Smith SCHOOL OF BUSINESS Queens University | Master of Management Analytics Toronto

MMA 823
Analytics for Financial Markets

Jon Aikman | Alex Scott

Project Report
21st August 2022 | 11:59PM

Team Adelaide

Student Name	Student Number
Ishaan Sinha	20290167
Jiali Bai	20316342
Juefei Chen	20311141
Kripa Shanker Nayak	20315710
Oluwaseun Adelana	20310041
Yichen Yuan	20315046

Order of files:

Filename	Pages	Comments and/or Instructions
MMA 823 Project Report_Final	17	

Additional Comments:

Algorithmic Trading

1. Executive Summary

2022 is a critical year in which the imbalances wrought by the pandemic will likely begin to resolve, and the business cycle will normalize. Given today's near-record price/earnings multiples on double-digit profit forecasts on the S&P 500 Index, investors might be forgiven for thinking they could return to the successful portfolio strategies of the past, anchored to U.S. mega-capitalization growth companies that dominate passive indices. We seem to be in excesses and extremes – high inflation and rising interest rates resulting in immense market volatility. Organizations are looking for ways to mitigate market volatility and improve their investment returns. Rags to Riches Capital, one of the leading asset management firms, hired Team Adelaide to help their Treasury Department explore various investment strategies to maximize their investment portfolio return.

Different strategies, such as Growth investing, Value investing, Dividend Growth Investing, Algorithmic Trading etc., can be implemented to maximize investment returns. We did a feasibility analysis to evaluate each strategy and found Algorithmic Trading to be the best fit for our client's needs. For our model development, we've utilized publicly available stock data from yahoo finance. We've used S&P 500 stocks data within the technology sector for the past two years to develop and implement our investment strategy. Pairs Trading is the investment strategy we've decided to implement within Algorithmic Trading. We've looked at historical means and returns of various co-moving technology stocks and selected Microsoft and Accenture for the initial model. Post model development, we back-tested our strategy before deploying it in the production environment.

2. Introduction

2022: The great Rebalancing begins - as economic imbalances wrought by the pandemic begin to ease, investors are in for hotter-than-expected growth and inflation. The economic and market environment in 2022 is decidedly reflationary, with higher economic growth, higher inflation, and eventually higher real interest rates—a hotter and faster business cycle.

We see two significant trends that could further drive higher-than-expected growth and inflation, with more lavish capital spending and improving productivity:

- Innovation - During pandemic-related shutdowns, service businesses were forced to innovate digitally. This has spurred not only investment but an explosion in start-ups, as well as historic levels of public and private market activity—from fintech and cryptocurrencies to autonomous vehicles and artificial intelligence.
- Deglobalization: Businesses were already contemplating supply-chain localization amid U.S.-China trade tensions before the pandemic. Today's inflation-driving supply imbalances and inventory shortages—not to mention increasing sensitivity around cybersecurity, public health, geopolitics and shifting regulatory frameworks in China—have all added momentum to this trend toward domestic sourcing.

These trends suggest that investors need to position themselves to retool their investment portfolios, and the focus should be on increased tech adoption. Investment firms are looking for tech-enabled investment strategies such as Algorithmic trading to help them hedge market volatility and improve profitability by identifying arbitrage opportunities. As per our feasibility analysis – Algo trading is legal and utilized by fund managers across the globe. It is technology ready – we have the tools to develop and implement algorithms. It helps businesses maximize returns and can be deployed within a shorter period. Algo trading combines programming and financial market insights to execute trades on our behalf effectively. Common Algo trading strategies include trend following, fund rebalancing and arbitrage opportunities. Trades are also executed based on volume or passage of time. Sentiments and similar transactions drive markets; hence Algo trading removes the emotional aspect of trading. Since an algorithm is trading for us, it increases our trading activity based on the defined strategy and is quick at executing trades. Also, we can back-test our strategies before deploying them in a real-world environment. Within Algorithmic Trading, a pairs trade is a trading strategy that involves matching a long position with a short position in two stocks with a high correlation. Usually, when a pair trade performs as expected, the investor would profit from the transaction. However, it relies on a high statistical correlation between the variables. Also, the past price index cannot always indicate future trends. But in general, pairs trading can typically achieve profit through simple and low-risk positions.

3. Trading Strategy

The trading strategy we've selected is mean reversion which is used as a statistical analysis of market conditions. This strategy assumes asset prices and historical returns gradually move towards the long-term mean. The fundamental principle is that the greater the deviation from the mean higher the probability that the next movement would be closer to the mean. We can make profitable trade when there's a signification deviation from the historical correlation between two shares.

