Index Tracking in the Structure of Fund of Funds based on Cointegration

by

Juntao Zhang

B.Sc.,The University of Western Ontario, 2016

Thesis Major Research Paper Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Business Economics in the

Department of Economics

Faculty of Graduate Studies

Brock University

August, 2018

Approval

Name: **Juntao Zhang**

Degree: **Master of Business Economics**

Title: **Index Tracking in the Structure of Fund of Funds based on Cointegration**

Supervisory Committee: **Ivan Medovikov**
Supervisor
Assistant Professor _____

B _____

Date Approved:

Agreement

In presenting this major research paper in partial fulfillment of the requirements for an advanced degree at Brock University, I agree that the Department of Economics shall have a right to make it freely available for reference and study. I further agree that permission for extensive copying of this research for scholarly purposes may be granted by the Graduate Program Director of the Master of Business Economics program or by his or her representatives. It is understood that copying or publication of the thesis for financial gain shall not be allowed without my written permission.

Department of Economics
Brock University
500 Glenridge Ave.
St. Catharines, Ontario
L2S 3A1 CANADA

Signature: _____ Date: _____

Abstract

Keywords:

Dedication

Thank God

Acknowledgements

Thank Prof.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the investment industry, there are two opposite equity portfolio management philosophies, active portfolio management, and passive portfolio management. Active investment is aiming to beat the benchmark index based on professional analytic and the fund manager's judgment in picking securities and determining the right moments to trade. A hedge fund is a typical active management portfolio. However, it is not as easy as planned in theory for active funds to outperform respective benchmarks. According to SPIVA® statistics & reports U.S. (S&P Dow Jones Indices), around 83.18% of all domestic funds under performed their benchmark for the 10-year period by the middle year of 2015 in the US. Besides us stock market, many active funds failed to beat the targeted indices over the 10-year period in other major capital markets.

On the contrary, passive investment pursues the same performance as a targeted index over a long period of time instead of trying to beat the benchmark. A tracking fund is a classic passive product whose mission is to mimic a specified benchmark passively with a buy-and-hold strategy, the benchmark can be a stock index, a commodity, bonds, even bitcoin. As opposed to under performed active funds, index tracking funds attracted more and more investors and grown significantly through nearly a decade bull market since the worst situation in early 2009. Plenty of capital flows into passive funds rapidly.

According Morningstar's Research (Morningstar, 2017), investors poured more than $692 billion into index funds across all asset classes in 2017. For the same period, actively managed funds experienced $7 billion in outflows. Now the total asset in index funds including index mutual funds and index ETFs is about 1112 trillion in the US.

There are two main conventional methods to track stock indices. One is called full replication in which the fund can take a long position on all the constituents of an index in the respective weights with a buy-and-hold strategy. The full replication approach is straightforward to implement and can achieve the precise tracking performance as long as fund managers rebalance the weights once a while. Even though full replication can closely track indices in theory, it has a few nonnegligible flaws. Full replication funds need to be rebalanced quite often with high volatility stock weights, which could lead to inflated costs. Liquidity is another issue, especially for some small capitalization stocks, this may affect fund construction and increase the transaction costs. In general, low cost is a signature characteristic of passive management funds, but full replications funds cannot exhibit this feature steadily. The other traditional tracking approach is known as sample replication. Some indexes may contain a large number of constituents, such as S&P Global 1200, Russell 2000. In such cases, full replication approach is not efficient to conduct, however, sample replication methodology can be appropriate. Sample replication funds are designed to long part of total stocks that could represent the underlying index based on correlations, risks, and returns. The sample replication funds trade comparatively fewer constituents, which could significantly reduce the costs, but this may potentially cause higher tracking errors.

In addition to traditional physical funds holding a portfolio of assets, there is another alternative approach so-called synthetic portfolio to replicate the performance of an index by using corresponding derivative and swaps rather than holding stocks directly.

Proponents claim that synthetic fund is a better financial instrument than traditional tracking funds to track illiquid indexes at a low cost and small tracking error. Whereas, the synthetic portfolios are born with a few risks involving counter party credit risk, liquidity risk, and collateral risk. Synthetic funds are not popular in the US markets due to strict regulations from the US Securities and Exchange Commission.

The goal is to construct a portfolio aiming to track S&P 500 Total Return Index (SPTR) on the fund of funds(FoF) structure by using cointegration analysis. Cointegration is a powerful econometric tool that could ensure the long-run equilibrium relationship between the portfolio and the underlying index. As an approach of sample replication, fund managers could buy sector and industry ETFs to mimic the SPTR index which consists of 11 different sectors and industries. There are numerous sector ETFs on the market, many of them have over a decade of history, large asset size, and high liquidity. I am going to select about 5 adequate ETFs from each sector to form an ETFs pool, and then design a portfolio based on lasso regression to select ETFs and find corresponding weights in the cointegration system.

This paper is organized as follows. Section 2 reviews other literature and compares different tracking methodologies. Section 3 describes the targeted index S&P 500 Total Return Index and sector ETFs on the market. Section 4 introduces the tracking methodology cointegration and variable selection asset allocation method lasso regression. Section 5 shows strategy implementation and empirical results for the tracking portfolios with varying numbers of ETFs and different rebalance strategies. In the end, section 6 makes the conclusion for the tracking strategy, in addition, I will discuss the limitations and potential research extensions based on this paper.

Chapter 2

Literature Review

Alexander and Dimitriu (2005) are pioneers who applied cointegration to passive portfolio management field. This paper deployed cointegration analysis to track a stock index, then built the long short market neutral strategy using enhanced index tracking. The object of using cointegration is to identify any common stochastic trends in stock prices and then achieve stationary tracking errors between a portfolio of stocks and the stock index over the long run.

The author divided the process of constructing index tracking portfolio into two parts, selection, and allocation. This paper took 'brute force' approach to select stocks. Firstly, pick the number of stocks to form the portfolio, then use all the combinations of stocks as possible portfolios. Next step is to optimize the weights of each stock from every possible combination by using the Engle-Granger cointegration regression.

Besides index tracking, this paper amplified cointegration methodology to long short market neutral strategy, which consists of a long portfolio tracking index plus, and a short portfolio tracking index minus. This long short strategy, as one of the statistical arbitrage strategies, could provide double alpha opportunities in stock markets. Vast

back testing results confirmed that Engle-Granger cointegration is a sound methodology to build index tracking portfolios with relative few stocks and fewer turnover rates.

Glova, Pastor, and Sabol(2015) explored cointegration and discovered its application in passive portfolio management. They discussed the statistical characteristics of cointegration and compared it with correlation from an asset management perspective. They noted that cointegration and correlation are related, both describe the relationship between assets. Cointegration is a long-term relationship among time series. If cointegration relationship exists, then it could ensure long-run equilibrium between stock prices. Correlation is a short time statistic based on assets' returns, that is not appropriate for constructing a long-term buy and hold portfolio.

This literature tracked Dow Jones Industrial Average Index and Dow Jones Composite Average Index by exploiting the mean-reverting property of cointegration. They used daily closed prices of indices and daily closing prices of component stocks adjusted for splits and dividends from 2000 to 2013. This paper conducted a lot of portfolios from the different selection process and compared returns and risk metrics of each portfolio. In the end, they proved that cointegration is a mature and stable methodology in passive portfolio management, which can create a comparable low volatility and low-cost tracking fund.

Sant'Anna, Filomena, and Caldeira (2017) compared cointegration and correlation approaches in index tracking and enhanced index tracking on the Brazil Ibovespa index and the US S&P 100 index. This paper pointed out that both methodologies are outperforming for index tracking portfolios, but no significant advantages turn towards neither method for enhanced index tracking.

The authors constructed a series of portfolios consists of at most 10 stocks by different

combinations between in sample and out of sample data intervals through both approaches. Then they found different patterns between Brazilian and US stock markets. There is a trade-off between tracking performance and costs in the Brazilian market, which is correlation based portfolios have larger average tracking errors, but smaller turnover values, on the other hand, cointegration based portfolios have smaller tracking errors, but higher turnover rates accompanied higher cost. However, no empirical evidence revealed the similar features on the S&P 100 index, tracking results did no favor either correlation no cointegration.

Overall, this paper failed to find robust evidence to demonstrate different characterizes of cointegration and correlation in the passive portfolio management area. It is worth noting that all portfolios have only 10 assets, which may be a potential reason why this paper did not generate strong findings.

Numerous studies proved that cointegration is a sound and robust methodology to track an index. However, there are another one crucial problems affect the tracking performance, construction costs, rebalance costs: asset selection. Asset selection is a picking art for fund managers. For the index-tracking funds, assets selection helps to selection appropriate subset of assets out of total assets pool to represent the index, moreover, funds can allocate different proportions of total capital to each asset in the portfolio. Alexander (2001)(cointegration and asset allocation) proposed the 'brute force' approach to select assets. The author tested all possible combinations of a fixed number of stocks in a portfolio.

Therefore, the 'brute force' method requires huge computing power. While tracking some other indexes contain large number of constitutes, it may cause explosive growth of computing, so that it is not applicable. For instance, the Russell 2000 index has 2000 stocks, if investors want to pick 1200 stocks to construct a tracking fund, they might

apply the combination formula $\frac{N!}{k!(N-k)!}$, where N is the total number of securities, and k is the number of ones selected. The number of possible portfolios is an incredible giant, then the 'brute force' becomes computationally infeasible even using modern supercomputers.

There are countless approaches to the asset selection problem. For linear regression, penalty methods are widely used as an effective statistical modelling technique. Regression with L1 penalty term is known as the least absolute shrinkage and selection operator (Tibshirani, 1996), lasso for short. For more details, I will discuss it in the methodology section. Many academies tested the tracking funds by using nonnegative-lasso method such as Wu et al. (2014) and Wu and Yang (2014). Yuehang Yang, Lan Wu (2016) proposed a two-stage nonnegative adaptive lasso method to do asset selection in ultra-high dimensional regression models based on adaptive lasso algorithm proposed by Zou (2006), which can deal with hundreds even thousands of stocks. They tracked the CSI 300 Index that is a major index in Chinese stock market by using long and hold sample replication strategy. They did not use cointegration to ensure the long run equilibrium between tracking fund and the CSI 300 Index. The first stage solved the asset selection problem, they used nonnegative adaptive lasso method to select 30 stocks out of total 300 stocks, the number 30 is a predetermined number. Once they determined the stocks, the second stage solved the asset allocation problem. They applied the nonnegative OLS method to estimate the weights of the 30 stocks in the tracking fund. The authors did not show a long time tracking performance, the results for short time were satisfactory.

Chapter 3

Methodology

Cointegration analysis is a widely used time series tool to identify and utilize the properties of many time series that share a common stochastic trend and the long run equilibrium relationship. The concept of cointegration was initially suggested by Granger in 1981, then in a seminal paper, Engle and Granger (1987) developed cointegration estimation procedures and tests.

In time series econometrics, the most important concept is stationarity, models are built on stationary time series processes. Unfortunately, the original stationary time series are very rare in finance. This paper deploy weakly stationary time series in the paper. The weakly stationary process is a stochastic process whose mean and variance are finite and do not change over time, denoted by $I(0)$. That is:
A series $X_t$is weakly stationary if it satisfies following conditions

- $E(X_t) = \mu$, where μ is independent of time t
- $Var(X_t) = \sigma_x^2$, where σ_x is finite constant and independent of t.
- $Cov(X_t, X_{t-s}) = \gamma_s$, where γ_s is independent of t for all s

Generally, the financial returns are weakly stationary time series process; nevertheless, many financial data like stock prices, interest rates, exchange rates are not stationary processes. A non-stationary time series X_t is called integrated of order d, if it can be made stationary by differencing d times, denoted by $X_t\ I(d)$.

Typically, log stock prices are a random walk with drift, integrated of order 1 $I(1)$, the models use log prices where the logarithm transformation could produce the continuous returns and make the price series to be more stable. It has the form:

$$X_t = \mu + X_{t-1} + \epsilon_t \tag{3.1}$$

where $\epsilon_t\ iid(0, \sigma^2)$ is the white noise.
The constant μ is called drift, and $\mu = E(X_t - X_t - 1)$, which is the time trend of log prices.

As for multivariate time series, each of them is integrated of the same order, but some linear combination of them has lower integration order, then these time series are cointegrated.
The more general form is for a N dimensional variable X, and $X_i\ I(d)$, i = 1,2,..N. Exist 1 or more linear combinations $Z_t = \alpha' X_t$ s.t. $Z_t\ I(d-b)$ where b > 0, then X is cointegrated $X\ CI(d, b)$

For example, both X_t, Y_t are random walker processes $I(1)$, $Z_t = X_t - \alpha' Y_t\ I(0)$. So X_t, Y_t are cointegrated, and the coefficients [1, α] is called the cointegration vector.
Cointegrated time series have may nice properties and characteristics, here X_t, Y_t have the common stochastic trends, they will move together in the long run equilibrium state. And the Z_t is the short-term deviation from the equilibrium, which is the mean-reverting property that index tracking funds benefit from.

When applying the non-stationary time series into a regression model, it may cause misleading statistical inferences among them, which is called spurious regression discussed by Granger and Newbold (1974). The regression model mistakenly provides a non-existing relationship between independent regressors and response variable with statistically significant coefficients and high R^2. Because two non-stationary stochastic processes move together does not mean that they are related, it raises the spurious correlation.

Cointegrated time series can avoid the problem of spurious correlation. Since the cointegrated time series share the common stochastic trend and the linear combination is a stationary process, they will not deviate far away.

There are many methods to test the presence of cointegration relationship in multiple time series, Engle and Granger (1987), Johansen (1988) and Johnsen and Juselius (1990). In this paper, I applied Engle-Granger test method. For more information about estimates and tests of cointegration system in not part of this paper, so not discuss it in details.

The Engle-Granger method tests the null hypothesis of no cointegration among the time series. Engle-Granger method requires all the time series have the same number of integration, then essentially perform unit root test on the residual term of the cointegration regression. If residual term is stationary process, then the corresponding time series are cointegrated, and vice versa. For instance, X and Y are $I(1)$, then regress X to Y by using ordinary least square.

$$Y_t = \beta_0 + \beta_1 Xt + \epsilon_t \tag{3.2}$$

Then apply augmented Dickey-Fuller root test on ϵ_t. Null hypothesis of ADF test is the

presence of unit root, which indicates that the time series is not stationary. Only residuals are stationary, the conclude that X and Y are cointegrated can be made.

Beside cointegration analysis ensures long run equilibrium state, least absolute shrinkage and selection operator (lasso) regression is a key technique to select proper assets and allocate the capital.

For a usual multiple linear regression:

$$y_i = \beta_0 + \beta_1 x_{1t} + \beta_2 x_{2t} + ... + \beta_p x_{pt} + \epsilon_t \tag{3.3}$$

for i = 1,2...n

The classic approach to estimate the unknown coefficients is ordinary least square(OLS) method, which is to minimize the sum of squares of the differences between a response variable and each predicted value by the estimated function. The explicit form is:

$$\begin{aligned} RSS &= \sum_{i=1}^{n} (y_i - \widehat{y}_i)^2 \\ &= \sum_{i=1}^{n} (y_i - \widehat{\beta}_0 - \widehat{\beta}_1 x_{i1} - ... - \widehat{\beta}_2 x_{i2} - \widehat{\beta}_t x_{it})^2 \end{aligned} \tag{3.4}$$

OLS is a classic and mature algorithm to solve multiple linear regression, however, for high dimensional data, the model that is easier to interpret and less complexity by shrinking some coefficient estimates to 0, which is equivalent to exclude those variables.

There are many alternative approaches to do variable selection and shrinkage: best subset selection, step-wise selection, ridge regression, lasso regression and elastic net.

This paper adopted lasso regression.

$$\begin{aligned}\beta_L &= \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{t}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{t}|\beta_j| \\ &= RSS + \lambda\sum_{j=1}^{t}|\beta_j|\end{aligned} \tag{3.5}$$

Notice that for any λ ¿0, there exists a s, equal to || where:

$$\beta_\lambda = \underset{b}{argmin}||y - Xb||^2, s.t.||b|| <= s_\lambda \tag{3.6}$$

This is the dual form of the optimization problem.
the only difference equation 3.5 and equation 3.6 is that the objective function contains a penalty term that is defined as the sum of the absolute value of coefficient estimates.

The $\lambda\sum_{j=1}^{t}|\beta_j|$ is called the L1-norm, and the λ is called a tuning parameter that can control the shrinkage degree to coefficient estimates. When λ is 0, the penalty term does not affect the objective functions, which is the same as normal OLS. On the contrary, as λ is getting larger, the impact of the shrinkage penalty grows, the coefficients estimates are approaching 0. With penalizing the coefficient estimates, some variables are removed out of the model, so a subset of variables is obtained. As $\lambda = \infty$, all the coefficients will be 0. Therefore, the choosing an appropriate λ is critical to the model.

In general, there is no simple closed-form solution to lasso regression. There are several optimization methods to solve lasso regression, such as coordinate descent, least angle regression. I applied coordinate descent algorithm by using Sci-kit Learn package in Python.

Chapter 4

Data

For empirical analysis, constructing funds to track the S&P 500 Total Return Index(SPTR) by using sector ETFs. All funds used daily close prices of SPTR and these ETFs adjusted for paying dividends and stock splits, and the adjusted closing prices are easy for us to perform analysis of historical returns. 10-year data from the beginning of 2008 to the end of 2017, this period contains 2518 trading days, downloaded data from Yahoo! Finance.

