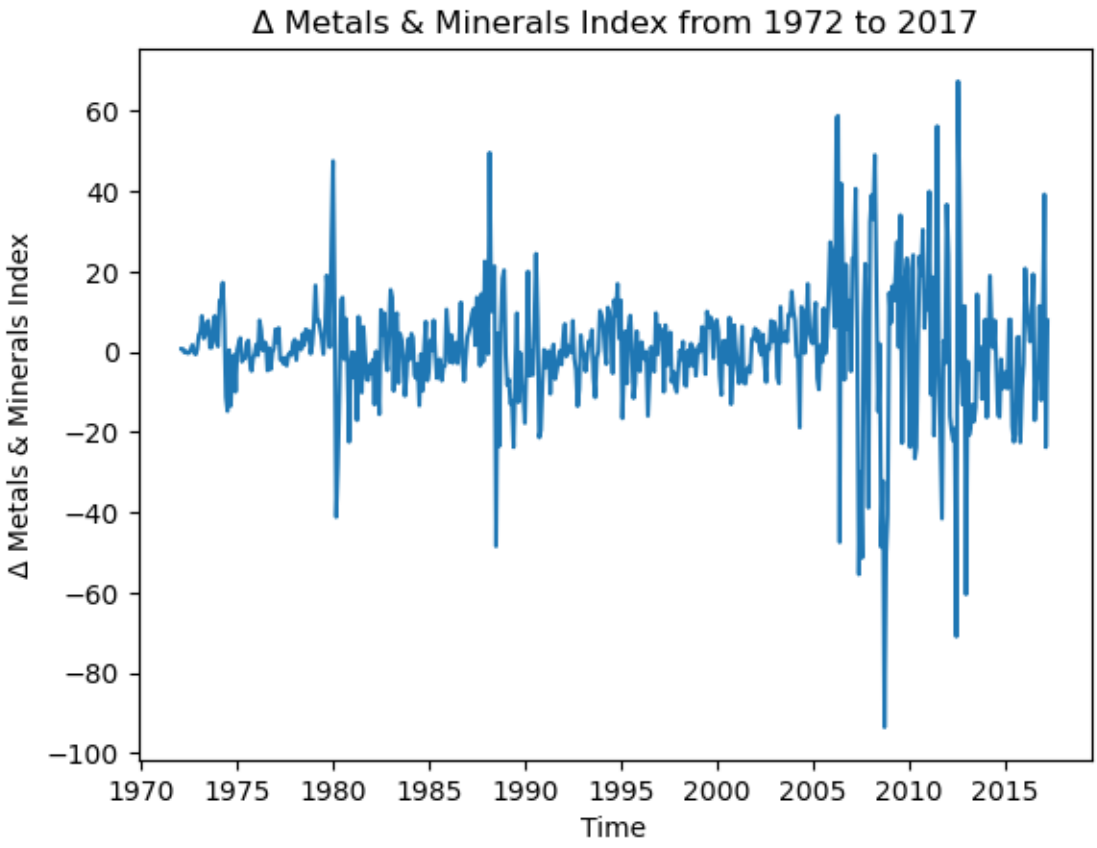

Forestry Index from 1972 to 2017
Forestry Index
400
350
300
250
200
150
100
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

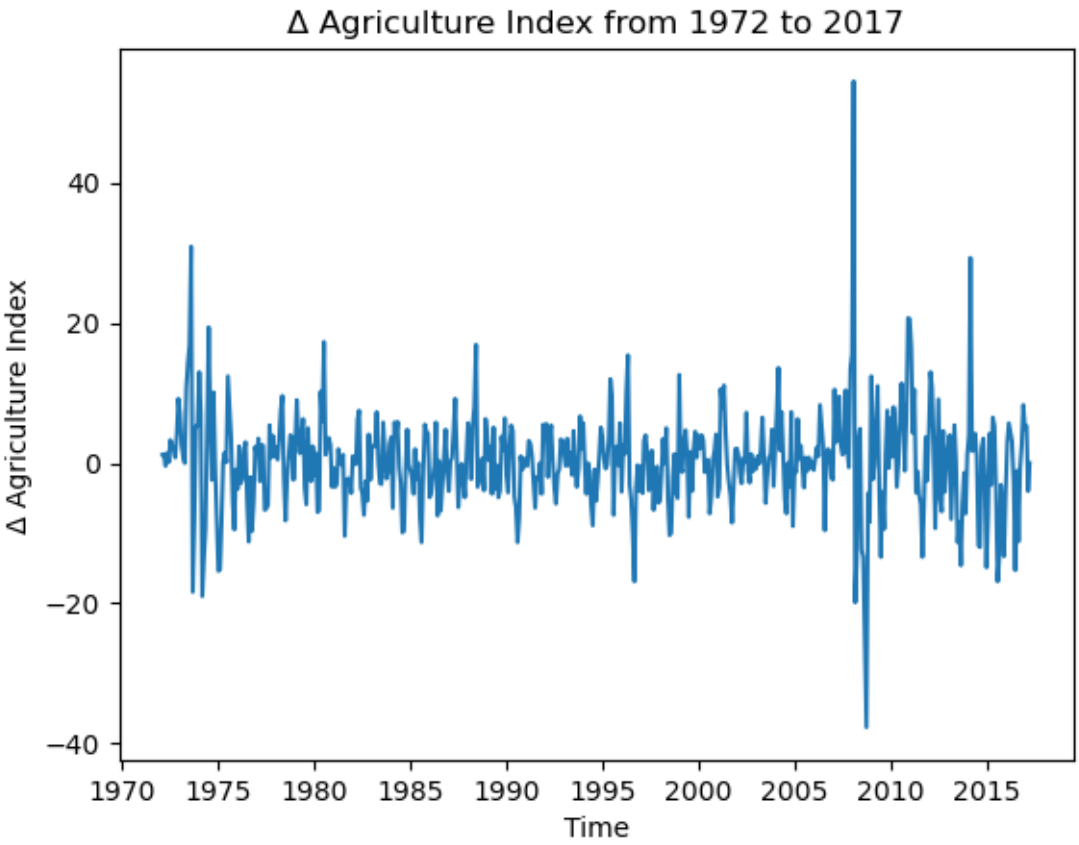

Δ US/CAD from 1972 to 2017
0.125
0.100
0.075
0.050
0.025
0.000
−0.025
−0.050
−0.075
Δ US/CAD
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

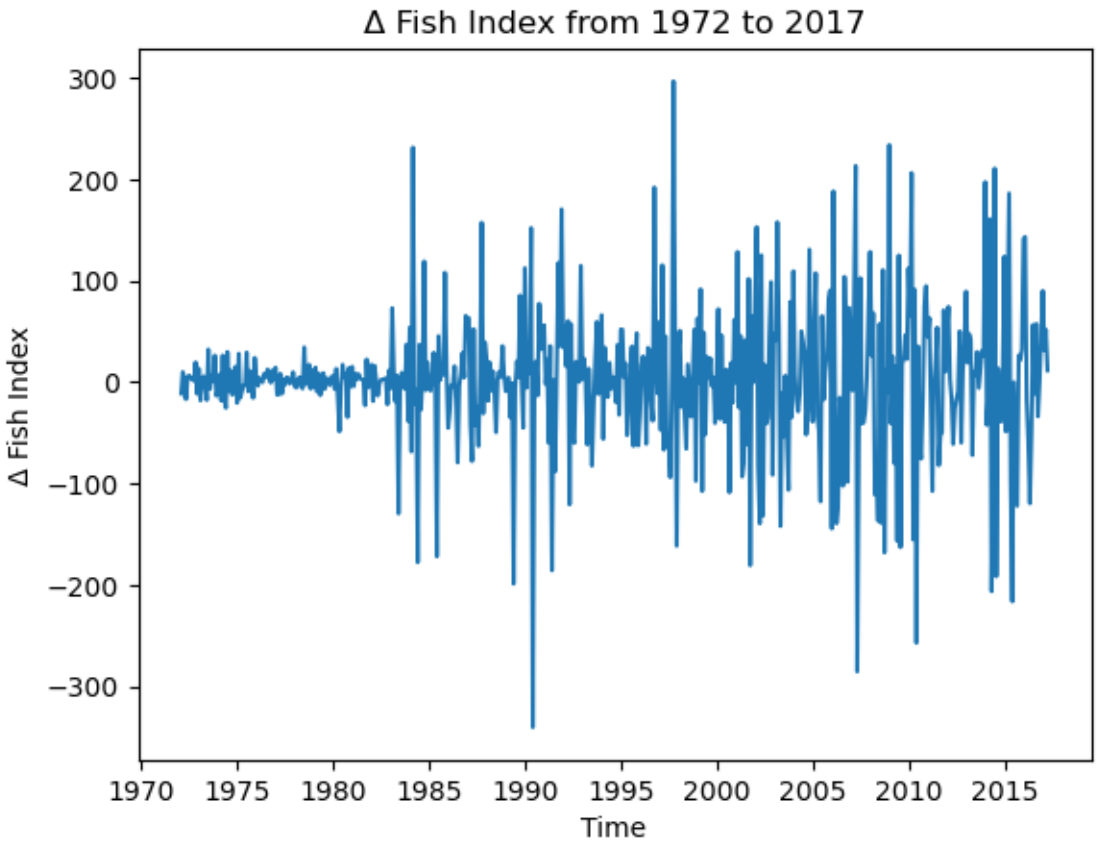

Δ Total Index from 1972 to 2017
Δ Total Index
50
25
0
−25
−50
−75
−100
−125
−150
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time

