Author: Haotian Zhang Date:2019.01.09

QUANTITATIVE REPORT 3 YIELD, RISK OF PORTFOLIO

1. Foundations

A. Background

There is a famous saying in investment science, "Don't put all the eggs in the same column."

As for the Markowitz mean-variance model, combining certain types of profitable products can minimize the systematic risk. Therefore, under a certain rate of return, the combination of multiple risk assets will be less than the single asset with the same rate of return. The risk, that is, the smaller fluctuations.
In this regard, this phenomenon is also very common in real life. In order to avoid risks, investors tend to diversify their assets into different financial instruments, such as trusts, bonds, funds, stocks, futures, options, and even real estate markets. So in so many financial products, how do we choose to get the highest possible return under risk-controlled conditions? Asset allocation is to solve this problem.

B. Three performance estimator

i. Return

$$\mathrm{E}(R_p) = \sum_i w_i \, \mathrm{E}(R_i)$$

where Rp is the return on the portfolio, Ri is the return on asset *i* and wi is the weighting of component asset *i* (that is, the proportion of asset "i" in the portfolio).

ii. Risk

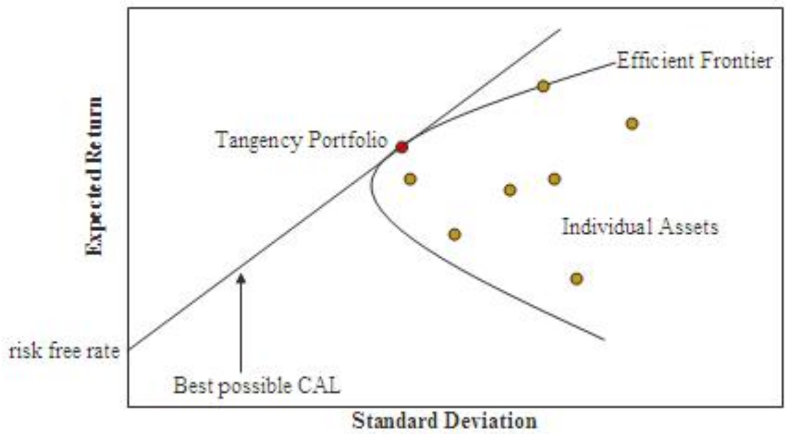
$$\sigma_p^2 = \sum_i w_i^2 \sigma_i^2 + \sum_i \sum_{j \neq i} w_i w_j \sigma_i \sigma_j \rho_{ij}$$

where σ is the (sample) standard deviation of the periodic returns on an asset, and ρij is the correlation coefficient between the returns on assets i and j.

iii. Sharpe Ratio

$$\frac{R_a - R_f}{\sigma_a}$$

Where Ra = Rp is the return on the portfolio, Rf is the return on risk-free asset, (Normally risk-free rate is LIBOR or treasury notes of USA for $)
σa is the (sample) standard deviation of the periodic returns on an asset.



SD-return graph

As the picture above, we can use Monte Carlo simulation to allocate different weights to the portfolio assets and collect the mean return and variance of the portfolio.

Basically, the bigger is Sharpe Ratio, the better is the performance of the portfolio. Then we can adjust the allocation of risk-free assets and portfolio.

C. What I will do?

In this paper, I will do two things separately with Python and Excel:

1. *Implement Python codes to process up-to-date "daily" data from Yahoo finance and mainly use Monte Carlo simulations to allocate portfolios, then use optimization to get the portfolio of the best performance.*
2. *Evaluate the covariance optimizing techniques such as shrinkage methods. Use data back testing, I used historical data from year 2011 to 2015 to computer the weight vector of portfolio and tested the weight vector with the data in year 2016. Then I separately tested the proportions in year 2016, 2017 and 2018 with each specific shrinkage method, and compared the performance of the three covariance techniques.*

The first project is a practical model which can help us easily and conveniently get data from website and allocate a best performance portfolio according to the historical data;

Whereas the second project is a research one that implement different shrinkage methods on both American stock markets and Chinese A share markets and use monthly return in latter months to data back test the portfolio, evaluate the performance of 3 main shrinkage methods.