Pairs Trading is a subset of mean reversion. It's a type of statistical arbitrage which aims to profit from price convergence between two assets. It takes advantage of mispricing between two or more co-moving assets and follows the Fundamental principle of investing, i.e., to buy undervalued and sell overvalued.

4. Statistical and Technical Analysis

The first step of Pairs Trading is to find pairs of stocks that are highly correlated with each other. Among S&P 500 index, we narrowed the scope to the information technology (IT) sector. Some tickers under the IT sector are Apple, Salesforce, IBM, Microsoft Et cetera. We used the cointegration test to determine if there is a correlation between time series data in the long term, then the cointegration test returned a p-value for each pair of stocks that we tested. When the p-value is less than 0.05, it means two stocks are significantly cointegrated with each other.

	S1	S2	pvalue
100	MA	WDC	0.000381
6	ACN	MSFT	0.000808
75	MA	MU	0.000817
85	MA	PTC	0.001172
80	MA	NXPI	0.001569

Figure 4.1： Top 5 pairs with the lowest p-value on the cointegration test

After randomly selecting a pair of stock Accenture (ACN) and Microsoft (MSFT) that have a p-value of 0.0008, A stationary test for the selected pair is needed to ensure the mean and variance stays constant over time and prices would keep returning to their mean. Therefore, we performed an augmented Dickey-Fuller test (ADF) to test the stationary of the ratio and spread between selected pairs, and it returned a p-value of 0.003 and 0.0002,

respectively. The tests indicated that the stock ACN and MSFT are highly correlated, and their relationship is stationary.

Figure 4.2

Accenture is a professional services provider specializing in IT consulting, and Microsoft is a world giant that produces computer software and consumer electronics. According to Avanade's official website[1], there has been a very solid partnership between Accenture and Microsoft since 2000, and they founded the joint venture Avanade to combine the best in strategy and technology and help clients unlock more value from IT. Figure 4.2 indicates that the two stocks had similar movement patterns in the past two years; therefore, we are confident in moving forward with these two stocks.

5. Modelling

Next, we needed to find the thresholds for the price to trigger trading. The presumption for pairs trading is that the price ratio would always go back to its mean, we first defined the price ratio as $X_t = S_t^1 / S_t^2$, and the z-score of X_t is defined as

$$Z_t = \frac{X_t - \mu_X}{\sigma(X)}$$

Technical indicators such as moving average, Bollinger bands, and standard deviation could apply to adjust the trading strategy for the market volatility. We started by using a

[1] https://www.avanade.com/en-ca/about-avanade/partnerships/accenture-avanade-microsoft-alliance

60-day moving average as the mean and a 5-day moving average as a raw score to compute a z-score ((5d MA - 60d MA) / 60d SD).

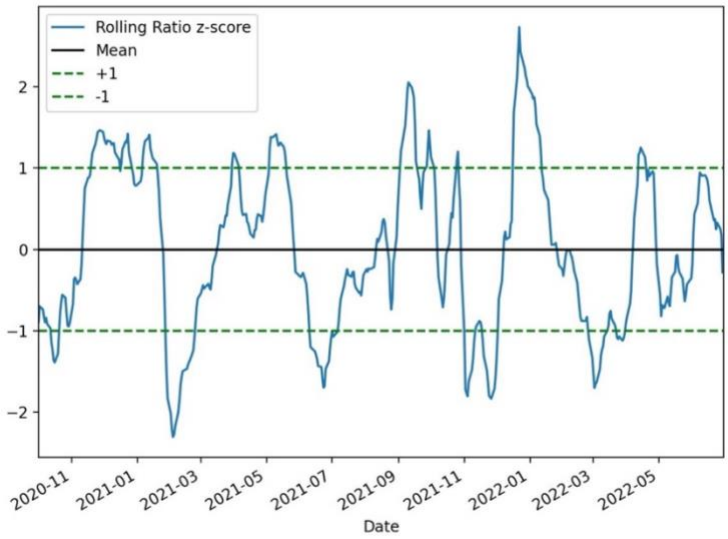


Figure 5.1

A trading opportunity is created when the z scores hit the top or bottom Bollinger band. Then we implemented the buy and sell trading strategy of shorting the outperforming stock and longing the underperforming stock.

- When $Z_t > 1$, ACN outperforms $\Longrightarrow$ short ACN and long $MSFT$
- When $Z_t < -1$, $MSFT$ outperforms $\Longrightarrow$ long ACN and short $MSFT$

We set up some conditions to reduce risk to stop trading and exit positions. If the absolute value of the z-score is lower than 0.5, we will exit the positions until the model triggers the next opportunity. In the case of market fluctuation or relationship breakage, we also set up a stop-loss threshold that if the daily loss reached 8% of our account size, we would exit the positions.

