Spread Analysis of Co-integrated pairs for Mean Reversion Trading

- Author: Ayush Patel
- Github: https://github.com/Ayush-Patel15
- LinkedIn: https://www.linkedin.com/in/ayush-15-patel/

In financial markets, prices often deviate from their typical patterns due to market noise, sentiment, or external factors. But what if we could identify relationships between assets that remain stable over time despite these short-term fluctuations? This post continues the journey into identifying such relationships using statistical methods and explores how these insights can be turned into actionable trading strategies.

Analysis of Cointegrated Pairs

To learn about cointegration, and related terms, you can read my previous post on cointegration.

The goal of this analysis is to build upon the results of the cointegration test conducted earlier by exploring and modeling the relationship between cointegrated stock pairs. We aim to examine their spread (any linear relationship between them), assess stationarity, and use statistical tools to generate actionable insights for pairs trading strategies.

Pairs Trading and the Mean Reversion Strategy

Pairs trading is a market-neutral strategy that profits from temporary deviations between two related assets. When two stocks exhibit cointegration, their prices move together in the long term, even if they temporarily deviate from their typical relationship. The key idea is to identify these deviations, assuming the spread between them will revert to its historical mean. Here's how it works:

- Identifying a Pair: Two cointegrated stocks, let's call them Stock A and Stock B, have a stable spread between their prices over time.
- Monitoring the Spread: If the spread widens significantly (e.g., Z-score > +2SD), Stock A may be considered overvalued, and Stock B undervalued. Spread is just any linear relationship between the Stock A and Stock B.
- Trading Signals: Go Long (Buy) on the undervalued stock and Go Short (Sell) on the overvalued stock when the spread deviates significantly. Close the trade when the spread reverts to the mean, realizing a profit.
- Risk Management: By pairing long and short positions, pairs trading minimizes exposure to broader market trends and focuses on the relationship between the two assets.

Importance in Finance

- Market Neutral Strategy: Pairs trading strategies are market-neutral, meaning they are unaffected by overall market movements.
- Identifying Arbitrage Opportunities: When the spread deviates significantly from its mean, it can signal opportunities for profitable trades.
- Risk Management: Stationary spreads allow for clearly defined entry and exit points, helping manage trade risks.

Python Implementation

Let's dive directly to the main topic, and learn the steps involved in research of finding Mean Reverting pairs. For this demonstration, I am using the output of one of my previous post, where I have saved a dataframe with the pairs and their verdict of `cointegrated` or `not-cointegrated`. I will filter out the pairs that are cointegrated and follow the below steps.

`STEP-1` : Use OLS (Ordinary Least Squares) Regression to find the relationship between the pairs.

- Apply an OLS regression to fit a linear model between the two stocks in each cointegrated pair.
- The regression equation models one stock as a function of the other, helping compute the spread between them.

`STEP-2` : Create the Spread using the Beta/Slope coeficient of the model.

- In pairs trading, this beta is called as the hedge ratio.
- Compute the spread as the linear equation: stock_y - (hedge_ratio * stock_x), where x and y are the dependent and independent variables.
- The spread represents the linear combination of their expected relationship.

`STEP-3` : Apply the stationarity test on the created spread series.

- Use the Augmented Dickey-Fuller (ADF) test to verify if the spread is stationary.
- A stationary spread is crucial for pairs trading, as it suggests the spread will revert to its historical mean over time.

`STEP-4` : If the spread is stationary, generate the Z-Score Series

- Normalize the spread by calculating its Z-score to measure how far it deviates from the mean.
- This step helps define boundaries for typical versus extreme deviations.

$$Z(t) = (Spread(t) - \mu_s)/\sigma_s$$

Where μ_s is the mean and σ_s is the standard deviation of the spread.

`STEP-5` : Plot the Z-Score Series, and save the output.

- Visualize the Z-score series with statistical boundaries at the mean, +2SD, and -2SD.
- Highlight key points where the spread crosses these thresholds to identify potential trading signals.

This plot have the pairs to analyze and decide, whether to trade or apply the mean reversion pairs trading logic or not.

