**Abstract**

Signal Frontier Analysis is used to evaluate how well a portfolio of stocks will perform given certain metrics. Forecasting how stocks will perform in the future is important for determining when to buy or sell investments in order to make as much profit as possible in a given period of time. Using Python, we will show the piece’s need to perform Signal Frontier Analysis including the concepts of momentum, holding periods, lookback periods, and how to use the Sharpe ratio. All data manipulation and analysis will be performed using Python 3.

**Introduction**

Time series analysis for stock prices is analyzing the prices of stocks in chronological order. Today, many different algorithms are deployed to predict how the price of stocks will change in what is called time series forecasting. One such method for analyzing the value of a portfolio of stocks is signal frontier analysis, which utilizes momentum and Sharpe’s ratio.

Momentum is the idea that stocks that have been known to historically increase will perform well in the future. The reverse is also true where stocks that have historically had the tendency to fall, will continue to have poor performances and decrease in value more according to John Verostek’s PowerPoint slide presentation on Signal Frontier Analysis. For this paper, we will utilize cross sectional momentum. To do so, we need to create a portfolio of stocks. We will create this portfolio using historical data of a particular stock and compare it to other stocks historical data at the same snapshot in time.

The Sharpe Ratio is a measurement for calculating the risk adjustment return. This means the Sharpe ratio calculates how risky an investment, such as one in a stock, will be. It is calculated by dividing the difference between the average portfolio return and the risk-free return by the standard deviation of the return on an investment.

We are using historical data for seven stocks. IBM, Texas Instruments, Cisco Systems Inc., Qualcomm Inc., Oracle Corporation, Nokia, and Applied Materials Inc. The data we are using is each of the stocks end of day price for each day the market was open. Our data starts at January 1st, 2009 and ends at June 1st, 2012. All of our analysis will be performed using the computer programming language Python.

**Methods**

Our first step is to collect historical data for each of the stocks. This is usually done by using an API such as Yahoo Finance, but today we are using a dataset containing historical time series data on 7 stocks. The data for each stock consists of its closing stock price for each day the market was open from our start date of January 1st, 2009 to our end date of June 1st, 2012.

To plot the cumulative returns for each stock we first, check for any missing data. If there is any then we can impute data in. Then, we create ‘rets’ which is the percent change between each data point. As in, this is the change in the price of the stock between two days as a percentage. We then perform a calculation of the percentage change plus 1 of the cumulative stock price and then subtract it by one.

We have to create a function in order to calculate momentum over a lookback period. We use the stock price, lookback, and lag as inputs. First the momentum is calculated over the inputted lookback period. The lookback is need so the percent change can be calculated from the price shift over the lag. Then, the different momentums calculated are ranked in descending order. After being ranked, our new function standardizes the data by subtracting it by the mean. Finally, we normalize the data by dividing it by its standard deviation. Therefore, our new function calc\_mom will output the momentum given a price, lookback, and lag period.

A lookback period is the length of time that returns are lagged, and the holding period is the number of months that we have had the portfolio since it has been created. A short lookback period, about 1 month, we expect an outcome of reversals. Reversal is volatility where a stock’s price could have increased one week and then decreased the next. An intermediate lookback period, 6 to 12 months, we expect to see more of a trend emerge such as a gradual rise in the stock price or a gradual decline. Long term outlooks, which are 3 to 5-year windows, can have an outcome of either an inversion or a reversal.

Now that momentum can be calculated with our new function calc\_mom, we can get use the momentum calculation towards a Sharpe ratio. Our next function ‘strat\_str’ computes the portfolio for specific lookback and holding periods and will output the Sharpe ratio of my portfolio. For input, the prices, lookback period, and holding periods desired. The frequencies are the modulus of the ‘%dB’ divided by the holding period. The portfolio is calculated using our momentum calculating equation using the prices as the prices input, the lb as the lookback input, and have lag=1. The daily\_rets is the daily returns of the stock from our historical data. It’s calculated by the percent change in each day’s end price. Now the function will calculate the returns of the given portfolio. It does this using the portfolio shifted once, and the daily returns multiplied together and then summed.

**Results**

This graph shows the cumulative returns for each of the stocks in our sample. According to the graph, at the end of our given time period, IBM is the stock with the highest value followed by Texas Instrument. Even though the graph shows that at the end of our period the stock prices of IBM and Texas Instruments are falling, along with many others, this should not be a concern because if we look at July 2009, 2010, and 2011 the cumulative returns of the stocks always seems to fall before they start to rise sharply in the fall. This is due to the effect of seasonality, so we can assume that in the Fall of 2012 the cumulative returns should rise. Nokia is the consistent worst performer amongst our sample of stocks over the given time period.



