**Systematic Trading Method with Fundamental Ranking and Optimization**

Trading stocks and other kind of securities is a very popular investing activity in the modern world. It can be a very rewarding opportunity that offers a substantial amount of return in a timely manner, though it may also serve as a ticking bomb in which it cannot far be distinguished from gambling, if not exercised wisely. Experience is the most precious gem that shapes one as a trader, and in time, either consciously or not, most develop strategies that they apply in hope that with discipline, one will be able to maximize returns without jeopardizing more than they should, while having a good control on how much they are willing to lose. In this paper, I will discuss a strategy, an algorithm I have developed and tested that assess many of the most crucial challenges in trading, which focuses on maximizing return while minimizing risk at the same time. The algorithm will also aid one in picking the best securities, as well as making trading decisions at the right timing. The result I received in the test I performed for this particular research was exceptional and past my expectations, it shows that with the right set up and decision making, this strategy can be successful more often than not.

**Introduction and Problem Statement**

What is an investment? The very basic idea of an investment is very simple, you spend what you can on something with hope that in time, you will get whatever you invest back, with an additional return, which we identify as profit. This general idea applies to all kinds of investment, whether it is real estate, stock trading, forex, or even starting a business, putting in time and money, can be viewed as a form of investment. Now, as we all know, almost all kind of investments involves risk in one way or another, no matter how small the probability is, there is almost always a chance that you may lose a portion or all of what you invest, with the exception of a handful of risk-free investment. Most of those risk-free

investment doesn’t offer significant return that can realistically be used to build wealth. On the other hand, stock trading offers one of the best, fastest return you can possibly get over most if not all investments that exists in the world, with a risk that is very much outweighed by its potential.

Stock trading have been a proven way to build wealth, legendary trader such as Rakesh Jhunjhunwala from India has a net worth of around $3 billion dollars, and he built it from scratch, starting from being a very good trader, and then an investor. Some may counter this intuition by saying that the stock market crashes and that you can lose money faster than you can gain. The idea in its essence is not wrong, but what causes people lose money in the stock market is not how the stock market behave itself, but more to how they use the money in the market and their strategies. How do I know this? The easiest way to answer this is by looking at the stock index. In the United States, they are the S&P 500 and NASDAQ. It is true that in times of a crisis, such as recently, the trade war, the market as a whole can crash and lose a significant amount of its value, but at the end of the day, as long as the government is pushing for growth in economy, their value always go back up and continue their momentum pushing new all-time highs not very long after the crash. This has been proven time and time again, as the S&P 500 and NASDAQ, just last week, fully recovered from the trade war crash that happened at the end of 2018, and now back in their all-time high once again with nothing to stop it. If you put your money in just the index long enough at any time in history and never pull it out, you would never have lost money. So why do a lot of people lose money in the first place? Other than that, most people do not invest in indexes, but in individual stocks themselves, the most general answer is that because of the lack of quality decision making, which I can explain with the following analogy.

Imagine being in a casino with $1000 and you are currently at $1300 with $300 in winnings, would you continue gambling, or would you be satisfied with a 30% profit? How about if we put it the other way, you are currently at $500, with $500 in losses, would you take your loss and walk away? Or would you continue gambling with hopes that you will eventually get back to $1000 and then, what? Gamble more to hopefully win and profit, which can certainly go the other way and end up to you losing money again, or walk away with your $1000, which most people will certainly not do because obviously, it makes your trip to the casino worthless and a waste of time. My point is, there is no certainty on when to walk away, or even when to gamble, and with the house having theoretically an unlimited amount of cash in hand, it is more than likely for you to eventually end up losing all of your money. This phenomenon applies in the stock market, especially to inexperienced traders, even if they are really good analysts. They buy a stock which based on their analysis will increase in value, but the problem here is, their analysis stops there without certainty on the other factors. When the price of the stock goes up, they will think that it will go even higher and hold on to the stock, and on the other hand if it goes down, they will tank, believing that their analysis is right and that the price will eventually rally, and hold until the stock goes back up. Which brings us to the next section to further discuss the problem and the solution I propose.

