Quantitative Trading by STFM (Statistical-Technical Forecasting with Momentum) Strategy

Authors: Group 11

Yichen Wang, MSc Financial System Engineering
Meng Zhang, MSc Financial System Engineering
Yinan Zhang, MSc Computational Finance
Gabriel Bila, MEng Computer Science
Chenrui Wang, MSc Financial Risk Management

December 14, 2015

Abstract

The STFM (Statistical-Technical Forecasting with Momentum) Strategy, implemented by python programming language on Quantopian platform, is composed of ARMA (Auto-Regression Moving Average) statistical models, Value Momentum Strategy and Stocks Technical indexes. The Performance Analysis is made by comparing benchmark with return and sharpe ratio, and the associated risks have been analyzed and discussed by VaR, Max Drawdown, Volatility and β return. The Code listing is attached at the end of the document with the allocation into several modules as well as the appropriate comment on demonstrating how the algorithm works.

Team Composition and Work Assignment

Team Leader: Yichen Wang
Strategy: Yichen Wang, Meng Zhang
Performance: Yinan Zhang
Risk: Chenrui Wang
Implementation: Meng Zhang, Gabriel Bila

Contents

1 Part 1 - STFM Strategy

1.1 Overview of the STFM Strategy

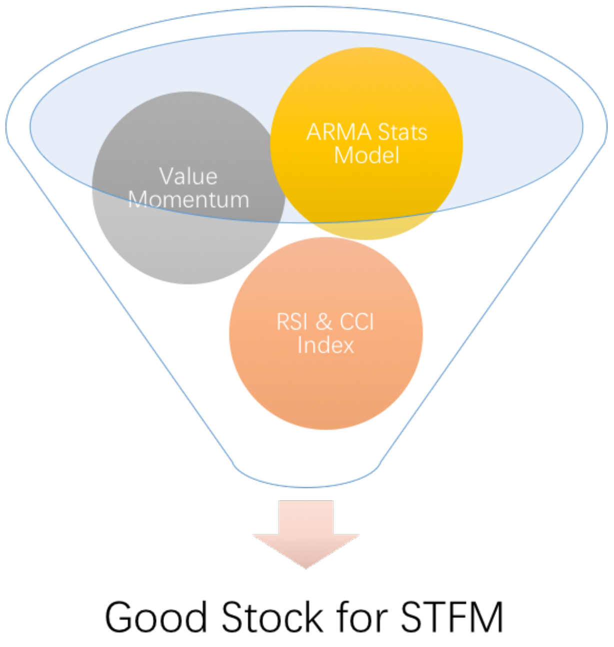


Figure 1: Filter logic of STFM Strategy

Overall, as it can be seen from figure 1, the STFM (Statistical-Technial Forecasting with Momentum) strategy is composed of 3 main components: Value Momentum strategy, ARMA (Auto-Regression Moving Average) Statistical Model and Technical Analysis Model, so as to combine the corresponding advantages of 3 totally-different kinds of strategies. All trades have to be executed under the permission after taking historical price trend with ARMA statistical models, buy/sell information with technical index and company finance data with value momentum strategy into consideration. In that term, the strategy define the stock as “good” one, which would be put into the stock pool, and wait for the signal matching for sell or buy.

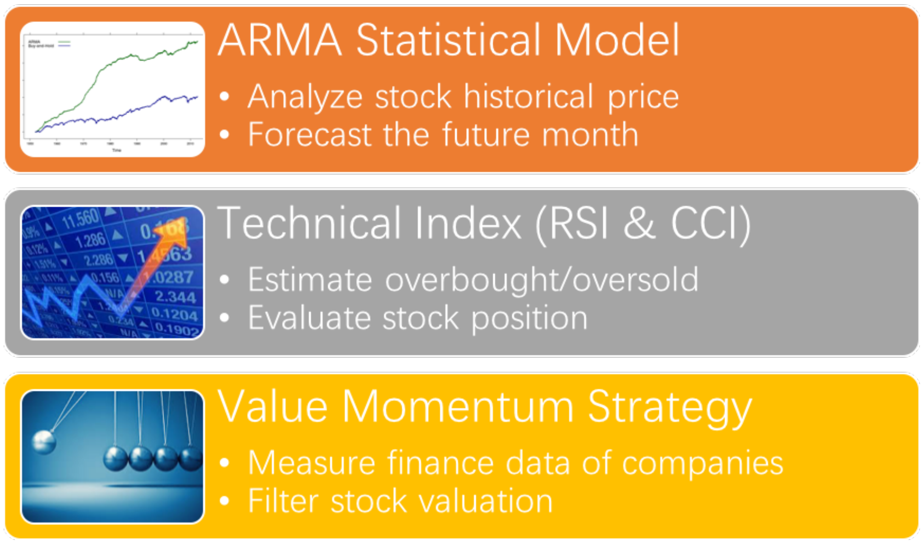

ARMA Statistical Model
Analyze stock historical price
Forecast the future month
Technical Index (RSI & CCI)
Estimate overbought/oversold
Evaluate stock position
Value Momentum Strategy
Measure finance data of companies
Filter stock valuation

Figure 2: Responsibility of main 3 Components of STFM Strategy

Figure 2 illustrates the main role of the 3 components: ARMA Statistical Model, Technical Index (RSI & CCI) and Value Momentum Strategy. The ARMA Statistical model will recur every month, analyzing the price history to give the prediction on next month. Technical Index method (RSI and CCI) can evaluate the relative strength and inspect whether stock are in healthy position or not. In addition, Value Momentum Strategy is able to measure the financial circumstances in comprehensive ways of data which could be obtained from Quantopian fundamental.

1.2 How STFM works

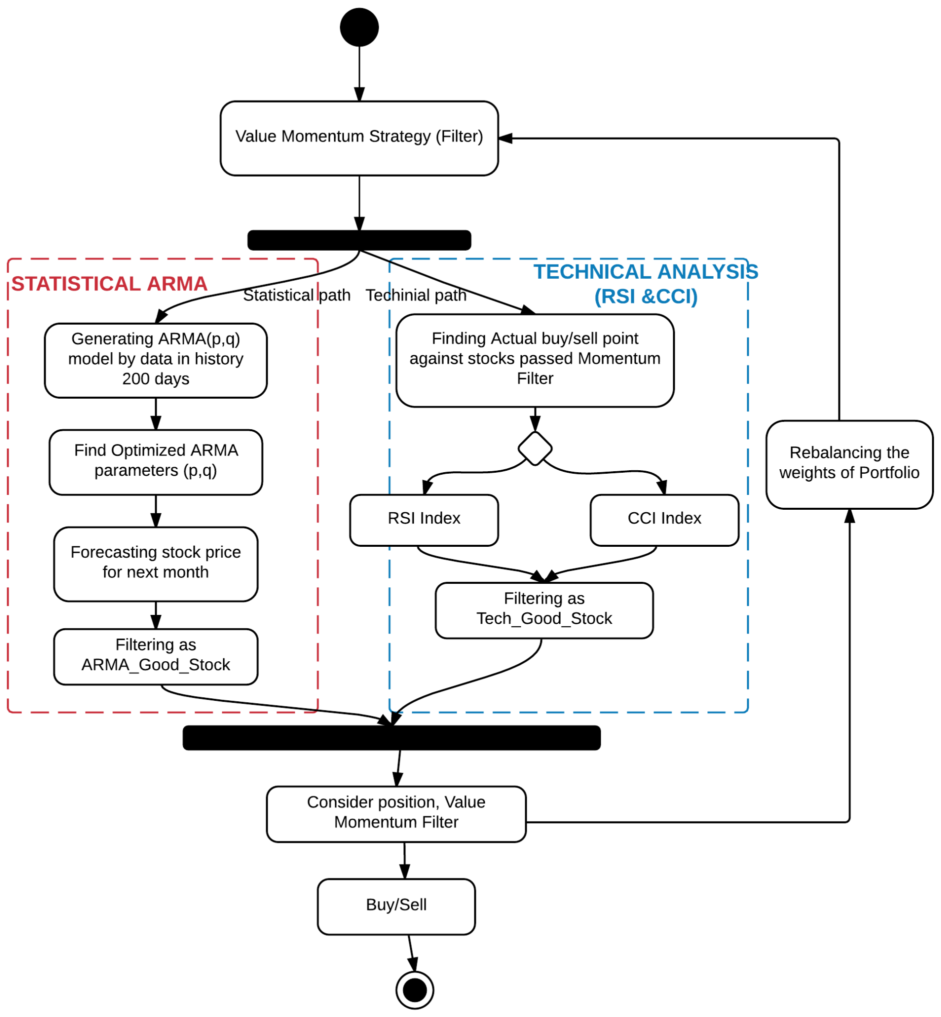

Value Momentum Strategy (Filter)
STATISTICAL ARMA
Statistical path
Techinial path
TECHNICAL ANALYSIS (RSI &CCI)
Generating ARMA(p,q) model by data in history 200 days
Find Optimized ARMA parameters (p,q)
Forecasting stock price for next month
Filtering as ARMA_Good_Stock
Finding Actual buy/sell point against stocks passed Momentum Filter
RSI Index
CCI Index
Filtering as Tech_Good_Stock
Rebalancing the weights of Portfolio
Consider position, Value Momentum Filter
Buy/Sell

Figure 3: UML Actitivity Diagram of STFM Strategy

As is can be seen from figure 3, the UML Activity diagram of the STFM Strategy illustrates on the logic and procedure of the STFM algorithm.