S&P 500 Total Return Index is a very similar index to the standard S&P 500 index. They have the same components that comprise 500 large capitalization companies listed on NYSE and NASDAQ and these constituents are categorized into 11 sectors: consumer discretionary, consumer staples, utilities, technology, health care, financial, energy, telecommunication, industrial, material, and real estate. The total market capitalization of the 500 companies was about 23 trillion dollars at the end of 2017, which could cover over 80 percent of US stock markets. This list could be considered as a proper representation of US stock markets and a critical indicator to reflect the business cycle.

Table 4.1: 10 Years SP 500 Dividend Yields

Year	2008	2009	2010	2011	2012
Dividend Yield	3.11%	2.00%	1.84%	2.07%	2.13%
Year	2013	2014	2015	2016	2017
Dividend Yield	1.89%	1.92%	2.11%	2.01%	1.86%

Both SPTR and SPX are calculated in a capitalization-weighted method in which the 500 constituents are weighted based on the market value of their outstanding shares. Especially, the SPX is calculated by taking the sum of the adjusted market capitalization of all the stocks and then dividing the summation with a particular index divisor that is made by Standard and Poor's. However, the only difference in calculation of SPTR is that all the dividends are reinvested into the entire index rather than being used to purchase the stocks that paid dividends. This is known as the index-wide reinvestment approach.

In the past 5 years, more than 400 companies out of the SP 500 paid cash dividends every year. Take 2017 as an example, total 432 companies returned about $420 billion to investors that is a record amount of dividend payments, and the yearly dividend yield is about 1.86%. According to Table 4.1, it has SP 500 dividend yields past 10 years, the dividend yield varied between 1.84% to 3.11%. Due to the enormous market capitalization, the small value of dividend yield still means huge capital. Therefore, the cumulative impact of dividends could be tremendous over a long period of time.

Figure 4.1 exhibits the adjusted closed prices of SP 500 and SPTR in the past 10 years. The SPTR index was 2273.4 in green and the SPX index was 1447.16 in red on the first trading day of 2008, the ratio of SPTR divided by SPX was 1.571. And then the indices were 5239.6 and 2687.5 respectively on 2017-12-29, the ratio was 1.95, which means the SPTR index was almost twice of the SPX index. SPTR is always on top of SPX, and the preponderance of SPTR is getting more and more distinct with stock market are stronger gradually.

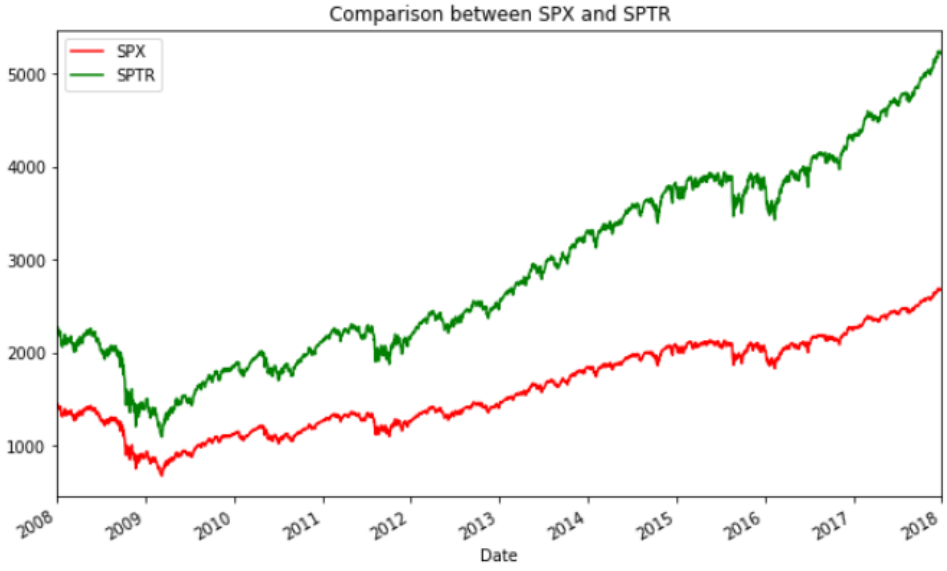


Figure 4.1: Comparison of Prices between SPX and SPTR

As shown in figure 4.2, the long-term return of the SPTR is remarkable indeed. Suppose 100 dollars were invested in the SPTR and the SPX at the beginning of 2008, and the investment was worth $183.98 and $161.38 by the end of 2017. The extra 22.6% returns of the SPTR over the SPX was from the dividends were reinvested into the SPX.

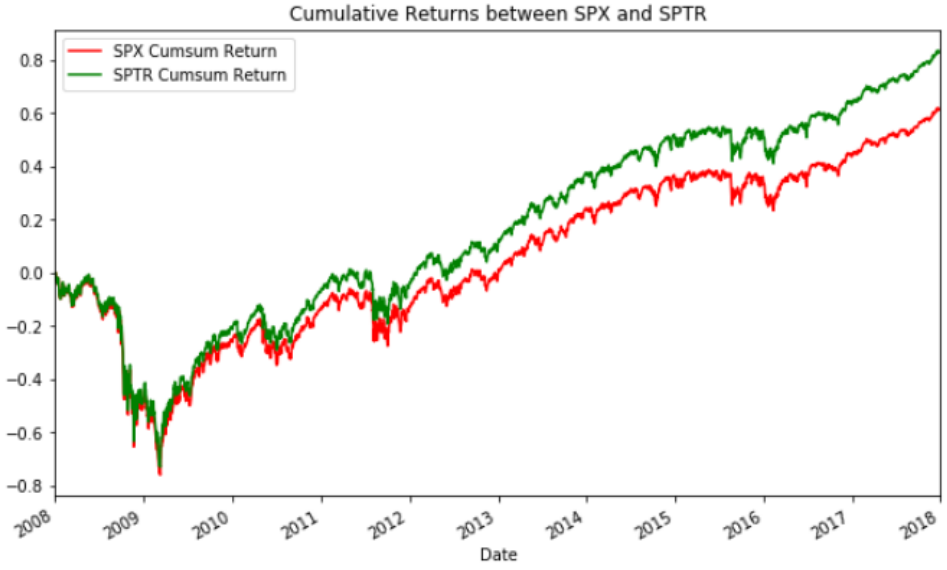
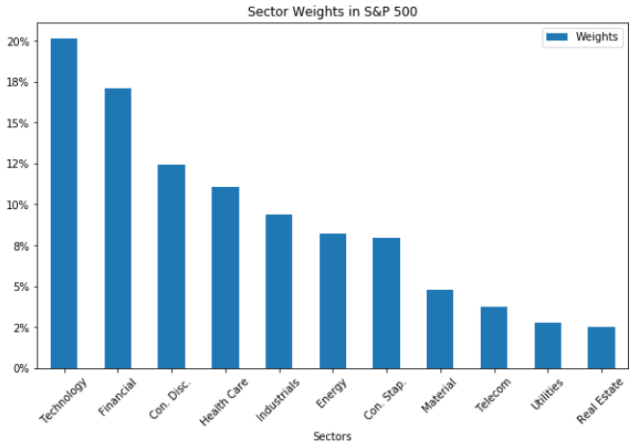


Figure 4.2: Comparison of Cumulative Returns between SPX and SPTR

The 500 constituent companies are divided into 11 sectors, and the weights of different sectors change over the time. Figure 1.2 is a snapshot of the sector's relative percentage of the total market capitalization of SPTR. The technology sector is the biggest, it has about 9 trillion dollars market capitalization, it weights up to 20%. The second position is the financial sector, the proportion is about 17%, and the financial sector used to be the largest for a long time. As in the table, the utility and real estate are the two smallest sectors, both sectors have about the same capitalization, only contribute about 5% of total market capitalization. It is worth noting that real estate was emancipated from the financial sector as a standalone sector since September 2016.

Figure 4.3: Weights for 11 Sectors in S&P 500



Benefit from the development of passive investment industry, there is plenty of EFTs focus on the single sector. For simplicity, chose about 5 ETFs from each sector to comprise the ETFs pool, and every ETF has outstanding tracking performance, good liquidity, relatively large asset size, and long history.

Below, table 4.2 is the ETFs pool.

Table 4.2: List of ETFs for Each Sector

Category	**Tickers**
Technology	XLK, VGT, IYW, RYT, IGM
Financial	VFH, KBE, IYF
Consumer discretionary	XLY, VCR, IYC, XRT, RCD, RXI
Health care	XLV, VHT, IBB, IYH, RYH
Industrial	XLI, VIS, IYJ, RGI
Energy	XLE, VDE, IYE, RYE, FXN
Consumer staples	XLP, VDC, IYK, RHS, FXG
Material	XLB, VAW, IYM, RTM, XME
Telecom	VOX, IXP, IYZ
Utilities	XLU, VPU, IDU, RYU, PUI
Real Estate	VNQ, IYR, RWR, USRT

Chapter 5

Research Design

In the beginning, the tracking funds need to determine the selections of ETFs in it and then optimize weights for each asset. By applying a lasso linear regression of log adjusted close prices from the beginning of 2008 to end of 2014: the dependent variable is the SPRT index, and the explanatory variables are the log prices of each asset from the ETFs pool. The model has the form:

$$ln(SPTR_t) = \beta_0 + \sum_{k=1}^{54} \beta_k ln(ETF_{kt}) + \epsilon_t \tag{5.1}$$

The coefficient estimates are estimated by the method of lasso, it has the equation:

$$\beta_L = argmin \sum_{t=1}^{T} (SPTR_t - \beta_o - \sum_{k=1}^{54} \beta_k ETF_{kt})^2 + \lambda \sum_{k=1}^{54} |\beta_k| \tag{5.2}$$

The λ is a tuning parameter in lasso L1 regularization, it could determine the values for coefficients estimates as motioned in previous sector. It is important to select an appropriate value for tuning parameter in lasso, however, the optimal tuning parameter is difficult to calibrate in practice (Lederer and Muller, 2015). Fang and Tang (2013) note

that "To the best of the knowledge, there is no existing work accommodating tuning parameter selection for general penalized likelihood methods."

The selection of tuning parameter is treated as an endogenous problem. Starting as $\lambda = 0$, then set a relative small step length τ=0.0001, every time add it to previous λ to get a new λ, until $\lambda = 1$.

$$\lambda_i = \lambda_{i-1} + \tau \tag{5.3}$$

where $i \geq 1$, $\lambda_0 = 0$, and $\tau = 0.0001$

For every λ, it will have a new lasso regression model. The new model will generate a different set of coefficient estimates, and the underlying residual series. Then test cointegration relationship between the SPTR index and the combination of non-zero ETFs adopt Engle-Granger cointegration method.

Augmented Dickey Fuller method is used to test the presence of unit root in residual series. If the residual series have not unit root, which means it is stationary and we get long run equilibrium from Engle Granger cointegration. In contrast, if the residual series has unit root existing, we do not go along with the corresponding combination of ETFs.

Different λ may lead to the same combination of ETFs but with slightly varying coefficient estimates, we chose the most cointegrated one as the fund. P-value from the ADF test on residual series is an indicator to reflect the stationarity, so it could be a good indicator to measure the degree of cointegration relationship between SPTR index and the assets combination. The smaller p-value, the stronger cointegration relationship.

Next issue is to allocate capital to each ETF in the portfolios. Because we adopted long and hold tracking strategy, cannot short sell ETFs, the coefficients cannot be negative.

From equation 5.3, the constraints require that all the coefficients estimates must be non-negative, zero means we exclude the ETFs.

As a common practice (Alexander 2008) to calculate the weights, we divided the coefficient estimates by the sum of the total value of coefficients except for the constant term β_0. The ratio of each component is the weight of it. In math form:

$$weight_i = \frac{\beta_i}{\sum_{k=1}^{n} \beta_k} \tag{5.4}$$

and

$$\sum_{j=1}^{m} w_j = 1$$

With stronger penalizing, more ETFs will be removed out of the combination. The regression model could provide cointegrated portfolios consists of a relatively small number of constituent ETFs. We will construct and compare three funds with 15, 10 and 6 ETFs.

Cointegration methodology offers a rationale for tracking targeted index over the long term. Based on the theory, the cointegrated portfolio will tie together with the index in the long run, it may only deviate away from the index temporarily. Therefore, we do not need to rebalance the weights of ETFs in the portfolio by the intrinsic characteristic of cointegration.

We will test the tracking performance of portfolios between no balance and different rebalance time intervals: annually rebalance, semi-annually rebalance and quarterly rebalance. For a cointegrated combination of ETFs with the initial weights, we will balance the weights of each asset by running regression equation, and then we also run ADF test on new residual series to make sure they are cointegrated. We have a test

period from the beginning of 2014 to the end of 2017. We will rebalance weights of constituents 3 times for annually rebalance, 7 times for semi-annually rebalance, and 13 times for quarterly rebalance.

Generally speaking, more frequently rebalance the portfolios could improve the tracking performance because we can adjust the funds short-term deviation from the index manually. At the same time, rebalancing will produce huge transaction costs. The enormous transaction costs could ruin the advantages of the tracking funds, and make the passive strategies less appealing to investors . For simplicity, we assume the transaction cost is 0.1% of the trading amount for institutions.

To sum up , we will build 3 portfolios based on a number of component ETFs, and for each portfolio, we have 4 different rebalance schemes, a total of 12 funds.

We fitted a regression model and estimated the weights for underlying ETFs in the portfolio within the training set from 2008-01-01 to 2013-12-31. The goal is to mimic the performance of SPTR and achieve the same profitability and volatility. We expected to minimize tracking errors and get a highly cointegration relationship between the portfolio and the underlying index. Then for the out-of-sample period from 2014-01-01 to 2017-12-31, we operated the portfolios in the market and then took the information ratio to test it as well. There are a few measures to assess the tracking funds' fitness.

- ADF test on residual series – measures the stationarity of errors from a regression model that is the proxy of cointegration relationship. Smaller P-value indicates more stationary residual terms and stronger cointegration relationship between log prices of portfolio and SPTR index.
- Tracking errors mean and volatility – the tracking error is defined as the deviation

of daily returns between portfolio and SPTR. We would like to have small mean and low volatility tracking errors, which implies a stable and robust tracking fund.

- Correlation coefficient of daily returns — We construct a cointegration relationship between the log prices of the portfolio and SPTR index, and the daily return is the first difference from log prices. This is used to directly reflect the tightness between daily returns from the portfolio and SPTR index.

- Information ratio – it is commonly used to measure the risk-adjusted returns of a portfolio above a benchmark. It has the formula:

$$IR = \frac{E(R_p - R_I)}{\sqrt{Var(R_p - R_I)}}$$

where R_p is the daily return of the portfolio and R_I is the daily return of the SPTR index. Positive and higher IR means outstanding risk-adjusted returns, and it is attractive for investors. However, we do not seek higher IR particularly as operating an index tracking fund.

Chapter 6

Empirical Results

Before running cointegration regression, we need to make sure all the time series have the same order of integration. We applied augmented Dickey-Fuller unit root test on the log transform of adjusted close prices of SPTR index and all ETFs from 2008 to 2017. Fortunately, every time series process is I(1), the order of integration is 1.

The results of the ADF tests can be found on Appendix Table 8.1. As planned in research design section, we need to find the tuning parameters λ in equation 5.2 that could select 15, 10, and 6 ETFs out of the ETFs pool to constitute the tracking funds.

When tuning parameter is 0.000101, lasso regression generates a cointegrated combination of 15 ETFs. The p–value of the ADF test on residual terms ϵ_t is 7.404e-5, so do not accept the null hypotheses, furthermore, we can conclude the combination of these positive variables is strongly cointegrated with the response variable SPTR index.

We have the 15 positive coefficients and corresponding weights in table 6.1 below.

Table 6.1: Portfolio of 15 ETFs

ETF Tickers	**Coefficients**	**Weights**
XLK	0.17645	18.47%
XLF	0.09428	9.87%
IYF	0.05618	5.88%
XLY	0.09494	9.94%
IYC	0.06336	6.63%
XLV	0.10934	11.45%
IBB	0.04119	4.31%
XLI	0.09956	10.42%
VIS	0.00901	0.94%
XLE	0.13247	13.87%
XLP	0.00029	0.03$
IYK	0.00142	0.15%
IXP	0.00062	0.06%
XLU	0.06863	7.18%
RWR	0.00753	0.79%

Table 6.2 shows the weights for each sector in the portfolio based on the weights of each ETF. It is worth noting that the weights in table 6.2 are very similar to the weights based on the market capitalization of each sector from SP 500 list. Which means the cointegration regression methodology could give us a fully diversified portfolio as the comprehensive index.However, for 4 smallest sectors material, telecom, utilities and real estate, each of them weights less than 3%, we do not allocate any capital on material and real estate sectors, and only 0.93% on utilities sector.