I hope that it has become obvious that even though you can perform a very good analysis and profit from a stock, you also have to be a good trader because being able to profit from a single stock is not enough. The idea is not only to make a profit but to be able to consistently make profits and hold on to your winnings, which are two completely different things to perform. To do that, I have to create some kind of consistency in the way I trade, I have to create a system with principles that I have to follow in order to consistently profit in not from individual trades, but for the entire portfolio. When I first recognized this and faced the problem myself, I created a simple system that consists of a single principle that I vow myself to always follow, and it is very simple. For every single stock or option, I buy, I set a target selling price and a cut loss price, and to always sell the security whenever one of the limits is reached. By doing this to every single security that I buy, I have eventually been able to slowly but surely profit consistently, even though some of my analysis might be wrong and resulted in losses here and there, but as long as there are more right predictions than wrong predictions, as a whole, my portfolio increases in value.

This is the challenge that everybody faces in trading stocks, and in this paper, I will propose a strategy to follow that addresses and tackles this problem, with a goal of making profits consistently with a risk that we can control. The strategy I propose is a period trading strategy that is built to aid us to make better decisions and to make sure that we pick the right stocks to buy, by having a ranking system that ranks companies fundamentals that allows us to pick the best stocks out of a pool of stock. The foundation of the method is to be able to consistently make profit while being able to control the amount of risk that I am willing to take. One of the steps that will be taken is by creating a portfolio by minimizing its risk for a target return that we decide, through mathematical optimization. The strategy will be based on making periodic decision that will eventually build up a consistency to follow, with setting up a buying period that limits our ability to buy stocks in an uncertain manner, while at the same time having a cut loss as a safety net to limit the maximum amount of money we can lose in every period. The goal that I am trying to achieve from this strategy is to not lose money in each of the year, and with the ultimate goal is to beat the S&P 500.

**Data Collection**

The strategy I developed requires many historical data involving the stocks. For each of the stock in the stock pool, and for each period, the data and the process of collecting data is as follows:

* First, I will need a collection of stocks that will be used in the pool for the period. Each of these stocks are hand-picked, preferably have a moderate amount of industrial diversity in them to provide a more flexible outcome from the ranking system.

*Stock screening and picking were performed in tradingview.com and zacks.com*

* For each of the stock, I will need their latest Free Cash Flow, Price-to-earnings Ratio, Price-to- book Ratio, Debt-to-Equity ratio, and Price-Earnings-to-Growth ratio. These fundamental data will be used to rank the stocks in the pool based on their fundamentals, which step will be discussed in detail later in this paper.

*All fundamental data was collected from the Bloomberg Terminal*

* I will then need to find the historical price data of the past quarter for each of the stock chosen after the fundamental ranking step to perform an optimization. This optimization is done in MATLAB, which aims for minimum risk given an expected return.
* Finally, I will also need to find the historical price data for the chosen stock for the period quarter to simulate the results.

*Historical price data are collected from Yahoo Finance and Bloomberg Terminal*

Processing the data themselves are all done in excel and MATLAB. Since most of the data collected comes in separate files, I will first need to combine them in excel into a format that I will use to run my code in MATLAB. The first and foremost will be the pool ranking system. I will need to combine the fundamental datas for each period into one table format to use it as a parameter in my code in MATLAB. The table and result look like the following:

Ranking the stocks will then result in a subset of the stocks from the pool (5 best stocks for this test run). For those stocks, I will then need to preprocess their historical price data in Excel just like the fundamentals above to achieve the format I want for the MATLAB code. Past quarter prices will be used to perform optimization and to calculate the weight of each stock to buy in this period, while next quarter prices will be used to simulate the portfolio and collect results.