Firstly, the Value Momentum Investment strategy is applied for the universe stocks, evaluating and inspecting the general markets in financial data, capitalization, PE Ratio and other several financial, marketing parameters to select the basic stock pool.

Then the strategy continues with the analysing and forecasting with double methodology - ARMA Statistical path and Technical index. Initially ARMA picks up history markets data to construct the model with statistical parameters, followed by the optimization and forecasting, and thus accomplishing the trend analysis to get the stocks with distinguished outlook.

Meanwhile, the Technical index models contains 2 commonly-applied methods: RSI and CCI, to estimate the relative strength, and evaluate whether the stock is out of the normal range in markets. Through RSI and CCI index methods, the filtering on the markets and trend of stocks can be derived as well, so that strategy is able to make sure all of the potential order would not lead to the loss.

Furthermore, the algorithm would keep recursive monthly due to the configuration of the ARMA forecasting, which will make prediction for the trend according to the historical month stock price.

1.3 Design of Strategies

1.3.1 Where our idea came from

In general, the idea of combining Value Momentum, ARMA Statistical Modelling and Technical indexes came from 3 main reasons. Firstly, we are fortunately enough to have really excellent members from different background.

The perfect allocation and team atmosphere is key to the great efficiency of project, in which everyone contributes a lot for each section, showing the strength and critical-thinking as well as the academic background. The Project Leader Yichen Wang is excellent at leadership, cooperating and coordinating with colleagues, as well as the mixed background from computer science and management skills. Meng Zhang is an experienced trader in Chinese stock markets who has gained profit for a long period, who has contributed a lot on implementing the strategy, addressing the requirement with realistic markets and optimizing the parameters. Yinan Zhang has graduated from UCL Statistics, who led the team on applying statistical modelling to make forecast to improve the efficiency and return rate.

Therefore, the background research went through quite well-efficient, so that we could push the process forward ahead of other teams. In the first week's meeting, team assigned the work into different sections from various perspective on strategy: Mathematics, Stock Markets and Technical Analysis.

Basically, our strategy's 3 main components synergised well; their combination helped minimise their individual drawbacks. The main feature for Statistical models is basing on the Stats theory with high accuracy on trend analysis as well as the forecasting. However, the cost of pure statistical methods is out of control which might made the efficiency of algorithm extremely low. The Stock

Index method relies on the finance index of the stocks, while it analyze the trend only with the stock circumstances that lacks the concern on history of price and forecasting.

Through effective combination on the statistical, technical and momentum strategy, the signals of buying/selling, weight re-balancing are realized to continuously improve the return rate as well as controlling the risk.

1.3.2 Buy/Sell Signals Determination

The STFM strategy equips the buying signal with 2 filters: Value Momentum filter and ARMA filter. All the buying orders have to acknowledge the permission with them, and only stocks healthy enough could be put into the stock pool to buy. The Technical index models RSI & CCI are not involved into the stock-buying filters, since the trigger of the strategy can only be activated when the stock price hit the bottom. The details of principal on applying RSI and CCI are about to be covered in the corresponding section.

Figure 4 indicates how the STFM filters all the stock market to get the qualified trade. When the ARMA model considers a stock as good and value momentum agrees, it will be added into the pool for the buying trigger.

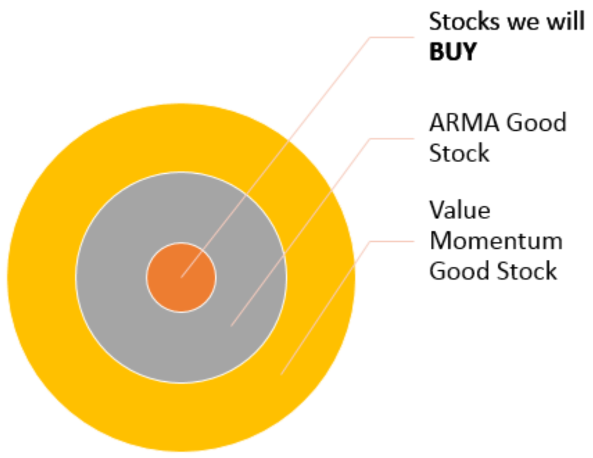


Figure 4: Buy Order determination of STFM strategy

As for the selling order, it can be seen from figure 5 that the STFM keep concerning on selling the unhealthy stocks with forecasting its future, analyzing the history and measuring their financial data. The objective stock will be removed from the pools with clearing position only if ARMA, Momentum and Tech index believe it to be terrible and may be worse then. Comparing to previous buying signals determination, the Technical index models RSI & CCI is included in the selling trigger because of its severity on the marginally-relative strength justification.

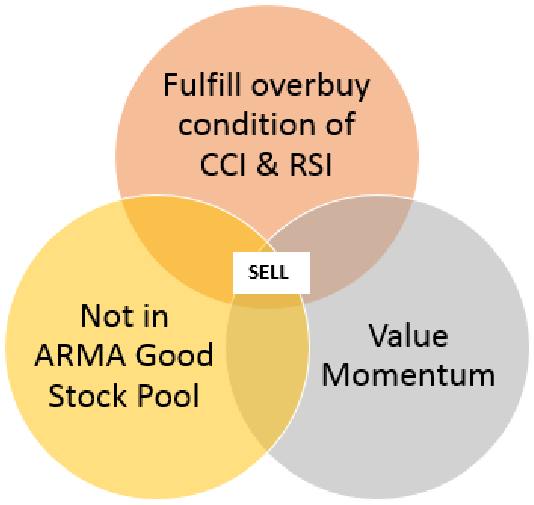

Fulfill overbuy condition of CCI & RSI
SELL
Not in ARMA Good Stock Pool
Value Momentum

Figure 5: Sell Order determination of STFM strategy

1.3.3 Asset Class

Basically the STFM is deployed under the American Stock Market with $\$100,000$ cash **without any leverage actions**. All of the team members agree with the opinion that leverage should not be included in the automatic trading simulation project. The main reason why the leverage cannot be included is that all members believe if we involve the leverage, testing with the historical market data, the Quantopian would bring an extremely huge return rate with our strategy, approximately $2,000,000\%$ which is not joking and enough for us to get into prison, so that it does not make any sense for using leverage only in order to make the huge return rate. The strategy has to be consolidate in realistic foundation, with potential ability to make profit and get real return in stock markets, rather than just make as large numbers of result as we can. Moreover, leverage will bring equally opportunity to lose money, while it cannot be measured and always ignored in the Quantopian project, which make the leverage more useless.

1.3.4 ARMA Statistical Model

The Auto Regression Moving Average (ARMA) models, known as one of the most commonly applied and popular statistical methodology in time series research, is an advanced model on statistical forecasting and trend analysis, since it combines the sweetness of AR and MA models to keep obtain accuracy of analyzing and predicting the trend on time series as well as eliminating the error. The ultimate objective for applying ARMA model is to make full use of the historical data on forecast as well as eliminating the noise.

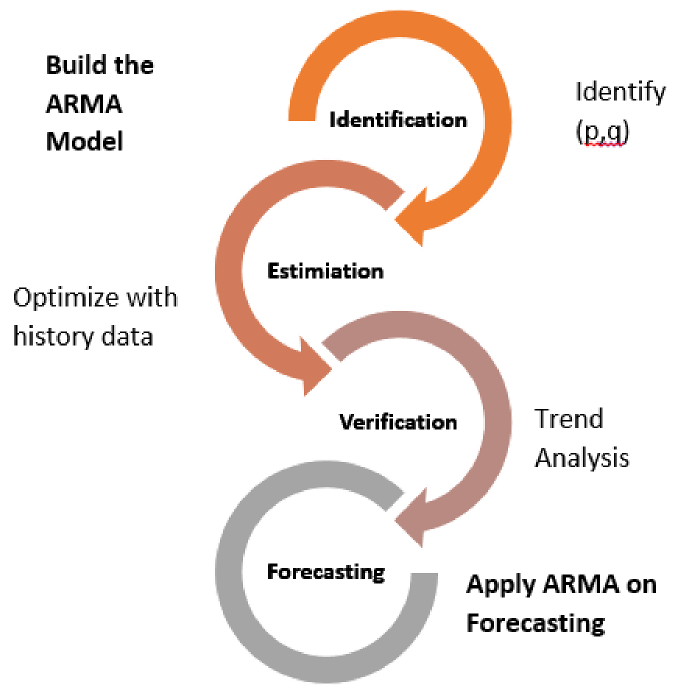


Figure 6: General Procedure of ARMA Statistical Models

The ARMA model application can be divided into 4 steps from figure 6. Firstly, it is essential to select a good model with unique parameter pairs (p for $AR(p)$ and q for $MA(q)$), which is decisive to the efficiency of models. After picking up key parameters, unknown parameters and noise has to be concerned and eliminated by appropriate noise reduction method in mathematics, followed by the verification stage, which is significant to inspect whether the model we built is reasonable as expected or not. Furthermore, the verified model is ready to work for forecast.