In [21]:
```python
## Necessary Import Statements
from statsmodels.tsa.stattools import adfuller
import matplotlib.pyplot as plt
import statsmodels.api as sm
import pandas as pd
import numpy as np
import os
```

In [22]:
```python
## FOLDER-PATHS
DATA_FOLDERPATH = "E:\\Market-Work\\All-Data\\Daily_Data-Stocks_Indices\\"
VERDICT_FILEPATH = "E:\\Market-Work\\My-Work\\Cointegration_Test\\Cointegration-Verdict.csv"
ZSCORE_FOLDERPATH = "E:\\Market-Work\\My-Work\\Spread_Analysis-and-Mean_Reversion_Technique\\Mea

## Reading of the dataframe, saved previously
verdict_df = pd.read_csv(VERDICT_FILEPATH, index_col=0)
```

In [23]:
```python
## Function to get the pair names, that are cointegrated, do remove the duplicates: apply sorting
def get_cointegrated_pairs(verdict_df):
        cointegrated_sets = set()
        for stock_1 in verdict_df.columns:
                cointegrated = verdict_df[(verdict_df[stock_1]=="cointegrated")][stock_1]
                for stock_2 in cointegrated.index:
                        pair_list = [stock_1, stock_2]
                        pair_list.sort()
                        cointegrated_sets.add(tuple(pair_list))
        cointegrated_pair = list(cointegrated_sets)
        return cointegrated_pair
```

In [24]:
```python
## Function to take a cointegrated pair, create a spread and then check for stationarity in the s
def save_zscore_plots(stock_pair, DATA_FOLDERPATH, ZSCORE_FOLDERPATH):
        stock_1, stock_2 = stock_pair
        stockname_1, stockname_2 = stock_1.split(".csv")[0], stock_2.split(".csv")[0]
        # print("==> Processing:", stock_1, stock_2)
        stock_filepath_1 = os.path.join(DATA_FOLDERPATH, stock_1)
        stock_filepath_2 = os.path.join(DATA_FOLDERPATH, stock_2)
        stock_df_1 = pd.read_csv(stock_filepath_1, index_col=0, usecols=["Date","Close"])
        stock_df_1 = stock_df_1.loc["2021":]
        stock_df_2 = pd.read_csv(stock_filepath_2, index_col=0, usecols=["Date","Close"])
        stock_df_2 = stock_df_2.loc["2021":]
        ## Fit the data with OLS method
        y_series = np.array(stock_df_1["Close"])
        x_series = np.array(stock_df_2["Close"])
        model = sm.OLS(y_series, x_series)
        result = model.fit()
        hedge_ratio = result.params[0]
        ## Build the equation for spread, and check for stationarity of that spread
        spread = y_series - (hedge_ratio * x_series)
        spread_pvalue = adfuller(spread)[1]
        ## If the spread is stationary, plot and save the Z-Score of the spread series
        if (spread_pvalue < 0.05):
                # print(f"{stockname_1}-{stockname_2} is stationary series.")
                z_spread = (spread - np.mean(spread)) / np.std(spread)
                zscore_mean, zscore_std = np.mean(z_spread), np.std(z_spread)
                plt.figure(figsize=(12, 10))
                plt.plot(z_spread, color="black", label="Z-score")
                plt.title(f"{stockname_1}-{stockname_2}, Hedge:{hedge_ratio}, p-value:{spread_pv
                plt.axhline(y=zscore_mean, color="blue", linestyle="dashed", label="Mean", linew
                plt.axhline(y=zscore_mean + (2 * zscore_std), color="red", linestyle="dotted", la
                plt.axhline(y=zscore_mean - (2 * zscore_std), color="green", linestyle="dotted",
                plt.legend()
                plt.tight_layout()
                saving_path = os.path.join(ZSCORE_FOLDERPATH, f"{stockname_1}-{stockname_2}.png"
                plt.savefig(saving_path)
                plt.close()
                ## Return the pair information, to store in a dataframe
                return [(stockname_1, stockname_2), hedge_ratio, spread_pvalue]
        return None
```

In [25]:
```
## Take out all the pairs from the dataframe
cointegrated_pair = get_cointegrated_pairs(verdict_df)
cointegrated_pair.sort()
```

