Pairs Trading Model Using Cointegration

Joao Pimentel

October 2025

1 Introduction

In financial markets, undestading the connection between various assets is crucial for creating successful trading strategies and controlling risk. Although correlation assesses the brief co-movement of assets, it frequently neglects to reflect long-term equilibrium connections. Cointegration,offers a good statistical approach to detect pairs or sets of stocks whose prices tend to move in together over time, despite temporary fluctuations. This idea is particularly beneficial for spotting possible arbitrage chances and developing mean-reverting trading approaches like pairs trading. This project utilizes cointegration analysis to investigate the dynamic connections among chosen stocks. By investigating if these assets exhibit a long-term stochastic trend, our goal is to reveal significant insights that standard correlation analysis may miss.

2 Theoretical Foundation

2.1 Time Series and Stationary Process

Time-series data is a type of data that organizations rely on to track trends and make predictions over specific periods. It is characterized by its chronological order, allowing businesses to uncover underlying patterns, observe changes over time, and forecast future events. Because stock data evolves sequentially over time and often exhibits trends, volatility clustering, and autocorrelation, it requires time series analysis tools such as cointegration, ARIMA models, or GARCH models to model and forecast behavior accurately.

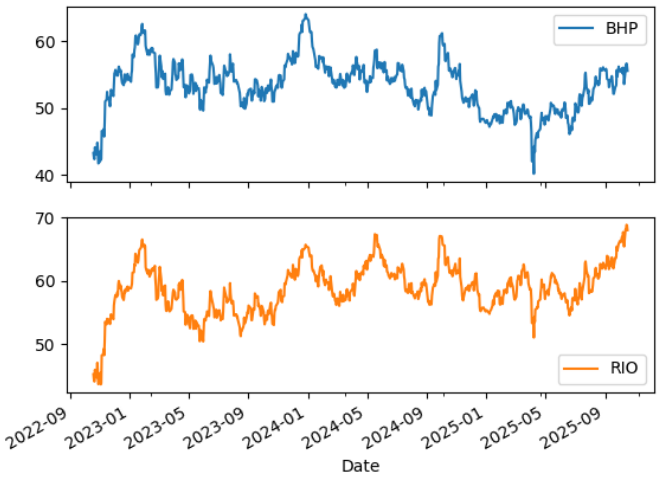
To understand cointegration, we need to also introduce the concept of stationary process. In mathematics and statistics, a stationary process is a stochastic process whose statistical properties, such as mean and variance, do not change over time. That will be one of the requirements to decide if two stocks cointegrate or not.

2.2 Engle–Granger two-step method

The method we are going to use to perform the cointegration is called Engle-Granger two-step method, here we will have two time series data points x_t and y_t both having order of integration $d = 1$, which represents the minimum number of differences required to obtain a stationary series. This is:

$$u_t = y_t - \beta x_t \tag{1}$$

where u_t is a stationary series. We can also see this graphically, below, you can see the historical data of two stocks, BHP and RIO, that are known to be cointegrated:



it is easy to that if BHP is multiplied by a factor β, the difference between the two stocks RIO and $\beta \cdot BHP$ becomes approximately stationary.

In most cases β is unknown, and we must estimate it first, this is usually done using Ordinary Least Squares by regressing y_t on x_t and an intercept, then we can run an Augmented Dickey-Fuller (ADF) test on u_t. If the variables are found to be cointegrated, a second-stage regression is conducted, the second stage regression is given as:

$$\Delta y_t = \Delta x_t b + \alpha u_{t-1} + \epsilon_t \tag{2}$$

If the variables are not cointegrated (if we cannot reject the null of no cointegration when testing u_t), then $\alpha = 0$ and we estimate a difference model: $\Delta y_t = \Delta x_t b + \epsilon_t$.

3 Methods

The first step is to choose our stocks, in this case, we are going to analyze Coca-Cola (KO) and PepsiCo (PEP). The second step is to gather historical data and also make sure that we are using clean data.

Listing 1: Downloading and plotting stock data using Python

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn as sk
import statsmodels
import yfinance as yf
from statsmodels.tsa.stattools import coint
import statsmodels.api as sm
from sklearn.linear_model import LinearRegression
from statsmodels.tsa.vector_ar.vecm import coint_johansen
import warnings
warnings.filterwarnings('ignore')

tickers = ['KO','PEP'] # selecting our stocks, Coca-Cola and PepsiCo
df = yf.download(tickers, period='3y', auto_adjust=False) #
    downloading our data
```

