# Project : Mean-reverting strategy

Kilian Wan

August 3, 2025

## 1 Introduction

This report presents a personal project aimed at exploring the logic and implementation of mean-reverting trading strategies, particularly in the context of pair trading. the objective is not to develop a high-performance trading algorithm, but rather to become more familiar with the practical steps involved in financial strategy research.

As one of my first projects in this area, I chose to focus on relatively simple strategy involving two highly correlated stocks. Throughout the process, I spent time reading papers and online resources to better understand key statistical concepts such as stationarity, cointegration etc. I also documented many definitions and mathematical derivations, which may make this report longer than a typical implementation report, but which reflect the pedagogical nature of the project.

Ultimately, this work helped me solidify both theoretical and coding foundations, and serves as a base for more complex or realistic trading models in the future.

The full coded in this project–including data loading, strategy logic and plotting, is available in the following Github repository : [link]

## 2 Data

The data is downloaded from Kaggle [1]. For this strategy, we choose the ticks `KO` (Coca-Cola) and `PEP` (Pepsi). These two companies operate in the same sector and often display highly correlated price movements, making them a natural candidate pair for testing a mean-reversion hypothesis.

While only this single pair is analyzed in this report, the same framework could be extended to other stock pairs to evaluate robustness and generalizability of the strategy.

## 3 Pairs trading strategy

First, to test if the pairs trading strategy works, we need to perform a *Augmented Dickey Fuller* (ADF) test. This test serves the purpose of finding out which stocks can be paired

together in the pairs trading strategy.

We first explain the essential terms used in the ADF test. The first term is *stationary time series*. Stationarity means that the statistical properties of a time series, i.e., mean, variance and covariance do not change over time, which implies that the time series has no unit root. Mathematically, it means that for any $t, s, \tau \in \mathcal{T}$ (time points), $\mathbb{E}[X_t] = \mu$, $\text{Var}(X_t) = \sigma^2 < \infty$ and $\mathbb{E}[X_t X_{t+\tau}] = \mathbb{E}[X_s X_{s+\tau}]$.

So in the case of stationarity, trading signals are generated assuming that the prices of both stocks will revert to the mean eventually. Hence, we can take the advantage of the prices that deviate from the mean for a short period of time.

A *unit root* is a characteristic of a time series that makes it non-stationary. Mathematically speaking, a unit root is said to exist in a time series of the value of $\alpha = 1$ in the below equation :

$$Y_t = \alpha Y_{t-1} + \beta X_e + \epsilon$$

where $Y_t$ is the value of the time series at time $t$ and $X_e$ is an exogenous variable.

The ADF test is based on the two following hypothesis : (i) the null hypothesis $H_0$ states that there exists a unit root in the time series and is non-stationary and (ii) the alternative hypothesis $H_a$ states that there exists no unit root in the time series and is stationary or trend stationary.

So if there is a unit root present in the time series, it implies that the time series is non-stationary and the stocks are not co-integrated. Hence, stocks cannot be traded together.

Alternatively, if the null hypothesis gets rejected and the stocks show co-integration; it implies that the time series is stationary and the stocks can be traded.

## 3.1 Checking for stationarity

We define the function `adftest` in Python, which returns whether the null hypothesis is rejected or not. We obtain the following values for both stocks :

| **Stock** | ***t*-statistic** | ***p*-value** |
|---|---|---|
| KO | 2.232 | 0.999 |
| PEP | 2.284 2 | 0.999 |

TABLE 1: Values of the ADF test

From Table 1, we clearly see that for both stocks, we fail to reject the null hypothesis.

We now need to check if both stocks are co-integrated. To this end, we have to check whether the *spread* (at time $t$), defined as $\texttt{KO}_t - \beta \cdot \texttt{PEP}_t$, is stationary or not. Here $\beta$ is the slope of the regression of PEP on KO, called the *hedge ratio*. Another way of defining

the spread is via the log-regression, rather than the simple linear regression, that is : $\log \texttt{KO}_t - \beta \cdot \log \texttt{PEP}_t$. For simplicity, we call the slope of the log-regression, the log-hedge ratio and the spread, the log-spread. We test both linear and log for the spread, and check which one gives us the co-integration. By performing the ADF test on both spread and log-spread, we obtain that only the spread is stationary ($p-$value $< 0.05$). So we use the traditional spread in our strategy.

This shows that both stocks are co-integrated.