In [26]:
```
## Save the plots, using the zscore function
all_pairs_result = []
for stock_pair in cointegrated_pair:
        result = save_zscore_plots(stock_pair, DATA_FOLDERPATH, ZSCORE_FOLDERPATH)
        if result:
                all_pairs_result.append(result)

## Create a dataframe, from the result
all_pairs_df = pd.DataFrame(all_pairs_result, columns=["Pair_Name", "Hedge_Ratio", "Spread_Pvalu
all_pairs_df.to_csv("mean_reverting_pairs.csv")
all_pairs_df
```

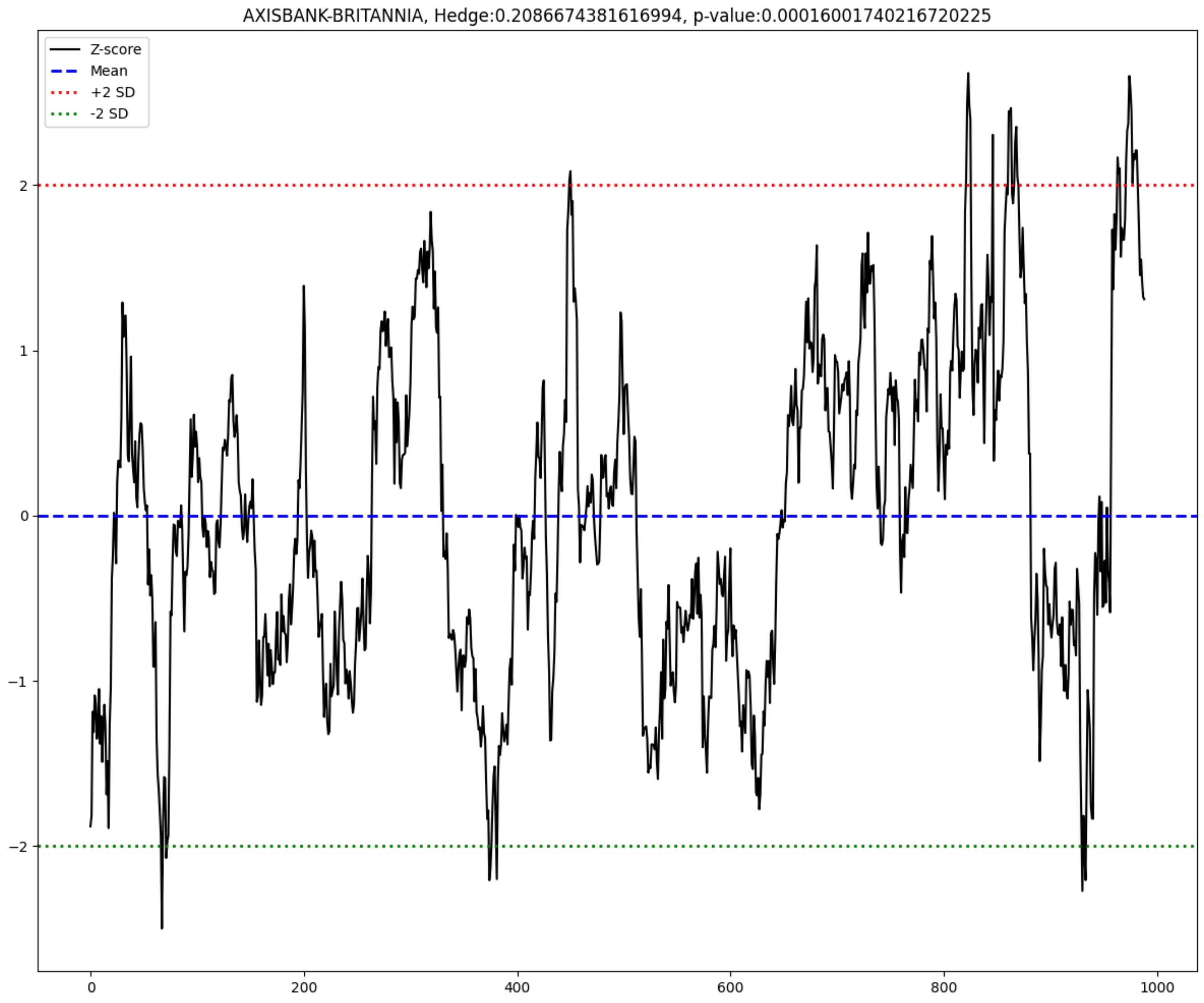
Out[26]:

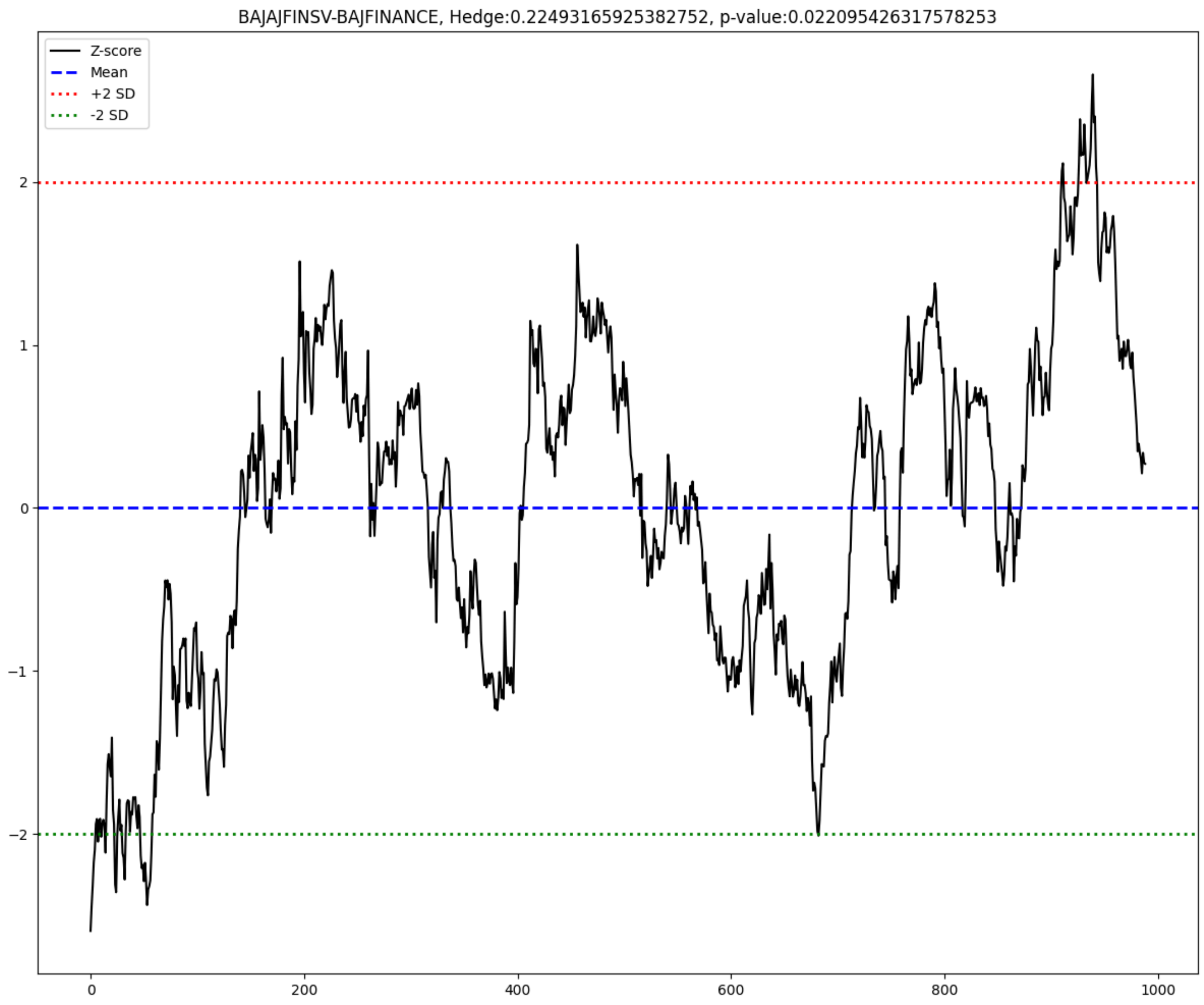
	Pair_Name	Hedge_Ratio	Spread_Pvalue
0	(ADANIPORTS, CIPLA)	0.817557	0.014739
1	(ADANIPORTS, HEROMOTOCO)	0.271965	0.000401
2	(ADANIPORTS, ONGC)	5.120778	0.016772
3	(APOLLOHOSP, CIPLA)	4.355830	0.011930
4	(APOLLOHOSP, DRREDDY)	4.755987	0.011459
5	(APOLLOHOSP, GRASIM)	2.652961	0.001459
6	(APOLLOHOSP, HCLTECH)	3.964081	0.001417
7	(APOLLOHOSP, ICICIBANK)	5.472318	0.002636
8	(APOLLOHOSP, NIFTY50)	0.260859	0.030433
9	(APOLLOHOSP, ULTRACEMCO)	0.604877	0.005304
10	(AXISBANK, BRITANNIA)	0.208667	0.000160
11	(AXISBANK, CIPLA)	0.796673	0.008501
12	(AXISBANK, DRREDDY)	0.873525	0.011421
13	(AXISBANK, MARUTI)	0.096936	0.002544
14	(BAJAJFINSV, BAJFINANCE)	0.224932	0.022095
15	(BANKNIFTY, BRITANNIA)	9.587851	0.007539
16	(BANKNIFTY, DRREDDY)	40.184071	0.016653
17	(BHARTIARTL, SUNPHARMA)	0.837146	0.018560
18	(CIPLA, EICHERMOT)	0.327559	0.007523
19	(CIPLA, GRASIM)	0.605480	0.000647
20	(CIPLA, ICICIBANK)	1.249313	0.009743
21	(CIPLA, SBIN)	1.921503	0.005455
22	(COALINDIA, NTPC)	1.162960	0.033485
23	(DRREDDY, ULTRACEMCO)	0.125488	0.003777
24	(EICHERMOT, ICICIBANK)	3.802502	0.002674
25	(EICHERMOT, SBIN)	5.843706	0.000953
26	(GRASIM, NIFTY50)	0.098119	0.025790
27	(GRASIM, SBIN)	3.168278	0.016624
28	(HCLTECH, ULTRACEMCO)	0.152057	0.006434
29	(HINDALCO, TCS)	0.137647	0.042066
30	(JSWSTEEL, MARUTI)	0.078550	0.027132
31	(JSWSTEEL, NESTLEIND)	0.351938	0.008969