6. Optimizing the strategy

In our approach, we used the historically observed ratio between Accenture and Microsoft stocks as the mean to apply the mean reversion strategy. We used a specified window length to calculate the moving average to find this ratio. We used the computed z-scores

to execute our buying and selling decisions. Hence, using the right window length at this stage is very important.

The window length determines how well the model fits the data. Choosing a window length that is too long can result in overfitting, which will cause the model to be more representative of the sample data rather than the actual values. On the other hand, choosing a window length that is too short could result in underfitting. This would mean that the model does not entirely represent the historically observed ratio. In either case, our model is unreliable, so we will not be able to trust our backtesting results to be an accurate estimate of future performance.

To optimize the window length being used, we performed hyperparameter tuning. Initially, we split the data into training and testing sets. The most recent 20% of the observations were included in the test set, while all other previous observations were included in the train set. We then decided to vary the window length between 1-200 days and ran all possible values. Finally, we chose the window where the highest profits were observed as the optimum length. From this, we found that the best window length for calculating the ratio between the two stock values is 56 days.

7. Backtesting the Trading Strategy

Post our model development, we did back testing to evaluate our model performance and test our trading strategy. For this, we simulated our trading model over the data we have had over two years. In the simulation, we started with $10,000 cash and did not take any positions on any of the two stocks, i.e., Accenture and Microsoft; hence our holding value until is nil. Our trading strategy also includes a stop loss criterion: exit all positions if the overall loss exceeds 8% of our cash.

Additionally, each time we encounter a buy or sell signal, we can trade 24 shares of Accenture and 29 shares of Microsoft. We can trade more shares of Microsoft because the maximum observed stock price of Accenture is higher than that of Microsoft. We are not taking any action in the simulation for the first 56 days. Then, we start making trades from the next day onwards. The profit or loss for any given day in the simulation is calculated using the stock prices and the total number of stocks we have longed or shorted at that given point.

Let's look at a small time frame within the 2-year simulation period to see how the trades are taking place, as shown in figure 7.1. On September 18, 2020, we took a short position on Accenture, resulting in a negative holding value of $5,526.67. We simultaneously took a long position on Microsoft, resulting in a positive holding value of $5,722.76. This makes the net holding value on this day $196.09, and the amount of cash remaining with us after these trades is $9803.91. The portfolio value remains at $10,000 as we have not made any profit or losses yet, but we have only lent and borrowed the stocks. Gain or loss is dependent on the fluctuations in the stock price only. For instance, on September 21 and 22, we continue to hold our position, i.e., do not make any trades, and our cash value remains the same. But we observed that the portfolio value is increasing on both these days and becomes $10,114.51 after the 21st and $10,183.44 after the 22nd based on the stock price volatility.

	Accenture (ACN)			Microsoft (MSFT)				
Date	**Price**	**Position**	**Holding**	**Price**	**Position**	**Holding**	**Cash**	**Portfolio Value**
2020-07-01	$208.46	-	$0.00	$201.09	-	$0.00	$10,00.00	$10,000.00
		…			…			
2020-09-18	$230.27	Short	($5,526.67)	$197.33	Long	$5,722.76	$9,803.91	$ 10,000.00
2020-09-21	$228.06	Hold	($5,473.55)	$199.45	Hold	$5,784.16	$9,803.91	$ 10,114.51
2020-09-22	$230.99	Hold	($5,543.99)	$204.25	Hold	$5,923.52	$9,803.91	$ 10,183.44
		…			…			

Figure 7.1: Trade simulation for the period Sept 18-22

Figure 7.2 details the implementation of our buy low and sell high strategy – this is a graph of the stock price of ACN & MSFT shares over the trading period of 2 years. The green trend line represents ACN while the blue represents MSFT, the purple marker signifies buying while the black signals sell.

Figure 7.3 shows our net portfolio value with time over the entire simulation. The markers used for the buy and sell signals are the same as in the previous figure.

