Cointegration Arbitrage: Minimum Profit Optimization Strategy

Juan Francisco Pérez Demšar

June 2024

1 Introduction

In the context of pairs trading, cointegration is used to exploit long term statistical relationships between the price of 2 assets.

Cointegration Arbitrage offers a more sophisticated approach to other strategies in statistical arbitrage, as it tries to profit from the stationarity of the spread between the 2 traded pairs.

This coding project is an overview of a cointegration based pairs trading strategy.

2 What's Cointegration?

The concept of correlation was first formalized by Robert Engle and Clive William John Granger in 1987.

To explain it simply, cointegration refers to a statistical relationship between two or more time series that might be correlated on the long term, despite being non-stationary individually (i.e. with changing variance)

Let X, Y be two separate time series.

If $X, Y \sim I(d)$ (Integrated of order d, a process is integrated to order d if taking repeated differences d times yields a stationary process), the collection (X, Y) is said to be cointegrated if:

$$\exists \alpha, \beta \text{ such that } Z = \alpha X + \beta Y \text{ with } Z \sim I(d-1)$$

In financial markets, we assume asset prices, interest rates, and yields integrated of order 1 $I \sim (1)$. Returns on the other hand are integrated of order 0 $I \sim (0)$, as they're obtained by differentiating the price.

Time series integrated of order 0 are weak-sense stationary, which means they're have a finite and time-invariant mean and standard deviation. These last two properties make these series more predictable and nicer to trade, however we cannot trade returns!

Here's where bringing the concept of cointegration in pairs trading becomes interesting.

3 Building the traded pair with cointegration

We are looking to build our "Z-Score". We want this constructed time series to be stationary as it is defined above.

Also we're looking to center the statistic at 0. Hence we want to build a Z of the following form:

$$Z = X_B - \beta X_A \quad \text{with } \beta \in \mathbb{R} \text{ and } Z \sim I(0)$$

β is the coefficient that allow the constructed statistic to be stationary. We don't know this number but we can estimate with an OLS regression.

In this case, we choose to regress Stock A on B such that:

$$X_B = \alpha + \beta X_B \quad \text{with } \beta, \alpha \in \mathbb{R}$$

for $\beta\&\alpha$ that solve the following minimization program:

$$\min_{\alpha,\beta} \sum_{i=1}^{n} (X_{b,i} - \alpha - \beta X_{a,i})^2$$

Having done the regression, we can deduce a formula for the Z-Score centered at 0 as:

$$Z = X_B - \beta X_A - \alpha$$

4 Determining upper and lower boundaries

This constructed Z-score will be bought or sold whenever it crosses a certain threshold U (symmetrical). We don't want to choose it arbitrarily thus we refer to minimum profit optimization.

Minimum profit optimization is a method used to determine the optimal threshold (U). The goal is to maximize the minimum profit by optimizing the intervals at which trading decisions are made.

We want to calculate the average time it takes for the Z-score to revert to 0 after crossing a threshold, and also the average time it takes for the Z-score to

go from 0 to a threshold.

When we take about "Time" we refer to the frequency of our data, so it might be days or hours depending on the user's choice.

$$Minimum\_Profit = U\left(\frac{T}{Avg\_Revert\_Time + Avg\_Cross\_Time} - 1\right)$$

To find the optimal U, we solve the following Maximization program.

$$Optimal\_U = \max_{U}\left(U\left(\frac{T}{Avg\_Revert\_Time + Avg\_Cross\_Time} - 1\right)\right)$$

5 Trading strategy

To recap, we have a Z-score we built as:

$$Z = X_B - \beta X_A - \alpha$$

and a boundary level U, which is symmetrical to the upside and downside.

The trading strategy consists of the following:

$$Z \geq U : SellB, Buy\beta A$$
$$Z \leq -U : BuyB, Sell\beta A$$

The position is held until the Z-score crosses 0 again.

The portfolio is self-financing, meaning that the proceeds from the shorted leg finance the long leg. For this is necessary to hold 150% of the value of the shorted leg at the time of the sale. Hence if we back-test the strategy with $1000 of portfolio value, we can only short $750.

6 Code implementation and Backtesting

I'll show the code implementation with an example pair of stocks.

I have chosen to test a pair 2 regional US Banks with less than $10B in total assets, TriCo Bancshares (TCBK) (Stock A) and Great Southern Bancorp (GSBC) (Stock B). We'll be looking at hourly data from May-December 2023 to "train" the model, and then we'll backtest the strategy with data from January-June 2024.

6.1 Downloading the data

We use yfinance to fetch the stock data.

Figure 1: Hourly Closing Prices of TCBK and GSBC

6.2 Linear regression

We use sci-kit learn to perform an OLS regression of stock A on Stock B (TCBK on GSBC). We use 100% of the data as training split.