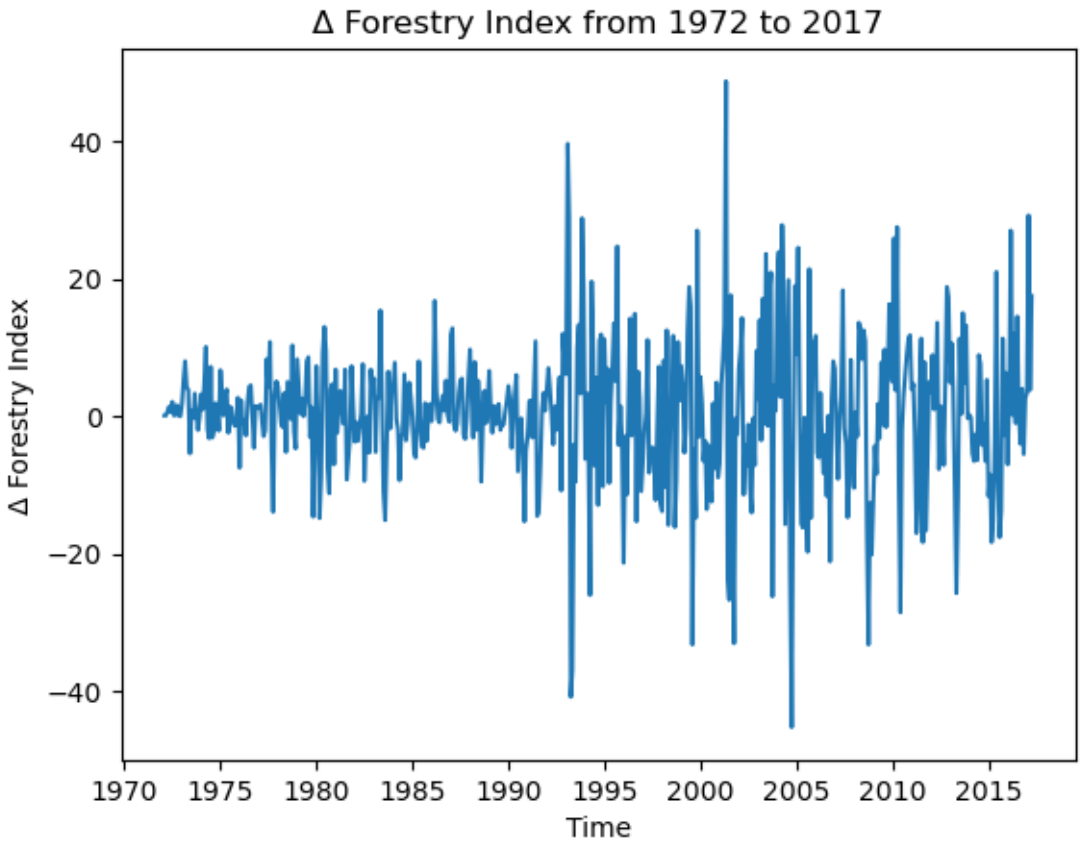

Δ Tot. Index (Ex. Energy) from 1972 to 2017
Δ Tot. Index (Ex. Energy)
Time
30
20
10
0
−10
−20
−30
−40
−50
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015


Δ Energy Index from 1972 to 2017
Δ Energy Index
Time
300
200
100
0
−100
−200
−300
−400
−500
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015


Δ Metals & Minerals Index from 1972 to 2017
Δ Metals & Minerals Index
60
40
20
0
−20
−40
−60
−80
−100
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time


Δ Agriculture Index from 1972 to 2017
Δ Agriculture Index
40
20
0
−20
−40
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time


Δ Fish Index from 1972 to 2017
Δ Fish Index
300
200
100
0
−100
−200
−300
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Time



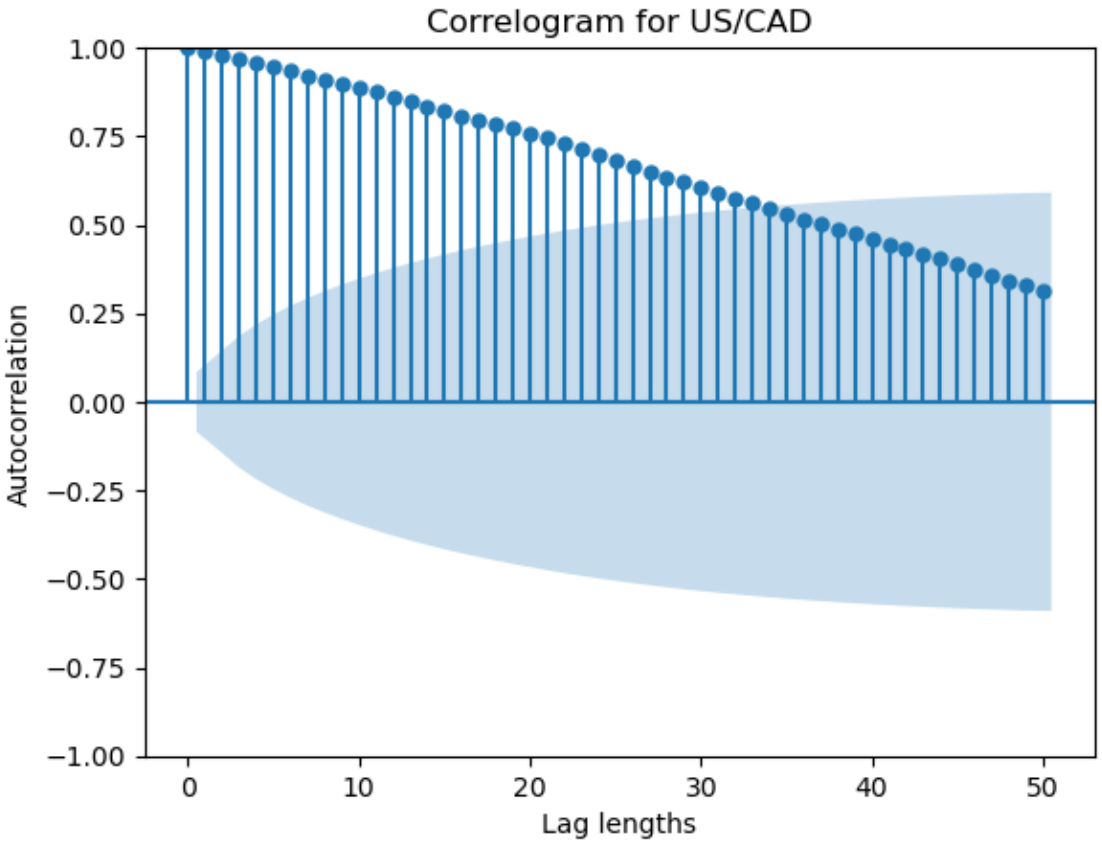
[23]:
```
levels = ['Total Index', 'Tot. Index (Ex. Energy)', 'Energy Index', 'Metals &␣
 ↪Minerals Index', 'Agriculture Index', 'Fish Index', 'Forestry Index']
first_differences = ['\u0394 Total Index', '\u0394 Tot. Index (Ex. Energy)',␣
 ↪'\u0394 Energy Index', '\u0394 Metals & Minerals Index', '\u0394 Agriculture␣
 ↪Index', '\u0394 Fish Index', '\u0394 Forestry Index']

def correl_table(i,j):
  correlations =[]
  p_values = []
  for variable in j:
    corr, p_val = pearsonr(merged_data[variable].dropna() ,merged_data[i].
 ↪dropna())
    correlations.append(corr)
    p_values.append(p_val)

  results = pd.DataFrame({'Correlations':correlations, 'P-values':p_values},␣
 ↪index = j)

  return results
```

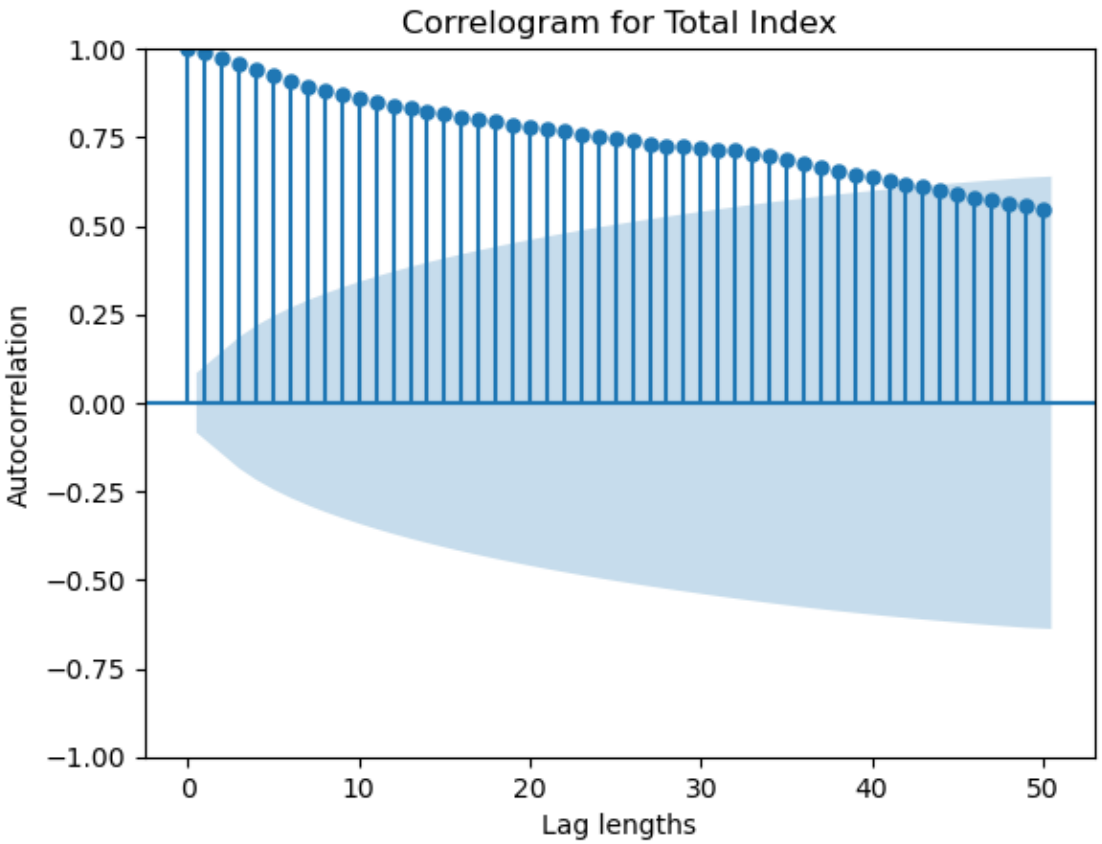
[24]:
```
correl_table('US/CAD', levels)
```