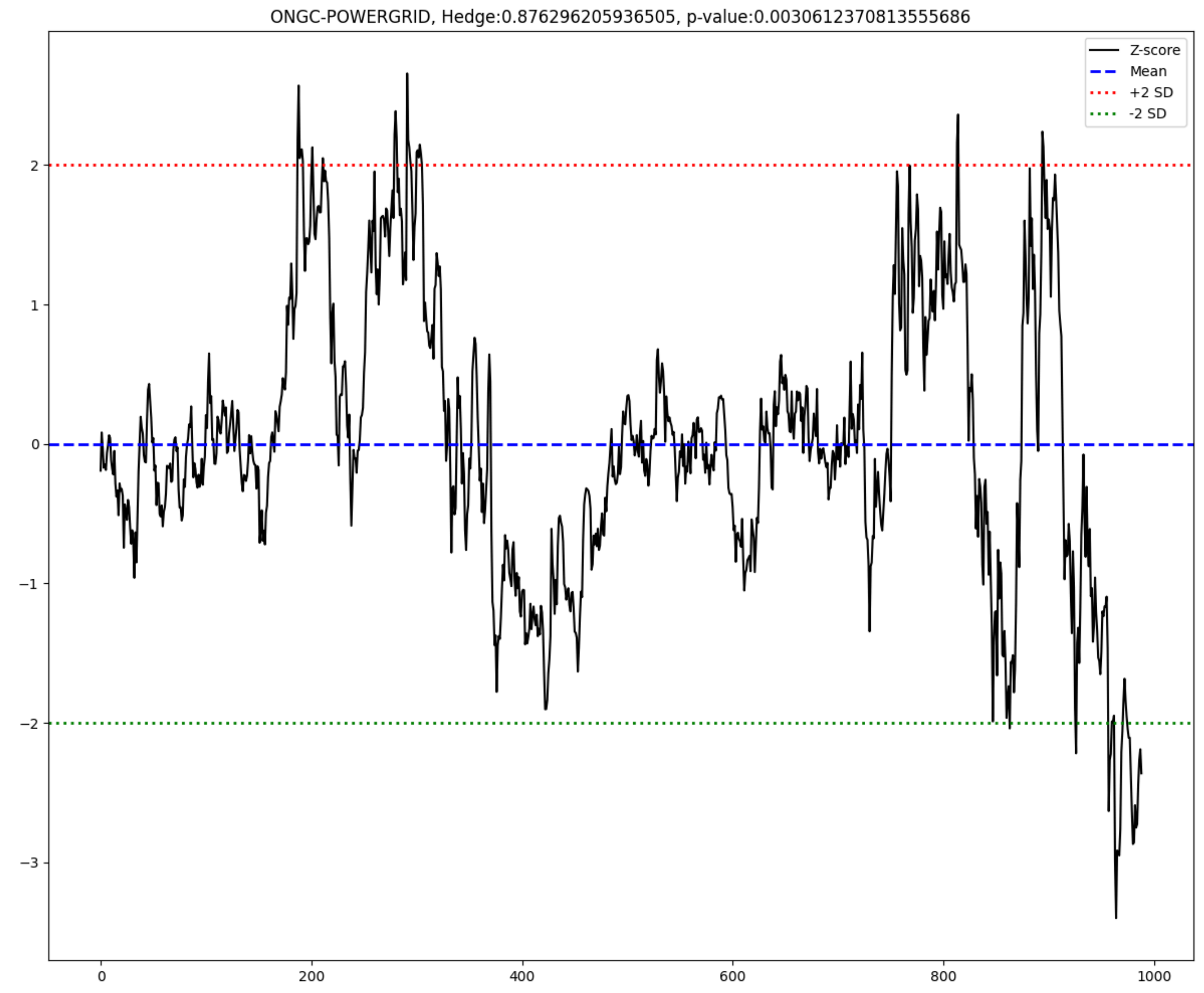
	Pair_Name	Hedge_Ratio	Spread_Pvalue
32	(JSWSTEEL, NIFTY50)	0.038828	0.012948
33	(MARUTI, SBIN)	15.803720	0.009736
34	(ONGC, POWERGRID)	0.876296	0.003061
35	(RELIANCE, SBILIFE)	0.952676	0.040494
36	(TATASTEEL, TCS)	0.033861	0.034059