2. Empirical results of the first project

A. Process

1. get stock historical data from Yahoo Finance with pandas-datareader package;

2. use python to plot standardize stock price in a picture and count daily log-returns of each stock;

3. compute the mean-return and covariance matrix of the assets;

4. use Monte Carlo to get 4000 weight vector and count each return and standard deviation;

5. optimize to get the portfolio with the biggest Sharpe ratio.

B. Results

Stock pools: Apple/Facebook/Google/Microsoft/Netflix

Pic1: I normalize the stock price into the same standard “1”

Date	AAPL	FB	GOOG	MSFT	NFLX
2017-01-03	1.000000	1.000000	1.000000	1.000000	1.000000
2017-01-04	0.998881	1.015660	1.000967	0.995526	1.015060
2017-01-05	1.003960	1.032603	1.010024	0.995526	1.033885
2017-01-06	1.015153	1.056050	1.025453	1.004155	1.028081
2017-01-09	1.024451	1.068800	1.026090	1.000959	1.027139

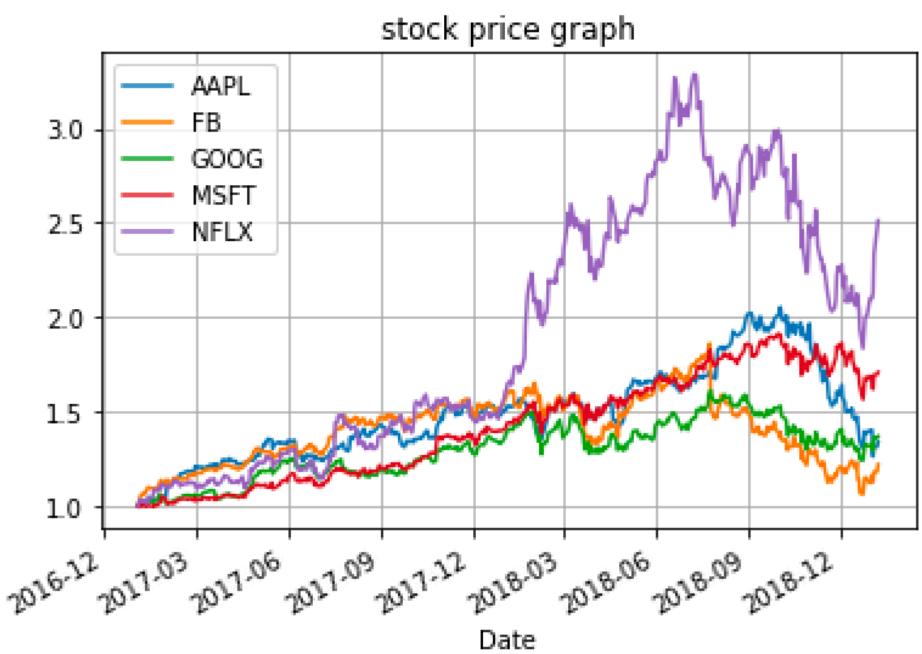
Pic2: Five year returns of five different stocks: Apple/Facebook/Google/Microsoft/Netflix