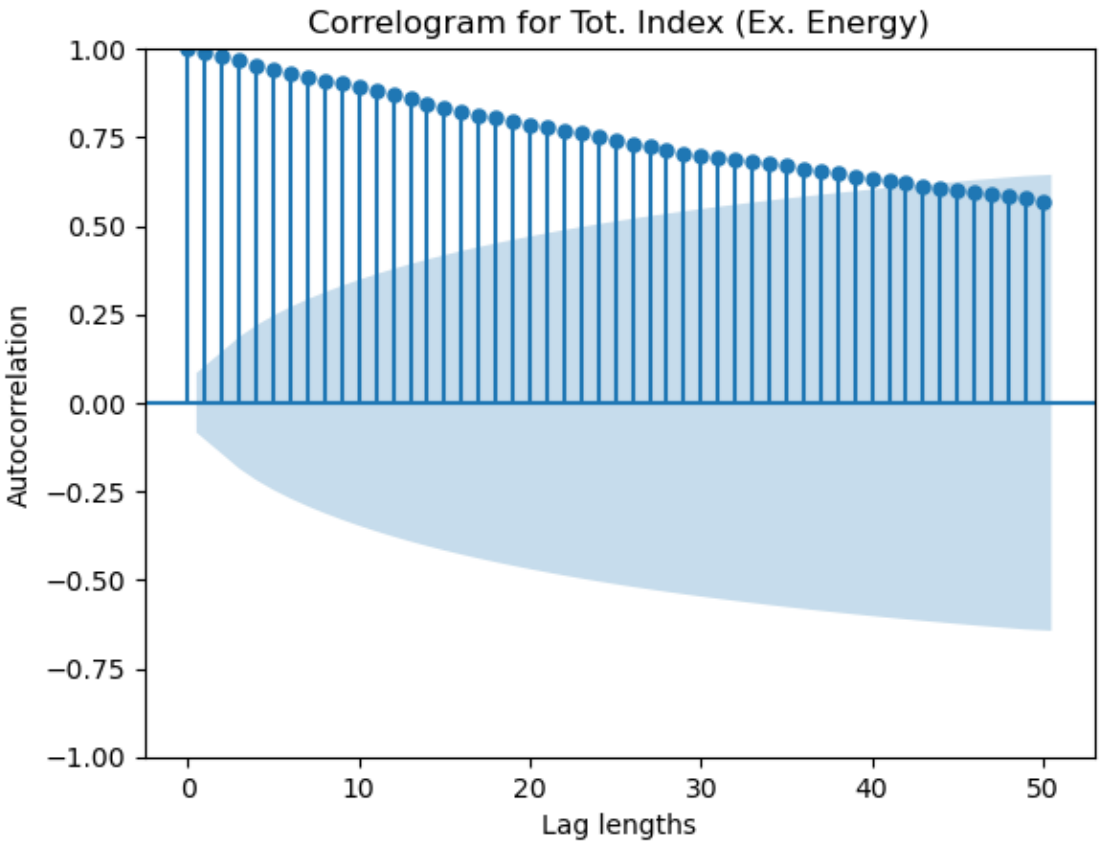
[24]:

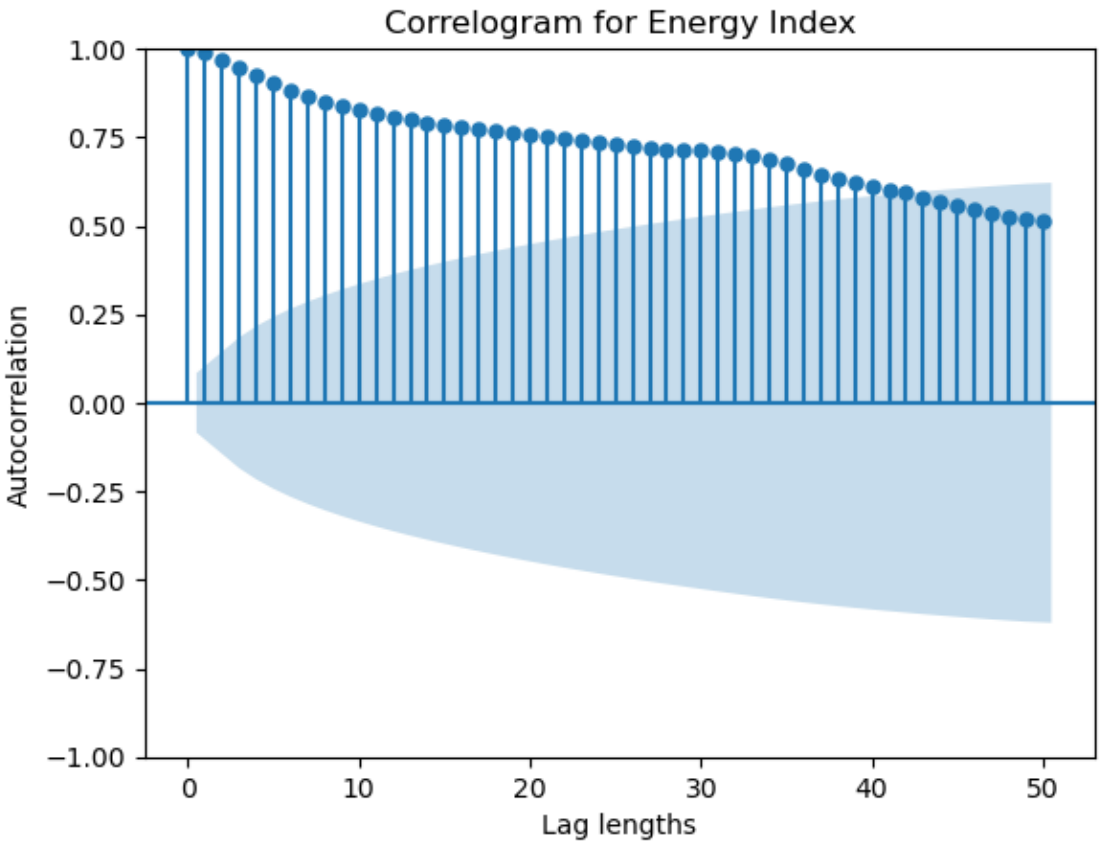
	Correlations	P-values
Total Index	-0.271408	1.219224e-10
Tot. Index (Ex. Energy)	-0.181758	1.997988e-05
Energy Index	-0.280557	2.684967e-11
Metals & Minerals Index	-0.308800	1.750228e-13
Agriculture Index	-0.416436	3.131778e-24
Fish Index	0.237805	1.971426e-08
Forestry Index	0.257792	1.045490e-09

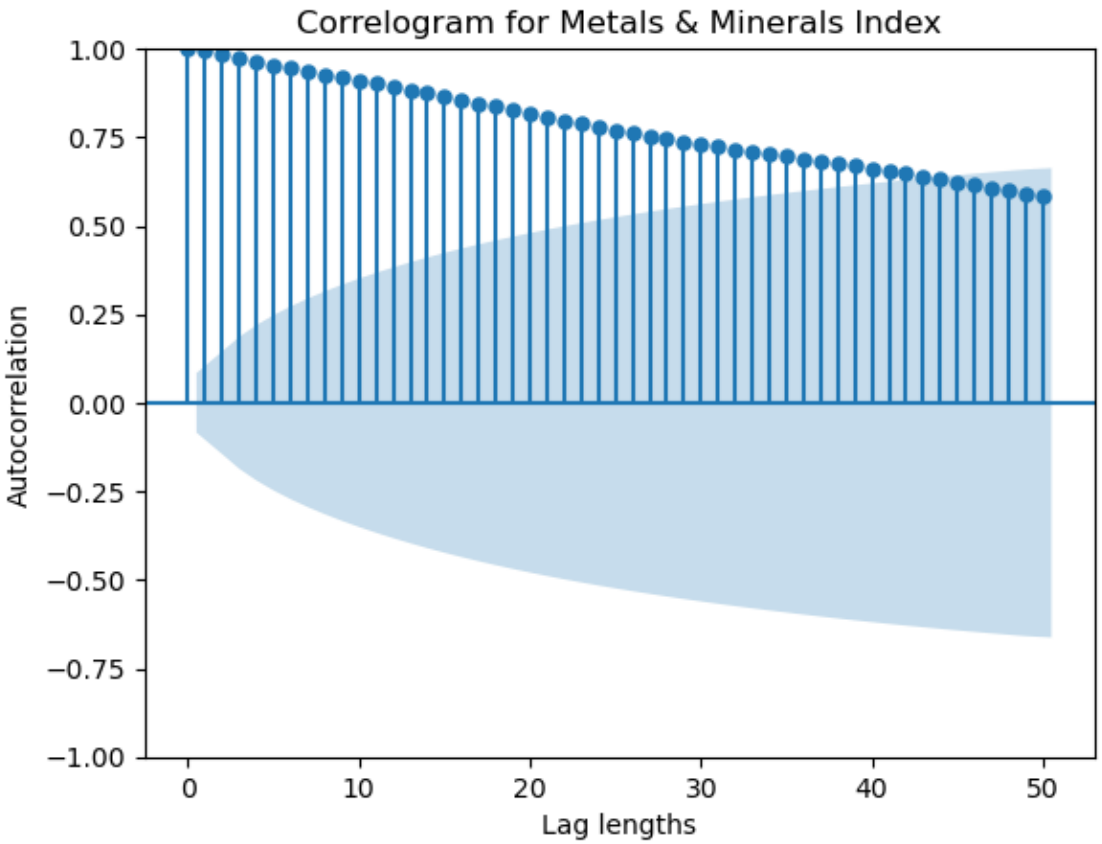
[25]:
```
for col in merged_data.columns:
  plot_acf(merged_data[col].dropna(), lags=50)
  plt.title(f'Correlogram for {col}')
  plt.xlabel('Lag lengths')
  plt.ylabel('Autocorrelation')
  plt.show()
```