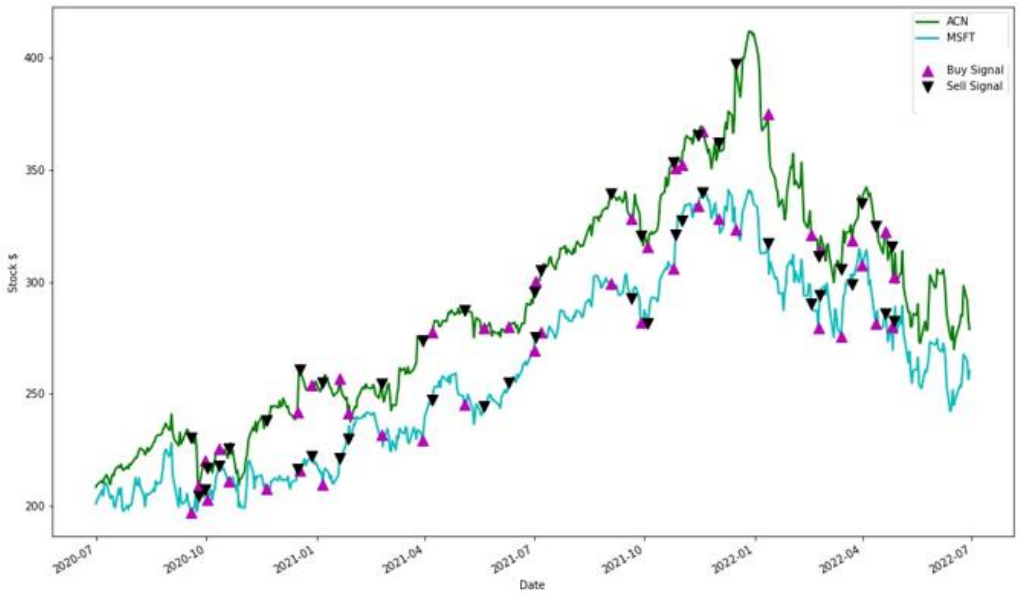


Figure 7.2: Trades over the 2-year simulation

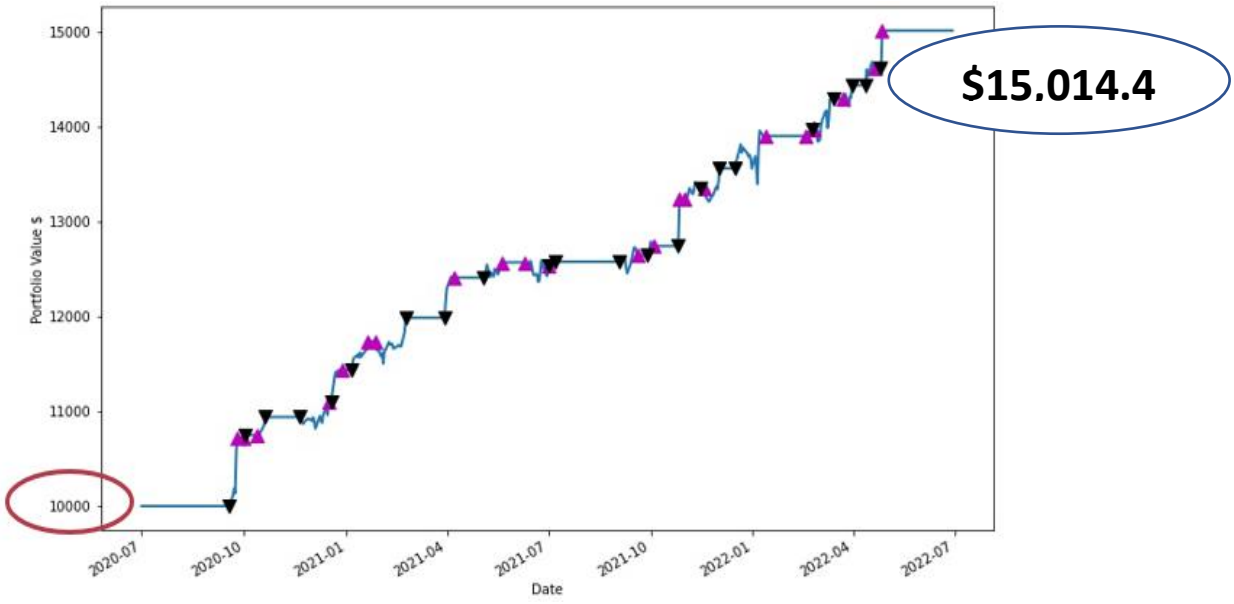


Figure 7.3: Net Portfolio Value

The results showed that over the entire trading simulation, the strategy managed to make an overall profit of $5,015. We are trading a total number of 44 times throughout this period. This means we are making many trades and have a hedged position most of the time, which means that this strategy works well if you are taking more risk and investing more in the stocks. But we can see that our net cash value in hand always remains close to $10,000, and so our model is not making the full use of our resources. Despite this

drawback, the model performs well, as can be seen by the portfolio value, which shows an increase of over 50% within the two years. Furthermore, we can say that the strategy is protected from the overall market movement as the portfolio value shows an always-increasing trend throughout the testing time. There is not a single point where a significant drop is observed in the net portfolio value.