AAPL	0.145221
FB	0.098895
GOOG	0.156445
MSFT	0.266734
NFLX	0.458743

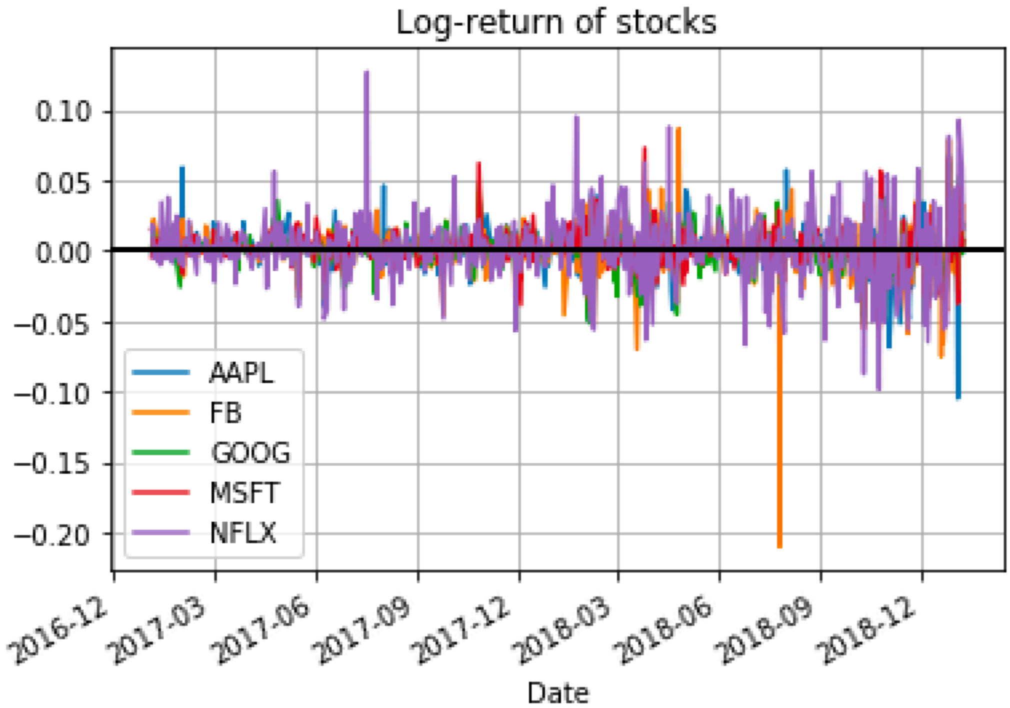
Pic3: Covariance matrix of five different stocks:

	AAPL	FB	GOOG	MSFT	NFLX
AAPL	0.062974	0.035512	0.036815	0.036732	0.046309
FB	0.035512	0.093237	0.044390	0.037910	0.056939
GOOG	0.036815	0.044390	0.053059	0.041186	0.054740
MSFT	0.036732	0.037910	0.041186	0.051971	0.054309
NFLX	0.046309	0.056939	0.054740	0.054309	0.149869

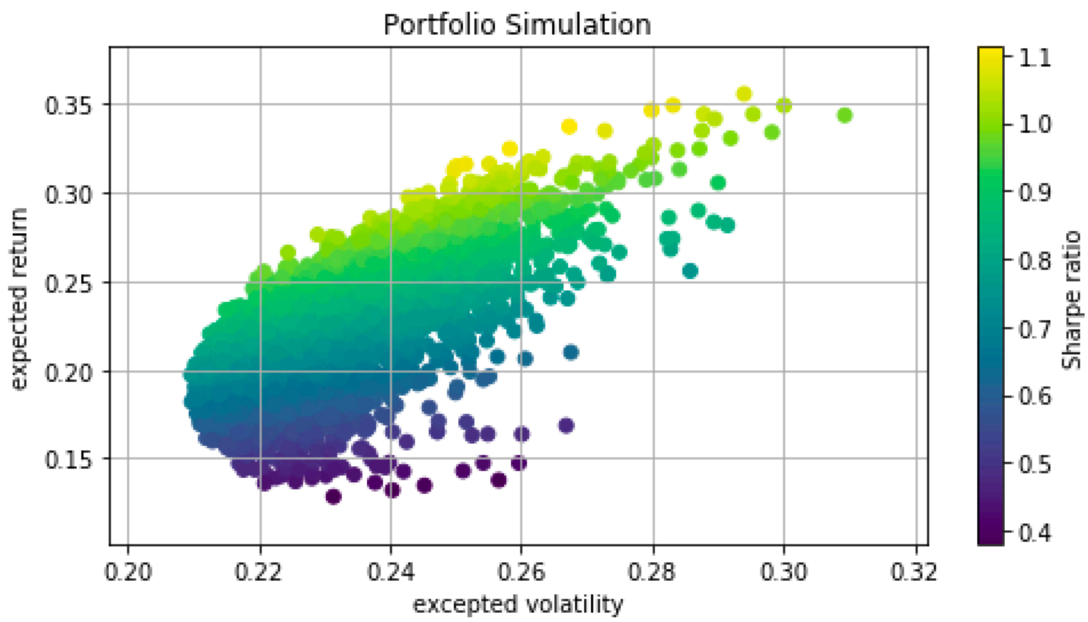
Pic4: stardardized price graph of five different stocks:



Pic5: Log-returns of five different stocks:



Pic6: Monte Carlo simulation with 4000 random-weighted portfolio



Pic7: Weight vector of "Max Sharpe Ratio" strategy

```
[0.       0.       0.       0.61657 0.38343]
```

Pic8: Weight vector of "Min Variance" strategy

```
[0.26639 0.0811  0.29275 0.35976 0.     ]
```

Pic9: return, SD and Sharpe Ratio of "Max Sharpe Ratio" strategy

```
[0.34036 0.25975 1.31033]
```

Pic10: return, SD and Sharpe Ratio of "Min Variance" strategy

```
[0.18847 0.20897 0.90188]
```

Remarks:

"Min Variance" Strategy means that we try to get the minimum variance of the portfolio in our target optimizing function, whereas "Max Sharpe Ratio" strategy means that we try to get maximum Sharpe ratio.

3. Empirical results of the second project

Following Markowitz (1952, 1959) we define the problem of portfolio selection as follows:

$$\begin{aligned}&\underset{\mathbf{w}}{\text{Min}}\ \mathbf{w}^{\mathbf{T}}\boldsymbol{\Sigma}\mathbf{w}\\&\text{subject to}\\&\mathbf{w}^{\mathbf{T}}\boldsymbol{\mu}=q\\&\mathbf{w}^{\mathbf{T}}\mathbf{1}=1\end{aligned}$$

where **w** denotes the vector of portfolio weights, q denotes the expected return that is required on the portfolio and **1** denotes a vector of ones.

$$\hat{\Sigma}=\frac{1}{T-1}R(I-\frac{1}{T}\iota\iota')R'$$

For this model, the estimator of covariance matrix is mostly concerned and simply we use sample covariance matrixs, whereas the sample matrixs have problems: mainly the covariance matrixs cannot be inversed when sample amounts are fixed while return datas can go to infinite.[1]However, Shrinkage methods improve performances.

For the second project, Evaluate the covariance optimizing techniques such as shrinkage methods. Use data back testing, I used historical data from year 2011 to 2015 to computer the weight vector of portfolio and tested the weight vector with the data in year 2016. Then I separately tested the proportions in year 2016, 2017 and 2018 with each specific shrinkage method, and compared the performance of the three covariance techniques：

i. Diagonal matrix
ii. Sample matrix
iii. Two-block covariance matrix

iv. Shrinkage convariance matrix

[1]Estimating the Covariance Matrix for Portfolio Optimization

Stock pools:

Amazon	Exxon	Microsoft	Alphabet(GOOG)	Booking	AT&T	Boeing	CVS

Times:2011.02-2015.12 / 2016.01-2016.12 data back testing

回报率	时间	Amazon	Exxon	Microsoft	Alphabet(GOOG	Booking	AT&T	Boeing	CVS
	2011/2/1	0.021516117	0.060114018	-0.041471331	0.021720326	0.059180474	0.031249964	0.03641334	-0.033333332
	2011/3/1	0.039471477	-0.016368549	-0.044770542	-0.043430133	0.115801526	0.078576536	0.026662935	0.038112491
	2011/4/1	0.087048202	0.045762583	0.0208744	-0.072704252	0.080108222	0.016661221	0.079128892	0.055361334
	2011/5/1	0.004494173	-0.051261671	-0.035108025	-0.02771559	-0.058170797	0.014138753	-0.021935322	0.068194311
	2011/6/1	0.039656327	-0.025038984	0.039584166	-0.042796081	-0.00632778	-0.00475282	-0.052543881	-0.028689507
	2011/7/1	0.088170564	-0.019537922	0.053846154	0.192167941	0.050241305	-0.068449538	-0.046801014	-0.03273028
	2011/8/1	-0.032761135	-0.072314876	-0.02919708	-0.103910962	-0.000725405	-0.026657553	-0.051227472	-0.012104485
	2011/9/1	0.004646193	-0.018778709	-0.064285752	-0.047914759	-0.163421839	0.001404494	-0.094974617	-0.064605959
	2011/10/1	-0.012579203	0.075175537	0.069907596	0.150667893	0.129622238	0.027699825	0.087258357	0.081571956
	2011/11/1	-0.099386457	0.03009356	-0.039429179	0.011389677	-0.042996112	-0.011258922	0.044079662	0.069088848
	2011/12/1	-0.099797117	0.053704623	0.014855317	0.077595591	-0.037415924	0.043478261	0.067840965	0.049948481
	2012/1/1	0.123281313	-0.012034025	0.137519343	-0.101857877	0.132090398	-0.02744709	0.011315638	0.023786195
	2012/2/1	-0.075858876	0.032959184	0.07483911	0.065746177	0.184196173	0.040122407	0.010380116	0.080239473
	2012/3/1	0.126996454	0.002658994	0.01638305	0.037185593	0.144301581	0.02092187	-0.007738413	-0.006651863
	2012/4/1	0.145128634	-0.004496794	-0.007439492	-0.05674942	0.060376316	0.053794428	0.032674464	-0.004017857
	2012/5/1	-0.081888704	-0.089298116	-0.08838223	-0.039662701	-0.177873894	0.038286174	-0.093619814	0.007171672
	2012/6/1	0.072518913	0.088261519	0.047961595	-0.001360082	0.062399128	0.043605563	0.067375405	0.039830909
	2012/7/1	0.021677236	0.01495849	-0.036613305	0.091195899	-0.004183516	0.063376276	-0.005248977	-0.031671303
	2012/8/1	0.064166313	0.005181405	0.045809333	0.082341979	-0.086393423	-0.03375525	-0.033960247	0.006629812
	2012/9/1	0.024368643	0.047537158	-0.034393251	0.101315085	0.023983988	0.028930186	-0.025210139	0.063007663
	2012/10/1	-0.084263949	-0.003061772	-0.04099459	-0.098343278	-0.073174256	-0.082493393	0.012069023	-0.041718217
	2012/11/1	0.082270617	-0.033234606	-0.067273999	0.026561833	0.155846382	-0.013298612	0.054514436	0.002370603
	2012/12/1	-0.004681642	-0.018039437	0.003380841	0.012901478	-0.064536538	-0.012305947	0.014539607	0.039561386
	2013/1/1	0.058317078	0.039514707	0.027705055	0.068294246	0.104901682	0.032038031	-0.019771735	0.058945256
	2013/2/1	-0.00463281	-0.0046682	0.012750382	0.060223146	0.005310246	0.032193129	0.041017989	-0.001562539
	2013/3/1	0.008400504	0.006253467	0.029136764	-0.008749402	-0.001378607	0.021720941	0.11638486	0.075704286
	2013/4/1	-0.047581495	-0.012429286	0.156938023	0.038252852	0.011378193	0.020986645	0.064764195	0.058010509
	2013/5/1	0.060635964	0.016631116	0.054380789	0.056574895	0.155088442	-0.065936921	0.083251205	-0.010312788
	2013/6/1	0.031537851	-0.00132644	-0.010315214	0.010502525	0.028286033	0.011717633	0.034538529	-0.006946891
	2013/7/1	0.084734772	0.037631456	-0.078170264	0.008382868	0.059273956	-0.003672373	0.02596638	0.075376041
	2013/8/1	-0.06719338	-0.070293291	0.048995038	-0.046015241	0.071785087	-0.040827842	-0.011227403	-0.055944103
	2013/9/1	0.112677069	-0.012849965	-0.003592904	0.034254379	0.077163203	-0.000295655	0.130677466	-0.022394471