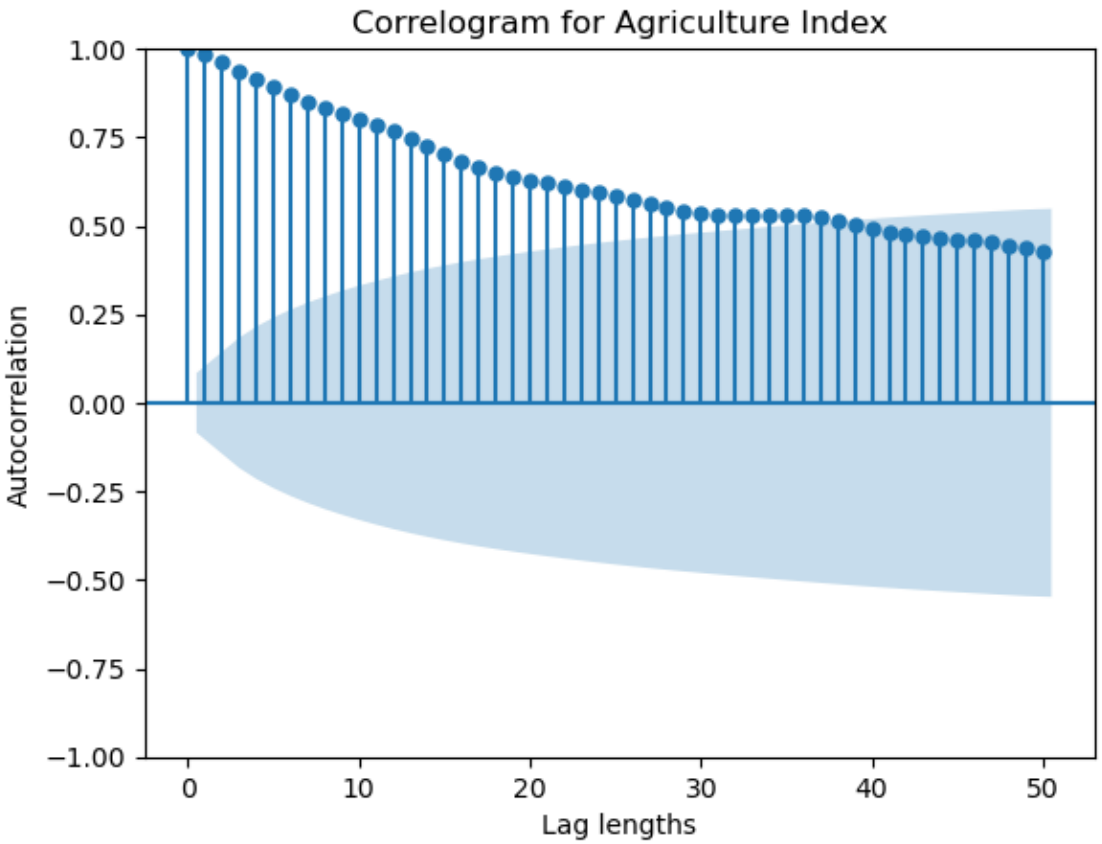


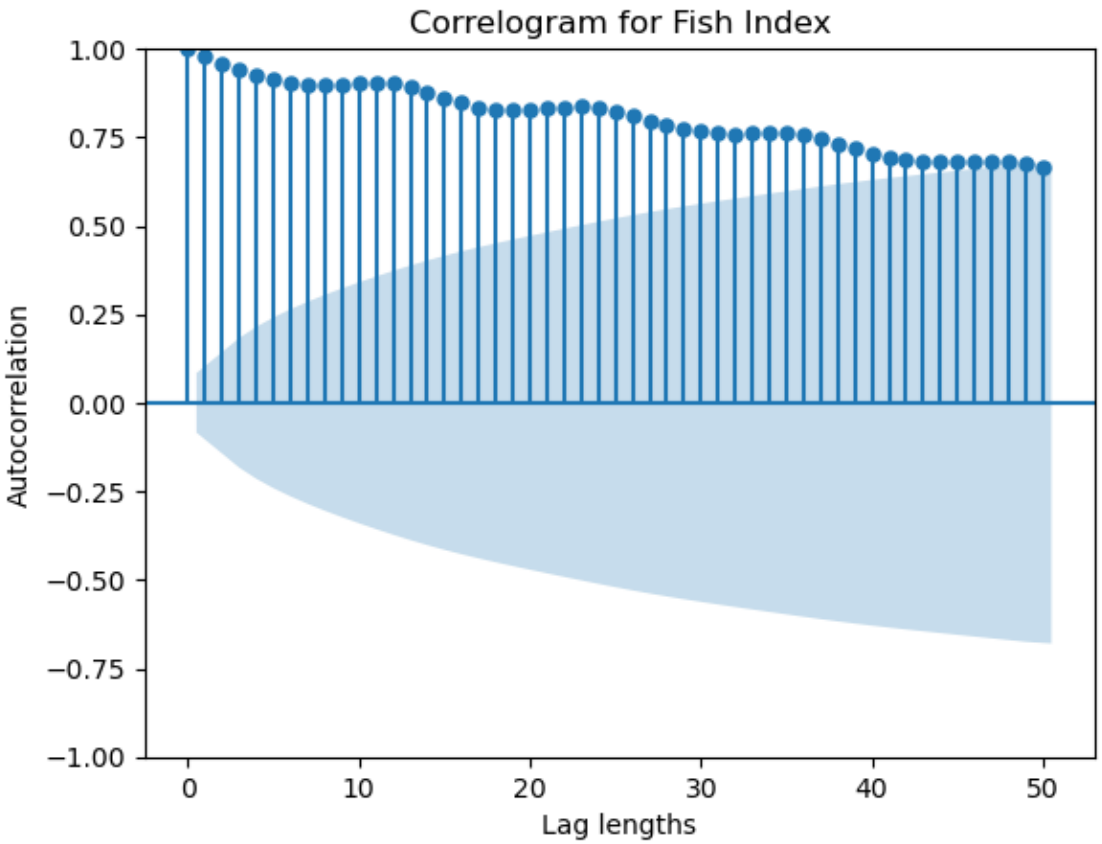

Correlogram for Total Index
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0
10
20
30
40
50
Lag lengths

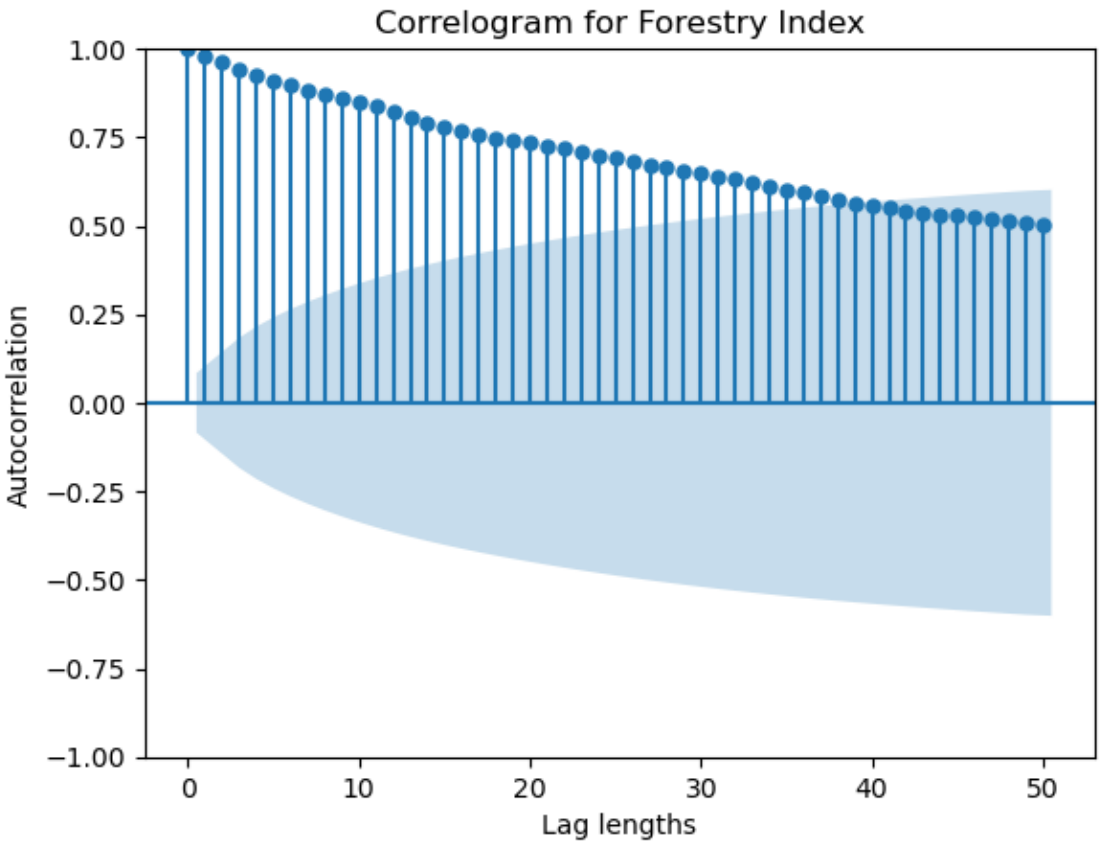

Correlogram for Tot. Index (Ex. Energy)
Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50
Lag lengths