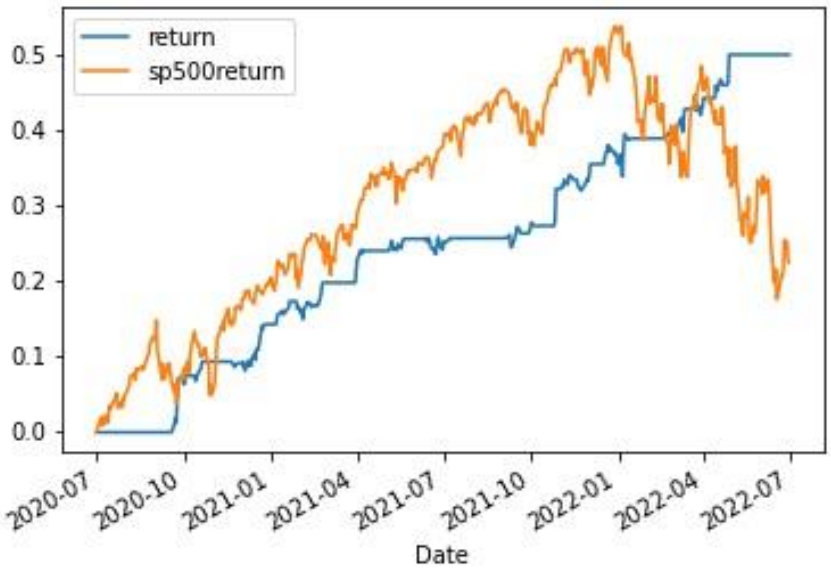


Figure 7.3: Strategy returns vs. S&P500 Index

Figure 7.3 compares our returns with that of the S&P 500 index. We can see how relatively close they perform until the vast variation between Apr 2021 and Jan 2022. The S&P 500 suffers a decline and makes a total return of about 25% at the end of the two years, while our model return consistently moves upwards until it gains an overall return close to 50%.

8. Next Steps

As part of our next steps, we plan to:

- Enhance our model by adding additional parameter tuning and including Machine Learning algorithms, such as decision trees or Support Vector Machines. We will also try to utilize unsupervised learning strategies such as clustering to develop more intelligent ways to identify stock pairs to trade.
- Implement additional trading strategies such as trend following and arbitrage

opportunities and include other industries from the S&P 500 within our scope as we are currently only limited to the technology sector.

- Carry out additional backtesting before full deployment. In future backtesting, we will include transactional costs and will also try to minimize the amount of cash we have in hand and, in turn, maximize our net holding value.

Appendix

1. Loading Data

Get a list of tech stocks from the S&P 500 companies

In [2]:
```
payload=pd.read_html('https://en.wikipedia.org/wiki/List_of_S%26P_500_companies')
first_table = payload[0]
df = first_table[first_table['GICS Sector']=='Information Technology']
tickers = df['Symbol'].values.tolist()
tickers
```

In [3]:
```
start_date = '2020-07-01'
end_date = '2022-06-30'
final_dataframe = pd.DataFrame()

for symbol in tickers:
    stock_daily=yf.download(symbol, start_date, end_date)
    stock_daily.insert(0,'Ticker', symbol)
    final_dataframe = pd.concat([final_dataframe, stock_daily])

final_dataframe.reset_index(level=0, inplace=True)
final_dataframe = final_dataframe.rename({'index': 'Date'}, axis=1)
```

Find pairs

In [6]:
```
from statsmodels.tsa.stattools import coint
import seaborn
```

In [7]:
```
def find_cointegrated_pairs(data):
    n = data.shape[1]
    pvalue_matrix = np.ones((n, n))
    keys = data.keys()
    pairs = {"S1":[], 'S2':[], 'pvalue':[]}

    for i in range(n):
        for j in range(i+1, n):
            S1 = data[keys[i]]
            S2 = data[keys[j]]
            result = coint(S1, S2)
            pvalue = result[1]
            pvalue_matrix[i, j] = pvalue
            if pvalue < 0.05:
                pairs['S1'].append(keys[i])
                pairs['S2'].append(keys[j])
                pairs['pvalue'].append(pvalue)

    return pvalue_matrix, pairs
```

p value for each pair of stocks

In [9]:
```
names = pt_df.keys().to_list()
df1 = pd.DataFrame(pairs[0], index=names, columns=names)
df1.head()
```