The main formula of $ARMA(p,q)$ is listed as follows:

$$X_t = c + + \sum_{i=1}^{p} \phi_i X_{t-i} + \epsilon_t + \sum_{i=1}^{q} \phi_i \epsilon_{t-i}$$

As can be seen from equation 1.3.4, the $ARMA(p,q)$ is composed of $AR(q)$ and $MA(q)$ to realize the trend analysis and forecasting.

On one hand, the $AR(p)$ (Auto-Regression) method, playing a fundamental role in the ARMA, is a statistical forecasting algorithm that can utilise the historical data of the series itself to forecast potential activities of next coming component, which is evolved from the linear regression method.

In statistics, the notation of $AR(p)$ is an auto regressive process of the order p, which defines a process which at time t, it depends linearly on past values of series itself as well as a random term. p can be considered as a "step" for the model, therefore as the optimization value can be found, the price of stock owned can be derived excellently.

The main formula of Auto-Regression method is listed as follows:

$$X_t = c + \sum_{i=1}^{p} \phi_i X_{t-i} + \epsilon_t$$

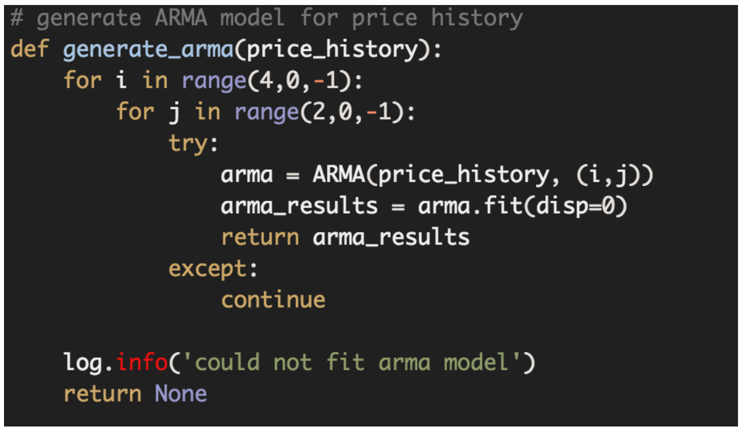
Assuming X_t is a time series, it can be seen simply in the figure 1.3.4 that X_t can be calculated by historical components in series $X_{t-1}, X_{t-2}, ..., X_{t-i}$. The C is a constant, so that the result of series in time t can be considered as the linear combination of previous historical data.

On the other hand, different from the $AR(p)$, the $MA(q)$ (Moving Average) method is applied to describe the accumulation of unknown factors, error series components and noises. The key parameter q can be thought as "step" as well, reflecting the numbers of moving average components. The ϵ_{t-i}, known as error components, can be mathematically assumed to be ignored, since it is sampled by standard normal distribution with mean $\mu = 0$ and variance 1.

The main formula of Moving Average method is listed as follows:

$$X_t = \mu + \epsilon_t + \sum_{i=1}^{q} \phi_i \epsilon_{t-i}$$

After discussing on the theory of ARMA, it is particularly vital to explain how ARMA is implemented in the STFM strategy.

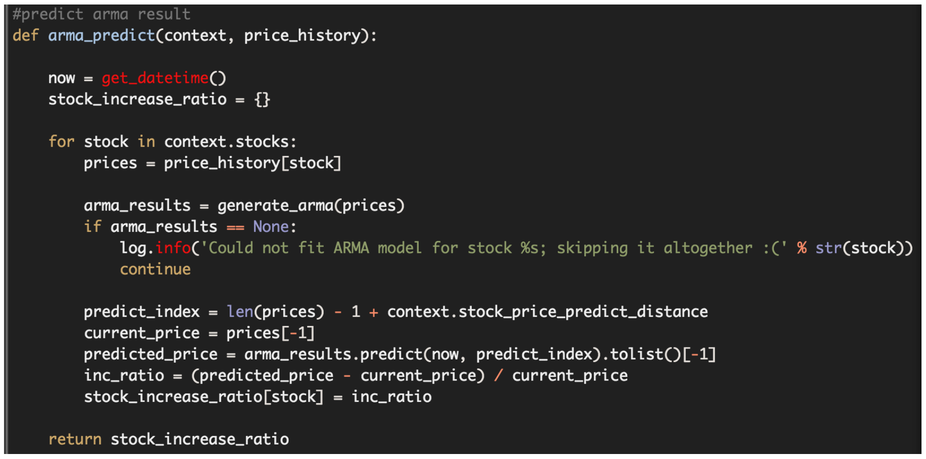


```
# generate ARMA model for price history
def generate_arma(price_history):
    for i in range(4,0,-1):
        for j in range(2,0,-1):
            try:
                arma = ARMA(price_history, (i,j))
                arma_results = arma.fit(disp=0)
                return arma_results
            except:
                continue

    log.info('could not fit arma model')
    return None
```

Figure 7: Generating ARMA Models

The figure 7 demonstrates how STFM generate the ARMA model. The parameter i and j refers to the statistical gradient p and q in $AR(p)$ and $MA(q)$. The for loop contains the model construction from 4 to 0 and 2 to 0 with the step of -1. The history dataset for the stock price is set to be 200 so as to make the trade-off on algorithm efficiency against the accuracy of the trend prediction.



```python
#predict arma result
def arma_predict(context, price_history):

    now = get_datetime()
    stock_increase_ratio = {}

    for stock in context.stocks:
        prices = price_history[stock]

        arma_results = generate_arma(prices)
        if arma_results == None:
            log.info('Could not fit ARMA model for stock %s; skipping it altogether :(' % str(stock))
            continue

        predict_index = len(prices) - 1 + context.stock_price_predict_distance
        current_price = prices[-1]
        predicted_price = arma_results.predict(now, predict_index).tolist()[-1]
        inc_ratio = (predicted_price - current_price) / current_price
        stock_increase_ratio[stock] = inc_ratio

    return stock_increase_ratio
```

Figure 8: Using ARMA on Prediction

As is mentioned above in application, the ARMA can be used for trend prediction after finding the optimal parameters pair p and q. Figure 8 shows the implementation of how STFM use ARMA to analyze the historical data to estimate the stock price in next month. The increase ratio of stock will be returned to be combined altogether with the technical index on determining the buying and selling trigger.

1.3.5 Momentum Investment Strategy

The Momentum investment strategy measures to what extent the stock price rise or fall in one direction, the "acceleration". The word "Momentum" in finance refers to the concept that it is more likely to keep the same trend rather than change the directions. Particularly, the stock would be allocated into 2 sections: long positions or short positions, in order to keep the profit as expected. By measuring the price differences over a past period of time, Momentum strategy allows us to determine the velocity of the trend, eventually leading to the filtering and sorting of the stocks. If the performance of a stock is well-qualified, it can be considered to continue the outstanding performance in the following time period through equipping with the Momentum investment strategy.

In these terms, what matters is the design of filters for momentum strategy. In order to filter and restrict the stock to fulfill the objectives of defining the trend, several financial parameters are involved to measure the foundations comprehensively. These parameters and their applications are listed as follows :

1. Market Capitalization - measures total market value of the company.
   $MarketCap = CurrentMarketPricexCompany'sshares$

2. Shares Outstanding - all shares of corporation that have been authorized, issued and purchased.

3. P/E Ratio - help valuation of the stocks in markets, the fraction of market value with earnings for each share.
   $P/ERatio = \frac{MarketValuePerShare}{EarningsperShare(EPS)}$

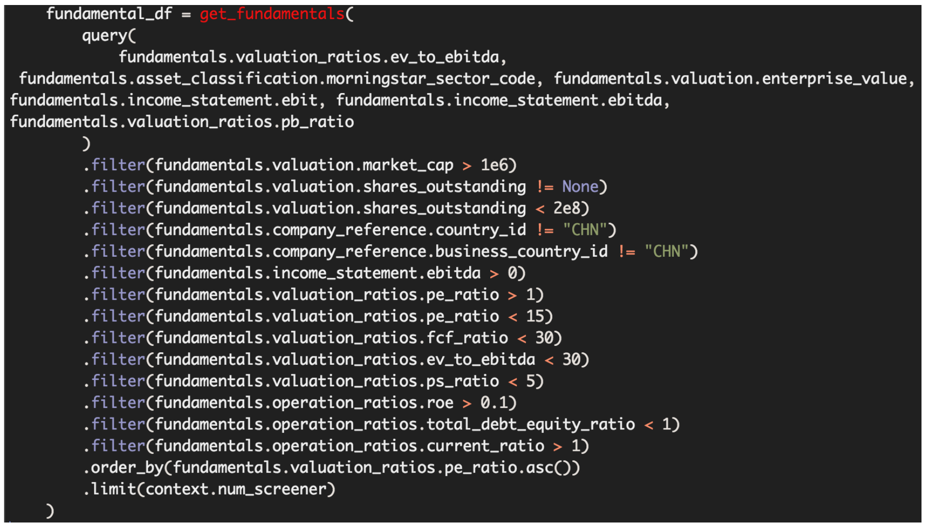
4. Return on Equity (ROE) - measures a corporation's profitability on how much profit a company can generate with the money shareholders have invested.
   $ReturnonEquity(ROE) = \frac{NetIncome}{Shareholder'sEquity}$

5. Debt/EBITDA - gives the investor the approximate amount of time for paying all the debts.
   $Debt/EBITDA = \frac{Debt}{EBITDA}$

6. P/S Ratio - help valuation of the stocks in markets, the fraction of market value with sales for each share. In different perspective than P/E Ratio.
   $P/SRatio = \frac{MarketValuePerShare}{SalesperShare(SPS)}$

7. Debt Equity Ratio - showing the relative proportion of shareholder's equity and debt used to finance the asset, known as risk and leverage as well.