**Method Used**

The algorithm I developed can be broken down into the following steps:

* Picking buying periods
* Stock Screening
* Forming a Stock Pool
* Ranking the fundamentals of the stock
* Picking the best ranked stock from step 4, with a few exceptions.
* From the stocks picked in step 5, process their historical data price to perform optimization
* Step 6 will result in a set of weight that will then be used for current period’s portfolio
* \*Optional: Set a target return for each of the stock for the period (This was in my plan at the beginning of the development of the strategy, but I finally decided to exclude it for reasons I will discuss shortly).
* Set a cut loss
* For this research, simulate the portfolio and collect results

**Picking Buying Periods**

As discussed previously in the paper, the strategy involves picking buying periods, in which all security purchases can only be performed on the first day of every period, while selling securities that have hit the cut loss can be done at any time during the period. In general, there is no limitation on how long each of the periods should be and when, although different lengths in periods have their own advantages and disadvantages. Picking a period too long increase risks of being exposed to external factors (or hence crashes), which most of the time is inevitable, while picking a period too short may reduce the reliability of the system, since the period length I use from the historical data price that calculates the weights that will later on be used in the portfolio will be equal to the length of the buying periods itself, using a very short period will leave me with a very small sample size, and for stock prices that can fluctuate unpredictably in any given day.

After much consideration, the period that I use for this research will be quarterly, with buying day the following dates:

* 1st of January
* 1st of April
* 1st of July
* 1st of October

Which I ran for 2 years, which are during 2017, when the stock market rose for the entire year, and 2018, when the stock market crashes and loses value for the year. This was done to test and evaluate the algorithm in times when the stock market performing well, and the opposite.

How did I come to the decision of picking the dates above?

The idea comes after multiple observations on the behavior of my ranking system and stock momentum. Each of the dates above are picked because they come right after almost every company released their most recent earnings statement. This will allow my ranking system to pick up their latest fundamentals. Because of the nature of my ranking system, most of the stocks that are ranked on top are stocks of companies that are having excellent performance, and hence have very strong, well-rounded

fundamentals. A lot of them are in their all-time high too. Which based on my observation, brings me to the following conclusion. Since stocks usually gain the best upward momentum after a very strong earnings report, the idea here is that we want to ride that momentum not very long after that momentum started to aim for as much return, we possibly can. Since the stocks picked are screened using fundamentals from their latest earnings report, the stocks that results from the ranking are mostly stocks that just had strong quarterly earnings. This timing also synergizes well because the fundamentals that we use will also be the latest we can possibly get, which amplifies the effectiveness of the ranking system.

**Stock Screening and forming a Stock Pool**

Forming a stock pool involves many considerations and can be subjective, but the general rule of thumb for this strategy to be successful are:

* Pick a substantial amount of Stocks from 3 or more industries, this will allow industrial diversification later in the portfolio, and reduces risk of getting much exposure into external factors that affects an entire industry. Additionally, because the ranking system at its base ranks the stocks by comparing them to other stocks that are in the pool,
* Pick stocks and industries that differs in volatility but pick those that are currently sought after and have room for growth.

The industries I used:

* Technology
* Retail
* Finance Sector
* Preferably pick stocks that have large market caps, have been in the market for a long time and have high volume (Blue-Chip stocks). The reason is because other than for their liquidity, these are the stocks that price mostly can are reflected from their fundamentals, and hence synergizes well with the ranking system. We do not want stocks that just had an IPO or that are valued based on their growth potential, such as Tesla or AMD (which I used in this project for research purposes), not necessarily because they are bad investment, but because the ranking system will not capture that potential as their fundamental are usually not very strong. For example, a lot of them still have negative cash flow, which will hurt their fundamental rank greatly and hence eventually become redundant in the process.

**Ranking Stock Fundamental and Picking the Stocks to Optimize**

I ranked the fundamentals of the stock by comparing each category of each stock to every other stock in the pool, each category will be given a score based on where the category of the stock rank compared to the same category of all of the other stocks in the pool. This can be best explained by looking at the following table (results from the first period):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| January 1  2017 | FCF | PE | PB | D/E | PEG |
| AMD | -45 | — | 25.48774 | 344.9519 | 90.72 |
| AMZN | 8646 | 152.5153 | 18.54747 | 105.8335 | 1.549057 |
| AAPL | 20592 | 13.86193 | 4.597652 | 66.12962 | 1.392801 |
| GOOGL | 6335 | 28.71818 | 3.940096 | 2.830202 | 1.240887 |
| FOOT | -14 | 14.43344 | 3.375644 | 4.861375 | 1.426386 |
| MS | -6137 | 14.24353 | 1.142089 | 448.5598 | 1.477931 |
| GS | 2516 | 14.12396 | 1.242116 | 461.3829 | 1.335663 |
| UAA | 333.569 | 60.8822 | 6.271432 | 40.24758 | 3.942468 |
| WFC | 2508 | 13.83155 | 1.579374 | 175.4929 | 1.501519 |
| NKE | 461 | 23.92898 | 6.72855 | 28.70243 | 2.34354 |
| COLM | 333.361 | 21.16446 | 2.609908 | 0.888581 | 2.4839 |
| ATVI | 770 | 29.04316 | 2.952026 | 53.5914 | 0.970423 |
| BAC | -33673 | 13.96905 | 0.919473 | 177.6679 | 1.519527 |
| JPM | 36624 | 14.14606 | 1.347065 | 244.6316 | 2.099667 |
| LULU | -37.564 | 29.23116 | 6.646474 | 0 | 1.552016 |