	AAPL	ACN	ADBE	ADI	ADP	ADSK	AKAM	AMAT	AMD	ANET	...	SWKS	TEL	TER
AAPL	1.0	0.091503	0.498870	0.303913	0.144977	0.330460	0.411365	0.481903	0.398258	0.006581	...	0.334309	0.419262	0.475464
ACN	1.0	1.000000	0.834347	0.307908	0.649731	0.741014	0.680705	0.585589	0.312760	0.371510	...	0.743453	0.295625	0.623457
ADBE	1.0	1.000000	1.000000	0.921077	0.923165	0.542329	0.533237	0.852066	0.852053	0.902882	...	0.610590	0.765666	0.817475
ADI	1.0	1.000000	1.000000	1.000000	0.436066	0.668932	0.538788	0.303056	0.659534	0.618293	...	0.696030	0.052142	0.538613
ADP	1.0	1.000000	1.000000	1.000000	1.000000	0.808753	0.815845	0.795721	0.815622	0.380619	...	0.803928	0.870212	0.889236

5 rows × 74 columns

Find top pairs with the lowest pvalue

In [10]:
```
df2 = pd.DataFrame.from_dict(pairs[1]).sort_values(by=['pvalue'])
df2.head()
```

	S1	S2	pvalue
100	MA	WDC	0.000381
6	ACN	MSFT	0.000808
75	MA	MU	0.000817
85	MA	PTC	0.001172
80	MA	NXPI	0.001569

Example

('ACN', 'MSFT')

In [11]:
```
from statsmodels.tsa.stattools import adfuller
```

Check for stationarity

In [12]:
```
S1 = pt_df['ACN']
S2 = pt_df['MSFT']
```

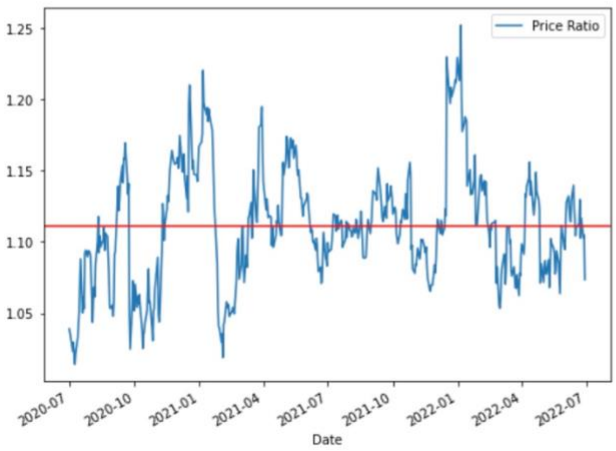
In [13]:
```
S1_ADF = adfuller(S1)
print('P value for the Augmented Dickey-Fuller Test is', S1_ADF[1])
S2_ADF = adfuller(S2)
print('P value for the Augmented Dickey-Fuller Test is', S2_ADF[1])
Spread_ADF = adfuller(S1 - S2)
print('P value for the Augmented Dickey-Fuller Test is', Spread_ADF[1])
Ratio_ADF = adfuller(S1 / S2)
print('P value for the Augmented Dickey-Fuller Test is', Ratio_ADF[1])
```

```
P value for the Augmented Dickey-Fuller Test is 0.43062722058486186
P value for the Augmented Dickey-Fuller Test is 0.5041410553894818
P value for the Augmented Dickey-Fuller Test is 0.003282069687849283
P value for the Augmented Dickey-Fuller Test is 0.00016499290297839346
```

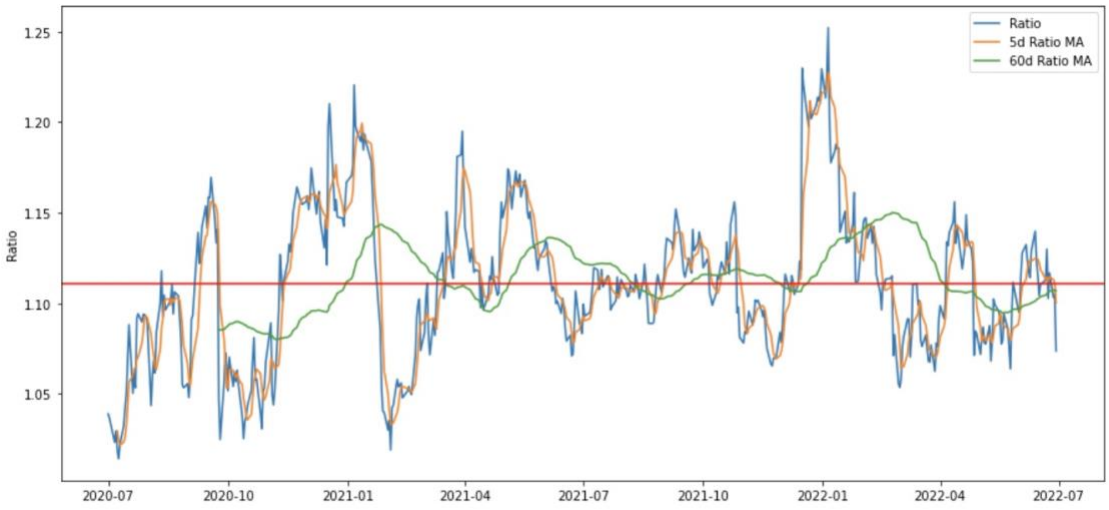
ACN/MSFT Ratio

In [17]:

```
ratio = S1/S2
ratio.plot(figsize=(8,6))
plt.axhline(ratio.mean(), color='red')
plt.legend(['Price Ratio'])
#plt.show()
plt.savefig('PriceRatio.jpg')
```