## 3.2 Building the Trading Signal

### 3.2.1 Computing the $Z$-score

Now that we have shown that both stocks are co-integrated, we calculate the $z$-score of the spread using a rolling window. Formally, we let $s_t := \text{spread}_t = \texttt{KO}_t - \beta \cdot \texttt{PEP}_t$ be the spread at time $t$. We choose a window size $\omega$ of 30 days (so one month). We compute the *rolling mean* and *rolling standard deviation*, defined as:

$$\mu_t = \text{mean}(s_{t-\omega+1}, \ldots, s_t), \quad \sigma_t = \text{std}(s_{t-\omega+1}, \ldots, s_t).$$

We then compute the $z$-score using the following formula :

$$z_t = \frac{s_t - \mu_t}{\sigma_t}$$

which yields a normal distribution with mean of 0 and standard deviation of 1.

Since we're using a rolling window of 30, we have to drop the first 30 elements of the vector $z_t$, since they have no value.

### 3.2.2 Trading Rules and Signal Logic

Now the goal is to use the $z$-scores to generate trading signals, that is *open positions* when the spread diverges significantly from its mean (i.e., when $|z_t|$ is large), and *close positions* when the $z$-score reverts close enough to zero (i.e., when $z_t \approx 0$). To see this, when the spread is large and positive, KO is too high relative to PEP. This means that KO is overvalued. When the spread is large and negative, KO is too low relative to PEP, and so is undervalued. The $z$-score just tells us how far the spread is from its typical rolling mean, in units of standard deviation.

More precisely, when we open a position, it means we *enter a trade*, that is we start holding stocks (either long or short) based on our signal. There are two types of actions. First, the Long KO / Short PEP (LK/SP), where we buy KO shares and sell PEP shares we don't own. The second is the Short KO / Long PEP (SK/LP), where we sell KO shares and buy PEP shares.

To close a position means that we *reverse the previous trades* to exist the market. So for example, if the previous position was LK/SP, now the action to close is sell KO, and buy back PEP. And if the previous position was SK/LP, we buy KO and sell PEP.

### 3.2.3 Encoding Positions

But how can we track the positions in the strategy? We can define a simple variable `position` where :

- 0 = flat (no trade),
- +1 = LK/SP,
- −1 = SK/LP.

The following table resumes the previous paragraphs :

| **Current $z$-score** | **Action** | **New position** |
|---|---|---|
| $z_t > 1$ | Open SK/LP | $-1$ |
| $z_t < -1$ | Open LK/SP | $+1$ |
| $-1 \leqslant z_t \leqslant 1$ and $z \neq 0$ | Hold current position | No change |

TABLE 2: Strategy

So it only enter trades when $z_t$ is beyond thresholds (in our case $\pm 1$). Only exit trades when $z_t$ crosses back toward 0 (e.g. $|z_t| < 0.1$). Otherwise hold the current position.

### 3.2.4 Strategy Returns

Once we have our position vector, we compute the strategy returns. At each time $t$, we want to know how much profit and loss did we make today, based on the position we held yesterday.

Let's define $r_t^{\text{KO}}$ as the return of KO from $t-1$ to $t$, and $r_t^{\text{PEP}}$ as the return of PEP from $t-1$ to $t$, with `position`$[t-1]$ is the position held on day $t-1$. So we have the following three cases:

- `position` = 1 (LK/SP) : strat_return$_t = r_t^{\text{KO}} - r_t^{\text{PEP}}$
- `position` = -1 (SK/LP) : strat_return$_t = -r_t^{\text{KO}} + r_t^{\text{PEP}}$
- `position` = 0 : strat_return$_t = 0$.

This assumes equal capital allocation to both legs of the trade.

Using the formulae above, we compute the daily returns of our strategy for each day based on the position held the day before. These returns are used to build the *cumulative performance* of the strategy, as well as to compute various metrics (in our case, total return and Sharpe Ratio), which are presented in the next section.

# 4 Performance

## 4.1 Equity curve

We first analyze the equity curve, which tracks the evolution of the cumulative returns over time. This curve is generated by compounding the daily strategy returns. It shows a hypothetical $1 investment would have grown if fully allocated to the strategy.

By plotting this curve, we can visually assess when the strategy performs well, when it stagnates, and whether it suffers significant drawdowns.

## 4.2 Sharpe Ratio

To evaluate the risk-adjusted performance of the strategy, we compute the *Sharpe Ratio*. Since we work with daily returns and assume zero risk-free, the Sharpe Ratio is computed as

$$\text{Sharpe Ratio} = \frac{\mathbb{E}[R_S]}{\text{std}(R_S)}$$

where $R_S$ is the strategy returns. A higher Sharpe Ratio indicates that the strategy produces more return per unit of risk.

## 4.3 Grid search for optimal parameters

To understand how different entry and exit thresholds affect the performance, we conduct a grid search over a range of thresholds pairs. For each combination, we compute the total return and Sharpe Ratio.