```
df = df['Adj Close'].dropna()
df.plot(subplots=True)
plt.show() # plotting the historical data
```

After this, the third step was to perform a cointegration analysis between the two stock price series KO and PEP. First, the Engle–Granger two-step method is applied using the `coint` function from the `statsmodels` library, which provides a test statistic (score) and the corresponding p-value. If the p-value is below a chosen significance level (typically 0.05), the null hypothesis of no cointegration is rejected, indicating a stable long-run relationship between the two assets.

Next, an ordinary least squares (OLS) regression is run with KO as the dependent variable and PEP as the independent variable to estimate the hedge ratio, denoted by β. This ratio represents how much of PEP would be needed to hedge one unit of KO in a pairs trading strategy. Finally, the spread is computed as the difference between KO and β times PEP, and it is plotted to visualize how the deviation from the equilibrium relationship evolves over time. A stationary spread suggests that the pair may offer trading opportunities based on mean reversion.

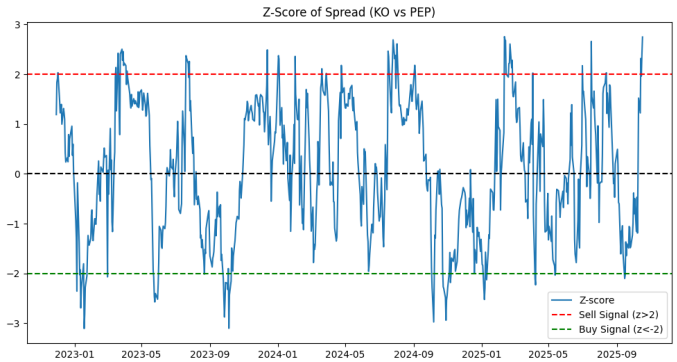
Listing 2: Cointegration test and hedge ratio estimation

```
score, p_value, _ = coint(df['KO'], df['PEP'])
print("score: ", round(score, 4))
print("p-value: ", round(p_value, 4))

X = sm.add_constant(df['PEP'])
model = sm.OLS(df['KO'], X).fit()
beta = model.params[1]
print("Estimated hedge ratio (beta):", round(beta, 4))

spread = df['KO'] - beta * df['PEP']
spread.plot(title='KO - beta * PEP (Spread)', figsize=(10,6))
plt.grid(True)
plt.show()
```

The fourth step computes the standardized spread, also known as the z-score, which helps to identify potential trading signals. A 30-day rolling window is applied to the spread to calculate its moving average and standard deviation over time. These rolling statistics are then used to normalize the spread, producing a z-score that measures how far the current spread deviates from its recent mean in units of standard deviations. When the z-score moves significantly above or below zero, it suggests that the spread has diverged from its equilibrium relationship, which may indicate a short-term trading opportunity if the spread is expected to revert to its mean.



The fifth step implements a simple pairs trading strategy based on the z-score of the spread between KO and PEP. Trading signals are generated according to threshold rules: a z-score above 2 triggers a short position, a z-score below -2 triggers a long position, and when the z-score is close to zero (between -0.5 and 0.5), the position is closed. The signals are forward-filled to maintain positions until a change occurs.

Next, daily returns for KO and PEP are calculated, and the return of the spread is computed using the previously estimated hedge ratio β. The strategy return is then calculated as the product of the lagged trading signal and the spread return. Cumulative returns are computed to track the overall performance over time.

Finally, the Sharpe ratio is calculated as a measure of risk-adjusted performance, annualized by multiplying the daily ratio by the square root of 252 trading days.

Listing 3: Trading signals and strategy performance evaluation

```
signal = np.where(zscore > 2, -1, np.nan)
signal = np.where(zscore < -2, 1, signal)
signal = np.where(abs(zscore) < 0.5, 0, signal)

signal = pd.Series(signal, index=zscore.index).ffill().fillna(0)

returns_ko = df['KO'].pct_change()
returns_pep = df['PEP'].pct_change()

spread_ret = returns_ko - beta * returns_pep
strategy_ret = signal.shift(1) * spread_ret

cumulative_ret = (1 + strategy_ret).cumprod() - 1

sharpe_ratio = np.sqrt(252) * (strategy_ret.mean() / strategy_ret.std
    ())

print("Sharpe Ratio:", round(sharpe_ratio, 4))
```

In the sixth step, we are going to compute the maximum drawdown of the trading strategy. First, the cumulative returns are calculated by compounding the daily strategy returns. The rolling maximum of these cumulative returns is then determined to identify the previous peaks. The drawdown at each point is calculated as the relative decline from the rolling maximum, and the maximum drawdown is the largest observed decline, expressed as a percentage. This metric provides a measure of the strategy's potential downside risk and the largest loss an investor could experience during the period.

Listing 4: Maximum drawdown calculation for the trading strategy

```
cum_ret = (1 + strategy_ret).cumprod()
rolling_max = cum_ret.cummax()
drawdown = (cum_ret - rolling_max) / rolling_max
max_drawdown = drawdown.min()
```

4 Results

The Engle–Granger cointegration test yielded a test statistic of -2.7771 with a corresponding p-value of 0.1731. Since the p-value exceeds the conventional significance level of 0.05, we fail

to reject the null hypothesis of no cointegration. This suggests that the stock prices of KO and PEP do not exhibit a statistically significant long-term equilibrium relationship over the analyzed period.

The hedge ratio (β) estimated from the ordinary least squares regression is -0.2576. This negative value indicates the relative position size of PEP required to hedge one unit of KO in a pairs trading strategy, reflecting the historical linear relationship between the two assets.

The trading strategy based on the z-score of the spread produced a Sharpe ratio of 0.337. While positive, this value indicates a modest risk-adjusted return over the analyzed period. Cumulative performance evaluation further reveals a maximum drawdown of -21.89%, highlighting the largest peak-to-trough decline experienced by the strategy and providing insight into its downside risk.