Table 6.2: Weights of Sectors in 15-Portfolio

Sector	**Weight**
Techonology	18.47%
Finance	15.75%
Consumer Discretionary	16.57%
Health Care	15.76%
Industry	11.36%
Energy	13.87%
Consumer Staples	0.15%
Telecommunication	0.06%
Utility	7.18%
Real Estate	0.79%

Within out-of-sample 1007 trading days from 2014-01-02 to 2017-12-29, we will explicate the tracking results of no balance, annual balance, semi-annual rebalance and quarterly rebalance in Table 6.3. For convenience, we define the portfolios as Regular 15-Fund, Annual 15-Fund, Semi-annual 15-Fund and Quarter 15-Fund. For all portfolios with 15 ETFs, they have overall good quality tracking performance, with above 99% correlation coefficients between portfolio daily returns and SPTR daily returns. Under this rebalance methodology, the annually rebalancing portfolio is the greatest.

Table 6.3: 15 ETFs Portfolio

	No Rebal.	**Yr. Rebal.**	**Semi-Yr. Rebal.**	**Qtr. Rebal.**
Mean of TE	-9.83e-6	-4.54e-6	1.16e-5	3.9e-7
Std. Deviation of TE	1.354e-3	9.501e-4	9.34e-4	1.125e-3
ρ_r	0.985866	0.992618	0.992799	0.989355
IR	-0.007	-0.005	-0.012	0

When λ in lasso regression is 0.000605, there is a combination of 10 ETFs that is cointegrated with SPTR index, and the p–value of the ADF test on residual terms ϵ_t is 4.29e-5.

The initial 10 ETFs with corresponding coefficients and weights are in Table 6.4, and the weights for each sector are in Table 6.5. Portfolios contain 10 ETFs from 8 different sectors except consumer staples, material, telecom. Comparing with 15-Portfolios, the sector allocations vary greatly. The 10 ETFs portfolios put more weights on consumer discretionary and energy sector instead of the original heavy sector technology. For previous small sectors real estate and telecom in 15 ETFs portfolios, 10 ETFs portfolios completely exclude telecom, but surprisingly allocate 5.55% capital on real estate sector.

Table 6.4: Portfolio of 10 ETFs

ETF Tickers	**Coefficients**	**Weights**
XLK	0.06107	7.13%
XLF	0.15907	18.58%
XLY	0.16606	19.4%
VCR	0.07785	9.09%
IBB	0.11882	13.88%
VIS	0.03534	4.13%
XLE	0.13592	15.88%
RYE	0.02546	2.97$
IDU	0.02902	3.39%
RWR	0.04741	5.54%

Even though 10 ETFs portfolios have quite different sector allocations from 15 ETFs portfolios, they achieveddecent tracking performance with high correlation coefficients between 0.944 and 0.969 from Table 6.6. Increasing rebalance frequency could obviously improve the tracking performance by enhancing correlation coefficients, decreasing the standard deviation of tracking errors.

Table 6.5: Weights of Sectors in 10 Portfolios

Sector	**Weight**
Technology	7.13%
Finance	18.58%
Consumer Discretionary	28.49%
Health Care	13.88%
dIndustry	4.13%
Energy	18.85%
Utility	3.39%
Real Estate	5.54%

Table 6.6: Performance of 10 ETFs Portfolio

	No Rebal.	**Yr. Rebal.**	**Semi-Yr. Rebal.**	**Qtr. Rebal.**
Mean of TE	-4.24e-5	-1.161e-5	-2.948e-5	-3.73e-5
Std. Deviation of TE	2.959e-3	2.4228e-3	2.196e-3	2.151e-3
ρ_r	0.943910	0.961805	0.968419	0.969016
IR	-0.014	-0.005	-0.013	-0.017

By applying 'trial and error' approach, the minimum number of ETFs in a cointegrated combination is 6. Some combinations of 5 ETFs are cointegrated with the SPTR index, but such portfolios are not robust, and the p-values of residual terms near the critical value 5%, the residual terms are not stationary at 95% confidence level.

From training data, cointegraton-lasso methodology generates portfolios of 6 ETFs with tuning parameter is 0.001571, and the p–value of ADF test on residuals terms is 0.03072. Those 6 ETFs and corresponding sector are in Table 6.7 and 6.8. 6 ETFs are from 5 sectors, and the biggest sector is consumer discretionary that has two ETFs in the portfolio and weights about 33%. The smallest sector is real estate, and it weights about 11%, however, real estate sector only weights 3% in original S&P 500. For no rebalance portfolio, the daily return series has the 91.9% correlation coefficient with SPTR daily returns, and with relatively low tracking error variance 1.519e-5.

Table 6.7: Portfolio of 5 ETFs

ETF Tickers	**Coefficients**	**Weights**
XLF	0.15621	19.92%
XLY	0.03599	4.59%
VCR	0.21792	27.79%
IBB	0.15505	19.77%
RYE	0.1354	17.27%
RWR	0.08369	10.67%

Table 6.8: 5 ETFs Portfolio

	No Rebal.	**Yr. Rebal.**	**Semi-Yr. Rebal.**	**Qtr. Rebal.**
Mean of TE	-8.809e-5	-7.531e-5	-6.69e-5	-8.55e-5
Std. Deviation of TE	3.783e-3	3.728e-3	3.532e-3	3.48e-3
ρ_r	0.919049	0.921228	0.928428	0.929425
IR	-0.023	-0.02	-0.019	-0.025

Table 6.9: Weights of Sectors in 5 Portfolios

Sector	**Weight**
Finance	19.92%
Consumer Discretionary	32.74%
Health Care	19.77%
Energy	17.27%
Real Estate	10.67%

Comparing all the portfolios, the results from three groups indicate that shrinking the rebalancing interval could improve the tracking performance by enhancing correlation coefficients and decreasing the standard deviation of tracking errors. Nevertheless, the improvements are apparently very limited. Correlation coefficient is a considerable metric to reflect the entire tracking performance, and it increased 0.69%, 2.51%, 1.038% for 15 ETFs portfolios, 10 ETFs portfolios, and 6 ETFs portfolios respectively. We do not calculate the rebalance costs and the payoffs from rebalance weights in this paper. From cost control and risk management perspective, rebalance seems like not very attractive.

Overall, no rebalance portfolios revealed decent and robust tracking performance within all three different groups 15 ETFs portfolios, 10 ETFs portfolios and 6 ETFs portfolios. Even the smallest portfolio has 6 ETFs could achieve over 90% similarity as SPTR daily returns. The reason is that the portfolios are cointegrated with SPTR, cointegration relationship could ensure the prices share common stochastic trends and are tied together in the long run. The portfolios are capable to mirror the behavior of SPTR index over a long timescale. The other reason is that the portfolios are created in fund of funds structure whose constituents are sector ETFs. The ETFs are tracking underlying sectors and they may rebalance and modify holdings regularly already. Therefore, the portfolios benefit from that structure, they do not require to be rebalanced often by their inherent characteristics.

Chapter 7

Conclusion

This paper constructed portfolios to tracking SPTR index by utilizing mature statistical tools cointegration and lasso regression to select different numbers of sector ETFs. Which fill the gap in the market, no such index fund targets SPTR. The fresh tracking methodology is the progress of passive investment strategies and embodies three features: cointegration, lasso regression, and fund of funds structure.

Cointegration relationship is fundamental for the portfolios to accurately mimic the performance of a benchmark. Because it guarantees the existence of common stochastic trends between prices of portfolio and benchmark, which helps them tie together in the long run. That is the reason why the portfolios do not need to rebalance frequently as well. lasso regression is a prevalent approach to perform the subset selection, and the implementation is straightforward. lasso approach possesses extensive flexibility, and it could generate adequate subsets of different numbers of components by changing the tuning parameter in regularization term. It is very impressive that lasso regression found a consistent cointegrated combination of the least 6 ETFs. Last but not least, fund of funds structure is an unique characteristic of the tracking portfolios. The portfolios

consisted of ETFs instead of stocks, therefore, there is completely no need to long hundreds of stocks to target SPTR index which contains 500 stocks. This ingenious structure allows us to build portfolios efficiently and econometrically by means of holding small number of representative ETFs.

Overall, Cointegration-lasso methodology leads to excellent portfolios that have high correlation and low tracking errors with underlying index. There are three groups of portfolios based on the number of ETFs, 15, 10 and 6. And 4 portfolios within a group based on various rebalance schemes: no rebalance, annually rebalance, semi-annually rebalance, and quarterly rebalance. 15 ETFs portfolios achieved the best tracking performance in general, they had over 99% correlation and the lowest tracking error variance of daily returns with SPTR index. Such tracking portfolios are very attractive in the market, and they demonstrate the success of cointegration lasso approach.

Bibliography

SPIVA® STATISTICS & REPORTS U.S.
https://us.spindices.com/spiva//reports

Morningstar (2017) "Morningstar Reports U.S. Mutual Fund and ETF Asset Flows"
https://www.morningstar.com/lp/fund-flows-direct?cid=CON$_D IR0030con = 12856$

Alexander, C. and Dimitriu, A. (2005) "Indexing, Cointegration and Equity Market Regimes", *International Journal Of Economics and Finance* Vol. 10: 213-231.

Glova.J., Pastor, D. and Sabol,T. (2015) "Cointegration and Its Specific Application in Portfolio Management",*Economic Computation and Economic Cybernetics Studies and Research* Issue 4: 193-207.

Sant'Anna. L., Filomena, T. and Caldeira, J. (2017) "Index Tracking and Enhanced Indexing Using Cointegration and Correlation with Endogenous Portfolio Selection", *The Quarterly Review of Economics and Finance* Vol. 65: 146-157

Alexander, C. (2001) "Cointegration and Asset Allocation: A New Active Hedge Fund Strtegy",*ISMA Centre Discussion Papers In Finance* 2001-03.

Tibshirani, R. (1996) "Regression Shrinkage and Selection via the Lasso", *Journal of the Royal Statistical Society* Vol. 58: 267-288.

Maurer T. (2008) "Cointegration in Finance: An Application to Index Tracking" SSRN: https://ssrn.com/abstract=1586997

Wu, L., Yang, Y. and Li, H. (2014) "Nonnegative-lasso and Application in Index Tracking", *Computational Statistics Data Analysis* 70: 116-126.

Wu, L. and Yang, Y. (2014) "Nonnegative Elastic Net and Application in Index Tracking", *Applied Mathematics and Computation* Vol. 227: 541-552.

Yang, Y. and Wu. L. (2016) "Nonnegative Adaptive Lasso for Ultra-high Dimensional Regression Models and a Two-stage Method Applied in Financial Modeling", *Journal of Statistical Planning and Inference* Vol 174: 52-67.

Zou, H. (2006) "The Adaptive Lasso and Its Oracle Properties" *Journal of the American Statistical Association* Vol. 101, 1418-1429.

Engle, R.F. and Granger C.W.J. (1987) "Co-Integration and Error Correction: Representation, Estimation, and Testing" *Econometrica* Vol. 55: 251-276.

Granger C.W.J, Newbold, P. (1974) "Spurious regressions in Econometrics" *Journal of Econometrics* Vol. 2: 111-120

Johansen Søren (1988) "Statistical Analysis of Cointegration Vectors" *Journal of Economic Dynamics and Control* Vol. 12: 231-254

Johansen, S., Juselius, K. (1990) "Maximum Likelihood Estimation and Inference on Cointegration–With Applications to the Demand for Money" *Oxford Bulletin of Economics and Statistics* Vol. 52: 169-210

Fan, Y. and Tang, C. (2013) "Tuning Parameter Selection in High Dimensional Penalized Likelihood" *Journal of the Royal Statistical Society* Series B: 531-552

Lederer, J., Muller, L.C., (2015) "Don't Fall for Tuning Parameters: Tuning-Free Variable Selection in High Dimensions With the TREX" *arXiv:Statistics-Methodology* https://arxiv.org/pdf/1410.0247.pdf

Appendix A

Table and Figures

Table A.1: ADF unit test on all ETFs

	XLK	**VGT**	**IYW**	**RYT**	**IGM**	**XLF**	**VFH**	**KBE**	**IYF**	**XLY**
ADF stats	0.583	0.581	0.504	0.461	0.516	-0.064	-0.34	-1.279	-0.482	0.157
P–value	0.987	0.987	0.985	0.984	0.985	0.953	0.919	0.638	0.895	0.969
diff(1) ADF stats	-10.766	-37.894	-37.841	-37.369	-10.652	-9.039	-9.325	-9.304	-9.357	-11.046
diff(1) P–value	2.458e-19	0.0	0.0	0.0	4.629e-19	5.199e-15	9.636e-16	1.094e-15	7.996e-16	5.212e-20
	VCR	**IYC**	**XRT**	**RCD**	**RXI**	**XLV**	**IBB**	**VHT**	**IYH**	**RYH**
ADF stats	-0.022	0.172	-0.838	-0.649	0.002	0.509	-0.434	0.447	0.447	0.208
P–value	0.957	0.971	0.808	0.859	0.959	0.985	0.904	0.983	0.983	0.973
diff(1) ADF stats	-37.322	-38.114	-10.938	-8.642	-10.808	-14.253	-37.357	-14.387	-14.298	-37.909
diff(1) P–value	0.0	0.0	9.435e-20	5.386e-14	1.939e-19	1.468e-26	0.0	8.955e-27	1.242e-26	0.0
	XLI	**VIS**	**IYJ**	**RGI**	**XLE**	**VDE**	**IYE**	**RYE**	**FXN**	**XLP**
ADF stats	0.177	0.044	0.103	-0.002	-1.669	-1.759	-1.723	-2.067	-2.269	0.038
P–value	0.971	0.962	0.966	0.958	0.447	0.401	0.419	0.258	0.182	0.962
diff(1) ADF stats	-10.973	-10.833	-10.687	-36.685	-11.767	-11.75	-11.939	-9.248	-9.018	-14.819
diff(1) P–value	7.806e-20	1.694e-19	3.802e-19	0.0	1.110	1.213e-21	4.599e-22	1.522e-15	5.865e-15	1.973e-27
	VDC	**IYK**	**RHS**	**FXG**	**XLB**	**VAW**	**IYM**	**RTM**	**VOX**	**IXP**
ADF stats	0.177	0.090	0.175	-0.081	-0.511	-0.570	-1.107	-0.454	-0.549	-0.706
P–value	0.971	0.965	0.971	0.951	0.889	0.878	0.712	0.901	0.882	0.845
diff(1) ADF stats	-26.967	-12.424	-38.364	-18.449	-11.216	-10.935	-10.896	-31.054	-11.171	-10.808
diff(1) P–value	0.0	4.084e-23	0.0	2.155e-30	2.069e-20	9.606e-20	1.193e-19	0.0	2.646e-20	1.939e-19
	IYZ	**XLU**	**VPU**	**IDU**	**RYU**	**PUI**	**VNQ**	**IYR**	**RWR**	**USRT**
ADF stats	-0.985	-0.119	-0.033	-0.089	-0.101	-0.114	-1.07	-1.040	-1.08	-1.334
P–value	0.759	0.948	0.956	0.950	0.949	0.948	0.727	0.738	0.722	0.614
diff(1) ADF stats	-10.788	-12.809	-17.679	-17.569	-17.948	-20.156	-9.155	-9.056	-9.091	-9.299
diff(1) P–value	2.167e-19	6.509e-24	3.614e-30	4.056e-30	2.846e-30	0.0	2.631e-15	4.709e-15	3.82e-15	1.119e-15