   Figure 9 lists the part of our code responsible for applying the value momentum method on evaluating the corporate finance. The aim is to filter the markets and get good, healthy stocks at the very beginning, before executing any trades.



```
    fundamental_df = get_fundamentals(
        query(
            fundamentals.valuation_ratios.ev_to_ebitda,
 fundamentals.asset_classification.morningstar_sector_code, fundamentals.valuation.enterprise_value,
fundamentals.income_statement.ebit, fundamentals.income_statement.ebitda,
fundamentals.valuation_ratios.pb_ratio
        )
        .filter(fundamentals.valuation.market_cap > 1e6)
        .filter(fundamentals.valuation.shares_outstanding != None)
        .filter(fundamentals.valuation.shares_outstanding < 2e8)
        .filter(fundamentals.company_reference.country_id != "CHN")
        .filter(fundamentals.company_reference.business_country_id != "CHN")
        .filter(fundamentals.income_statement.ebitda > 0)
        .filter(fundamentals.valuation_ratios.pe_ratio > 1)
        .filter(fundamentals.valuation_ratios.pe_ratio < 15)
        .filter(fundamentals.valuation_ratios.fcf_ratio < 30)
        .filter(fundamentals.valuation_ratios.ev_to_ebitda < 30)
        .filter(fundamentals.valuation_ratios.ps_ratio < 5)
        .filter(fundamentals.operation_ratios.roe > 0.1)
        .filter(fundamentals.operation_ratios.total_debt_equity_ratio < 1)
        .filter(fundamentals.operation_ratios.current_ratio > 1)
        .order_by(fundamentals.valuation_ratios.pe_ratio.asc())
        .limit(context.num_screener)
    )
```

Figure 9: Value Momentum Filter on stock pool

1.3.6 Technical Index (RSI & CCI)

The STFM strategy involves stock markets technical index method to cooperate with the statistical model and value momentum strategy.

The traditional popular stocks method MACD, which we believe most groups applied, are not eventually included. The reason why RSI and CCI are involved rather than popular index MACD is that we have to guarantee all components of our strategy work smoothly and complement each other instead of conflicting. The main responsibility of RSI and CCI is to make up of the drawbacks for statistical model on evaluating the relative performance and position.

RSI (Relative Strength Index) mainly refers to the comparison in mathematics on the buyer power and seller power, the “relative strength”. RSI value volatilises within the range of (-100,100), which can be considered as the measurement of strength of markets confidence. When RSI is extremely high (e.g 90), then the market price will naturally drop down, so the RSI indicator will reflect the overbought signal notifying controller to sell the stocks. Correspondingly, the market will have a strong rebound if RSI hit the bottom, thus indicating what may be a great opportunity for us to buy the stocks.

The formula of calculating RSI is:

$$RSI = 100 - \frac{100}{1+RS}$$

in which RS refers to relative Average Gain or Average Loss.

The figure 10 is an example for demonstrating the overbought and oversold position of a stock, in STFM the threshold is set to be 70 and 30.

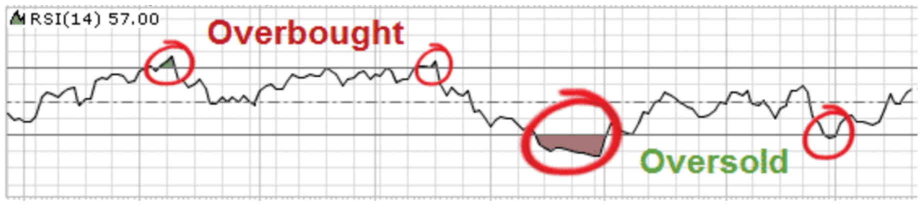
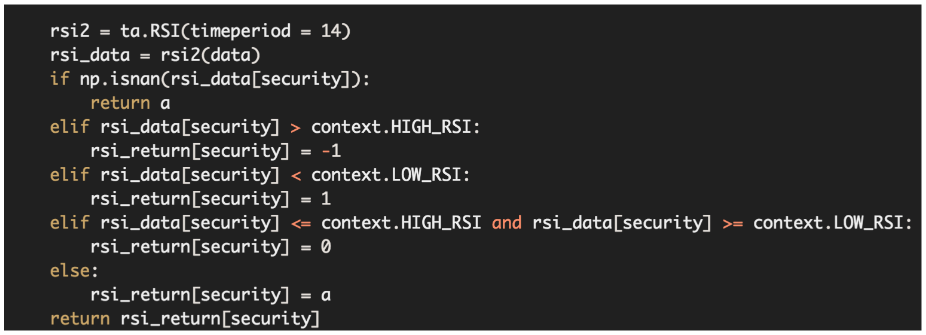


Figure 10: RSI example (overbought/oversold) for a single stock



```
    rsi2 = ta.RSI(timeperiod = 14)
    rsi_data = rsi2(data)
    if np.isnan(rsi_data[security]):
        return a
    elif rsi_data[security] > context.HIGH_RSI:
        rsi_return[security] = -1
    elif rsi_data[security] < context.LOW_RSI:
        rsi_return[security] = 1
    elif rsi_data[security] <= context.HIGH_RSI and rsi_data[security] >= context.LOW_RSI:
        rsi_return[security] = 0
    else:
        rsi_return[security] = a
    return rsi_return[security]
```

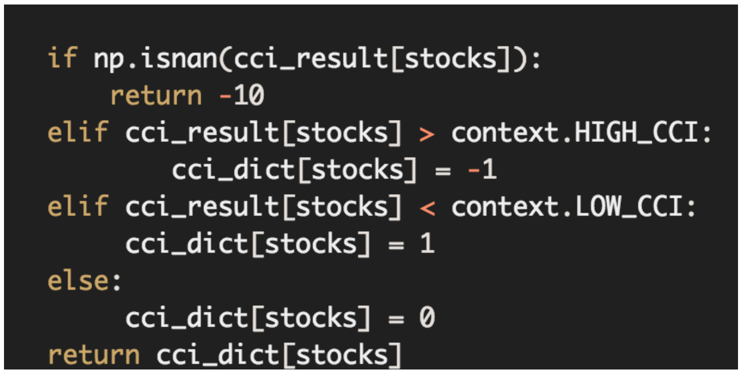
Figure 11: RSI (Relative Strength Index) on order trigger

Figure 11 shows how STFM utilizes the RSI on helping to determine the order trigger. When relative position is in over height, the stock is in overbought position, STFM pull the sell trigger and vice versa for hitting the oversold line.

In addition to the RSI, the CCI (Commodity Channel Index) is another method to evaluate the bias and degree of deviation of the stock price. Compared to RSI for reflecting the strength and confidence of investors for the stocks, CCI concentrates on determining whether the stock price is in normal distribution with healthy position or being biased. The threshold for CCI is set to be 100 and -100 in the STFM.

Despite the difference between the CSI and RSI, the way they are used in our strategy is analogous - the buying and selling triggers are set when the stock price hits the maximum and minimum thresholds respectively. Figure 12 contains a code segment illustrating this similarity.

. . . .
Printed name: YICHEN WANG
. . . .
MENG ZHANG
. . . .
Signature: YICHEN WANG
. . . .
MENG ZHANG



```
if np.isnan(cci_result[stocks]):
    return -10
elif cci_result[stocks] > context.HIGH_CCI:
        cci_dict[stocks] = -1
elif cci_result[stocks] < context.LOW_CCI:
     cci_dict[stocks] = 1
else:
     cci_dict[stocks] = 0
return cci_dict[stocks]
```

Figure 12: CCI (Commodity Channel Index) on order trigger

2 Part 2 - Performance Analysis

As mentioned in Part1, STFM strategy is set to trade monthly, and it needs to run in long term. Specifically we generated the strategy for 12 years, as shown in figure 13 steady increasing performance is ideal for investment and successfully beat the benchmark over the whole period. In this section, the performance of STFM strategy will be firstly discussed in terms of return and Sharpe ratio, and then the comparison with benchmarking will be displayed from the respective of return comparison and information ratio.

2.1 Return for STFM strategy

2.1.1 Cumulative Performance

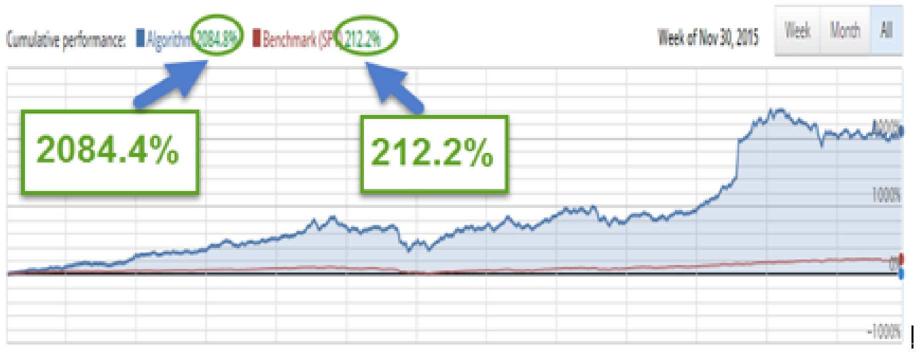


Figure 13: Cumulative Performace

In general, the return of STFM strategy keeps raising over 12-years period.