Two stocks tend to move around together around the mean



Model Tuning

```
test_size = 0.2
train_size = int(len(S1) * (1-test_size))
S1_train, S1_test = S1[0:train_size], S1[train_size:len(S1)]
S2_train, S2_test = S2[0:train_size], S2[train_size:len(S1)]
```

In [26]:

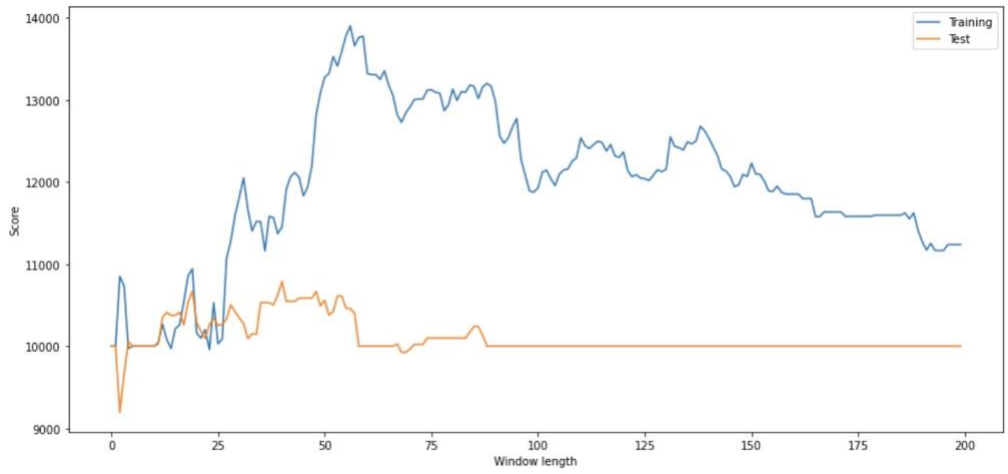
```
# Find the window length 0-200
# that gives the highest returns using this strategy
length_scores = [test(S1_train,
                      S2_train, 5, l)
                      for l in range(200)]
best_length = np.argmax(length_scores)
print ('Best window length:', best_length)
```

```
Best window length: 56
```

In [27]:

```
# Find the returns for test data
# using what we think is the best window length
length_scores2 = [test(S1_test,
                       S2_test, 5, l)
                       for l in range(200)]
print (best_length, 'day window:', length_scores2[best_length])
# Find the best window length based on this dataset,
# and the returns using this window length
best_length2 = np.argmax(length_scores2)
print (best_length2, 'day window:', length_scores2[best_length2])
```

```
56 day window: 10459.954467773438
40 day window: 10787.050277709961
```



In [35]:

```
portfolio_df = pd.DataFrame.from_dict(portfolio_).set_index(S1.index)
```

In [36]:

```
portfolio_df.insert(0,'S2', S2)
portfolio_df.insert(0,'S1', S1)
portfolio_df.insert(4,"Holding2", portfolio_df['Position2'].cumsum()*portfolio_df['S2']*size2)
portfolio_df.insert(4,"Holding1", portfolio_df['Position1'].cumsum()*portfolio_df['S1']*size1)
portfolio_df['TotalAsset'] = portfolio_df['Holding1'] + portfolio_df['Holding2'] + portfolio_df
```