Figure A.1: SPTR and 15 ETFs portfolio no rebalance

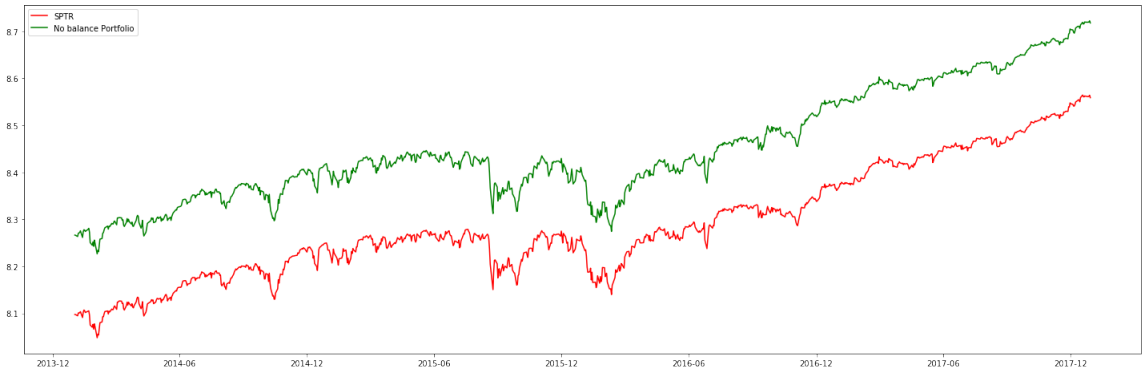


Figure A.2: Daily Returns of SPTR and 15 ETFs portfolio no rebalance

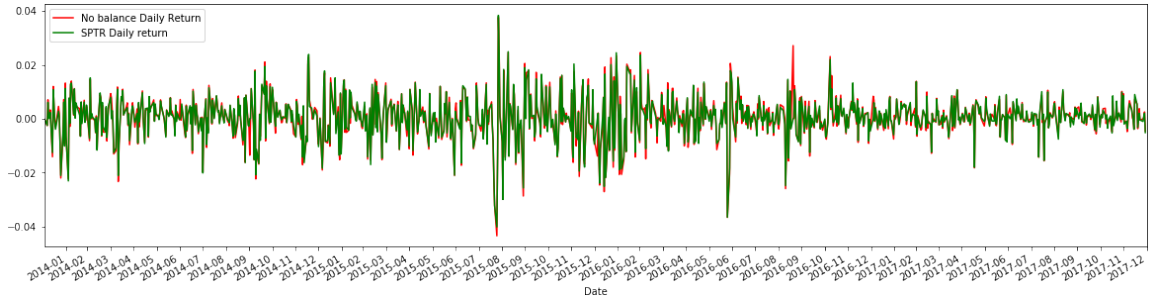


Figure A.3: Cumsum Returns of SPTR and 15 ETFs portfolio no rebalance

Figure A.4: SPTR and 15 ETFs portfolio annual rebalance

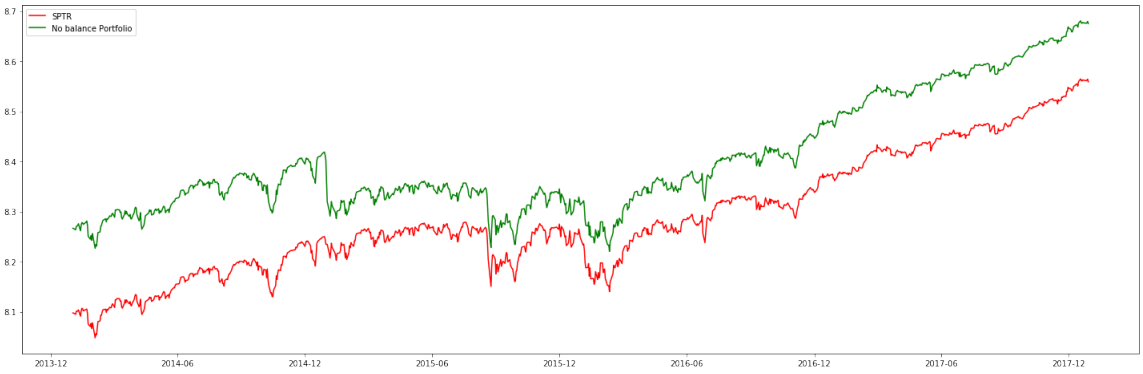


Figure A.5: Daily Returns of SPTR and 15 ETFs portfolio annual rebalance

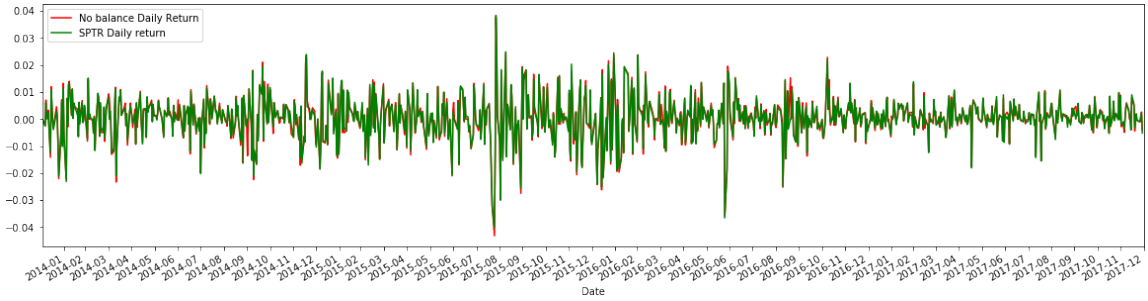


Figure A.6: Cumsum Returns of SPTR and 15 ETFs portfolio annual rebalance

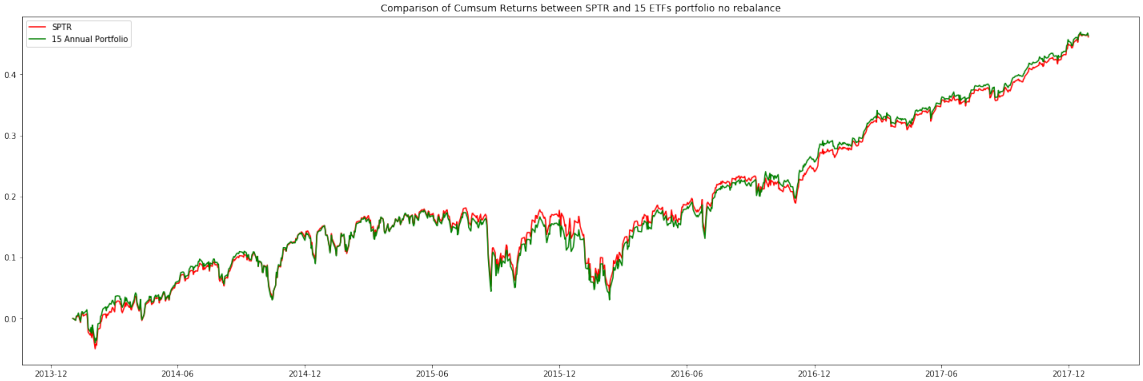


Figure A.7: SPTR and 15 ETFs portfolio semi-annual rebalance

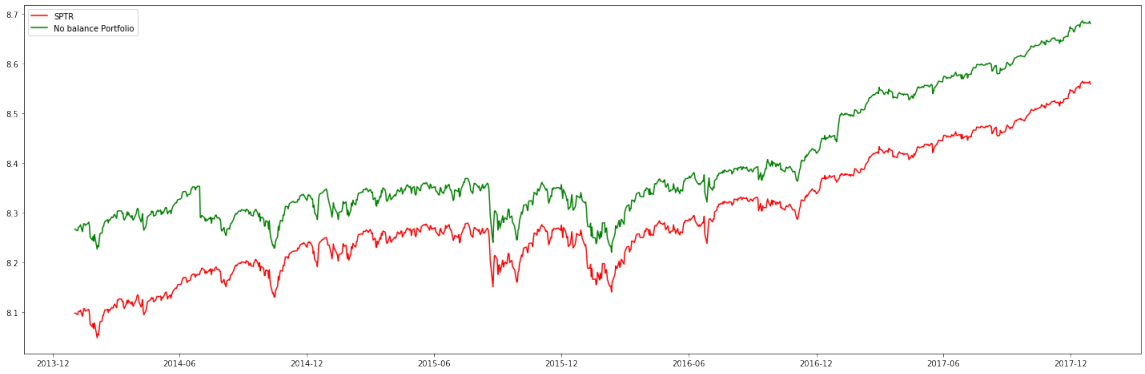


Figure A.8: Daily Returns of SPTR and 15 ETFs portfolio semi-annual rebalance

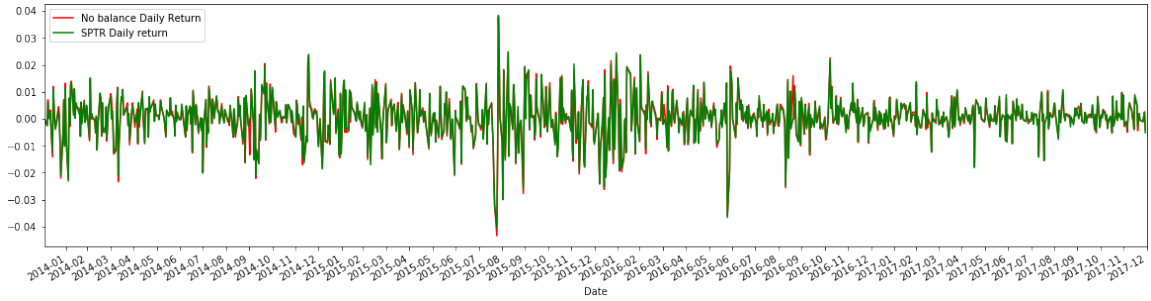


Figure A.9: Cumsum Returns of SPTR and 15 ETFs portfolio Semi-annual rebalance

Figure A.10: SPTR and 15 ETFs portfolio quarterly rebalance

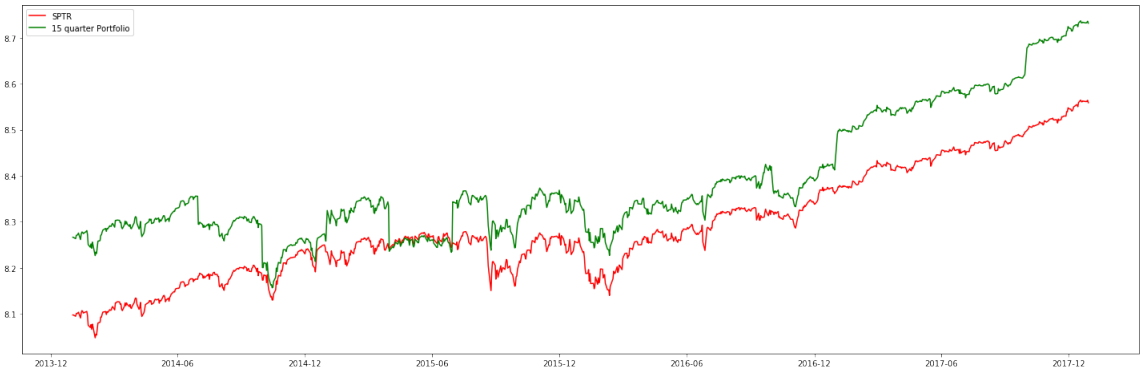


Figure A.11: Daily Returns of SPTR and 15 ETFs portfolio Quarterly rebalance

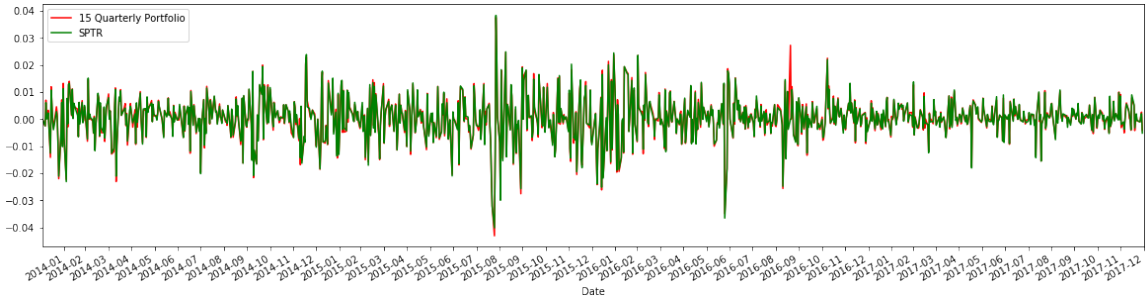


Figure A.12: Cumsum Returns of SPTR and 15 ETFs portfolio Quarterly rebalance

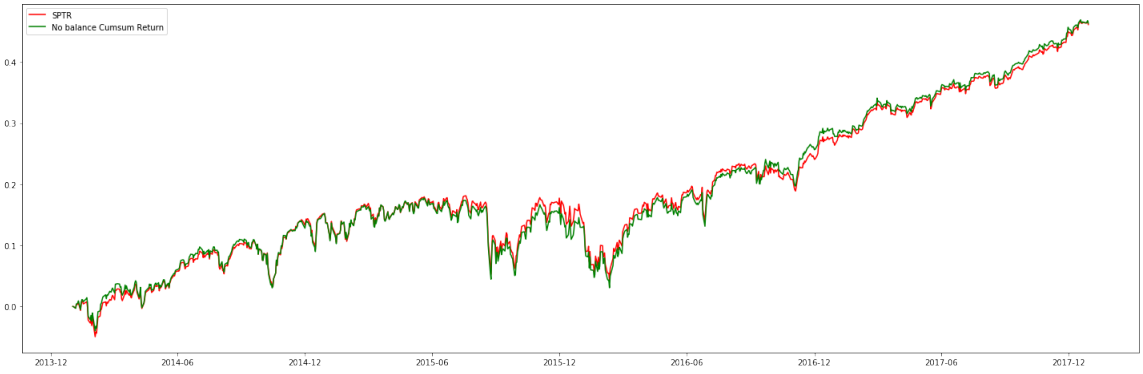


Figure A.13: SPTR and 10 ETFs no rebalance portfolio

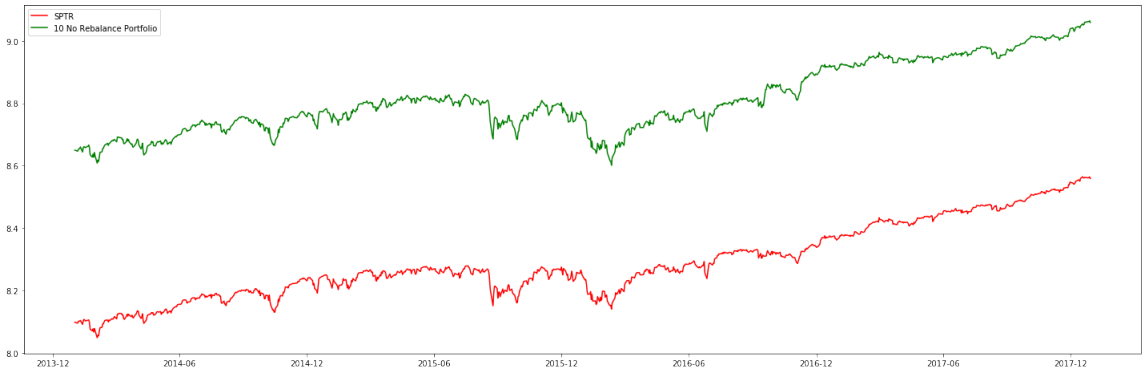


Figure A.14: Daily Returns of SPTR and 10 ETFs portfolio no rebalance

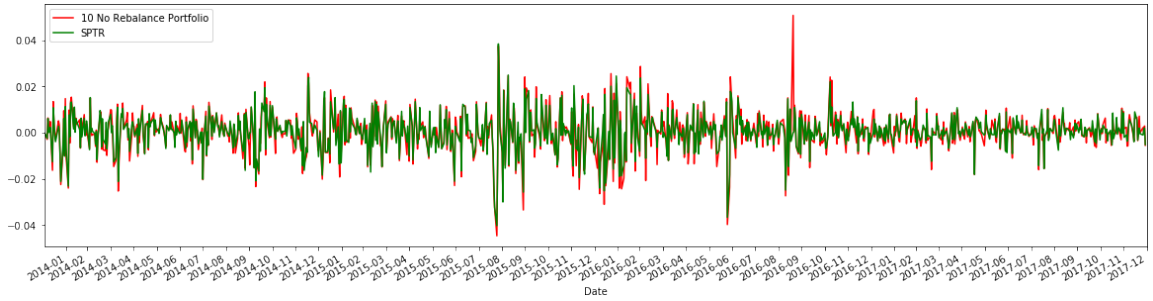


Figure A.15: Cumsum Returns of SPTR and 10 ETFs portfolio no rebalance

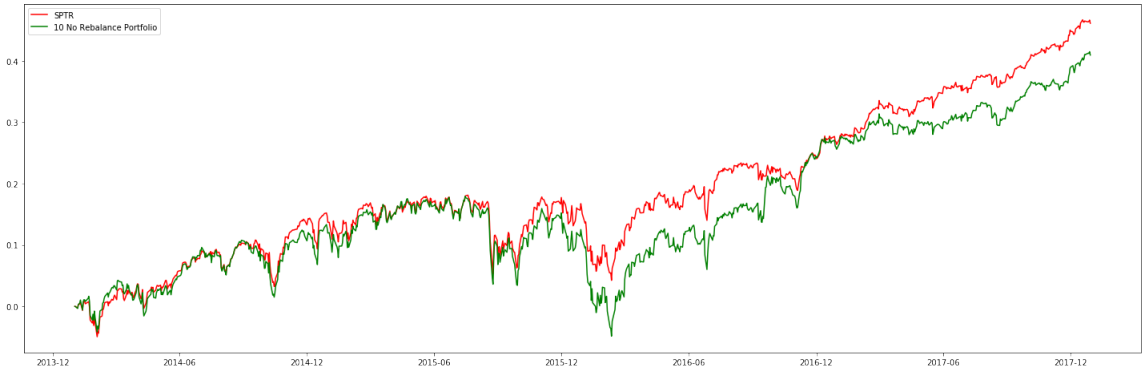


Figure A.16: SPTR and 10 ETFs portfolio annual rebalance

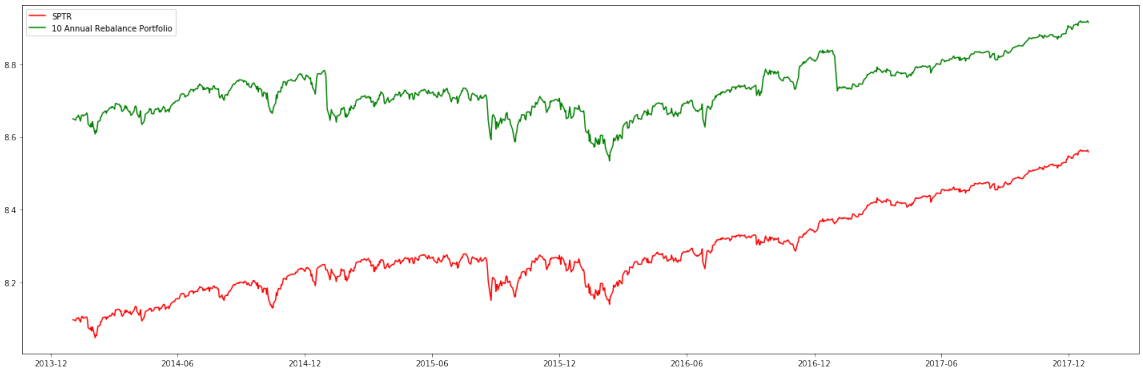


Figure A.17: Daily Returns of SPTR and 10 ETFs portfolio annual rebalance

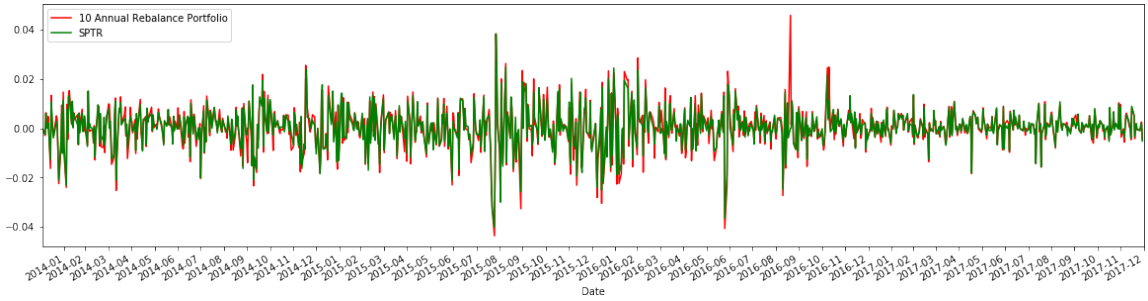


Figure A.18: Cumsum Returns of SPTR and 10 ETFs portfolio annual rebalance

Figure A.19: SPTR and 10 ETFs portfolio semi-annual rebalance

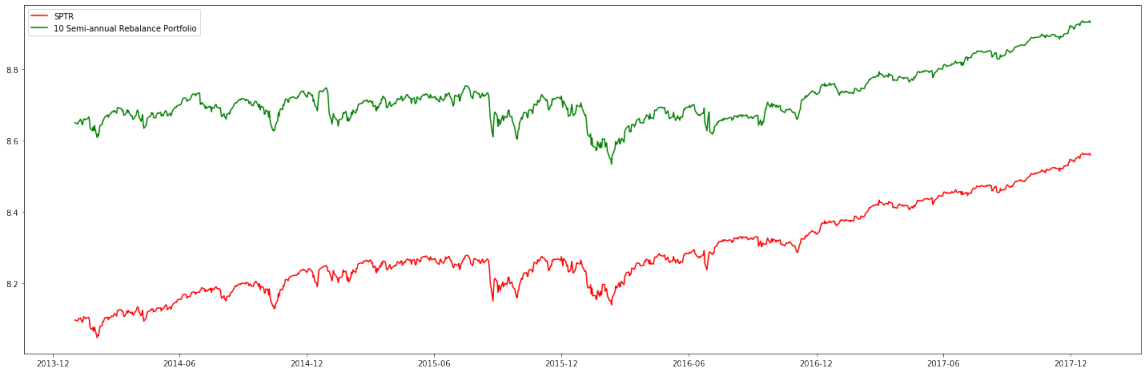


Figure A.20: Daily Returns of SPTR and 10 ETFs portfolio semi-annual rebalance

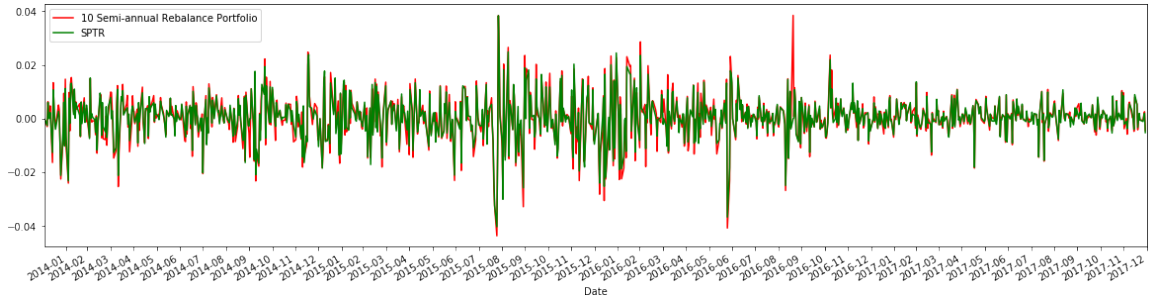


Figure A.21: Cumsum Returns of SPTR and 10 ETFs portfolio Semi-annual rebalance

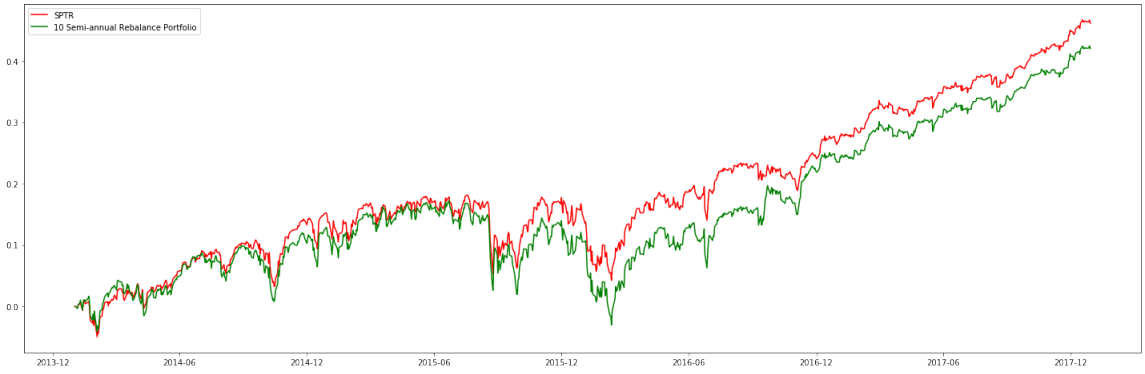