Based on the cumulative performance figure 13, before 2008, the cumulative return increase continuously and steadily with a minor fluctuation happened in the second half of 2007. Due to economic crisis in 2008 and 2009, year 2008 started with a flat trend, then followed by two successive declines of 294.63% and 90.88%. However the return was still above 350% while the benchmark returns fell to negative. After this, the return quickly back to 517.39% in May 2009, and continue to grow with some fluctuations within the range 660% to 944%. At the period of Nov 2012 to Nov 2013, the return experienced a dramatic raise of 1349.31% while benchmark's cumulative return only increased by 59.%. After Nov 2013, the cumulative return first had a small fall to 2085.52%, and then kept this extraordinary performance until end. It is notable that the annual rate of return has reached a remarkable level 178%.

2.1.2 Return

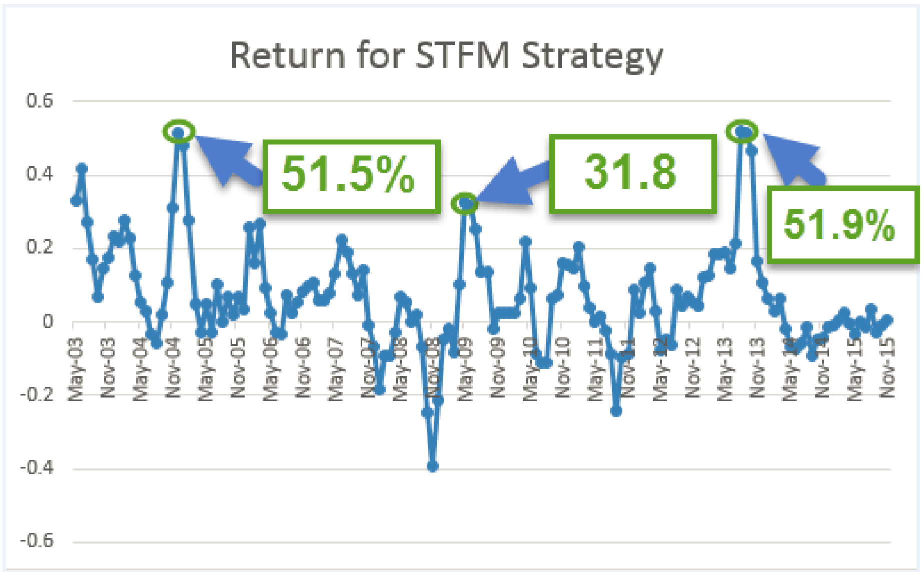


Figure 14: Return for STFM startegy

We use the data returned in the Quantopian simulation to draw a scatter plot figure 14. From the plot, we can see that STFM strategy has a spectacularly strong performance in return. It is positive in most of times over 12-year period, and the third of them had the return above 20% with two peaks of 51.5% and 51.9% in 2004 and 2013 respectively. Although there were some losses in 2007 and 2008 due to the strike of the economic crisis, the return immediately recover from the hard time and hit 31.8% in 2009. All this impressive data shows that STFM is a profitable strategy.

2.2 Return vs Risk

2.2.1 Sharpe Ratio

In this section, we will analyze the performance of STFM strategy compared to the risk taken. In our case, we choose Sharpe ratio instead of Sortino ratio because our strategy is low-volatility. The higher the Sharpe ratio, the better the performance and the greater the profits for taking on additional risk. A Sharpe ratio of 1 is considered good, while 2 is considered great and 3 is considered exceptional.

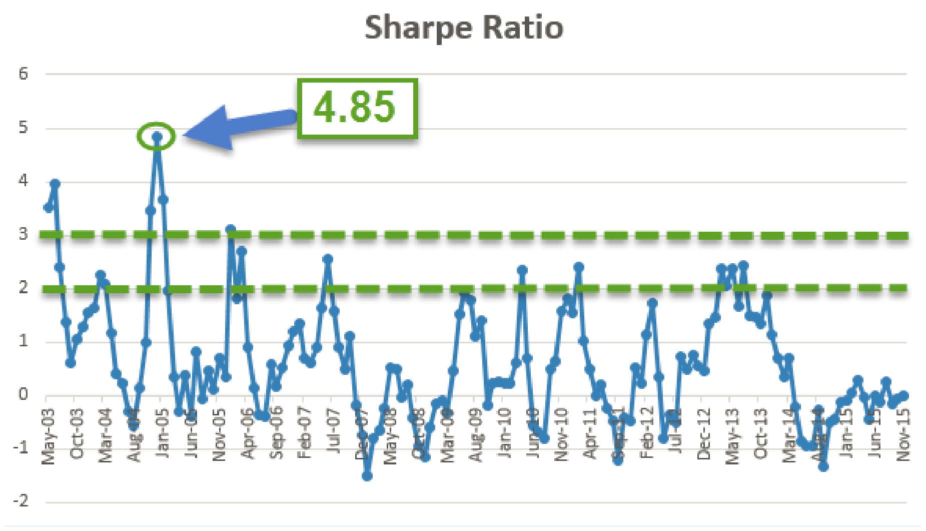


Figure 15: Sharpe Ratio

From figure 15 we can see that the Sharpe ratio for STFM strategy ranged from -1.504 to 4.85 with large fluctuations over 12-year period. Before second half of 2007, most months had positive Sharpe ratios, 11 of them exceed 2, and 6 of them over 3, which mean most times STFM strategy had great returns per unit risk and STFM strategy holds exceptional return versus return for several months. Significantly, the Sharpe ratio peaked to a remarkable level (4.85) on Dec 2004. Then influenced by economic crisis, the Sharpe ratio dropped down to negative. After this serious strike, the global economy start recovery, and Sharpe ratio of STFM strategy back to positive although there are still some fluctuations. This is because that the economy was still unstable, and investors were less confident in the market compared with the time before the crisis. In 2014 and 2015 the Sharpe ratio kept a low level with minor fluctuations corresponding to the low return over this period.

Furthermore, it is worthwhile to note that for overall 12-year trades the Sharpe ratio is 6.01, which is significantly high. This shows that STFM strategy has an exceptional return relative to risk.

2.2.2 Return/Risk Ratio

Return/Risk ratio is one of the most important judgment for evaluating a strategy in commodities futures market and stock market. Paying too much attention to high risk of withdrawal will probably indicates losing the chance of gaining more profit, especially in value investing strategy. The balance of controlling risk and acquiring more profit is reflected in the Return/Risk ratio.

A strategy with a Return/Risk ratio which is more than one is considered as a profitable strategy. The ratio is derived from annual rate of return dividing max drawdown. In our case, the ratio is 178%/59.5% approximately to 3, which means that for one unit of risk STFM can bring 3 unit of profit. This also proves the conclusion based on Sharpe ratio that STFM has extraordinary return relative to risk.

2.3 Comparison with Benchmark

2.3.1 Return Comparison

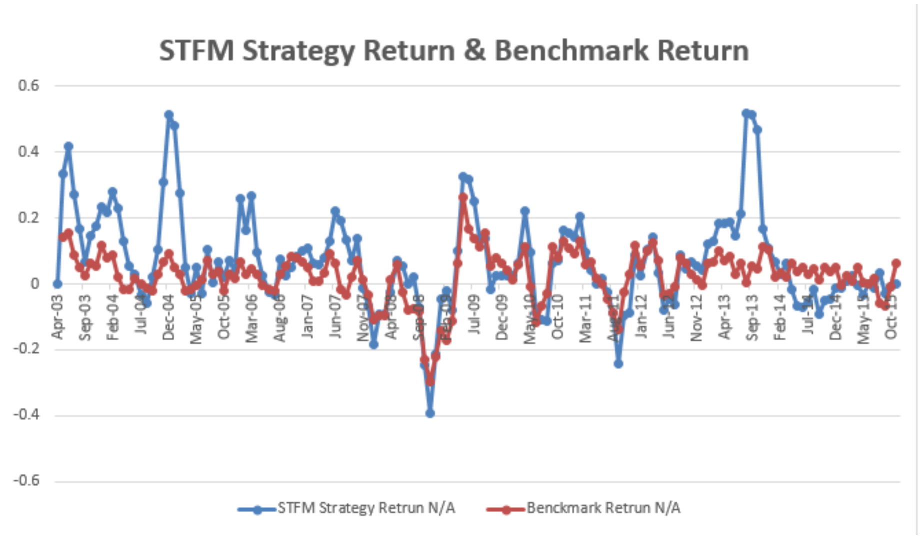


Figure 16: STFM Return & Benchmark Return

As shown by the cumulative performance figure 13, STFM had an extraordinary return with 2084.4% while the Benchmark only had 212.2% as return, so from this prospective STFM completely beat the Benchmark. We draw the plot for STFM Return & Benchmark Return. Looking at figure 16, we found that our strategy has similar trend as Benchmark, which both have higher return before and after economic crisis. Before 2007, global economy kept steady increasing, the return for STFM is significantly higher than the return for Benchmark. This fact went to be prominent in Dec 2004 where STFM's return reached a highest level five times of Benchmark's return. And STFM has larger fluctuation, so during this time STFM kept a high unstable return compared with Benchmark, in general STFM completely beat Benchmark during this period. Then in 2007 and 2008 both STFM and Benchmark experienced a serious decline, because global economy had a hard time over recession period. After that hard time, both STFM and Benchmark recover from the negative return, while in most times STFM's return was above Benchmark's return, so STFM beat Benchmark again. It is worthwhile to note that STFM had a surprising high return in middle of the year 2013. This shows that the filters of SFTM worked well during this time. From the view of 12-year period, STFM strategy beat the Benchmark.