Some Z-Score plots for reference

All the plots are present at my github, in the folder Mean_Reverting_Pairs.




BAJAJFINSV-BAJFINANCE, Hedge:0.22493165925382752, p-value:0.022095426317578253
Z-score
Mean
+2 SD
-2 SD
2
1
0
−1
−2
0
200
400
600
800
1000



Limitations of Mean Reversion Pairs Trading

- Assumption of Stationarity: The success of pairs trading hinges on the assumption that the spread between cointegrated pairs is stationary and will revert to its mean over time. However, market conditions can change, causing previously stationary relationships to break down.
- Selection Bias: Choosing the right pairs is critical. Poor pair selection can lead to false signals and suboptimal trading outcomes. The process often involves extensive data analysis and rigorous testing to ensure robust results.
- Market Regimes and Structural Shifts: Unexpected macroeconomic events or structural changes in industries can permanently shift relationships between pairs, rendering historical patterns invalid.
- Execution Risk: Delays in trade execution can result in missed opportunities or increased losses. High-frequency market movements may reduce the efficiency of pairs trading strategies.
- Correlation vs. Cointegration: Pairs trading requires cointegration, not just correlation. Two highly correlated stocks may not maintain a stable spread, leading to potential strategy failure if cointegration is not carefully verified.

- Transaction Costs and Slippage: Frequent trading to capture small deviations can result in significant transaction costs and slippage, eroding potential profits.

Recommendation for Better Results

To achieve more reliable and optimal outcomes, consider applying this technique to ETFs, related commodities, or sector-wise stock constituents, as these groups often share common factors that influence their movements, increasing the likelihood of stable relationships. Please note that this document focuses primarily on explaining the technique and steps involved, therefore I have used the data of all Nifty50 stocks that I had from the past cointegration test.

`Disclaimer` : This is just for the purpose of knowledge and my own learning, and not any advice to create any kind of trading strategy based on the post. Do you own analysis and invest in markets. However, this is just for my own learning purpose, so need not to worry. For learning purpose, I can test for any weird possibilities, even if doesn't make any sense in the practical world.

Do let me know, if you want more details on the spread and mean reversion pair trading strategy.