Figure A.22: SPTR and 10 ETFs portfolio quarterly rebalance

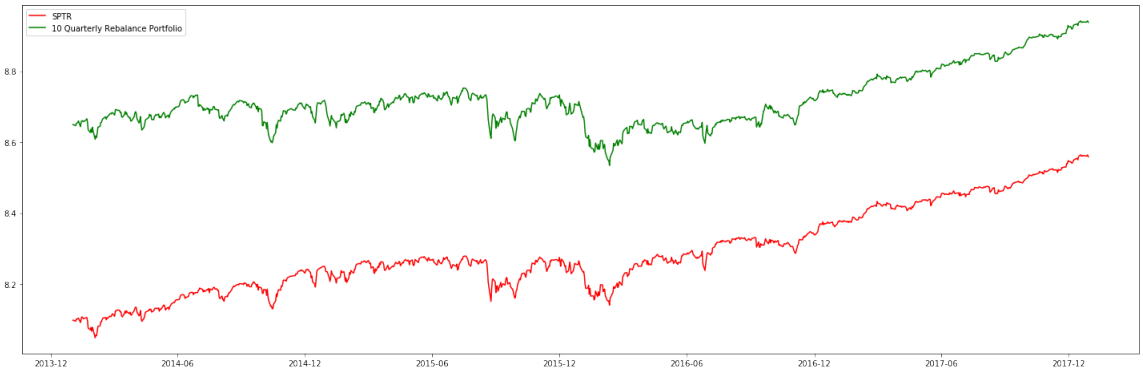


Figure A.23: Daily Returns of SPTR and 10 ETFs portfolio Quarterly rebalance

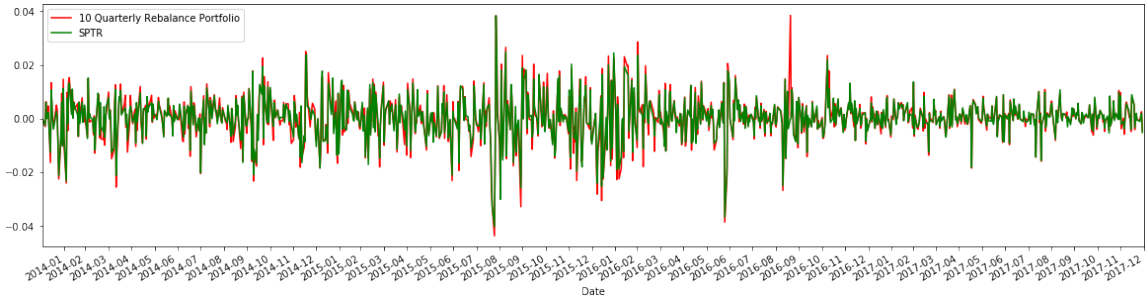


Figure A.24: Cumsum Returns of SPTR and 10 ETFs portfolio Quarterly rebalance

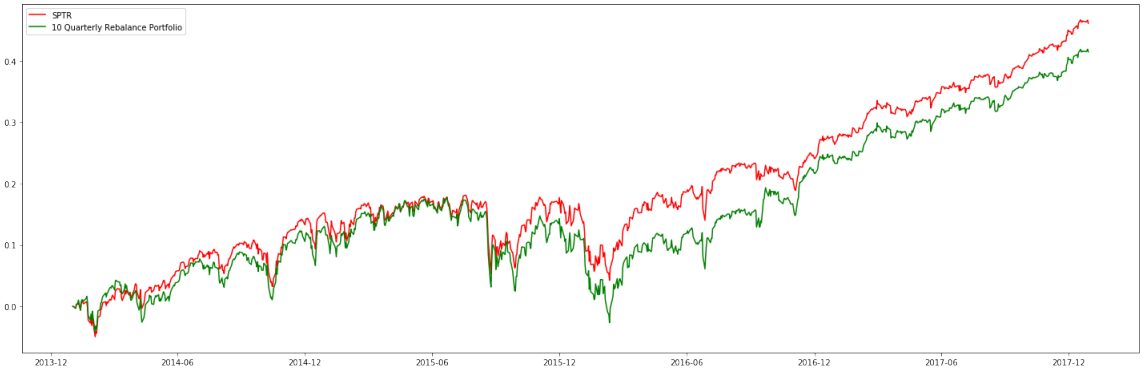


Figure A.25: SPTR and 5 ETFs no rebalance portfolio

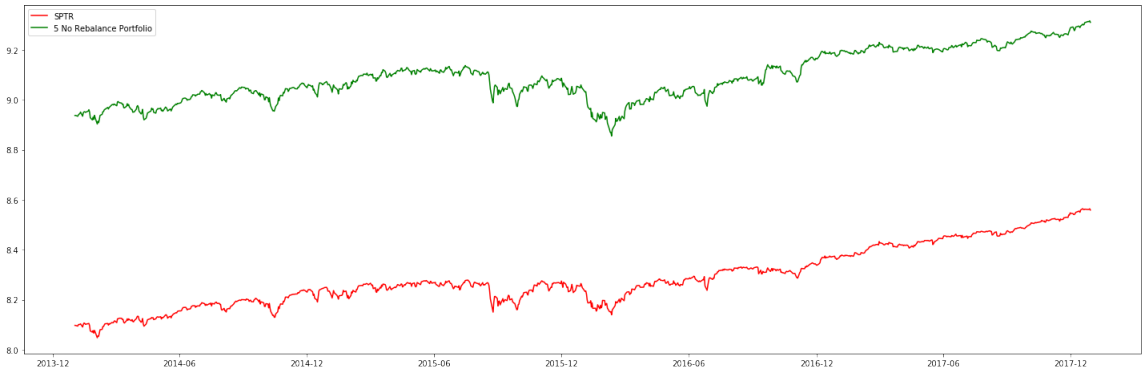


Figure A.26: Daily Returns of SPTR and 5 ETFs portfolio no rebalance

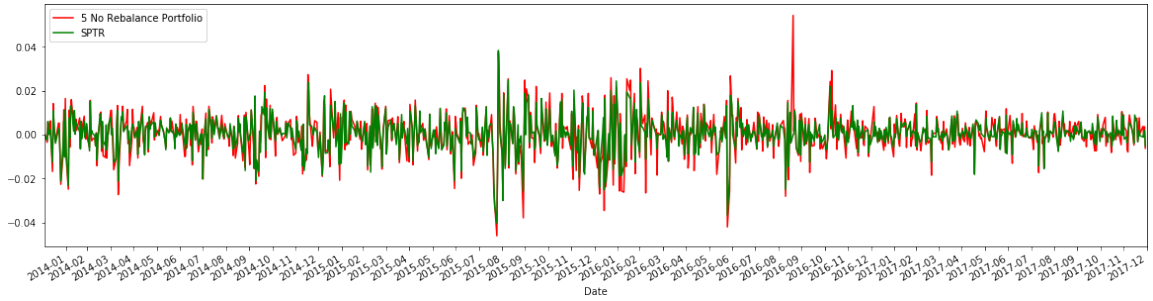


Figure A.27: Cumsum Returns of SPTR and 5 ETFs portfolio no rebalance

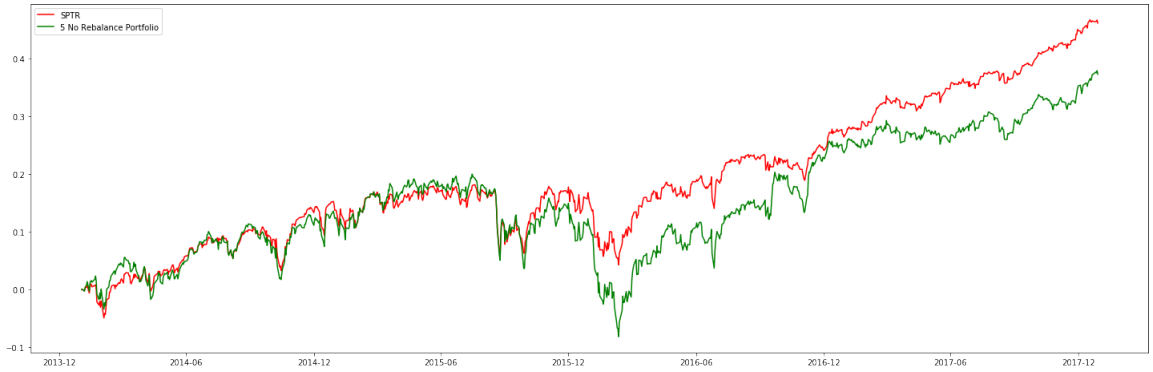


Figure A.28: SPTR and 5 ETFs portfolio annual rebalance

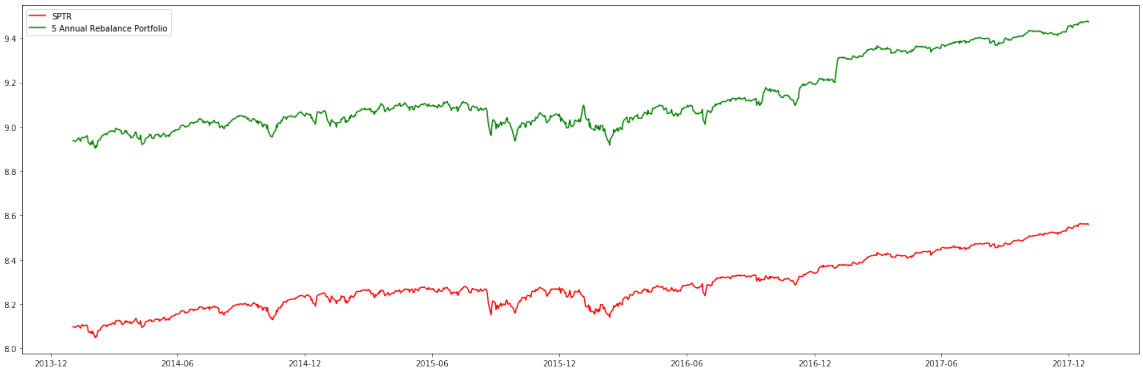


Figure A.29: Daily Returns of SPTR and 5 ETFs portfolio annual rebalance

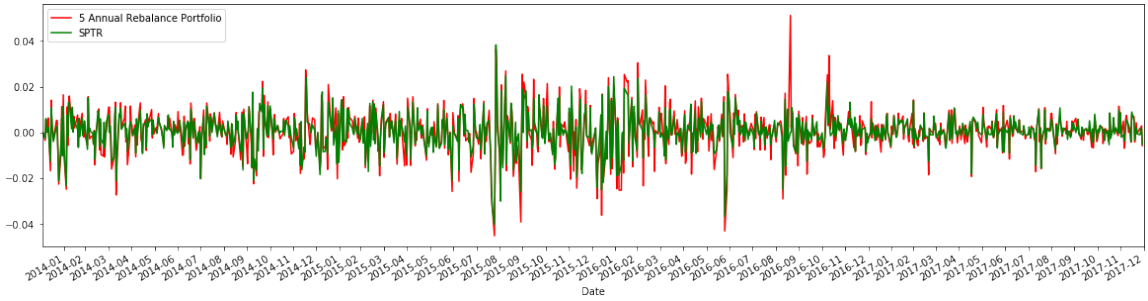


Figure A.30: Cumsum Returns of SPTR and 5 ETFs portfolio annual rebalance

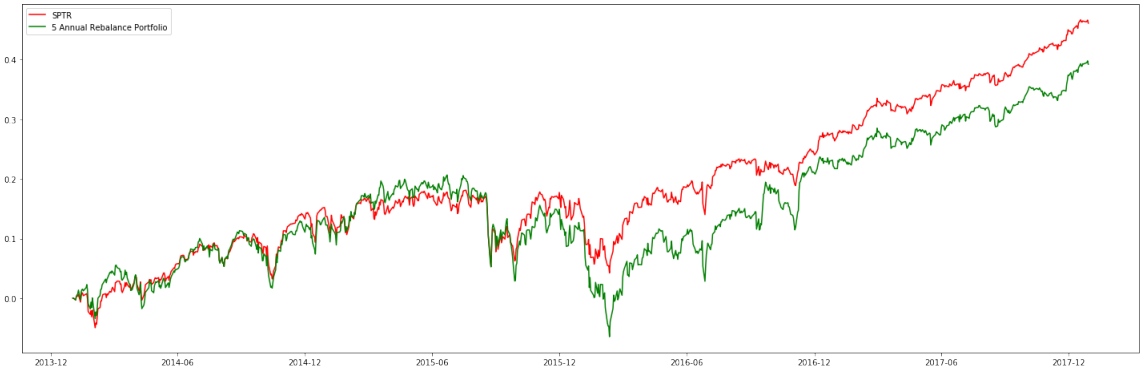


Figure A.31: SPTR and 5 ETFs portfolio semi-annual rebalance

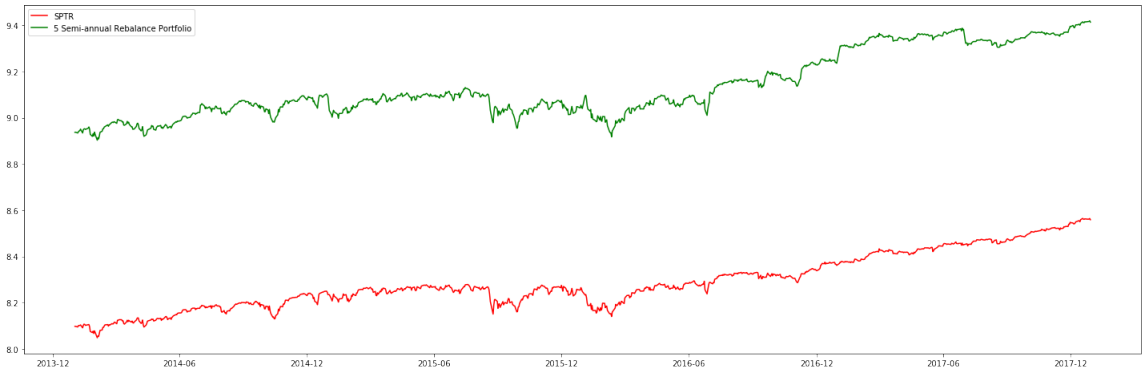


Figure A.32: Daily Returns of SPTR and 5 ETFs portfolio semi-annual rebalance

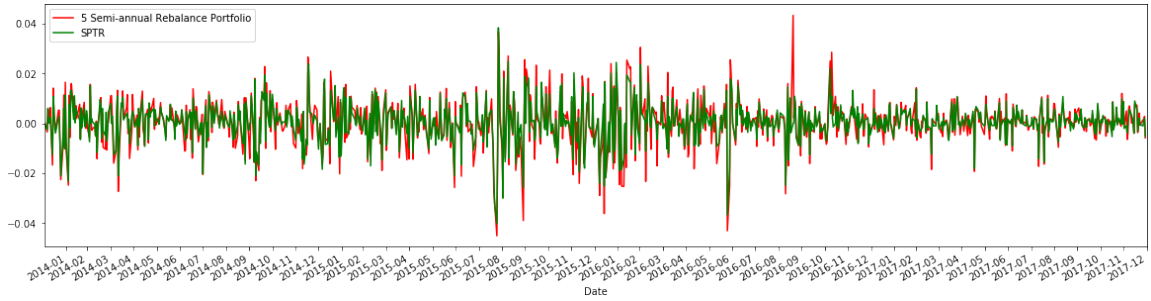


Figure A.33: Cumsum Returns of SPTR and 5 ETFs portfolio Semi-annual rebalance

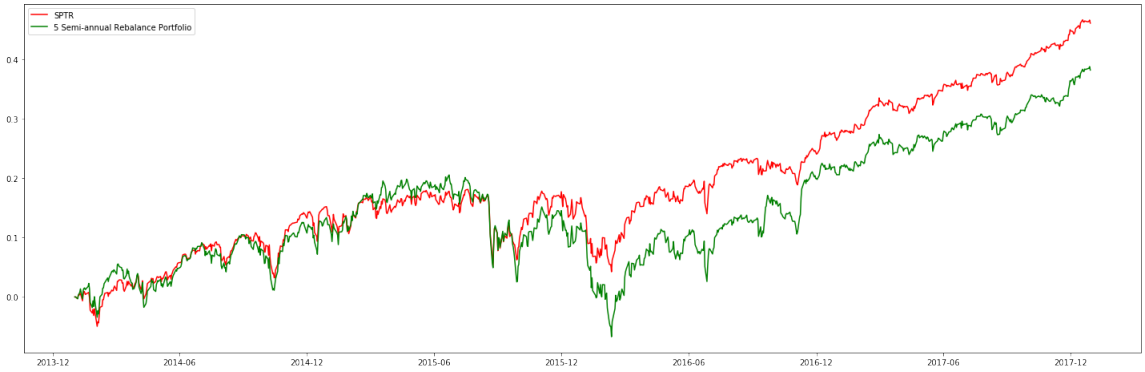


Figure A.34: SPTR and 5 ETFs portfolio quarterly rebalance

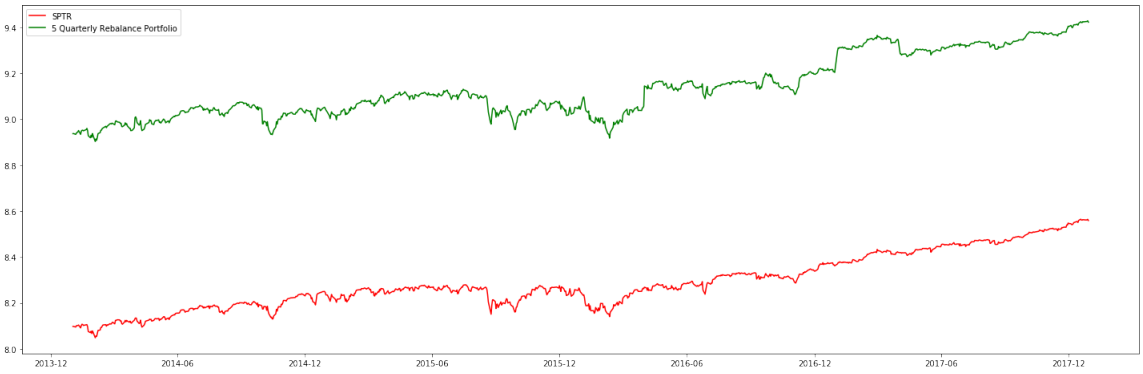


Figure A.35: Daily Returns of SPTR and 5 ETFs portfolio Quarterly rebalance

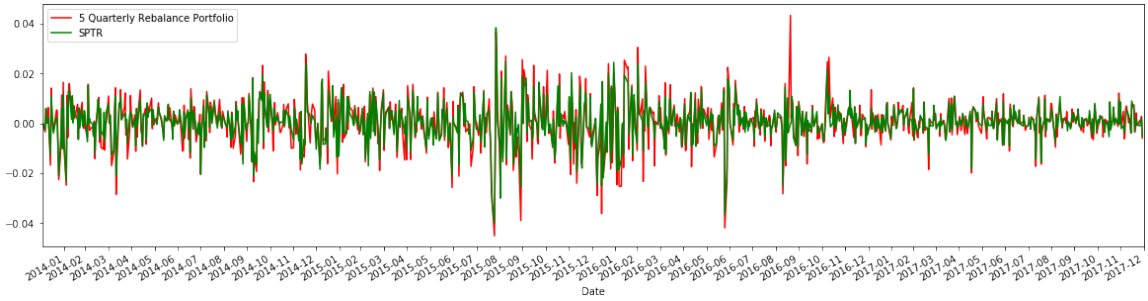
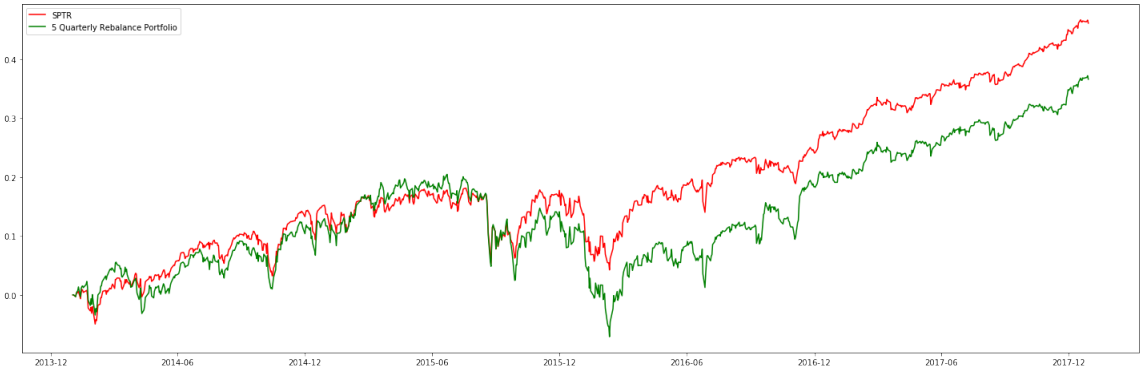


Figure A.36: Cumsum Returns of SPTR and 5 ETFs portfolio Quarterly rebalance



Appendix B

Code

```
import pandas_datareader.data as web
import pandas as pd
import matplotlib.pyplot as plt
import datetime as dt
import numpy as np
import gzip, cPickle
from tqdm import tqdm

from scipy import stats
import statsmodels.api as sm
import statsmodels.tsa as tsa
from statsmodels.tsa.stattools import coint

from statsmodels.formula.api import ols
import statsmodels.formula.api as smf
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from sklearn.linear_model import LassoLars