2.3.2 Information Ratio

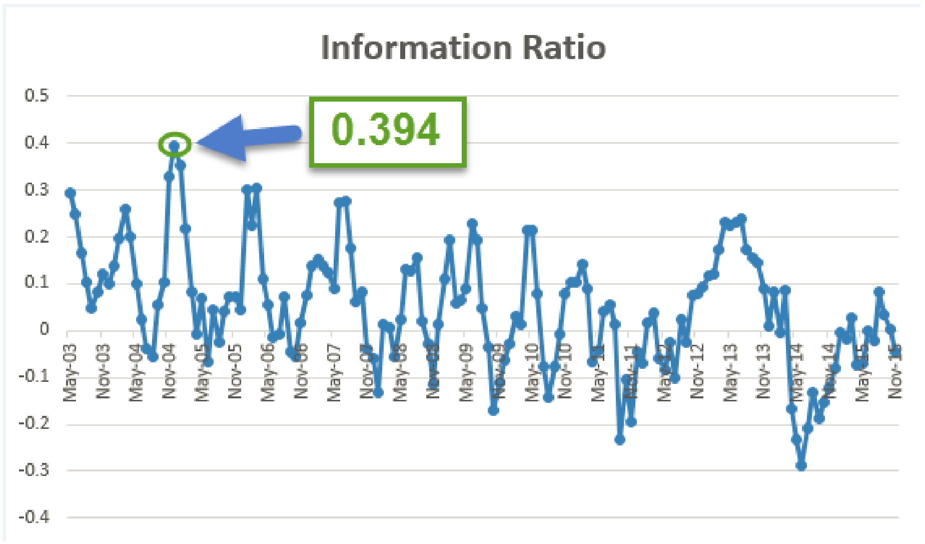


Figure 17: Information Ratio

Comparison of return is not enough, so now we start to talk about information ratio shown in figure 17, which measures STFM to generate excess returns relative to Benchmark. In most of times, information ratio is positive and even peaked at 0.394 Dec 2004, so compared with Benchmark STFM strategy performed better. By looking at the general trend of the plot, we found that it was in a decreasing trend with frequent fluctuations. This shows that the difference in performance between STFM and Benchmark decreased over 12-year period. Furthermore, we noticed that there is an unexpected high peak in 2013 corresponding to the high return of STFM in 2013 mentioned before.

2.4 Summary of Performance Analysis

To sum up, STFM is a high-return strategy. Looking through the 12-year simulation, we found that STFM performed best in 2004 and 2005 with remarkable level in both return and Sharpe ratio. During the past 12 years, STFM performed great and completely beat the benchmark in terms of return, so in long-term STFM is profitable.

....
Printed name: YINAN ZHANG
....
Signature: YINAN ZHANG

3 Part 3 - Analysis of associated risks

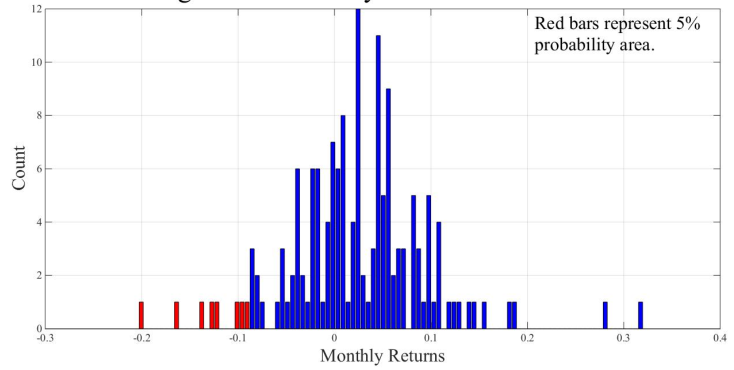
3.1 Value at Risk (VaR)

Generally, there are two types of methods to calculate VaR. Local-valuation methods, of which Delta-normal method is a perfect example, measure portfolio risk by valuing the assets at one point. The second category, Full-valuation methods fully reprice the portfolio under a set of scenarios over a time period. The Historical Simulation method and Monte Carlo method are pretty popular among Full-valuation methods. Compared with Local-valuation methods, Full-valuation methods don't assume the distribution and better solve fat tail problems.

After thorough comparisons, Historical Simulation is chosen to calculate VaR in this report. In a wide-range fluctuating market, Historical Simulation method allows us to better capture all kinds of risks. On the other hand, Monte Carlo method also has a good performance in VaR calculation, while it is difficult to select the accurate model without enough information during such a long period of 12 years. According to a McKinsey report, more than half of VaR calculations are done by Historical Simulation method, so it should be safe to follow the mainstream.

Use Matlab to distribute monthly returns and to mark the corresponding quantiles when confidence level is 99% and 95%. The figure Figure 3.1 shows the monthly return distribution in the last 12 years, while the y label implies the frequency of each bin. Red bars on the left hand represent the worst return with total 5% possibility area. The shape of the distribution is quite similar to that of normal distributions and it suggests the stability of strategy.

Figure 3.1 Monthly Returns Distribution



As shown in the Table 3.1, there is 1% chance for us to lose more than 16128.3 dollars and 5% chance to lose more than 9025 dollars in the next month, with the initial capital as 100000 dollars. The column for Average Return is the product

Table 3.1 VaR Calculation Result (Historical Simulation Method)

	VaR Confidence Level		Average Return
	99%	95%	
1 Month	$16,128.30	$9,025	$2,320
3 Months	$24,795.00	$9,775	$7,040
6 Months	$37,754.00	$20,840	$14,568
12 Months	$39,473.00	$20,000	$30,700

of average returns for different time interval and 100,000 dollars. Through comparisons, we can see that STFM is more suitable to long term trade than short term trade, with less gaps between VaR and Average Return proportionally.

3.2 Max Draw-down

The higher Max Drawdown is, the greater the risk is. Max Drawdown measures the largest single drop from peak to bottom in the value of a portfolio in a specific period and is even more important than volatility to quantitative investments and hedge funds.

Figure 3.2 Cumulative Return with Max Drawdown

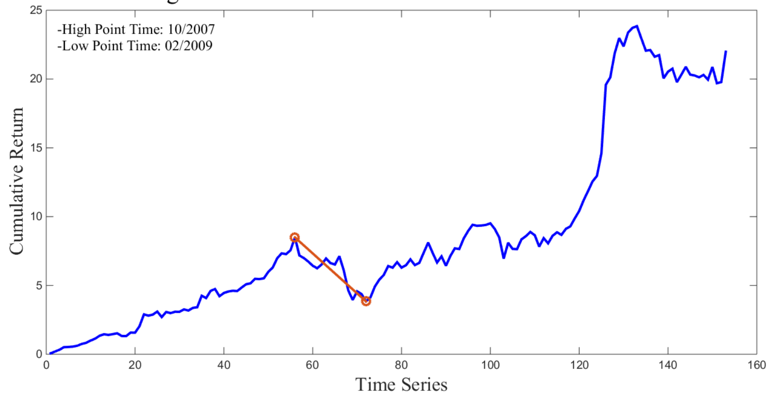


Figure 3.2 displays the movement of STFM's portfolio return and Max Drawdown, which is 59.5%, marked out by Matlab in red color, with the time period from 2003 to 2015. To analyze Max Drawdown, it is important to calculate the time point of Max Drawdown. Based on the Matlab results, the return plunge started from Oct 2007 to Feb 2009, coinciding with subprime crisis and financial crisis. Therefore, STFM strategy cannot exclude all the market risks in an extreme weak market.

However, as a combination of Momentum Investing and Value Investing, STFM has the reason for a high Max Drawdown, because Value Investing strategies always have higher Max Drawdown and the loss can be made up later as long as it chose the right stocks. As is shown in Figure 3.3, the Max Drawdown for 3 Months fluctuated around 10% with the average 10.73%. There is only one large rise from 2008 to 2009, the period of financial crisis. On the whole, Max Drawdown is stable except the situation of an extreme bear market.