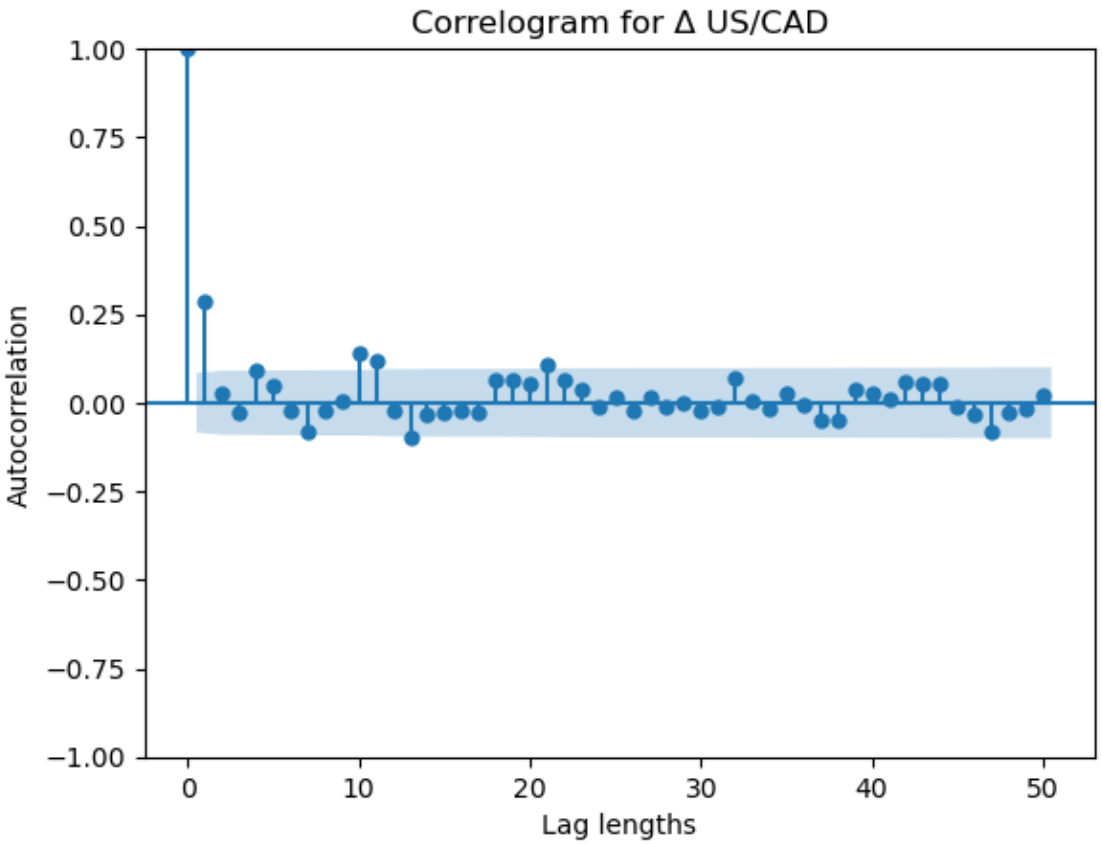

Correlogram for Energy Index
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0
10
20
30
40
50
Lag lengths

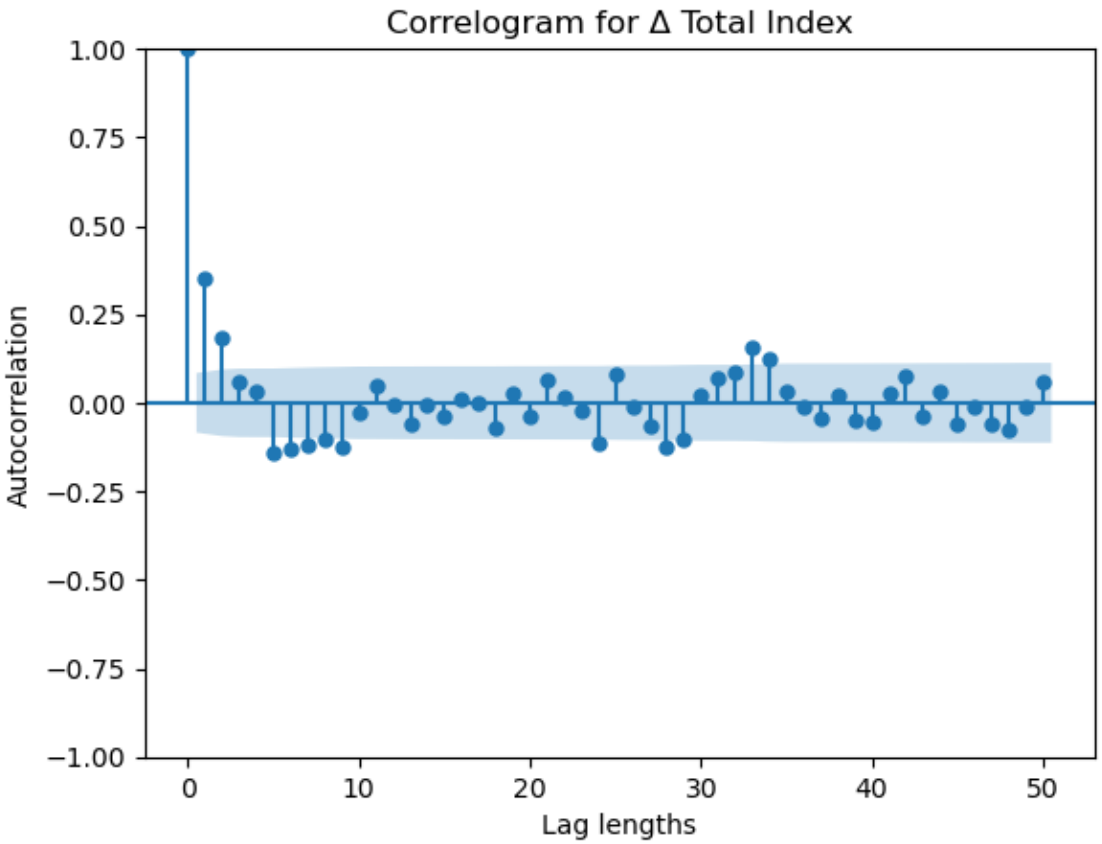

Correlogram for Metals & Minerals Index
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0
10
20
30
40
50
Lag lengths

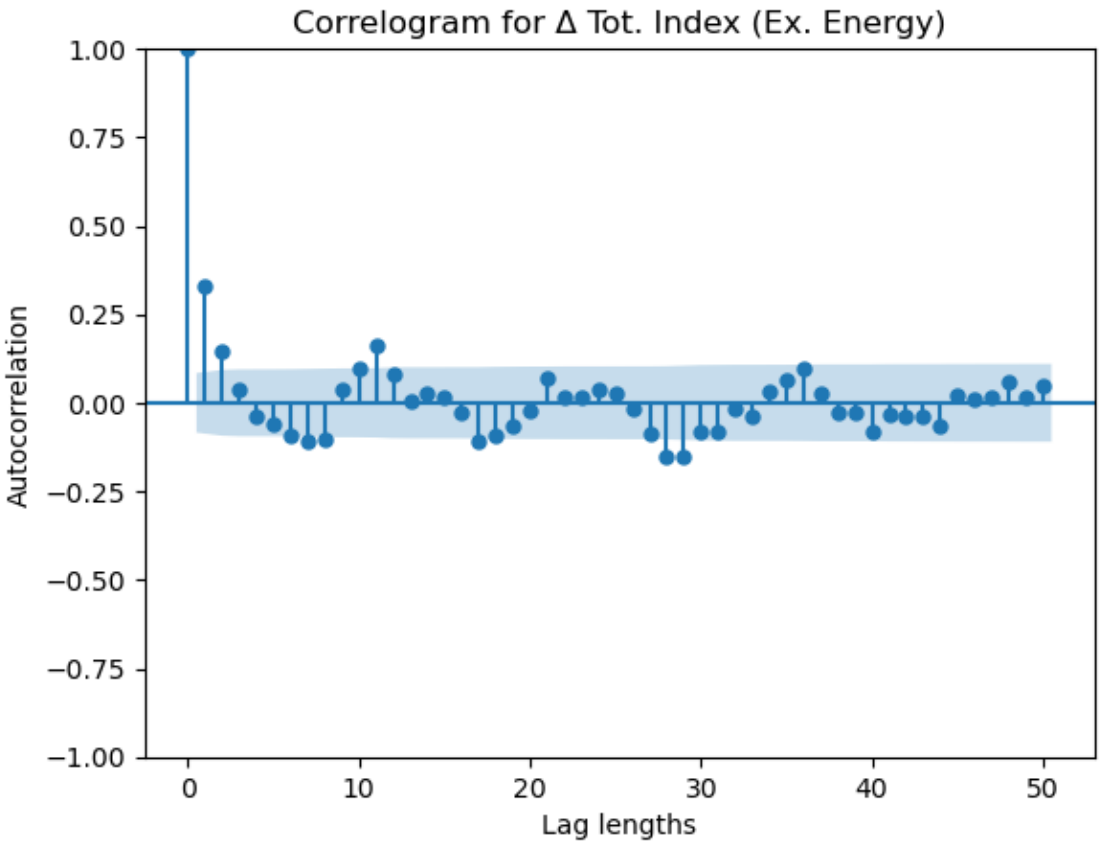

Correlogram for Agriculture Index
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0
10
20
30
40
50
Lag lengths

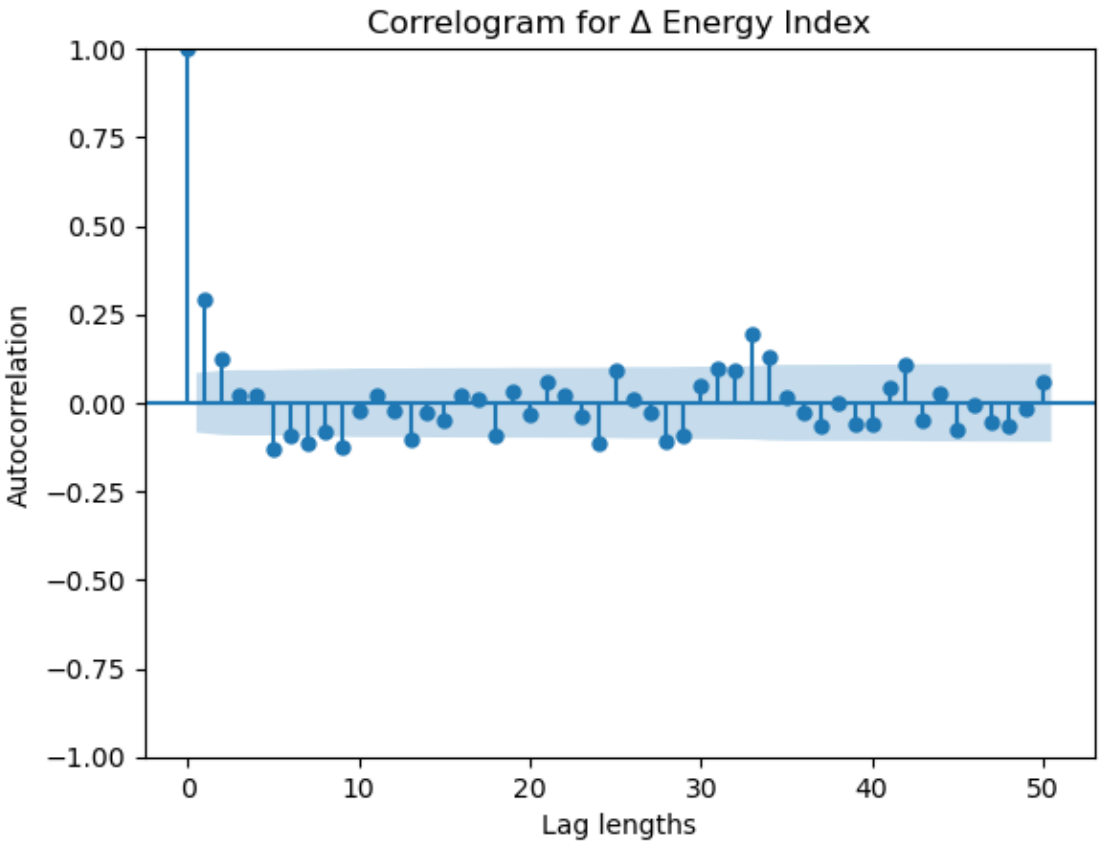

Correlogram for Fish Index
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0
10
20
30
40
50
Lag lengths