from statsmodels.tsa.stattools import adfuller as ADF
from sklearn.decomposition import PCA
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from itertools import combinations
from tqdm import tqdm
import pickle
from datetime import datetime as dt
import math

## This step is to download and load all the ETF data
```

```
tickers = '''XLK,VGT,IYW,RYT,IGM,XLF,VFH,KBE,IYF,RYF,XLY,VCR,IYC,XRT,RCD,XLV,
IBB,VHT,IYH,RYH,XLI,VIS,IYJ,RGI,UXI,XLE,VDE,IYE,RYE,FXN,XLP,VDC,IYK,RHS,FXG,
XLB,VAW,IYM,RTM,XME,VOX,IXP,IYZ,IGN,XLU,VPU,IDU,RYU,PUI,VNQ,
IYR,RWR,FRI,USRT'''.split(',')


X = pd.DataFrame()
start_date = '2008-01-01'
end_date = '2018-01-31'
data_source = 'yahoo'
# User pandas_reader.data.DataReader to load the desired data.
for i in tickers:
print i
etf_data= web.DataReader(i, data_source, start_date, end_date)
new_col = etf_data[['Adj Close']]
new_col.columns = [i]
X =  pd.concat([X, new_col], axis=1)

with gzip.open('ETFs_GSPC_MRP_Production.pkl.gz','w') as f:
cPickle.dump(X,f)

ETFs_GSPC = X.copy()

All_ETFs = ETFs_GSPC.iloc[:, :-1]


with gzip.open('ETFs_GSPC_MRP_Production.pkl.gz','r') as f:
   ETFs_GSPC = cPickle.load(f)
```

```
All_ETFs = ETFs_GSPC.iloc[:, :-1]

All_ETFs.head(2)

All_ETFs_log = All_ETFs.apply(np.log)

All_ETFs_log .drop(['RYF','FTXO','PNQI', 'JHMC','PSCH','PSCI','PSCC',
                    'XNTK','PSCM','XTL','REM','RYF','XME'], axis=1, inplace=True) # 51Z ETFs left

All_ETFs_log.columns

len(All_ETFs_log.columns)

All_ETFs_log.shape

# SPTR

# pickle.dump(ETFs_GSPC, open("ETFs_GSPC_MRP_Production.p", "w"))
with gzip.open('sptr.pkl.gz','r') as f:
   SPTR = cPickle.load(f)

SPTR_log = np.log(SPTR[['Adj Close']].copy())
SPTR_log.head()

SPTR_log_train = SPTR_log.loc["2008-01": "2013-12-31"].copy()
SPTR_log_test = SPTR_log.loc["2014-01-01": "2017-12-31"].copy()

SPTR_log_test['Daily_Return'] = SPTR_log_test['Adj Close'].diff()
```

```
SPTR_log_test['Daily_Return'][0] = 0
SPTR_log_test['Cumsum Daily_Return'] = SPTR_log_test['Daily_Return'].cumsum()

j = 0
for i in All_ETFs_log.columns:
    etf = All_ETFs_log[[i]].dropna(axis=0,how = 'any')
    print i
    print ADF(etf.values.flatten())
    etf = etf.diff(1).dropna()
    print '\n'
    print ADF(etf.values.flatten())
    print '\n\n'
```