This produces a table of results, which we visualize as a heatmap (See Figure 4.1 below). The heatmap helps identify which combinations of parameters yield the best performance.

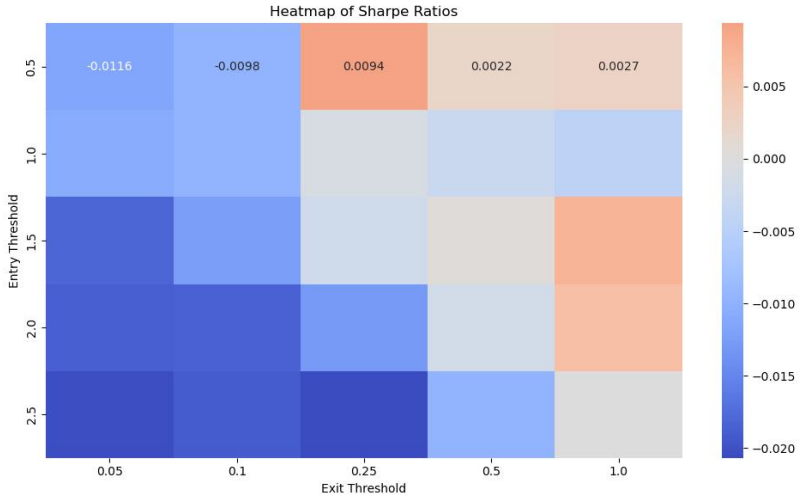


FIG. 4.1: Sharpe Ratios for different entry/exit threshold combinations

From Figure 4.1, we observe that the pair of parameters that maximize the Sharpe Ratio is the pair $[2.5, 0.05]$. We now plot the equity curve of the strategy using these optimal parameters:

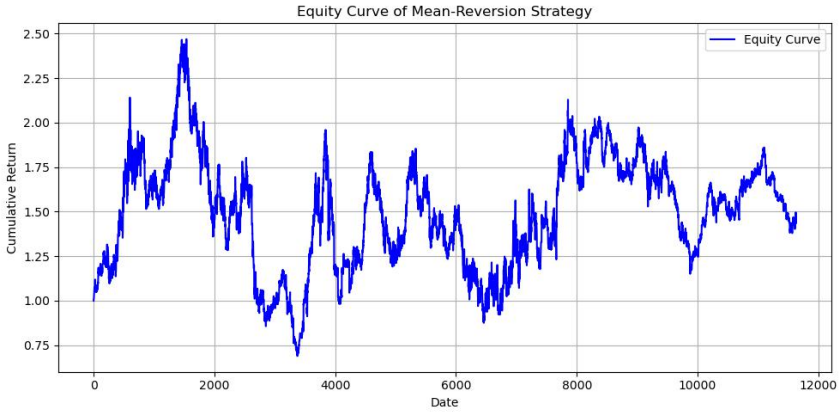


FIG. 4.2: Equity curve of the mean-reversion strategy using optimal parameters

Despite some fluctuations, we can see periods of strong performance with some drawdowns, highlighting the risks and limitations of a simple threshold-based strategy.

# 5 Discussion and Conclusion

In this project we have implemented a basic mean-reversion strategy using a pairs trading approach on Coca-Cola (KO) and PepsiCo (PEP). By testing for stationarity with the ADF test and confirming cointegration, we justified the use of a spread-based signal. A rolling $z$-score was used to identify trading opportunities when the spread diverged significantly from its historical mean.

We encoded a simple position management system and evaluated strategy returns under various entry and exit thresholds. The grid search over hyperparameters revealed that the most profitable setup, in terms of Sharpe Ratio, corresponded to a entry/exit threshold of $(2.5, 0.05)$. the corresponding equity curve demonstrates some profitability but also notable drawdowns and volatility.

While the strategy performs reasonably under certain conditions, several limitations should be acknowledged:

- Transaction costs and slippage[1] were not accounted for and would likely reduce profitability in a real-world setting.
- The strategy assumes instant execution and equal capital allocation, which may not hold in practice.

[1]Difference between the expected price of a trade and the price at which the trade is executed.

- The strategy is based on historical price behavior of only two stocks–further analysis with larger dynamic pair selection would be necessary for generalization.
- The thresholds were optimized on past data; without walk-forward testing or OOS validation, there is risk of overfitting.

# References

[1] Erik Hallmar. *Daily Historical Stock Prices (1970-2018)*. https://www.kaggle.com/datasets/ehallmar/daily-historical-stock-prices-1970-2018. Accessed July 30, 2025. 2018.