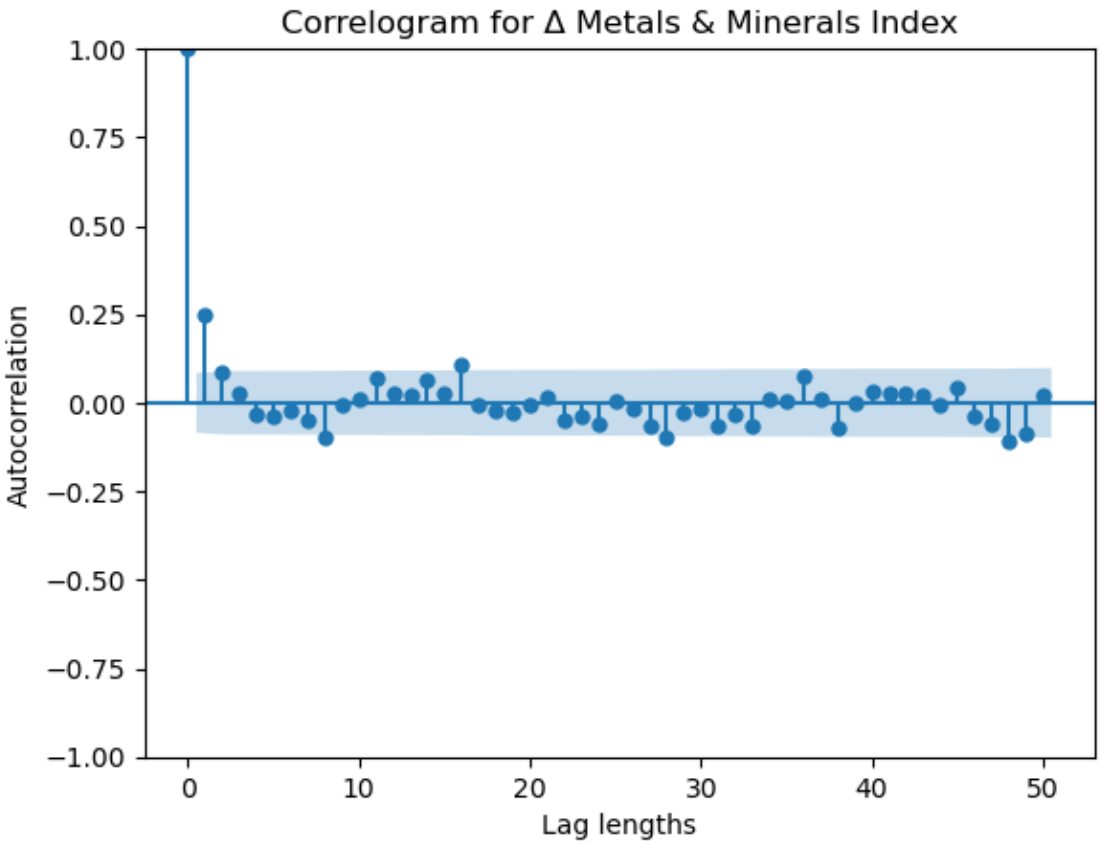

Correlogram for Forestry Index
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0
10
20
30
40
50
Lag lengths

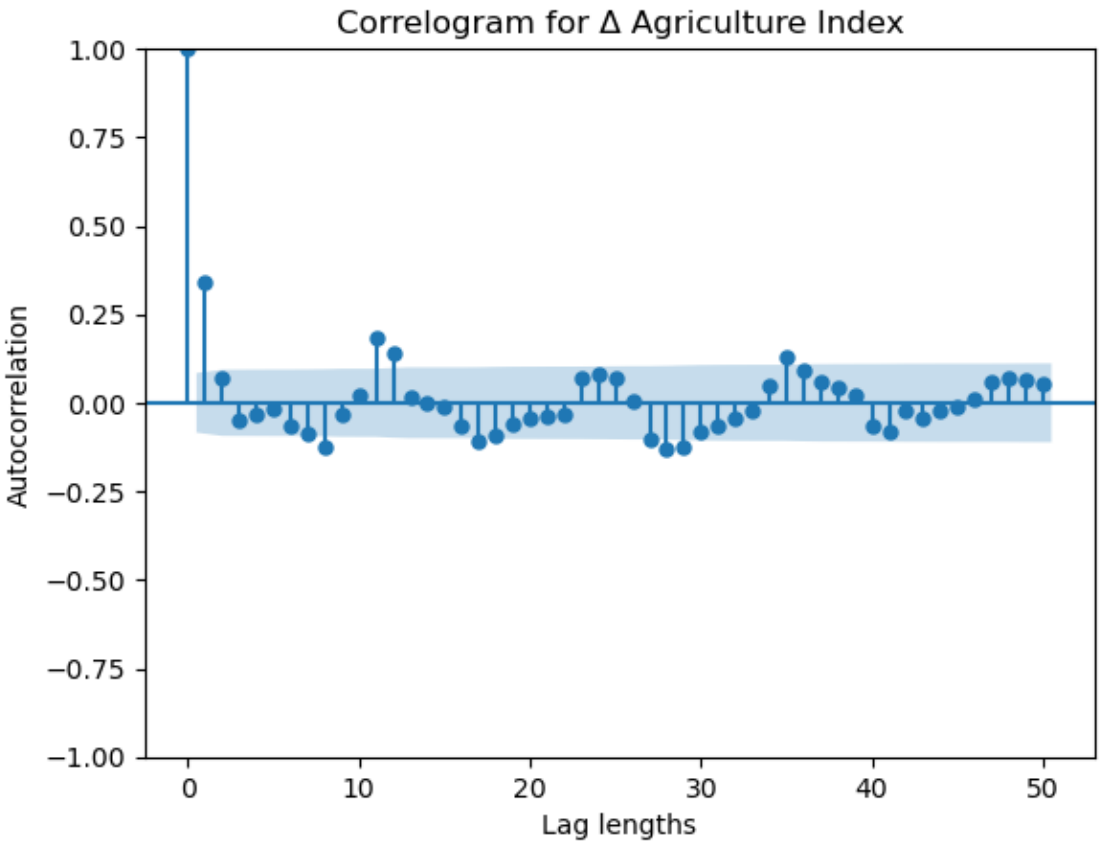

Correlogram for Δ US/CAD
Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50
Lag lengths

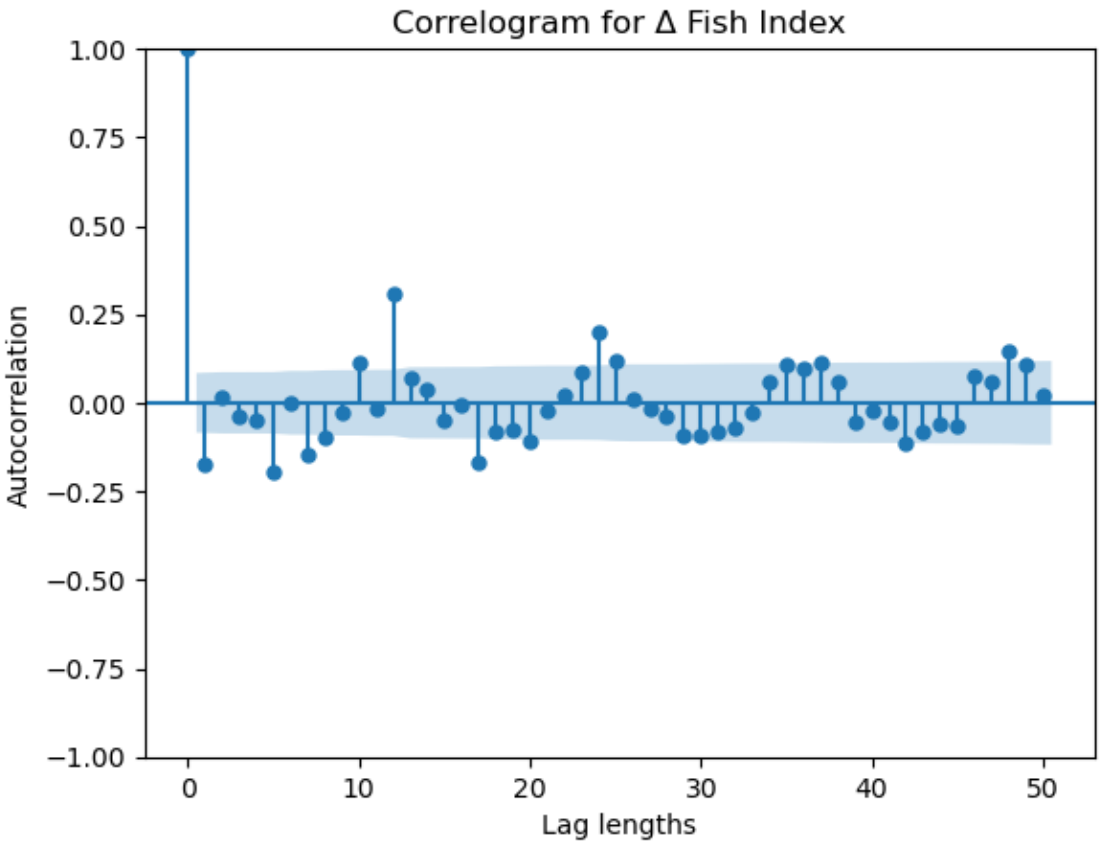

Correlogram for Δ Total Index
Autocorrelation
Lag lengths
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50