Lasso Alpha finding

```
%%time
j = 0
m = 0

stationary_alpha = []

SPTR_log_train = SPTR_log.loc["2008-01": "2014-12-31"].copy()
All_ETFs_log_train = All_ETFs_log.loc["2008-01": "2014-12-31"].copy()

for i in tqdm(np.arange(0.000841, 0.021, 0.00001)):
    m+=1
```

```
    LR_lasso = Lasso(alpha=i, fit_intercept=True,normalize=None,positive=True,random_state=123)
    LR_lasso.fit(All_ETFs_log_train,SPTR_log_train)
    LR_lasso_residual = SPTR_log_train.values.flatten() - LR_lasso.predict(All_ETFs_log_train)
    if check_for_stationarity_no_print(LR_lasso_residual.flatten(),cutoff=0.05):
        j+= 1
        stationary_alpha.append(LR_lasso.alpha)
        positive_number = sum(LR_lasso.coef_>0)
        negative_number = sum(LR_lasso.coef_<0)
        print 'we get the right alpha'
        print '\nthe alpha in LASSO is %s and the P-value for ADF is %s'%(i,
ADF(LR_lasso_residual.flatten())[1])
        print 'with in all coefficients, there are %s ETFs are positive'%(positive_number)
        print 'with in all coefficients, there are %s ETFs are negative'%(negative_number)



print '\n\ntotal %s LR models are stationary'%(j)
print 'total we tested %s models'%(m)

LR_lasso = Lasso(alpha= 9.13e-05, fit_intercept=True,normalize=None, random_state=123)
LR_lasso.fit(All_ETFs_log_train,SPTR_log_train)
LR_lasso_residual = SPTR_log_train.values.flatten() - LR_lasso.predict(All_ETFs_log_train)


def alpha_finding(X=All_ETFs_log_train, Y=SPTR_log_train, pos_no = 15):
    for i in np.arange(0.000001, 0.000611, 0.000001):
        LR_lasso = Lasso(alpha=i, fit_intercept=True,normalize=None,positive=True, random_state=123)
        #LR_lasso = Lasso(alpha=i, fit_intercept=True,normalize=None, random_state=123)
        LR_lasso.fit(X,Y)
```

```
        LR_lasso_residual = Y.values.flatten() - LR_lasso.predict(X)
        if check_for_stationarity_no_print(LR_lasso_residual.flatten(),cutoff=0.05):
            positive_number = sum(LR_lasso.coef_>0)
            negative_number = sum(LR_lasso.coef_<0)
            if (positive_number == pos_no and negative_number == 0):
                print 'we get the right alpha'
                print 'the alpha in LASSO is %s and the P-value for ADF is %s'%(i,
ADF(LR_lasso_residual.flatten())[1])
                print 'with in all coefficients, there are %s ETFs are positive'%(positive_number)
                print 'with in all coefficients, there are %s ETFs are negative'%(negative_number)
                break

    return round(i,8)

SPTR_log_train = SPTR_log.loc["2008-01": "2013-12-31"].copy()
All_ETFs_log_train = All_ETFs_log.loc["2008-01": "2013-12-31"].copy()

alpha_finding(X=All_ETFs_log_train, Y= SPTR_log_train, pos_no=6 )



## SPTR no balance

def SPTR_no_balance(alpha_input = 0.000001):
    SPTR_log_train = SPTR_log.loc["2008-01": "2013-12-31"].copy()

    All_ETFs_log_train = All_ETFs_log.loc["2008-01": "2013-12-31"].copy()
```

```
   portfolio_test =
pd.DataFrame(0,columns=['portf_forest','portfolio_test_intercept'] ,index=SPTR_log.loc['2014':'2017'].in
dex)  # this is to store the test data from ETFs


   LR_lasso_year = Lasso(alpha=alpha_input,
fit_intercept=True,normalize=None,positive=True,random_state=123)


   LR_lasso_year.fit(All_ETFs_log_train,SPTR_log_train[['Adj Close']])

   print('\n\n ')

   print('for LASSO alpha',LR_lasso_year.alpha)

   print('original coeff greater than 0 is ',np.sum(LR_lasso_year.coef_ > 0 ))

   print('original coeff leass than 0 is ' ,np.sum(LR_lasso_year.coef_ < 0 ))

   #print('original coeff equal to 0 is ',np.sum(LR_lasso_year.coef_ == 0) )

   print('we select ETFs tickers are', All_ETFs_log_train.columns[LR_lasso_year.coef_>0])


   index_of_etfs = LR_lasso_year.coef_ > 0 # these etfs are our portfolio etfs, we need those to build the
portfolio,and to rebalance


   All_ETFs_log_year = All_ETFs_log.loc[: , index_of_etfs] .copy()  # this contains all the etfs we used to
build portfolio


   coefficients= LR_lasso_year.coef_.copy()


   nonzero_coeff  = coefficients[~(coefficients==0)].copy()


   nonzero_coeff = nonzero_coeff/nonzero_coeff.sum()


   portfolio_test['portf_forest'] = All_ETFs_log_year.dot(nonzero_coeff).copy()


   portfolio_test['portfolio_test_intercept'] = portfolio_test['portf_forest']+LR_lasso_year.intercept_
```

```
plt.figure(figsize=(25,8))

plt.plot(SPTR_log_test.index,SPTR_log_test['Adj Close'], 'r-', label='Real data' )

plt.plot(SPTR_log_test.index,portfolio_test['portfolio_test_intercept'].loc["2014-01-01": "2017-12-31"], 'g-', label='No balance Portfolio' )

plt.legend(loc='upper left')

plt.title('Comparison between real S&P500 and portfolio constructed on test data')

plt.show()


portfolio_test['Daily_Return'] = portfolio_test['portf_forest'].diff()

portfolio_test['Daily_Return'][0] = 0

portfolio_test['Cumsum Daily_Return'] = portfolio_test['Daily_Return'].cumsum()


portfolio_test['Daily_Return'].plot(kind='line',figsize=(20,5),label = 'annual rebalance portfolio cumsum return',style='r-')

SPTR_log_test['Daily_Return'].loc["2014-01-01": "2017-12-31"].plot(kind='line',figsize=(20,5),label = 'SPTR cumsum return',style='g-')

plt.legend(loc='upper left')

plt.xticks( pd.date_range(start='2014-1-01', end='2018-01-01', freq = 'M'))

plt.title('Cumsum Return SPTR and Portfolio')

plt.show()


plt.figure(figsize=(25,8))

plt.title('Cumsum daily return')

plt.plot(SPTR_log_test.index,SPTR_log_test['Cumsum Daily_Return'], 'r-', label='Real data' )

plt.plot(SPTR_log_test.index,portfolio_test['Cumsum Daily_Return'].loc["2014-01-01": "2017-12-31"], 'g-', label='No balance Portfolio' )

plt.legend(loc='upper left')

plt.show()
```

```
    Track_error_annual = portfolio_test['Daily_Return'] - SPTR_log_test['Daily_Return']


    #print('Sum of Tracking error is %.8f') %(np.sum(np.absolute(Track_error_annual)))
    print('Mean of Tracking Error is %.8f') %(np.mean(Track_error_annual))


    print('Tracking Error std is %.8f') %(np.std(Track_error_annual))


   # print('Sum of square Tracking Error is %.8f') %(np.sum(Track_error_annual**2)) # this is the sum of
square of tracking error


    print('Information ratio of the annual portfolio
is %.3f')%(np.mean(Track_error_annual)/np.std(Track_error_annual))


    print('Annual Correlation coefficient between portfolio and SPTR is %.6f')%(

np.corrcoef(SPTR_log_test['Daily_Return'].values.flatten(),portfolio_test['Daily_Return'].values.flatten())
[0,1])


    return portfolio_test


portfolio_test_nobalance_15 =  SPTR_no_balance(alpha_input = 0.000101)


portfolio_test_nobalance_15.head(2)


plt.figure(figsize=(25,8))
plt.plot(SPTR_log_test.index,SPTR_log_test['Adj Close'], 'r-', label='SPTR' )
plt.plot(SPTR_log_test.index,portfolio_test_nobalance_15['portfolio_test_intercept'].loc["2014-01-01":
"2017-12-31"], 'g-', label='No balance Portfolio' )
plt.legend(loc='upper left')
```

```
plt.title('Comparison between SPTR and 15 ETFs portfolio no rebalance')

plt.show()


plt.figure(figsize=(25,8))

portfolio_test_nobalance_15['Daily_Return'].plot(kind='line',figsize=(20,5),label = 'No balance Daily Return',style='r-')

SPTR_log_test['Daily_Return'].loc["2014-01-01": "2017-12-31"].plot(kind='line',figsize=(20,5),label = 'SPTR Daily return',style='g-')

plt.legend(loc='upper left')

plt.xticks( pd.date_range(start='2014-1-01', end='2018-01-01', freq = 'M'))

plt.title('Comparison of Daily Returns between SPTR and 15 ETFs portfolio no rebalance')

plt.show()


plt.figure(figsize=(25,8))

plt.plot(SPTR_log_test.index,SPTR_log_test['Cumsum Daily_Return'], 'r-', label='SPTR' )

plt.plot(SPTR_log_test.index,portfolio_test_nobalance_15['Cumsum Daily_Return'].loc["2014-01-01": "2017-12-31"], 'g-', label='No balance Cumsum Return' )

plt.legend(loc='upper left')

plt.title('Comparison of Cumsum Returns between SPTR and 15 ETFs portfolio no rebalance')

plt.show()




SPTR_log_train = SPTR_log.loc["2008-01": "2013-12-31"].copy()


All_ETFs_log_train = All_ETFs_log.loc["2008-01": "2013-12-31"].copy()


LR_lasso_year = Lasso(alpha=0.001571,
fit_intercept=True,normalize=None,positive=True,random_state=123)
```

```
LR_lasso_year.fit(All_ETFs_log_train,SPTR_log_train[['Adj Close']])

print('\n\n ')

print('for LASSO alpha',LR_lasso_year.alpha)

print('original coeff greater than 0 is ',np.sum(LR_lasso_year.coef_ > 0 ))

print('original coeff leass than 0 is ' ,np.sum(LR_lasso_year.coef_ < 0 ))

#print('original coeff equal to 0 is ',np.sum(LR_lasso_year.coef_ == 0) )

print('we select ETFs tickers are', All_ETFs_log_train.columns[LR_lasso_year.coef_>0])


index_of_etfs = LR_lasso_year.coef_ > 0 # these etfs are our portfolio etfs, we need those to build the portfolio,and to rebalance


All_ETFs_log_year = All_ETFs_log.loc[: , index_of_etfs] .copy()  # this contains all the etfs we used to build portfolio


coefficients= LR_lasso_year.coef_.copy()



coefficients


nonzero_coeff  = coefficients[~(coefficients==0)].copy()


nonzero_coeff


for i in nonzero_coeff:
    print round(i,5)


nonzero_coeff = nonzero_coeff/nonzero_coeff.sum()


for i in nonzero_coeff:
```

```
    print round(i*100,2)




### SPTR year rebalance


All_ETFs_log.shape


pd.date_range(start='2014', end='2017', freq='A')


def SPTR_annual_balance(alpha_input = 0.000001):
    SPTR_log_train = SPTR_log.loc["2008-01": "2013-12-31"].copy()


    All_ETFs_log_train = All_ETFs_log.loc["2008-01": "2013-12-31"].copy()


    portfolio_test =
pd.DataFrame(0,columns=['portf_forest','portfolio_test_intercept'] ,index=SPTR_log.loc['2014':'2017'].in
dex)  # this is to store the test data from ETFs


    LR_lasso_year = Lasso(alpha=alpha_input,
fit_intercept=True,normalize=None,positive=True,random_state=123)


    LR_lasso_year.fit(All_ETFs_log_train,SPTR_log_train[['Adj Close']])


    print('original coeff greater than 0 is ',np.sum(LR_lasso_year.coef_ > 0 ))
    print('original coeff leass than 0 is ' ,np.sum(LR_lasso_year.coef_ < 0 ))
    #print('original coeff equal to 0 is ',np.sum(LR_lasso_year.coef_ == 0) )
    print('we select ETFs tickers are', All_ETFs_log_train.columns[LR_lasso_year.coef_>0])
```

```
    index_of_etfs = LR_lasso_year.coef_ > 0 # these etfs are our portfolio etfs, we need those to build the portfolio,and to rebalance

    All_ETFs_log_year = All_ETFs_log.loc[: , index_of_etfs] .copy()  # this contains all the etfs we used to build portfolio

    coefficients= LR_lasso_year.coef_.copy()

    nonzero_coeff  = coefficients[~(coefficients==0)].copy()

    nonzero_coeff = nonzero_coeff/nonzero_coeff.sum()

    portfolio_test.loc['2014','portf_forest'] = All_ETFs_log_year.dot(nonzero_coeff).loc['2014'].copy()

    portfolio_test.loc['2014','portfolio_test_intercept'] = portfolio_test.loc['2014','portf_forest']+LR_lasso_year.intercept_

    j = 1

    for i in pd.date_range(start='2014', end='2017', freq='A'):

        All_ETFs_log_train = All_ETFs_log.loc['2008': i].copy()

        SPTR_log_train = SPTR_log.loc['2008': i].copy()

        print('\n\nfor range in 2008 to'+str(i))

        new_alpha = alpha_finding(X=All_ETFs_log_train, Y= SPTR_log_train, pos_no = 15)

        Lasso_Model = Lasso(alpha= new_alpha,fit_intercept=True,normalize=None,random_state=123)

        Lasso_Model.fit(All_ETFs_log_train,SPTR_log_train[['Adj Close']])
```

```
#       print('number of coeff greater than 0 is ',np.sum(Lasso_Model.coef_ > 0 ))

#       print('number of coeff leass than 0 is ' ,np.sum(Lasso_Model.coef_ < 0 ))

#       print('new alpha is ',new_alpha )

      print('we select ETFs tickers are', All_ETFs_log_train.columns[Lasso_Model.coef_>0])

      Lasso_Model_residual = SPTR_log_train[['Adj Close']].values.flatten() -
Lasso_Model.predict(All_ETFs_log_train)

      check_for_stationarity(Lasso_Model_residual.flatten(),cutoff=0.05)

      index_of_etfs = Lasso_Model.coef_ > 0 # these etfs are our portfolio etfs, we need those to build
the portfolio,and to rebalance

      All_ETFs_log_year = All_ETFs_log.loc[: , index_of_etfs] .copy()  # this contains all the etfs we used to
build portfolio

      coefficients= Lasso_Model.coef_.copy()

      nonzero_coeff  = coefficients[~(coefficients==0)].copy()

      nonzero_coeff = nonzero_coeff/nonzero_coeff.sum()

      portfolio_test.loc[str(2014+j),'portf_forest'] =
All_ETFs_log_year.dot(nonzero_coeff).loc[str(2014+j)].copy()
```

```
        portfolio_test.loc[str(2014+j),'portfolio_test_intercept'] =
portfolio_test.loc[str(2014+j),'portf_forest']+Lasso_Model.intercept_



        j+=1


        print('%sth loop is good'%(j))


    SPTR_log_test = SPTR_log.loc["2014-01-01": "2017-12-31"].copy()

    plt.figure(figsize=(25,8))

    plt.plot(SPTR_log_test.index,SPTR_log_test['Adj Close'], 'r-', label='Real data' )

    plt.plot(SPTR_log_test.index,portfolio_test['portfolio_test_intercept'].loc["2014-01-01": "2017-12-
31"], 'g-', label='annual Portfolio' )

    plt.legend(loc='upper left')

    plt.title('Comparison between real S&P500 and portfolio constructed on test data')

    plt.show()


    SPTR_log_test['Daily_Return'] = SPTR_log_test['Adj Close'].diff()

    SPTR_log_test['Daily_Return'][0] = 0

    SPTR_log_test['Cumsum Daily_Return'] = SPTR_log_test['Daily_Return'].cumsum()


    portfolio_test['Daily_Return'] = portfolio_test['portf_forest'].diff()

    portfolio_test['Daily_Return'][0] = 0

    # to adjust after rebalancing, the price change. we don't want to count the daily return after
rebalancing

    for i in ['2015-01-02','2016-01-04','2017-01-03']:

        portfolio_test.loc[i,'Daily_Return'] = 0

        SPTR_log_test.loc[i,'Daily_Return'] = 0

    portfolio_test['Cumsum Daily_Return'] = portfolio_test['Daily_Return'].cumsum()
```

```
portfolio_test['Daily_Return'].plot(kind='line',figsize=(20,5),label = 'annual rebalance portfolio cumsum return',style='r-')

SPTR_log_test['Daily_Return'].loc["2014-01-01": "2017-12-31"].plot(kind='line',figsize=(20,5),label = 'SPTR cumsum return',style='g-')

plt.legend(loc='upper left')

plt.xticks( pd.date_range(start='2014-1-01', end='2018-01-01', freq = 'M'))

plt.title('Daily Return SPTR and Portfolio')

plt.show()


plt.figure(figsize=(25,8))

plt.title('Cumsum daily return')

plt.plot(SPTR_log_test.index,SPTR_log_test['Cumsum Daily_Return'], 'r-', label='Real data' )

plt.plot(SPTR_log_test.index,portfolio_test['Cumsum Daily_Return'].loc["2014-01-01": "2017-12-31"], 'g-', label='Annual balance Portfolio' )

plt.legend(loc='upper left')

plt.show()




Track_error_annual = portfolio_test['Daily_Return'] - SPTR_log_test['Daily_Return']


# to adjust after rebalancing, the price change.


for i in ['2015-01-02','2016-01-04','2017-01-03']:

    Track_error_annual.loc[i] = 0


print('Mean of Tracking Error is %.8f') %(np.mean(Track_error_annual))


print('std of Tracking Error is %.8f') %(np.std(Track_error_annual))

#print('Sum of  Tracking Error is %.8f') %(np.sum(np.abs(Track_error_annual)))
```

```
    print('Information ratio of the annual portfolio
is %.3f')%(np.mean(Track_error_annual)/np.std(Track_error_annual))


    print('Annual Correlation coefficient between portfolio and SPTR is %.6f')%(

np.corrcoef(SPTR_log_test['Daily_Return'].values.flatten(),portfolio_test['Daily_Return'].values.flatten())
[0,1])


    return portfolio_test


portfolio_test_annual_balance_15 =  SPTR_annual_balance(alpha_input=0.000101)


# portfolio_test_annual_balance_10 =  SPTR_annual_balance(alpha_input=0.000605)


# portfolio_test_annual_balance_5ETFs =  SPTR_annual_balance(alpha_input=0.001571)


plt.figure(figsize=(25,8))

plt.plot(SPTR_log_test.index,SPTR_log_test['Adj Close'], 'r-', label='SPTR' )

plt.plot(SPTR_log_test.index,portfolio_test_annual_balance_15['portfolio_test_intercept'].loc["2014-01-
01": "2017-12-31"], 'g-', label='15 Annual Portfolio' )

plt.legend(loc='upper left')

plt.title('Comparison between SPTR and 15 ETFs portfolio Annual rebalance')

plt.show()


plt.figure(figsize=(25,8))

portfolio_test_annual_balance_15['Daily_Return'].plot(kind='line',figsize=(20,5),label = '15 Annual
Portfolio',style='r-')

SPTR_log_test['Daily_Return'].loc["2014-01-01": "2017-12-31"].plot(kind='line',figsize=(20,5),label =
'SPTR',style='g-')

plt.legend(loc='upper left')
```

```
plt.xticks( pd.date_range(start='2014-1-01', end='2018-01-01', freq = 'M'))

plt.title('Comparison of Daily Returns between SPTR and 15 ETFs portfolio Annual rebalance')

plt.show()



plt.figure(figsize=(25,8))

plt.plot(SPTR_log_test.index,SPTR_log_test['Cumsum Daily_Return'], 'r-', label='SPTR' )

plt.plot(SPTR_log_test.index,portfolio_test_annual_balance_15['Cumsum Daily_Return'].loc["2014-01-
01": "2017-12-31"], 'g-', label='15 Annual Portfolio' )

plt.legend(loc='upper left')

plt.title('Comparison of Cumsum Returns between SPTR and 15 ETFs portfolio no rebalance')

plt.show()
```