[3]:
```
# Split data into features (X) and target (y)
X = data[['Stock_A']]
y = data['Stock_B']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=None, random_state=42)

# Create a linear regression model
model = LinearRegression()

# Fit the model to the training data
model.fit(X_train, y_train)

# Print model coefficients
print("Coefficient:", model.coef_)
print("Intercept:", model.intercept_)

#Store the results
b = model.coef_
a = model.intercept_
```

```
Coefficient: [0.91816706]
Intercept: 19.7211431481159
```

6.3 Building the Z-Score

We use the regression results from the previous section to calculate the Z-Score data series.

$$Z = X_B - \beta X_A - \alpha$$

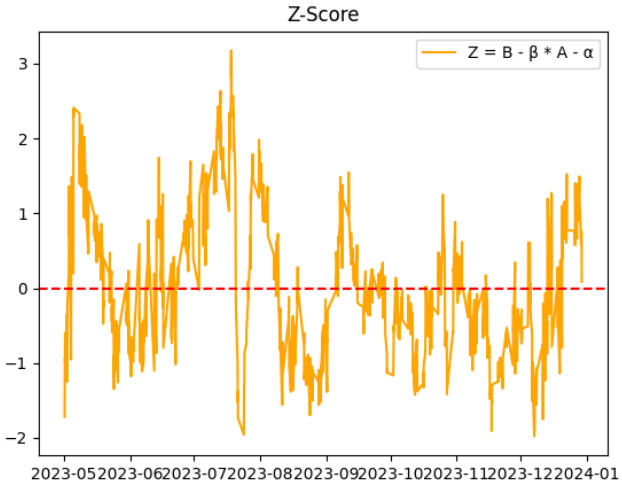

Figure 2: Z-Score build from Stock A and B

6.4 Testing for stationarity

We use the Augmented Dickey-Fuller Test to evaluate whether our Z-Score exhibits stationarity (we recall that a series integrated of order 0 is weak sense stationary). This test helps determine if there is a unit root in the time series data.

[5]:
```
adf = adfuller(Z)[1]

#Print the results
print("p-value is:", adfuller(Z)[1])
if adf>0.05:
    print('Non-Stationary process')
elif adf<0.05:
    print('Stationary process')
```

```
p-value is: 0.0005372664441636305
Stationary process
```

6.5 Determining the trading thresholds

We will determine the buying and selling boundaries given the minimum profit optimization approach defined above. You can check the "Functions.ipynb" notebook to see the implementation.

```
[6]: U = Minimum_Optimized_Profit(Z)
     print("Trading threshold is: ", U)
```

```
/home/juanfrancisco/Juan-Programaci
eprecated. In a future version, int
tion, use `ser.iloc[pos]`
  "    above_U_intervals = []\n",
Trading threshold is:  0.93
```

6.6 Backtesting the strategy

We have found a correlated pair of stocks. The constructed Z-Score exhibits stationarity over the training period as per the ADF test.

Next step is fetching data from January to June 2024, we'll use the values from the linear regression and the calculated boundaries.

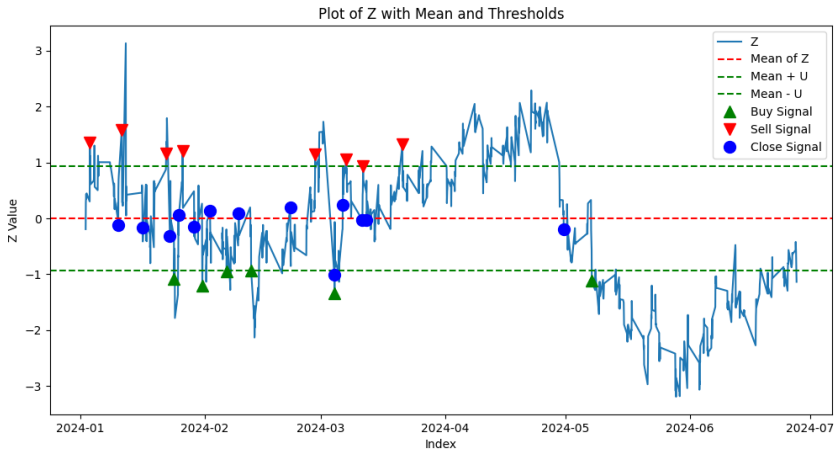

Figure 3: Generated trading signals

Finally with a margin requirement of 150% we get the following performance.

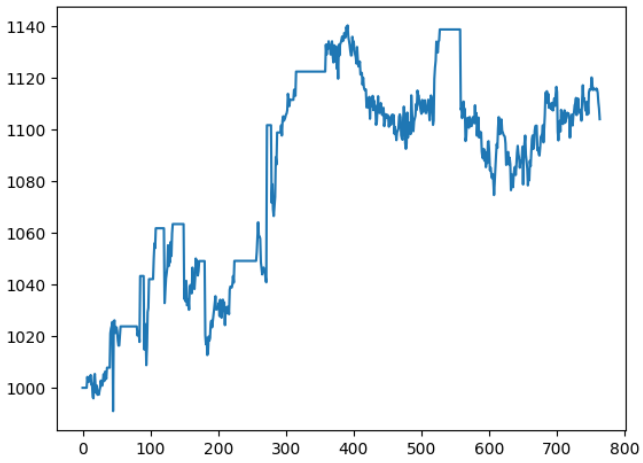

Figure 4: $1000 Portfolio value overtime

Metric	**Value**
Annualized Return	32.79%
Annualized Volatility	20.03%
Sharpe Ratio	1.43

Table 1: Performance Metrics