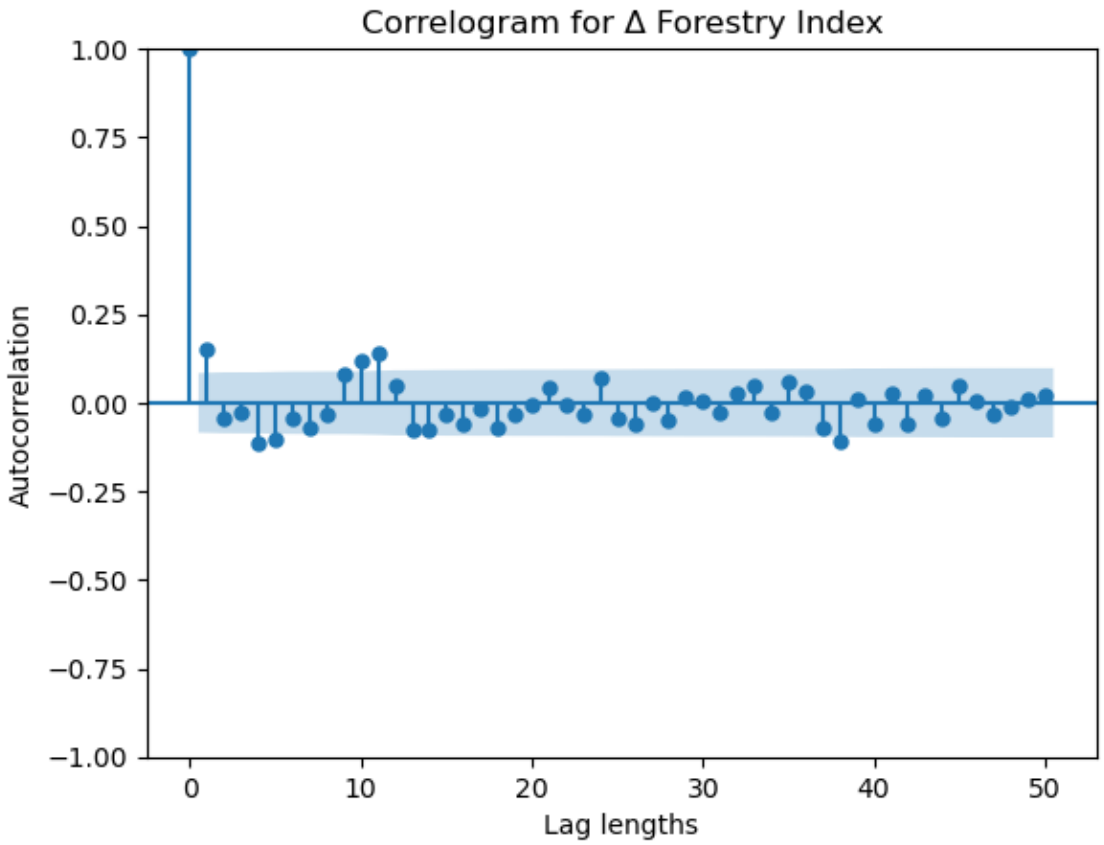

Correlogram for Δ Tot. Index (Ex. Energy)
Autocorrelation
Lag lengths
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50


Correlogram for Δ Energy Index
Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50
Lag lengths


Correlogram for Δ Metals & Minerals Index
Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
10
20
30
40
50
Lag lengths


Correlogram for Δ Agriculture Index
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0
10
20
30
40
50
Lag lengths


Correlogram for Δ Fish Index
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0
10
20
30
40
50
Lag lengths



[26]:
```
all_levels = ['US/CAD', 'Total Index', 'Tot. Index (Ex. Energy)', 'Energy␣
 ↪Index', 'Metals & Minerals Index', 'Agriculture Index', 'Fish Index',␣
 ↪'Forestry Index']
all_first_differences = ['\u0394 US/CAD','\u0394 Total Index', '\u0394 Tot.␣
 ↪Index (Ex. Energy)', '\u0394 Energy Index', '\u0394 Metals & Minerals␣
 ↪Index', '\u0394 Agriculture Index', '\u0394 Fish Index', '\u0394 Forestry␣
 ↪Index']


def adf(i):
  test_statistic= []
  p_value = []
  lag_order = []
  for j in i:
    test_result = adfuller(merged_data[j].dropna(), regression='ct', autolag =␣
 ↪'BIC')
    test_stat, p_val, lag = test_result[:3]
    test_statistic.append(test_stat)
    p_value.append(p_val)
    lag_order.append(lag)
```

```
    aggregated_results = pd.DataFrame({'Test Statistic': test_statistic,
                                       'P-value': p_value,
                                       'Optimal Lags': lag_order},
                                      index = i)

    return aggregated_results
```

[27]:
```
adf(all_levels)
```

[27]:

	Test Statistic	P-value	Optimal Lags
US/CAD	-1.820350	0.694864	1
Total Index	-3.189798	0.086454	1
Tot. Index (Ex. Energy)	-3.073165	0.112760	1
Energy Index	-3.318634	0.063246	1
Metals & Minerals Index	-2.320616	0.422582	1
Agriculture Index	-3.348556	0.058650	1
Fish Index	-3.011191	0.128979	13
Forestry Index	-4.098506	0.006339	1

[28]:
```
adf(all_first_differences)
```

[28]:

	Test Statistic	P-value	Optimal Lags
Δ US/CAD	-17.278915	0.000000e+00	0
Δ Total Index	-16.045115	1.121003e-22	0
Δ Tot. Index (Ex. Energy)	-16.476556	0.000000e+00	0
Δ Energy Index	-17.161346	0.000000e+00	0
Δ Metals & Minerals Index	-18.015887	0.000000e+00	0
Δ Agriculture Index	-16.281335	0.000000e+00	0
Δ Fish Index	-7.574259	6.874987e-10	12
Δ Forestry Index	-19.908161	0.000000e+00	0

[29]:
```
data = merged_data[['Δ Total Index', 'Δ US/CAD']]
data['lagged Δ US/CAD'] = data['Δ US/CAD'].shift(1)
data = data.dropna()
X = sm.add_constant(data['lagged Δ US/CAD'])
regression = sm.OLS(data['Δ Total Index'], X)
results = regression.fit()

results.summary()
```

```
C:\Programs\Anaconda3\TEMP\ipykernel_10168\206188953.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
```

```
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  data['lagged Δ US/CAD'] = data['Δ US/CAD'].shift(1)
```

[29]:

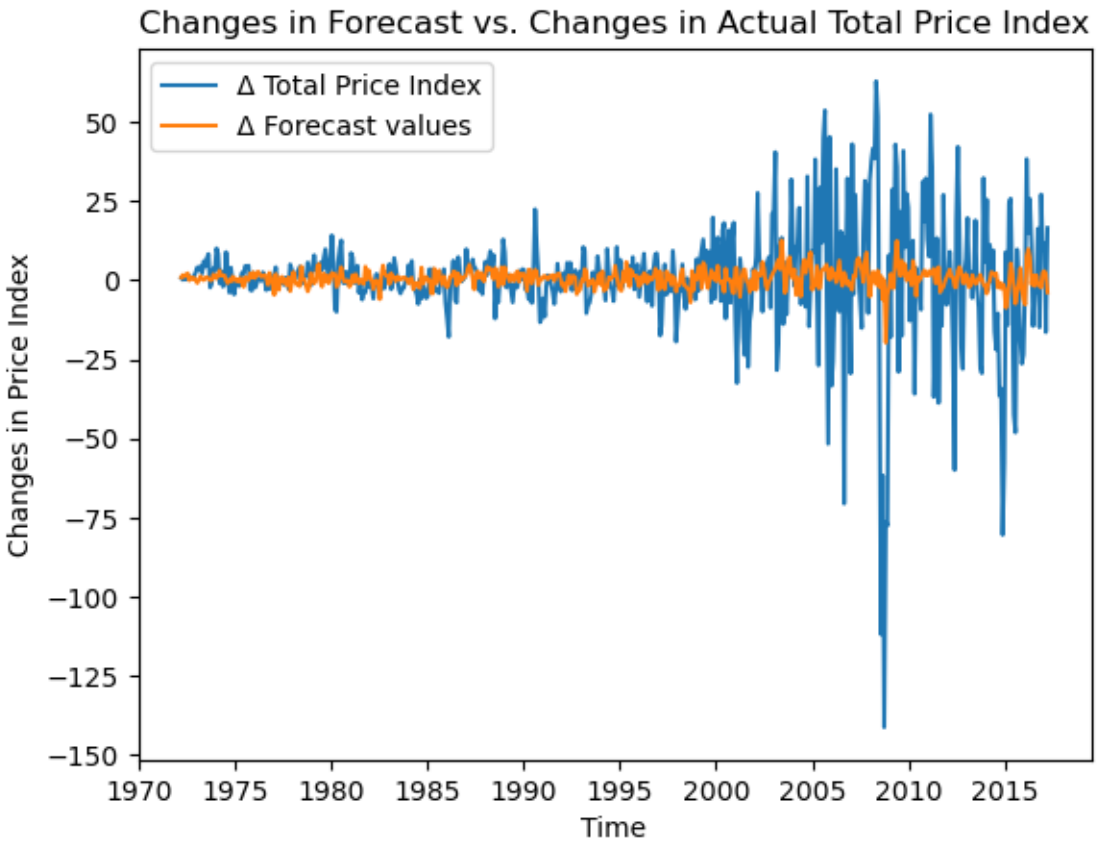
Dep. Variable:	Δ Total Index	**R-squared:**	0.026
Model:	OLS	**Adj. R-squared:**	0.024
Method:	Least Squares	**F-statistic:**	14.33
Date:	Tue, 06 Feb 2024	**Prob (F-statistic):**	0.000171
Time:	09:42:07	**Log-Likelihood:**	-2323.1
No. Observations:	542	**AIC:**	4650.
Df Residuals:	540	**BIC:**	4659.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	0.6703	0.757	0.885	0.377	-0.817	2.158
lagged Δ US/CAD	-161.3003	42.614	-3.785	0.000	-245.009	-77.591