Here, each of the fundamentals are ranked based on how they perform compared to others. For example, for the FCF (Free Cash Flow), companies with higher FCF will receive a higher score, since it implies a healthy financial situation, on the other hand, stocks with a lower P/E ratio are those receiving the higher score, since the P/E ratio indicates how over-priced a stock is. As for the result:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | FCF | PE | PB |  | D/E | PEG | Total |
| AAPL | 14 |  | 14 | 6 | 8 | 12 | 54 |
| GOOGL | 12 |  | 6 | 7 | 13 | 14 | 52 |
| WFC | 10 |  | 15 | 11 | 6 | 9 | 51 |
| GS | 11 |  | 12 | 13 | 1 | 13 | 50 |
| ATVI | 9 |  | 5 | 9 | 9 | 15 | 47 |
| JPM | 15 |  | 11 | 12 | 4 | 5 | 47 |
| FOOT | 5 |  | 9 | 8 | 12 | 11 | 45 |
| BAC | 1 |  | 13 | 15 | 5 | 8 | 42 |
| COLM | 6 |  | 8 | 10 | 14 | 3 | 41 |
| MS | 2 |  | 10 | 14 | 2 | 10 | 38 |
| NKE | 8 |  | 7 | 3 | 11 | 4 | 33 |
| LULU | 4 |  | 4 | 4 | 15 | 6 | 33 |
| AMZN | 13 |  | 2 | 2 | 7 | 7 | 31 |
| UAA | 7 |  | 3 | 5 | 10 | 2 | 27 |
| AMD | 3 |  | 1 | 1 | 3 | 1 | 9 |

Each of the category will receive a score based on their own category rank in the pool, for example,

AAPL’s FCF receives a score of 14, which means out of the 15 stocks in the pool above, AAPL’s FCF is the second best in the pool, while JPM’s FCF receives the best out of all. This ranking system allows companies that performs best in the overall category to be at the top, which allow us to filter companies that performing significantly well during last period. We then pick several of the best stocks from the result of the ranking. For this research, I picked the top 5 stocks with an exception that we can only pick

a maximum of 2 stocks from each industry, If more than 2 stocks from the same industry are on the top 5, I picked the 2 best stocks from that industry and pick the next best available from another.

**Optimization (Minimum Risk Mean-Variance Portfolio)**

The optimization method I use is by minimizing the variance formula with an expected return (I picked 8

% for the expected quarterly return in this research) as a constraint. The formula is the following:



I use the historical prices from the past quarter for each period to come up with the covariance matrix, average return and volatility for each of the stock. This optimization problem can be performed by using Lagrange and be solved numerically, but for this research the process is done by using MATLAB’s *fmincon* function to minimize the variance formula above. This allows me to come up with a reliable result while having the option to modify the optimization process easily, such as adding a lower bound for each of the weight if necessary. The optimization will produce a set of weight that I will follow in applying the portfolio for the next quarter.

**Target Return and Cut Loss**

After much consideration, I decided not to have a target return in this strategy. The main reason behind this is because many of the stocks that are resulted from the ranking system are stocks that have been performing very well in every fundamental category, and because the period I chose are right after the earnings date, we want to pick up as much of that upward momentum possible without having to limit them. Additionally, I don’t think it makes sense to set up a target return that is generalized through the entire stock pool, since the chosen stock will come from different industries, they will differ greatly in volatility and what they can achieve in a single quarterly period.