The way to read the heat map is the darker the color on the map the higher rate of return. Our axis are the lookback period and the holding period. According to the heat map of our data, a lookback period of 35 days along with a holding period of 30 days will provide the highest Sharpe ratio. This could be deemed a short lookback period. Though it may not be as optimal if you wanted longer lookback and/or holding periods it would be beneficial to have a lookback period of 55 days and a holding period of 55 days. Long lookback periods of three to five years can either result in inversion or reversal. According to our data, it would not be beneficial to have a portfolio with a long lookback period, but instead a short term one.



**Future Work / Discussion / Conclusion**

In future work I would like to look at how my time series looks under different lags such as the AR(1), AR(2), or ARIMA models. These autoregressive models are other alternatives to time series analysis that deal with lags that I believe would be beneficial to compare to. I also would like to do further research into the Sharpe ratio and its uses. Finally, I would like to see a heat map of the data for a time covering longer than an intermediate lookback period and see the heat map comparing the holding period to a long lookback period.

Appendix A

Bibliography

Baz, Jamil and Granger, Nicolas and Harvey, Campbell R. and Le Roux, Nicolas and Rattray, Sandy, Dissecting Investment Strategies in the Cross Section and Time Series (December 4, 2015). Available at SSRN: <https://ssrn.com/abstract=2695101> or [http://dx.doi.org/10.2139/ssrn.2695101](https://dx.doi.org/10.2139/ssrn.2695101)

Moskowitz, Tobias J., Y. Ooi, L. Pedersen, *Time Series Momentum*. Journal of Financial Economics. Elsevier. May 2012. <https://www.sciencedirect.com/science/article/pii/S0304405X11002613>

All code is curtesy of Celia Taylor and Professor Slater except the import statement.

Appendix B

import datetime

import pandas as pd

start\_dt=datetime.datetime(2009,1,1)

end\_dt=datetime.datetime(2012,6,1)

from pandas\_datareader import data as web

stock='IBM'

#px=pd.read\_csv('/Users/Phillip/Downloads/casestudy4/example.csv') #index='Date'

px=pd.DataFrame.from\_csv('/Users/Phillip/Downloads/casestudy4/case\_study.csv')

names=['TXN','CSCO', 'QCOM', 'ORCL', 'NOK', 'AMAT']

px.head()

for stock in names:

while True:

try:

px[stock] #=web.get\_data\_yahoo(stock, start\_dt, end\_dt)['Adj Close']

break

except:

print('Unable to read stock: {0}'.format(stock))

print('trying again')

px['AMAT'].head()

from \_\_future\_\_ import division

from pandas import Series, DataFrame

import pandas as pd

from numpy.random import randn

import numpy as np

pd.options.display.max\_rows = 12

np.set\_printoptions(precision=4, suppress=True)

import matplotlib.pyplot as plt

plt.rc('figure', figsize=(12, 6))

plt.close('all')

px = px.asfreq('B').fillna(method='pad')

rets = px.pct\_change()

((1 + rets).cumprod() - 1).plot()

def calc\_mom(price, lookback, lag):

mom\_ret = price.shift(lag).pct\_change(lookback)

ranks = mom\_ret.rank(axis=1, ascending=False)

demeaned = ranks.subtract(ranks.mean(axis=1), axis=0)

return demeaned.divide(demeaned.std(axis=1), axis=0)

compound = lambda x : (1 + x).prod() - 1

daily\_sr = lambda x: x.mean() / x.std()

def strat\_sr(prices, lb, hold):

# Compute portfolio weights

freq = '%dB' % hold

port = calc\_mom(prices, lb, lag=1)

daily\_rets = prices.pct\_change()

# Compute portfolio returns

port = port.shift(1).resample(freq, how='first')

returns = daily\_rets.resample(freq, how=compound)

port\_rets = (port \* returns).sum(axis=1)

return daily\_sr(port\_rets) \* np.sqrt(252 / hold)

strat\_sr(px, 70, 30)

from collections import defaultdict

lookbacks = range(20, 90, 5)

holdings = range(20, 90, 5)

dd = defaultdict(dict)

for lb in lookbacks:

for hold in holdings:

dd[lb][hold] = strat\_sr(px, lb, hold)

ddf = DataFrame(dd)

ddf.index.name = 'Holding Period'

ddf.columns.name = 'Lookback Period'

import matplotlib.pyplot as plt

def heatmap(df, cmap=plt.cm.gray\_r):

fig = plt.figure()

ax = fig.add\_subplot(111)

axim = ax.imshow(df.values, cmap=cmap, interpolation='nearest')

ax.set\_xlabel(df.columns.name)

ax.set\_xticks(np.arange(len(df.columns)))

ax.set\_xticklabels(list(df.columns))

ax.set\_ylabel(df.index.name)

ax.set\_yticks(np.arange(len(df.index)))

ax.set\_yticklabels(list(df.index))

plt.colorbar(axim)

heatmap(ddf)

plt.show()