Omnibus:	268.260	**Durbin-Watson:**	1.439
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	4060.132
Skew:	-1.782	**Prob(JB):**	0.00
Kurtosis:	15.926	**Cond. No.**	56.3

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[30]:
```
predicted_values = results.predict(X)
plt.plot(data['Δ Total Index'], label='Δ Total Price Index')
plt.plot(predicted_values, label='Δ Forecast values')
plt.xlabel('Time')
plt.ylabel('Changes in Price Index')
plt.title('Changes in Forecast vs. Changes in Actual Total Price Index')
plt.legend()

plt.show()
```



[31]:
```
coint(merged_data['US/CAD'],merged_data['Total Index'], trend='ct', autolag =␣
↪'BIC')
```

[31]:
```
(-2.984403958647333,
 0.27057735153066076,
 array([-4.35617324, -3.79812587, -3.50937687]))
```

[32]:
```
test_data = data[['Δ Total Index','lagged Δ US/CAD']]
grangercausalitytests(test_data, 10)
```

```
Granger Causality
number of lags (no zero) 1
ssr based F test:         F=1.7408  , p=0.1876  , df_denom=538, df_num=1
ssr based chi2 test:   chi2=1.7505  , p=0.1858  , df=1
likelihood ratio test: chi2=1.7477  , p=0.1862  , df=1
parameter F test:         F=1.7408  , p=0.1876  , df_denom=538, df_num=1

Granger Causality
number of lags (no zero) 2
```

```
ssr based F test:         F=1.0963  , p=0.3348  , df_denom=535, df_num=2
ssr based chi2 test:   chi2=2.2131  , p=0.3307  , df=2
likelihood ratio test: chi2=2.2086  , p=0.3314  , df=2
parameter F test:         F=1.0963  , p=0.3348  , df_denom=535, df_num=2

Granger Causality
number of lags (no zero) 3
ssr based F test:         F=1.0974  , p=0.3497  , df_denom=532, df_num=3
ssr based chi2 test:   chi2=3.3355  , p=0.3427  , df=3
likelihood ratio test: chi2=3.3253  , p=0.3441  , df=3
parameter F test:         F=1.0974  , p=0.3497  , df_denom=532, df_num=3

Granger Causality
number of lags (no zero) 4
ssr based F test:         F=0.9347  , p=0.4434  , df_denom=529, df_num=4
ssr based chi2 test:   chi2=3.8025  , p=0.4334  , df=4
likelihood ratio test: chi2=3.7891  , p=0.4353  , df=4
parameter F test:         F=0.9347  , p=0.4434  , df_denom=529, df_num=4

Granger Causality
number of lags (no zero) 5
ssr based F test:         F=2.0709  , p=0.0676  , df_denom=526, df_num=5
ssr based chi2 test:   chi2=10.5711 , p=0.0606  , df=5
likelihood ratio test: chi2=10.4684 , p=0.0630  , df=5
parameter F test:         F=2.0709  , p=0.0676  , df_denom=526, df_num=5

Granger Causality
number of lags (no zero) 6
ssr based F test:         F=2.5373  , p=0.0198  , df_denom=523, df_num=6
ssr based chi2 test:   chi2=15.6020 , p=0.0161  , df=6
likelihood ratio test: chi2=15.3793 , p=0.0175  , df=6
parameter F test:         F=2.5373  , p=0.0198  , df_denom=523, df_num=6

Granger Causality
number of lags (no zero) 7
ssr based F test:         F=2.5582  , p=0.0134  , df_denom=520, df_num=7
ssr based chi2 test:   chi2=18.4242 , p=0.0102  , df=7
likelihood ratio test: chi2=18.1140 , p=0.0115  , df=7
parameter F test:         F=2.5582  , p=0.0134  , df_denom=520, df_num=7

Granger Causality
number of lags (no zero) 8
ssr based F test:         F=3.1865  , p=0.0015  , df_denom=517, df_num=8
ssr based chi2 test:   chi2=26.3304 , p=0.0009  , df=8
likelihood ratio test: chi2=25.7019 , p=0.0012  , df=8
parameter F test:         F=3.1865  , p=0.0015  , df_denom=517, df_num=8

Granger Causality
```

```
number of lags (no zero) 9
ssr based F test:         F=3.7176  , p=0.0002  , df_denom=514, df_num=9
ssr based chi2 test:   chi2=34.6956 , p=0.0001  , df=9
likelihood ratio test: chi2=33.6130 , p=0.0001  , df=9
parameter F test:         F=3.7176  , p=0.0002  , df_denom=514, df_num=9

Granger Causality
number of lags (no zero) 10
ssr based F test:         F=3.6129  , p=0.0001  , df_denom=511, df_num=10
ssr based chi2 test:   chi2=37.6143 , p=0.0000  , df=10
likelihood ratio test: chi2=36.3441 , p=0.0001  , df=10
parameter F test:         F=3.6129  , p=0.0001  , df_denom=511, df_num=10
```

[32]:
```
{1: ({'ssr_ftest': (1.740812448134465, 0.18759713386206153, 538.0, 1),
   'ssr_chi2test': (1.7505195807448801, 0.185811427164105, 1),
   'lrtest': (1.7476935875511117, 0.1861669434368003, 1),
   'params_ftest': (1.7408124481343334, 0.18759713386206275, 538.0, 1.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64eb984d0>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64e63ca50>,
   array([[0., 1., 0.]])]),
 2: ({'ssr_ftest': (1.096327198337354, 0.3348452770045843, 535.0, 2),
   'ssr_chi2test': (2.213146493839892, 0.33069021541777216, 2),
   'lrtest': (2.20862364601453, 0.3314388923975564, 2),
   'params_ftest': (1.0963271983373595, 0.3348452770045843, 535.0, 2.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64cf7fe10>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64eee1590>,
   array([[0., 0., 1., 0., 0.],
          [0., 0., 0., 1., 0.]])]),
 3: ({'ssr_ftest': (1.0974037936484997, 0.349715239829664, 532.0, 3),
   'ssr_chi2test': (3.335529951747413, 0.3427280723997282, 3),
   'lrtest': (3.3252515919584766, 0.34414352867898007, 3),
   'params_ftest': (1.0974037936485423, 0.3497152398296306, 532.0, 3.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64eedc050>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64ec2eb50>,
   array([[0., 0., 0., 1., 0., 0., 0.],
          [0., 0., 0., 0., 1., 0., 0.],
          [0., 0., 0., 0., 0., 1., 0.]])]),
 4: ({'ssr_ftest': (0.9347258681799805, 0.4433755626338173, 529.0, 4),
   'ssr_chi2test': (3.802514306849372, 0.4333918437895512, 4),
   'lrtest': (3.7891394517628214, 0.4352941573916168, 4),
   'params_ftest': (0.934725868179952, 0.44337556263383304, 529.0, 4.0)},
```

```
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64e91a1d0>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64eb56090>,
   array([[0., 0., 0., 0., 1., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 1., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 1., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 1., 0.]])]),
 5: ({'ssr_ftest': (2.070909752567062, 0.06759951387921952, 526.0, 5),
   'ssr_chi2test': (10.57108875597445, 0.06057936690520133, 5),
   'lrtest': (10.4683860603227, 0.06300077418311895, 5),
   'params_ftest': (2.0709097525670623, 0.06759951387921952, 526.0, 5.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64eebc910>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64eebce90>,
   array([[0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.]])]),
 6: ({'ssr_ftest': (2.537273041969586, 0.019768882862297756, 523.0, 6),
   'ssr_chi2test': (15.602046085992713, 0.016057059375412623, 6),
   'lrtest': (15.379284094917239, 0.017503238475923992, 6),
   'params_ftest': (2.537273041969575, 0.01976888286229831, 523.0, 6.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64eebccd0>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64e698290>,
   array([[0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.]])]),
 7: ({'ssr_ftest': (2.5582278072482727, 0.013447579702982333, 520.0, 7),
   'ssr_chi2test': (18.424159881047657, 0.010196025091347082, 7),
   'lrtest': (18.114017523585062, 0.011466032862644769, 7),
   'params_ftest': (2.5582278072482603, 0.013447579702983001, 520.0, 7.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64eba2ed0>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64e5e9c50>,
   array([[0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
```

```
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.]])]),
 8: ({'ssr_ftest': (3.1865259877830225, 0.0015381265094788687, 517.0, 8),
   'ssr_chi2test': (26.33044297835411, 0.0009218960176477123, 8),
   'lrtest': (25.70187255925066, 0.0011810160798400645, 8),
   'params_ftest': (3.1865259877830256, 0.0015381265094788687, 517.0, 8.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64e68a610>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64e689450>,
   array([[0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0.,
           0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
           0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.,
           0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.,
           0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.,
           0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.,
           0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.,
           0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
           0.]])]),
 9: ({'ssr_ftest': (3.717638957794304, 0.0001546288074270872, 514.0, 9),
   'ssr_chi2test': (34.695552685874084, 6.741780799082168e-05, 9),
   'lrtest': (33.613033150179945, 0.0001043994589725665, 9),
   'params_ftest': (3.717638957794307, 0.0001546288074270872, 514.0, 9.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64ef018d0>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64ef00650>,
   array([[0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
           0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.,
           0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.,
           0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.,
           0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.,
           0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.,
           0., 0., 0.],
```

```
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
           0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
           1., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
           0., 1., 0.]])]),
 10: ({'ssr_ftest': (3.6129490156234936, 0.00011630574502336793, 511.0, 10),
   'ssr_chi2test': (37.61426372429939, 4.428924714779609e-05, 10),
   'lrtest': (36.34406523368489, 7.345023139421463e-05, 10),
   'params_ftest': (3.6129490156234993, 0.00011630574502336793, 511.0, 10.0)},
  [<statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64ef03810>,
   <statsmodels.regression.linear_model.RegressionResultsWrapper at
0x2a64ef03bd0>,
   array([[0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0.,
           0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0.,
           0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.,
           0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0.,
           0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0.,
           0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
           0., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
           1., 0., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
           0., 1., 0., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
           0., 0., 1., 0., 0.],
          [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
           0., 0., 0., 1., 0.]])])}
```