The cut loss is determined based on the return expected, for this test run I used an expected return of 8% and a cut loss of 5%.

**Results**

After ranking the fundamentals of the stocks for the first period, which results can be observed earlier in the previous section, I ran the optimization using MATLAB for the top 5 ranked stocks, and here is a result example from the first period.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1/1/2017 | | | | | |
| **Results (E[R] = 8%)** | **AAPL** | **GOOGL** | **WFC** | **GS** | **ATVI** |
| Weights | 0.412925828 | 0.146509805 | 0.252063781 | 0.03198588 | 0.156514706 |
| Volatility | 2.11E-01 |  |  |  |  |
| **Cash** | $ 1,000,000.00 |  |  |  |  |
| **Position** | $ 412,925.83 | $ 146,509.80 | $ 252,063.78 | $ 31,985.88 | $ 156,514.71 |
| **Buy** | 115.82 | 792.450012 | 55.110001 | 239.449997 | 36.110001 |
| **Cut Loss** | 110.029 | 752.8275114 | 52.35450095 | 227.4774972 | 34.30450095 |
| **Period Decision** | Hold | Hold | Cut | Cut | Hold |
| **End Period** | Sell at end | Sell et end | $ 239,460.59 | $ 30,386.59 | Sell at end |
| **Sell Price** | 143.660004 | 847.799988 | 13.89123523 | 7.659018902 | 49.61 |
| **End Value** | $ 512,182.06 | $ 156,743.02 | $ 239,460.59 | $ 30,386.59 | $ 215,028.92 |
| **Return** | 24.0373% | 6.9847% | -5.0000% | -5.0000% | 37.3858% |
| **Portoflio Value** | $ 1,153,801.18 |  |  |  |  |
| **Portfolio Return** | 15.3801% |  |  |  |  |

The stocks that were picked in the first period were AAPL, GOOGL, WFC, GS, and ATVI, with weights calculated through the optimization process and can be observed above. The period decision for each of the stock is based on whether or not the stock ever reaches a price that goes below its cut-loss, which from the above example, are performed for WFC and GS.

The stocks used and weights generated for this simulation is the following:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Period** | **E[R] = 8%** |  |  |  |  |
| **1** | **AAPL** | **GOOGL** | **WFC** | **GS** | **ATVI** |
| **Weight** | 0.412926 | 0.14651 | 0.252064 | 0.031986 | 0.156515 |
| **2** | **AAPL** | **GOOGL** | **WFC** | **MS** | **COLM** |
| **Weight** | 0.083815 | 0.613373 | 0.19297 | 0.018437 | 0.091406 |
| **3** | **BAC** | **GOOGL** | **JPM** | **COLM** | **AAPL** |
| **Weight** | 0.007335 | 0.47115 | 0.278294 | 0.196266 | 0.046955 |
| **4** | **BAC** | **WFC** | **GOOGL** | **AAPL** | **LULU** |
| **Weight** | 0.026776 | 0.200325 | 0.471304 | 0.206192 | 0.095403 |
| **5** | **MS** | **AAPL** | **GOOGL** | **WFC** | **JPM** |
| **Weight** | 0.06109 | 0.206192 | 0.471304 | 0.200325 | 0.06109 |
| **6** | **BAC** | **WFC** | **AAPL** | **GOOGL** | **COLM** |
| **Weight** | 0.061089 | 0.2003245 | 0.206192 | 0.471303 | 0.061089 |
| **7** | **AAPL** | **GOOGL** | **GS** | **JPM** | **COLM** |
| **Weight** | 0.149069 | 0.110805 | 0.309005 | 0.115457 | 0.315665 |
| **8** | **AAPL** | **GOOGL** | **MS** | **COLM** | **BAC** |
| **Weight** | 0.149069 | 0.110805 | 0.309005 | 0.115457 | 0.315665 |

For this simulation, I tested the system by running simulation through 2 different years, which are 2017 and 2018. This is done to observe the behavior of the strategy through different kind of markets, one is when the market is having a good momentum and consistently goes up for the entire year (2017), and one when the market crashes and experience a loss in value for the year (2018). As my period lengths are a quarter year, here are the results for each of the 8 periods in the simulation compared to the S&P 500.