SPTR semi-annual rebalance

```
def  SPTR_semi_annual_balance(alpha_input= 6.1e-05):

   SPTR_log_train = SPTR_log.loc["2008-01": "2013-12-31"].copy()

   All_ETFs_log_train = All_ETFs_log.loc["2008-01": "2013-12-31"].copy()

   portfolio_test_semi =
pd.DataFrame(0,columns=['portf_forest','portfolio_test_intercept'] ,index=SPTR_log.loc['2014':'2017'].in
dex)  # this is to store the test data from ETFs

   LR_lasso_semi = Lasso(alpha=alpha_input, fit_intercept=True,normalize=None,positive=True,
random_state=123)

   LR_lasso_semi.fit(All_ETFs_log_train,SPTR_log_train[['Adj Close']])

   print('original coeff greater than 0 is ',np.sum(LR_lasso_semi.coef_ > 0 ))

   print('original coeff leass than 0 is ' ,np.sum(LR_lasso_semi.coef_ < 0 ))

   print('original coeff equal to 0 is ',np.sum(LR_lasso_semi.coef_ == 0) )

   index_of_etfs = LR_lasso_semi.coef_ > 0 # these etfs are our portfolio etfs, we need those to build the
portfolio,and to rebalance

   All_ETFs_log_semi = All_ETFs_log.loc[: , index_of_etfs] .copy()  # this contains all the etfs we used to
build portfolio

   coefficients= LR_lasso_semi.coef_.copy()

   nonzero_coeff  = coefficients[~(coefficients==0)].copy()

   nonzero_coeff = nonzero_coeff/nonzero_coeff.sum()
```

```
    portfolio_test_semi.loc['20140101':'20140630','portf_forest'] =
All_ETFs_log_semi.dot(nonzero_coeff).loc['20140101':'20140630'].copy()

    portfolio_test_semi.loc['20140101':'20140630','portfolio_test_intercept'] =
portfolio_test_semi.loc['20140101':'20140630','portf_forest']+LR_lasso_semi.intercept_

    j = 0

    semi_dates_begin = pd.date_range(start='2014-07-01', end='2017-07-01', freq='6MS')

    semi_dates_end = pd.date_range(start='2014-06-30', end='2017-12-31', freq='6M',closed='right')

    for i in pd.date_range(start='2013-12-31', end='2017-06-30', freq='6M',closed='right'):

        All_ETFs_log_train = All_ETFs_log.loc['2008': i].copy()

        SPTR_log_train = SPTR_log.loc['2008': i].copy()

        print('\n\nfor range in 2008 to'+str(i))

        new_alpha = alpha_finding(X=All_ETFs_log_train, Y= SPTR_log_train, pos_no=15)

        Lasso_Model = Lasso(alpha=new_alpha , fit_intercept=True,normalize=None,random_state=123)

        Lasso_Model.fit(All_ETFs_log_train,SPTR_log_train[['Adj Close']])

        print('we select ETFs tickers are  ',All_ETFs_log_train.columns[Lasso_Model.coef_>0] )
```

```
        Lasso_Model_residual = SPTR_log_train[['Adj Close']].values.flatten() -
Lasso_Model.predict(All_ETFs_log_train)


        check_for_stationarity(Lasso_Model_residual.flatten(),cutoff=0.05)


        index_of_etfs = Lasso_Model.coef_ > 0 # these etfs are our portfolio etfs, we need those to build
the portfolio,and to rebalance


        All_ETFs_log_semi = All_ETFs_log.loc[: , index_of_etfs] .copy()  # this contains all the etfs we used to
build portfolio


        coefficients= Lasso_Model.coef_.copy()


        nonzero_coeff  = coefficients[~(coefficients==0)].copy()


        nonzero_coeff = nonzero_coeff/nonzero_coeff.sum()


        portfolio_test_semi.loc[semi_dates_begin[j]:semi_dates_end[j],'portf_forest'] =
All_ETFs_log_semi.dot(nonzero_coeff).loc[semi_dates_begin[j]:semi_dates_end[j]].copy()


        portfolio_test_semi.loc[semi_dates_begin[j]:semi_dates_end[j],'portfolio_test_intercept'] =
portfolio_test_semi.loc[semi_dates_begin[j]:semi_dates_end[j],'portf_forest']+Lasso_Model.intercept_




        j+=1


        print('%sth loop is good'%(j))


    plt.figure(figsize=(25,8))
    plt.plot(SPTR_log_test.index,SPTR_log_test['Adj Close'], 'r-', label='Real data' )
```

```
plt.plot(SPTR_log_test.index,portfolio_test_semi['portfolio_test_intercept'].loc["2014-01-01": "2017-12-31"], 'g-', label='semi Portfolio' )

plt.legend(loc='upper left')

plt.title('Semi annaual Comparison between real SPTR and portfolio constructed on test data')

plt.show()


portfolio_test_semi['Daily_Return'] = portfolio_test_semi['portf_forest'].diff()

portfolio_test_semi['Daily_Return'][0] = 0

# to adjust after rebalancing, the price change. we don't want to count the daily return after rebalancing

for i in ['2014-07-01','2015-01-02','2015-07-01','2016-01-04','2016-07-01','2017-01-03','2017-07-03']:

    portfolio_test_semi.loc[i,'Daily_Return'] = 0

    SPTR_log_test.loc[i,'Daily_Return'] = 0


portfolio_test_semi['Cumsum Daily_Return'] = portfolio_test_semi['Daily_Return'].cumsum()


portfolio_test_semi['Daily_Return'].plot(kind='line',figsize=(20,5),label = 'semi rebalance cumsum return',style='r-')

SPTR_log_test['Daily_Return'].loc["2014-01-01": "2017-12-31"].plot(kind='line',figsize=(20,5),label = 'SPTR cumsum return',style='g-')

plt.legend(loc='upper left')

plt.xticks( pd.date_range(start='2014-1-01', end='2018-01-01', freq = 'M'))

plt.title('Semi-annual Daily Return SPTR and Portfolio')

plt.show()


plt.figure(figsize=(25,8))

plt.title('Cumsum daily return')

plt.plot(SPTR_log_test.index,SPTR_log_test['Cumsum Daily_Return'], 'r-', label='Real data' )

plt.plot(SPTR_log_test.index,portfolio_test_semi['Cumsum Daily_Return'].loc["2014-01-01": "2017-12-31"], 'g-', label='Semi-annual balance Portfolio' )
```

```
    plt.legend(loc='upper left')
    plt.show()



    Track_error_semi = portfolio_test_semi['Daily_Return'] - SPTR_log_test['Daily_Return']


    print('Mean of Tracking Error is %.8f') %(np.mean(Track_error_semi))


    print('Std of Tracking Error is %.8f') %(np.std(Track_error_semi))


    #print('Sum of  Tracking Error is %.8f') %(np.sum(np.abs(Track_error_semi)))


    print('Semi-annual information ratio of the annual portfolio
is %.3f')%(np.mean(Track_error_semi)/np.std(Track_error_semi))


    print('Semi-annual  Correlation coefficient between portfolio and SPTR is %.6f')%(

np.corrcoef(SPTR_log_test['Daily_Return'].values.flatten(),portfolio_test_semi['Daily_Return'].values.fla
tten())[0,1])
    return portfolio_test_semi


portfolio_test_semi_annual_balance_15 =  SPTR_semi_annual_balance(alpha_input=0.000101)




plt.figure(figsize=(25,8))
plt.plot(SPTR_log_test.index,SPTR_log_test['Adj Close'], 'r-', label='SPTR' )
plt.plot(SPTR_log_test.index,portfolio_test_semi_annual_balance_15['portfolio_test_intercept'].loc["20
14-01-01": "2017-12-31"], 'g-', label='15 Semi-annual Portfolio' )
plt.legend(loc='upper left')
```

```
plt.title('Comparison between SPTR and 15 ETFs portfolio Semi-annual rebalance')

plt.show()


plt.figure(figsize=(25,8))

portfolio_test_semi_annual_balance_15['Daily_Return'].plot(kind='line',figsize=(20,5),label = '15 Semi-annual Portfolio',style='r-')

SPTR_log_test['Daily_Return'].loc["2014-01-01": "2017-12-31"].plot(kind='line',figsize=(20,5),label = 'SPTR',style='g-')

plt.legend(loc='upper left')

plt.xticks( pd.date_range(start='2014-1-01', end='2018-01-01', freq = 'M'))

plt.title('Comparison of Daily Returns between SPTR and 15 ETFs portfolio semi-annual rebalance')

plt.show()



plt.figure(figsize=(25,8))

plt.plot(SPTR_log_test.index,SPTR_log_test['Cumsum Daily_Return'], 'r-', label='SPTR' )

plt.plot(SPTR_log_test.index,portfolio_test_semi_annual_balance_15['Cumsum Daily_Return'].loc["2014-01-01": "2017-12-31"], 'g-', label='15 Semi-annual Portfolio' )

plt.legend(loc='upper left')

plt.title('Comparison of Cumsum Returns between SPTR and 15 ETFs portfolio semi-annual rebalance')

plt.show()



## rebalance quarterly


def SPTR_quarter_balance(alpha_input= 6.1e-05):

    SPTR_log_train = SPTR_log.loc["2008-01": "2013-12-31"].copy()


    All_ETFs_log_train = All_ETFs_log.loc["2008-01": "2013-12-31"].copy()
```

```
   portfolio_test_quarter =
pd.DataFrame(0,columns=['portf_forest','portfolio_test_intercept'] ,index=SPTR_log.loc['2014':'2017'].in
dex)  # this is to store the test data from ETFs


   LR_lasso_quarter = Lasso(alpha= alpha_input,fit_intercept=True,positive=True,normalize=None)


   LR_lasso_quarter.fit(All_ETFs_log_train,SPTR_log_train[['Adj Close']])


   print('original coeff greater than 0 is ',np.sum(LR_lasso_quarter.coef_ > 0 ))

   print('original coeff leass than 0 is ' ,np.sum(LR_lasso_quarter.coef_ < 0 ))

   print('original coeff equal to 0 is ',np.sum(LR_lasso_quarter.coef_ == 0) )


   index_of_etfs = LR_lasso_quarter.coef_ > 0 # these etfs are our portfolio etfs, we need those to build
the portfolio,and to rebalance


   All_ETFs_log_quarter = All_ETFs_log.loc[: , index_of_etfs] .copy()  # this contains all the etfs we used
to build portfolio


   coefficients= LR_lasso_quarter.coef_.copy()


   nonzero_coeff  = coefficients[~(coefficients==0)].copy()


   nonzero_coeff = nonzero_coeff/nonzero_coeff.sum()


   portfolio_test_quarter.loc['2014Q1','portf_forest'] =
All_ETFs_log_quarter.dot(nonzero_coeff).loc['2014Q1'].copy()


   portfolio_test_quarter.loc['2014Q1','portfolio_test_intercept'] =
portfolio_test_quarter.loc['2014Q1','portf_forest']+LR_lasso_quarter.intercept_


   quarter_dates = [str(i)+j for i in range(2014,2018) for j in ['Q1','Q2','Q3','Q4']]
```

```
    j = 1


    for i in pd.date_range(start='2014Q1', end='2017Q4', freq='Q'):

        All_ETFs_log_train = All_ETFs_log.loc['2008': i].copy()

        SPTR_log_train = SPTR_log.loc['2008': i].copy()

        print('\n\nfor range in 2008 to'+str(i))

        new_alpha = alpha_finding(X=All_ETFs_log_train, Y= SPTR_log_train, pos_no=15)

        Lasso_Model = Lasso(alpha=new_alpha,
fit_intercept=True,positive=True,normalize=None,random_state=123)

        Lasso_Model.fit(All_ETFs_log_train,SPTR_log_train[['Adj Close']])

        print('we select ETF tickers are ',All_ETFs_log_train.columns[Lasso_Model.coef_>0] )

        Lasso_Model_residual = SPTR_log_train[['Adj Close']].values.flatten() -
Lasso_Model.predict(All_ETFs_log_train)

        check_for_stationarity(Lasso_Model_residual.flatten(),cutoff=0.05)

        index_of_etfs = Lasso_Model.coef_ > 0 # these etfs are our portfolio etfs, we need those to build
the portfolio,and to rebalance
```

```
        All_ETFs_log_quarter = All_ETFs_log.loc[: , index_of_etfs] .copy()  # this contains all the etfs we used to build portfolio


        coefficients= Lasso_Model.coef_.copy()


        nonzero_coeff  = coefficients[~(coefficients==0)].copy()


        nonzero_coeff = nonzero_coeff/nonzero_coeff.sum()


        portfolio_test_quarter.loc[quarter_dates[j],'portf_forest'] = All_ETFs_log_quarter.dot(nonzero_coeff).loc[quarter_dates[j]].copy()


        portfolio_test_quarter.loc[quarter_dates[j],'portfolio_test_intercept'] = portfolio_test_quarter.loc[quarter_dates[j],'portf_forest']+Lasso_Model.intercept_


        j+=1


        print('%sth loop is good'%(j))




    plt.figure(figsize=(25,8))

    plt.plot(SPTR_log_test.index,SPTR_log_test['Adj Close'], 'r-', label='Real data' )

    plt.plot(SPTR_log_test.index,portfolio_test_quarter['portfolio_test_intercept'].loc["2014-01-01":"2017-12-31"], 'g-', label='quarter Portfolio' )

    plt.legend(loc='upper left')

    plt.title('Comparison between real S&P500 and portfolio constructed on test data')

    plt.show()


    portfolio_test_quarter['Daily_Return'] = portfolio_test_quarter['portf_forest'].diff()
```

```
    portfolio_test_quarter['Daily_Return'][0] = 0

    # to adjust after rebalancing, the price change. we don't want to count the daily return after rebalancing

    for i in ['2014-04-01','2014-07-01','2014-10-01','2015-01-02','2015-04-01','2015-07-01','2015-10-01','2016-01-04',

              '2016-04-01','2016-07-01','2016-10-03','2017-01-03','2017-04-03','2017-07-03','2017-10-02']:

        portfolio_test_quarter.loc[i,'Daily_Return'] = 0

        SPTR_log_test.loc[i,'Daily_Return'] = 0

    portfolio_test_quarter['Cumsum Daily_Return'] = portfolio_test_quarter['Daily_Return'].cumsum()


    portfolio_test_quarter['Daily_Return'].plot(kind='line',figsize=(20,5),label = 'quarterly rebalance portfolio cumsum return',style='r-')

    SPTR_log_test['Daily_Return'].loc["2014-01-01": "2017-12-31"].plot(kind='line',figsize=(20,5),label = 'SPTR cumsum return',style='g-')

    plt.legend(loc='upper left')

    plt.xticks( pd.date_range(start='2014-1-01', end='2018-01-01', freq = 'M'))

    plt.title('Quarterly Cumsum Return SPTR and Portfolio')

    plt.show()


    plt.figure(figsize=(25,8))

    plt.title('Cumsum daily return')

    plt.plot(SPTR_log_test.index,SPTR_log_test['Cumsum Daily_Return'], 'r-', label='Real data' )

    plt.plot(SPTR_log_test.index,portfolio_test_quarter['Cumsum Daily_Return'].loc["2014-01-01": "2017-12-31"], 'g-', label='Quarterly balance Portfolio' )

    plt.legend(loc='upper left')

    plt.show()


    Track_error_quarter = portfolio_test_quarter['Daily_Return'] - SPTR_log_test['Daily_Return']


    print('Mean of Tracking Error is %.8f') %(np.mean(Track_error_quarter))
```

```
    print('std of Tracking Error is %.8f') %(np.std(Track_error_quarter))

    #print('Sum of  Tracking Error is %.8f，') %(np.sum(np.abs(Track_error_quarter)))

    print('Quarter information ratio of the portfolio
is %.3f')%(np.mean(Track_error_quarter)/np.std(Track_error_quarter))

    print('Quarterly  Correlation coefficient between portfolio and SPTR is %.6f')%(

np.corrcoef(SPTR_log_test['Daily_Return'].values.flatten(),portfolio_test_quarter['Daily_Return'].values.
flatten())[0,1])

    return portfolio_test_quarter

portfolio_test_quarter_balance_15 =  SPTR_quarter_balance(alpha_input=0.000101)

plt.figure(figsize=(25,8))
plt.plot(SPTR_log_test.index,SPTR_log_test['Adj Close'], 'r-', label='SPTR' )
plt.plot(SPTR_log_test.index,portfolio_test_quarter_balance_15['portfolio_test_intercept'].loc["2014-
01-01": "2017-12-31"], 'g-', label='15 quarter Portfolio' )
plt.legend(loc='upper left')
#plt.title('Comparison between SPTR and 15 ETFs portfolio Quarterly-annual rebalance')
plt.show()

plt.figure(figsize=(25,8))
portfolio_test_quarter_balance_15['Daily_Return'].plot(kind='line',figsize=(20,5),label = '15 Quarterly
Portfolio',style='r-')
SPTR_log_test['Daily_Return'].loc["2014-01-01": "2017-12-31"].plot(kind='line',figsize=(20,5),label =
'SPTR',style='g-')
plt.legend(loc='upper left')
```

```
plt.xticks( pd.date_range(start='2014-1-01', end='2018-01-01', freq = 'M'))
plt.show()


plt.figure(figsize=(25,8))
plt.plot(SPTR_log_test.index,SPTR_log_test['Cumsum Daily_Return'], 'r-', label='SPTR' )
plt.plot(SPTR_log_test.index,portfolio_test_quarter_balance_15['Cumsum Daily_Return'].loc["2014-01-01": "2017-12-31"], 'g-', label='15 Quarterly Portfolio' )
plt.legend(loc='upper left')
#plt.title('Comparison of Cumsum Returns between SPTR and 15 ETFs portfolio semi-annual rebalance')
plt.show()



def check_for_stationarity(X, cutoff=0.01):
    # H_0 in adfuller is unit root exists (non-stationary)
    # We must observe significant p-value to convince ourselves that the series is stationary
    pvalue = ADF(X)[1]
    if pvalue < cutoff:
        print 'p-value = ' + str(pvalue) + ' The series '  +' is likely stationary.'
        #return True
    else:
        print 'p-value = ' + str(pvalue) + ' The series '+' is likely non-stationary.'
        #return False

def check_for_stationarity_no_print(X, cutoff=0.01):
    # H_0 in adfuller is unit root exists (non-stationary)
    # We must observe significant p-value to convince ourselves that the series is stationary
    pvalue = ADF(X)[1]
    if pvalue < cutoff:
        #print 'p-value = ' + str(pvalue) + ' The series '  +' is likely stationary.'
```

```
    return True
else:
    #print 'p-value = ' + str(pvalue) + ' The series '+' is likely non-stationary.'
    return False
```