Backtesting

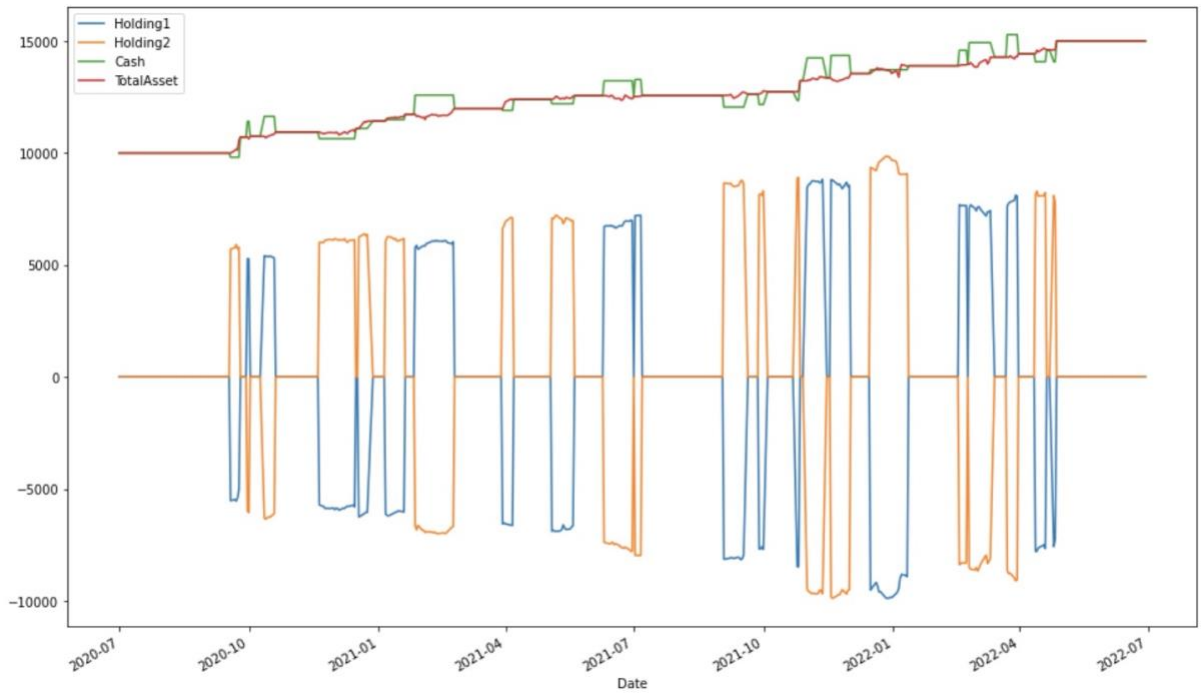
In [33]:

```
def trade(S1, S2, window1, window2):
    # Compute rolling mean and rolling standard deviation
    ratios = S1/S2
    ma1 = ratios.rolling(window=window1, center=False).mean()
    ma2 = ratios.rolling(window=window2, center=False).mean()
    std = ratios.rolling(window=window2, center=False).std()
    zscore = (ma1 - ma2)/std

    # Simulate trading
    # Start with $10,000 and no positions
    cash = float(10000)
    signal1 = 0
    signal2 = 0
    position1 = 0
    position2 = 0
    signals1 = np.array([])
    signals2 = np.array([])
    date = S1.index
    size1 = cash//max(S1)
    size2 = cash//max(S2)
    stop_loss = cash * 0.08 * -1
    portfolio = {"Position1":[], "Position2":[], "Cash":[]}

    for i in range(len(ratios)):
        # Sell short if the z-score is > 1
        if zscore[i] > 1:
            signal1 = -1
        # Buy long if the z-score is < 1
        elif zscore[i] < -1:
            signal1 = 1
        else:
            signal1 = 0

        signal2 = -signal1
        signals1 = np.append(signals1, signal1)
        signals2 = np.append(signals2, signal2)
        position1 = signals1[i] - signals1[i-1]
        position2 = signals2[i] - signals2[i-1]
        pnl = - position1*S1[i]*size1 - position2*S2[i]*size2
        if pnl < stop_loss or abs(zscore[i]) < 0.5:
            signals1[i] = 0
            signals2[i] = 0
            position1 = signals1[i] - signals1[i-1]
            position2 = signals2[i] - signals2[i-1]
            cash -= position1*S1[i]*size1 + position2*S2[i]*size2
            print("stop-loss triggered")
        else:
            cash += pnl

        portfolio["Position1"].append(position1)
        portfolio["Position2"].append(position2)
        portfolio["Cash"].append(cash)
        print(f"Date {date[i]}, Position 1 {position1}, Position 2 {position2}, PNL {pnl}, Cash

    return portfolio
```

Portfolio

In [36]:

```
portfolio_df.insert(0,'S2', S2)
portfolio_df.insert(0,'S1', S1)
portfolio_df.insert(4,"Holding2", portfolio_df['Position2'].cumsum()*portfolio_df['S2']*size2)
portfolio_df.insert(4,"Holding1", portfolio_df['Position1'].cumsum()*portfolio_df['S1']*size1)
portfolio_df['TotalAsset'] = portfolio_df['Holding1'] + portfolio_df['Holding2'] + portfolio_df
```




```
portfolio_df['sp500'] = sp500_df['Adj Close']
portfolio_df['return'] = (portfolio_df['TotalAsset'] - portfolio_df['TotalAsset'][0])/portfolio
portfolio_df['sp500return'] = (portfolio_df['sp500']-portfolio_df['sp500'][0])/portfolio_df['sp
```