|  |  |  |  |
| --- | --- | --- | --- |
| **Period** | **Portfolio Value** | **Portfolio Return** | **S&P 500 Return** |
| 0 | $ 1,000,000.00 | - | - |
| 1 | $ 1,153,801.18 | 15.380% | 5.53% |
| 2 | $ 1,187,205.50 | 2.895% | 3.165% |
| 3 | $ 1,250,907.57 | 5.366% | 1.57% |
| 4 | $ 1,376,705.19 | 10.057% | 4.198 |
| 5 | $ 1,321,051.38 | -4.043% | -0.096% |
| 6 | $ 1,431,591.40 | 8.368% | 6.640% |
| 7 | $ 1,487,833.12 | 3.929% | -0.240% |
| 8 | $ 1,413,441.46 | -5.000% | -7.200% |
|  |  |  |  |
| 2017 |  | **37.671%** | **21.830%** |
| 2018 |  | **6.994%** | **-4.380%** |

From the table above, we can observe the return from the portfolio generated by the algorithm for each period, and the return of the S&P 500 from the same period. The result I got meets my expectation, with the return in 2017 almost double the S&P 500. The 38% return in 2017 exceeds my expectation, as the parameter I used for this run was 8 % a quarter, a cumulative return of 8 % per quarter should result in a 36 % return, 2 % below the return I got from my simulation.

Although, the more important significant result here is the one from 2018, in the year when the S&P 500 loses around 4% of its value, the algorithm still managed to beat the trade war crisis and generate a

return of 7%. This is what I’m trying to achieve and was looking for running the system in 2018, to see how the algorithm behave in a dire situation and to observe if it is able to control the amount of risk that I am willing to take as a trader, while protecting the portfolio from exposure to external factor as much possible. The periodic system is built on the foundation of creating consistency in making profits and minimizing lost, and I believe that based on the result above, I can conclude that the system can be successful more often than not, given the right setup. As can be observed above, the system profits in 6 out of 8 of the periods. Although we lose value in period 5 and 8, the only period that I consider a failure is period 5, because period 8 is when the worst of the market happen, and there is not a single stock in the stock pool that don’t result in a cut loss. With that being said though, the result from 2018 was still satisfying, being able to still make profit when the entire market crashes.

**Conclusion**

Based on the result, the first thing that I can conclude is that the method serves its number one purpose, which is to not lose money, while I consider the substantial return received in both 2017 and 2018 proves the effectiveness of the algorithm. Given more time, there are many more initiatives that I would love to try to apply to improve the algorithm, such as creating separate pool of stocks for different industries so that the fundamental ranking system can work even better by comparing fundamentals of companies that belong in the same industry, making the comparison more reliable, and creating a dynamic cut loss that goes up and reset as the current value goes up.

**References**

“Yahoo Finance - Business Finance, Stock Market, Quotes, News.” *Yahoo! Finance*, Yahoo!, finance.yahoo.com/.

“Stock Market Quotes & Financial News.” *Investing.com*, [www.investing.com/.](http://www.investing.com/)

Bloomberg Terminal

**Software**

classdef OptimizeResearch methods

function [weights,portVar,targetPrices] = findMinimumWeight(obj,prices,pos,ret)

returns = price2ret(prices); avgRet = mean(returns)\*100; covMat = cov(returns);

n = size(prices); n = n(2);

weight = ones(1,n); target = ret;

fun = @(x) x \* covMat \* x'; x0 = pos;

A = [];

b = [];

Aeq = [weight;avgRet]; beq = [1;target];

lb = zeros(1,n); ub = [];

weights = 0;

portVar = 0;