Figure 3.3 Max Drawdown for 3 Months

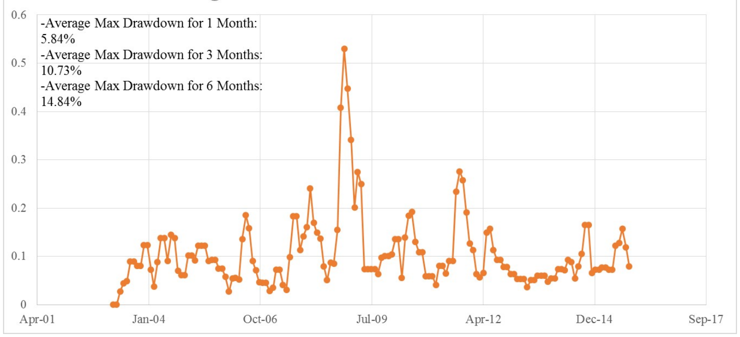


Figure 3.4 Retrace Ratio & Benchmark Return

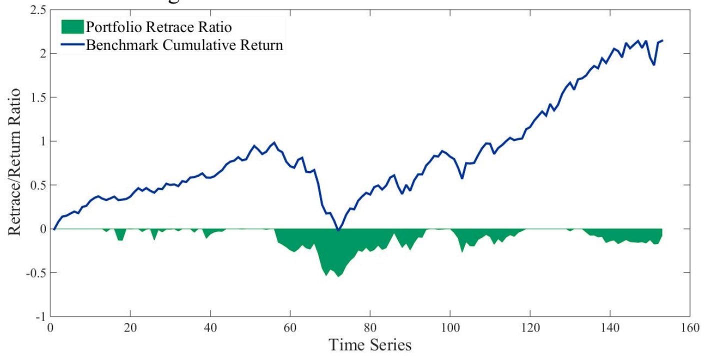


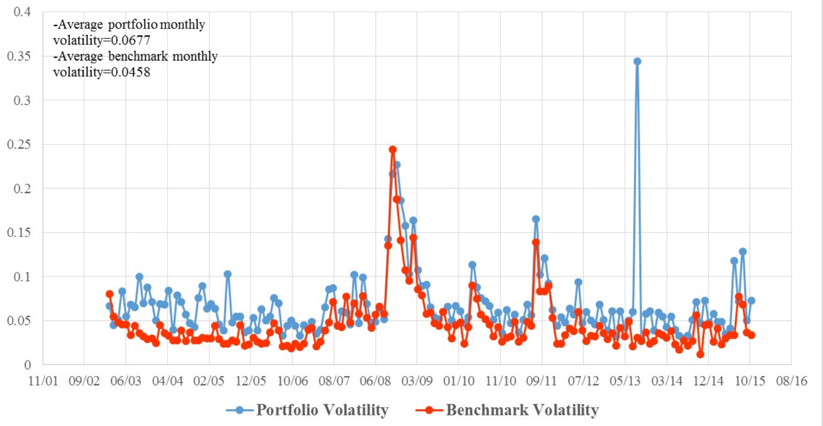
Figure 3.4 is plotted to better examine the relation between overall market situations and the STFM strategy's risk. The retrace ratio is set to be negative so that the cumulative return and retrace ratio will move with the same direction. Obviously, the portfolio's retrace ratio almost simultaneously fluctuates with the performance of the overall market, especially in a bear market. In other words, we should pay more attention to hedge market risks during recession.

3.3 Volatility

By scattering portfolio volatility and benchmark volatility in the same graph, we can see that the portfolio volatility has the similar movements with and are close to benchmark volatility, with the corresponding average monthly volatility 0.0677 and 0.0458. However, there is a huge upswing around 2013 and 2014. This fluctuation is caused by a sudden increase of the portfolio return, so it can be recognized as a good volatility.

This upswing can also be explained by Sharpe Ratio and Sortino Ratio. Sharpe Ratio and Sortino Ratio are widely used to check a portfolio's return generated from per unit risk. The two ratios' difference is that Sortino Ratio excludes the volatility caused by increasing porfolio's value. As is shown by Quantopian, STFM's Shape Ratio and Sortino Ratio are 6.30 and 9.24 respectively. Without the good volatility, the return of every unit risk increases by 46.7%. Therefore, STFM's volatility is decent and acceptable.

Figure 3.5 Portfolio Volatility & Benchmark Volatility

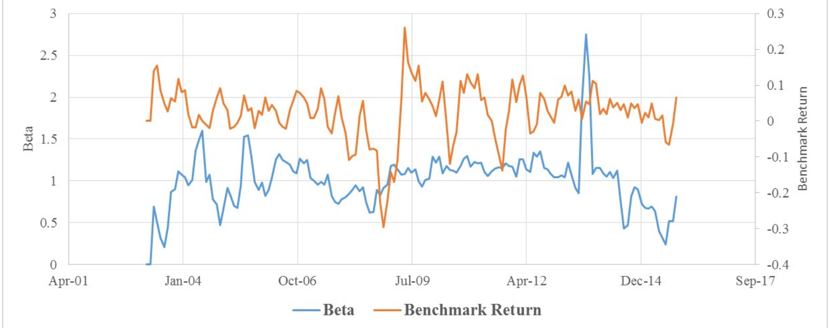


3.4 Beta and Benchmark Return

As discussed in Max Drawdown, the portfolio of STFM seems to move corresponding with the market in a high level. To have a further insight of this problem, the analysis for Beta is essential. With the time period of 12 years, the portfolio Beta equals to 1, which implies that the portfolio is a market portfolio. From the average aspect, Beta as 1 is a moderate and safe number.

In Figure 3.6, Beta is plotted with Benchmark Return with different coordinates. Apparently, Beta fluctuates in a wide range from 0 to 3. In an ideal situation, Beta should be high in a bull market and be low in a bear market. In other words, Beta should move with Benchmark in the same trend. However, STFM's Beta rises during recession period around 2007 to 2009 and drops when Benchmark keeps increasing, for example in the second half of 2004. Therefore, STFM does not seem very good at estimating the market trend and choosing the right time point occasionally even though it has a really good performance in the long run.

Figure 3.6 Beta & Benchmark Return



To verify the conclusion above, apply H-M (Henriksson and Merton) Model. The model assumes a good hedge fund manager should adjust the capital allocation to reduce Beta in advance so that loss will be diminished when $Rm < Rf$. In other words, there are two Betas for a bull market and a bear market respectively. D in the model is a dummy variable, which equals to 1 when $Rm > Rf$. If $\beta_{i2>}0$, it means the manager make a higher Beta in a bull market and a lower Beta in a bear market.

$$R_i - R_f = \alpha + \beta_{i1}(R_m - R_f) + \beta_{i12}(R_m - R_f)D + \epsilon_i$$

Table 3.2 Regression Summary Output for H-M Model

Regression Statistics	
Multiple R	0.625903693
R Square	0.391755433
Adjusted R Square	0.383645506
Standard Error	0.056429883
Observations	153

ANOVA

	df	SS	MS	F	Significance F
Regression	2	0.307642509	0.153821	48.30566	6.40E-17
Residual	150	0.47764975	0.003184		
Total	152	0.785292259			

	Coefficients	Standadr Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept	0.021105745	0.007089459	2.97706	0.003394	0.007097645	0.03511384	0.00709764	0.035113844
Rm-Rf	1.314204851	0.203002821	6.473826	1.28E-09	0.913090494	1.71531921	0.91309049	1.715319208
(Rm-Rf)*D	-0.442234674	0.340985203	-1.29693	0.196646	-1.115989154	0.23151981	-1.1159892	0.231519807

Table 3.2 shows the summary of regression result. β_{i2} is -0.4422 which means STFM indeed has some drawbacks in the ability to choose the opportunity in market. However, the tStat and P-value of β_{i2} implies the H-M model doesn't fit the data perfectly. So the conclusion is still indeterminate. But if we can obtain more data, H-M Model may provide a more reliable answer for β_{i2} because the sample size is only 153 and with the time interval as one month here.

3.5 Summary of Risk Analysis

In brief, STFM has a rather low risk compared with strategies that use leverage and have narrow instruments selections. As mentioned in Max Drawdown and Beta part, STFM's main risk is market risk, which can never be eliminated. Holding defensive stocks is one way to reduce market risk when there is an expectation for a bear market.

STFM can stand out because it operates without any leverage. No leverage has already reduced a large amount of risks. So even if the Max Drawdown during financial crisis reaches 59.5%, there's no enlargement effects of loss due to leverage at least. Usually the leverage should be limited within 3, so STFM is excellent to beat the benchmark without leverage.

STFM also operates a wide stocks selection, which helps to diversify risks. ARMA Model and Technical Model are applied to the whole universe of market. According to the backtest, 490 stocks are traded during the last 12 years. Stocks from different industries will reduce the effects of industrial risks.

However, as discussed in the Max Drawdown and Beta parts before, STFM has drawbacks in following the market trend and resisting market risk. Beta fails to adjust based on market performance. In other words, STFM may have a dissatisfactory return in a bear market. So STFM should be carefully applied during recession and be compensated by remedial actions.

Suggestion for controlling market risks under STFM is to hold defensive stocks when there is an expectation for a bear market. ARMA Model and Technical Model cannot eliminate all the lags in price estimation. So we can supplement STFM actively according to our judgement of market trends. With the expectation for a bear market, we should buy and hold stocks with negative Beta or small Beta (lower than 0.5), for example the stocks in food, medical and entertainment industries. Defensive stocks are less related to macro-economic climate index or even negative related to the performance of market. It would be helpful to solve the problem that STFM's portfolio Beta rises when market worsens sometimes.

....
Printed name: CHENRUI WANG
....
Signature: CHENRUI WANG

4 Part 4 - Algorithm Implementation

4.1 The Code Listing

The Code of the STFM strategy can be divided in the following sequence:

- Header
- RSI & CCI Definition
- ARMA Genaration
- Value Momentum Filter
- Initialization on Trade
- Reblance the Portfolio
- Buy&Sell Trigger
- ARMA Prediction

Header

```
# GS06 Financial Instutition and Markets Quantitative Trading
    Project
# Group 11 - STFM (Statistical-Technical Forecasting with
    Momentum) Strategy
# Project Leader: Yichen Wang
# Code Author: Meng Zhang, Gabriel Bila
# Group Strategy: Yichen Wang, Meng Zhang
# Performance Analysis: Yinan Zhang
# Risk Management: Chenrui Wang


import pandas as pd
import numpy as np
import datetime
import math
import talib
from statsmodels.tsa.arima_model import ARMA
from statsmodels.tsa.arima_model import ARIMA
```

. . . .
Printed name: MENG ZHANG
. . . .
GABRIEL BILA
. . . .
Signature: MENG ZHANG
. . . .
GABRIEL BILA

RSI & CCI Definition

```python
#calculating rsi for a security
def rsi_det(context, data, security):

    a=-10
    rsi_return = {}
    if security in data and not np.isnan(data[security].price)
        and 'price' in data[security]:

        rsi2 = ta.RSI(timeperiod = 14)
        rsi_data = rsi2(data)
        if np.isnan(rsi_data[security]):
            return a
        elif rsi_data[security] > context.HIGH_RSI:
            rsi_return[security] = -1
        elif rsi_data[security] < context.LOW_RSI:
            rsi_return[security] = 1
        elif rsi_data[security] <= context.HIGH_RSI and
            rsi_data[security] >= context.LOW_RSI:
            rsi_return[security] = 0
        else:
            rsi_return[security] = a
        return rsi_return[security]
    else:
        return a

#calculating cci for a security
def cci_det(context, data, stocks):
    cci_dict = {}

    if stocks in data and not np.isnan(data[stocks].price):
        cci2 = ta.CCI(timeperiod = 14)
        cci_result = cci2(data)

        if np.isnan(cci_result[stocks]):
            return -10
        elif cci_result[stocks] > context.HIGH_CCI:
                cci_dict[stocks] = -1
        elif cci_result[stocks] < context.LOW_CCI:
             cci_dict[stocks] = 1
        else:
             cci_dict[stocks] = 0
        return cci_dict[stocks]
    else:
        return -10
```

ARMA Generation

```python
# generate ARMA model for price history
def generate_arma(price_history):
    for i in range(4,0,-1):
        for j in range(2,0,-1):
            try:
                arma = ARMA(price_history, (i,j))
                arma_results = arma.fit(disp=0)
                return arma_results
            except:
                continue

    log.info('could not fit arma model')
    return None
```

Value Momentum Filter

```python

#define the universe using value momentum trading strategy
def before_trading_start(context, data):

    #this code prevents query every day
    if context.month_count != context.holding_months:
        return

    fundamental_df = get_fundamentals(
        query(
            fundamentals.valuation_ratios.ev_to_ebitda,
 fundamentals.asset_classification.morningstar_sector_code, fundamentals.valuation.enterprise_value, fundamentals.income_statement.ebit, fundamentals.income_statement.ebitda, fundamentals.valuation_ratios.pb_ratio
        )
        .filter(fundamentals.valuation.market_cap > 1e6)
        .filter(fundamentals.valuation.shares_outstanding != None)
        .filter(fundamentals.valuation.shares_outstanding < 2e8)
        .filter(fundamentals.company_reference.country_id != "CHN")
        .filter(fundamentals.company_reference.business_country_id != "CHN")
        .filter(fundamentals.income_statement.ebitda > 0)
        .filter(fundamentals.valuation_ratios.pe_ratio > 1)
        .filter(fundamentals.valuation_ratios.pe_ratio < 15)
        .filter(fundamentals.valuation_ratios.fcf_ratio < 30)
        .filter(fundamentals.valuation_ratios.ev_to_ebitda < 30)
        .filter(fundamentals.valuation_ratios.ps_ratio < 5)
        .filter(fundamentals.operation_ratios.roe > 0.1)
        .filter(fundamentals.operation_ratios.total_debt_equity_ratio < 1)
        .filter(fundamentals.operation_ratios.current_ratio > 1)
        .order_by(fundamentals.valuation_ratios.pe_ratio.asc())
        .limit(context.num_screener)
    )

    # Filter out only stocks that fits in criteria
    context.stocks = [stock for stock in fundamental_df  ]
    # Update context.fundamental_df with the securities that we need
    context.fundamental_df = fundamental_df[context.stocks]

    update_universe(context.fundamental_df.columns.values)
```

Initialization on Trade

```python
def initialize(context):
    #the constant for portfolio turnover rate
    context.holding_months = 1
    #number of stocks to pass through the fundamental screener
    context.num_screener = 800
    #number of stocks in portfolio at any time
    context.num_stock = 50
    #number of days to "look back" if employing momentum. ie formation
    context.formation_days = 200
    #set False if you want the highest momentum, True if you want low
    context.lowmom = False
    # high and low CCI
    context.HIGH_CCI = 100
    context.LOW_CCI = -100
    #high and low RSI
    context.LOW_RSI = 30
    context.HIGH_RSI = 70

    context.history_look_back = 180 # look back by 180 days when generating ARMA model
    context.stock_price_predict_distance = 30 # predict stock price for 5 days

    context.price_increase_threshold = 0.009 # threshold for deciding whether to buy or sell or keep stock when predicting price increase

    #month counter for holding period logic.
    context.month_count = context.holding_months

    # Rebalance monthly on the first day of the month at market open
    schedule_function(rebalance,
                      date_rule=date_rules.month_start(),
                      time_rule=time_rules.market_open())
```

Rebalance on Portfolio

```python

def rebalance(context, data):
    #This condition block is to skip every "holding_months"
    price_history = history(bar_count=context.formation_days,
        frequency="1d", field='price')
    if context.month_count >= context.holding_months:
        context.month_count = 1
    else:
        context.month_count += 1
        return

    chosen_df = calc_return(context, price_history)
    price_arma = arma_predict(context, price_history)
    if context.num_stock < context.num_screener:
        chosen_df = sort_return(chosen_df, context.lowmom)

    log.info('price arma: \%s' \% price_arma.values())
    chosen_df = chosen_df.iloc[:,:(context.num_stock-1)]

    arma_good_stock_increase_ratio = {stock: inc_ratio for
        stock, inc_ratio in price_arma.items() if inc_ratio >
        context.price_increase_threshold}
    #get good stock using Arma
    arma_good_stock = arma_good_stock_increase_ratio.keys()

    # Create weights for each stock
    a = 0
    weight = 0.99/len(chosen_df.columns)
    # strategy for selling stocks
    for stock in context.portfolio.positions :
        if context.portfolio.positions[stock].amount>0:
            if stock in data :

                rsi_dict = rsi_det(context, data, stock)
                cci_dict = cci_det(context, data, stock)
                #Sell the stock using technical strategy and
                    arma strategy
                if (stock not in chosen_df  or (cci_dict > -5
                    and cci_dict < 0 and rsi_dict < 0 and
                    rsi_dict > -5) or (stock.end_date -
                    get_datetime()).days < 30) and stock not
                    in arma_good_stock:
                    order_target(stock, 0)
```

Buy&Sell Trigger

```

    # strategy for buying stocks
    for stock in chosen_df:

        if weight != 0 and stock in data and (stock.end_date -
             get_datetime()).days > 30 and (stock.start_date -
              get_datetime()).days < 30:
            a = a + 1
    if a!= 0:
        weight2 = 0.99/a
    else:
        weight2 = weight
        #using arma strategy for buying
    for stock in chosen_df:
        if (weight != 0 and stock in data and (stock.end_date -
              get_datetime()).days > 30 and (stock.start_date
            -  get_datetime()).days < 30) or stock in
            arma_good_stock:
            order_target_percent(stock, weight2)

def sort_return(df, lowmom):

    df = df.T
    df = df.sort(columns='return', ascending = lowmom)
    df = df.T

    return df

def calc_return(context, price_history):

    temp = context.fundamental_df.copy()

    for s in temp:
        now = price_history[s].ix[-20]
        old = price_history[s].ix[0]
        pct_change = (now - old) / old
        if np.isnan(pct_change):
            temp = temp.drop(s,1)
        else:
            temp.loc['return', s] = pct_change
            #calculate percent change

    return temp
```

ARMA Prediction

```python
#predict arma result
def arma_predict(context, price_history):

    now = get_datetime()
    stock_increase_ratio = {}

    for stock in context.stocks:
        prices = price_history[stock]

        arma_results = generate_arma(prices)
        if arma_results == None:
            log.info('Could not fit ARMA model for stock \%s;
                skipping it altogether :(' \% str(stock))
            continue

        predict_index = len(prices) - 1 + context.
            stock_price_predict_distance
        current_price = prices[-1]
        predicted_price = arma_results.predict(now,
            predict_index).tolist()[-1]
        inc_ratio = (predicted_price - current_price) /
            current_price
        stock_increase_ratio[stock] = inc_ratio

    return stock_increase_ratio

def handle_data(context, data):

    record(num_positions = len(context.portfolio.positions))
```