[weights,portVar] = fmincon(fun,x0,A,b,Aeq,beq,lb,ub);

end

function [scores,totals] = scoreStocks(obj,data) n = size(data);

nStocks = n(1); nIndicators = n(2);

sortedData = zeros(nStocks,nIndicators);

sortedData(:,1) = sortrows(data(:,1));

sortedData(:,2) = sortrows(data(:,2),'descend');

sortedData(:,3) = sortrows(data(:,3),'descend');

sortedData(:,4) = sortrows(data(:,4),'descend');

sortedData(:,5) = sortrows(data(:,5),'descend');

scores = zeros(nStocks,nIndicators); totals = zeros(nStocks,2);

for j = 1:nStocks

for k = 1:nIndicators currData = data(j,k); counter = 1;

while currData ~= sortedData(counter,k) counter = counter+1;

end

scores(j,k) = counter;

end

end

totals(j,2) =sum(scores(j,:));

end

%names = table2array(names);

%for y = 1:nStocks

% totals(y,1) = names(y,1);

%end

function dailyReturns = dailyReturn(obj,prices) dailyReturns = [];

dimension = size(prices); days = dimension(1);

n = dimension(2); for current = 1:n

for currDay = 2:days

today = prices(currDay,current); yesterday = prices(currDay-1,current); dayReturn = (today/yesterday) - 1;

dailyReturns(currDay-1,current) = dayReturn;

end

end

end

function [posVal,posRet,portVal] = simPort(obj,cash,weights,freturn,cutLoss)

sizes = size(freturn); nStock = sizes(2); nDays = sizes(1);

posVal = zeros(nDays,nStock); portVal = zeros(nDays+1,1); portVal(1) = cash;

for i = 1 : nStock

posVal(1,i) = cash \* weights(1,i);

end

returns = 1 + freturn; for j = 2 : nDays+1

for y = 1 : nStock

posVal(j,y) = posVal(j-1,y) \* returns(j-1,y) portVal(j) = portVal(j) + posVal(j,y)

end

end

posRet = zeros(1,nStock); for z = 1 : nStock

posRet(z) = (posVal(nDays+1,z) / posVal(1,z)) - 1 if(posRet(z) <= -cutLoss)

posRet(z) = -cutLoss

end

end

end

end

end

classdef ProcessReturns methods

function dailyReturns = dailyReturn(obj,prices) dailyReturns = [];

dimension = size(prices);

days = dimension(1); n = dimension(2); for current = 1:n

for currDay = 2:days

today = prices(currDay,current); yesterday = prices(currDay-1,current); dayReturn = (today/yesterday) - 1;

dailyReturns(currDay-1,current) = dayReturn;

end

end

end

function averageReturns = averageReturn(obj,returns) averageReturns = [];

dimension = size(returns); days = dimension(1);

n = dimension(2); for current = 1:n

averageReturns(current) = mean(returns(:,current));

end

end

% function PeriodAverageReturns = PeriodAverageReturns(obj,prices)

% PeriodReturns = [];

% dimension = size(prices);

% days = dimension(1);

% n = dimension(2);

% period = 10;

% periods = days/10;

% currPeriod = 1;

% while currPeriod <= periods

%

%

function[minimumVarianceWeights,minimumVariance] = minVariance(obj,returns)

covMatrix = [];

covMatrix = cov(returns); dimension = size(returns); position = ones(1,dimension(2));

minimumVarianceWeights = mtimes(position,inv(covMatrix)); minimumVarianceWeights =

minimumVarianceWeights/sum(minimumVarianceWeights); weights = minimumVarianceWeights;

minimumVariance = weights\*covMatrix\*transpose(weights);

end

function [MVPortWithReturn,lagrange,constraintMatrix] = minVarRet(obj,returns,covariances,averageReturn,desiredReturn)

dimension = size(returns); position = ones(dimension(2),1);

tReturn = transpose(averageReturn);

%lagrangeMatrix = [(2\*covariances) transpose(position) transpose(averageReturn)];

%lagrangeMatrix = [lagrangeMatrix ; position 0 0 ];

%lagrangeMatrix = [lagrangeMatrix ; averageReturn 0 0]; cov2 = 2 \* covariances;

lagrange = [cov2 position tReturn ; transpose(position) 0 0 ; transpose(tReturn) 0 0];

end

end

end

weightConstraint = 1;

nPos = zeros(dimension(2),1);

constraintMatrix = [nPos; weightConstraint; desiredReturn]; MVPortWithReturn = (lagrange \* constraintMatrix) MVPortWithReturn = MVPortWithReturn/sum(MVPortWithReturn);