	2013/10/1	0.164374301	0.041608577	0.064002436	0.176582078	0.042415494	0.07037259	0.110638298	0.097092476
	2013/11/1	0.081284499	0.043070742	0.076814487	0.02814917	0.131425378	-0.027348121	0.028735632	0.0754899
	2013/12/1	0.013134531	0.082584443	-0.018882795	0.057682742	-0.025102055	-0.001420023	0.016685326	0.068847089
	2014/1/1	-0.100554192	-0.089327997	0.011494253	0.053769489	-0.015063669	-0.052332196	-0.082277109	-0.053793475
	2014/2/1	0.009506828	0.044596276	0.012420745	0.029365615	0.178139339	-0.041716687	0.029219192	0.080035409
	2014/3/1	-0.071057748	0.014646339	0.069955649	-0.080232544	-0.116359209	0.098340119	-0.026605647	0.023516571
	2014/4/1	-0.095846807	0.048423464	-0.014393754	-0.057008414	-0.028643595	0.0179641	0.02812978	-0.028586695
	2014/5/1	0.027685473	-0.01835763	0.013366262	0.063095729	0.104409419	-0.006442577	0.048287055	0.077007686
	2014/6/1	0.039129776	0.001492102	0.018563801	0.027487555	-0.059149249	-0.003101212	-0.059297575	-0.037665947
	2014/7/1	-0.036301524	-0.017282459	0.035011966	-0.006396841	0.032792977	0.006504496	-0.053053524	0.013135173
	2014/8/1	0.08322956	0.00525568	0.052594995	0	0.001497129	-0.01770163	0.052456838	0.04046616
	2014/9/1	-0.048961794	-0.054393686	0.020471076	0.010076895	-0.068897701	0.008009239	0.004574085	0.001762102
	2014/10/1	-0.052660994	0.028282785	0.012726488	-0.031661281	0.041110676	-0.011350794	-0.019390745	0.07815055
	2014/11/1	0.108623142	-0.063798967	0.018317359	-0.030854317	-0.038152578	0.015499455	0.075654445	0.064677813
	2014/12/1	-0.083540065	0.021095604	-0.028445931	-0.028477492	-0.017221301	-0.050593583	-0.032599025	0.054181228
	2015/1/1	0.14235538	-0.054407779	-0.130247554	0.015420918	-0.114654305	-0.019946472	0.118402827	0.019208868
	2015/2/1	0.072292909	0.012811748	0.085395936	0.04467564	0.225858863	0.049817834	0.037696988	0.058170322
	2015/3/1	-0.021201594	-0.03998194	-0.072747962	-0.018624637	-0.059257489	-0.055266173	-0.005104435	-0.006354135
	2015/4/1	0.133512476	0.027882388	0.196261658	-0.014056541	0.063273662	0.060949368	-0.044909421	-0.037980797
	2015/5/1	0.017663265	-0.024836968	-0.036595354	-0.009733208	-0.053134178	-0.002886778	-0.019673448	0.031120918
	2015/6/1	0.011322566	-0.023474179	-0.057831817	-0.021799957	-0.017635954	0.028372871	-0.012809585	0.024418833
	2015/7/1	0.235112601	-0.047956708	0.057757619	0.201917298	0.080078473	-0.021959403	0.039287752	0.072368461
	2015/8/1	-0.043383396	-0.050119948	-0.068094238	0.01918128	0.004077028	-0.037420925	-0.093570127	-0.089534977
	2015/9/1	-0.001949736	-0.011828815	0.017003631	-0.04578034	-0.009434288	-0.025717614	0.002066146	-0.057812489
	2015/10/1	0.222723643	0.112844657	0.189335774	0.168288383	0.152636548	0.028544995	0.130737002	0.023839095
	2015/11/1	0.062150443	-0.013052865	0.032484784	0.044723594	0.03080001	0.004774694	-0.0176944	-0.047479278
	2015/12/1	0.01668175	-0.045432364	0.020791206	0.021923013	-0.132427461	0.021978083	-0.005912692	0.039111501
	期望收益率	0.026718897	0.000368605	0.013745094	0.018199634	0.022510482	0.004502951	0.014004543	0.019044407
	方差	0.006267435	0.001909145	0.003974143	0.004688622	0.007844324	0.001421475	0.00305817	0.00218239
	标准差	0.079167135	0.043693759	0.063040806	0.068473511	0.088568191	0.037702453	0.055300721	0.046716053

Remarks: the data are from Yahoo Finance

i. Diagonal matrix

P1: mean return and diagonal matrix with data from 2011-2015

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Mean return	0.026718897	0.0003686	0.01374509	0.01819963	0.02251048	0.00450295	0.01400454	0.01904441

Covariance matrix

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Amazon	0.006267435	0	0	0	0	0	0	0
Exxon	0	0.00190914	0	0	0	0	0	0
Microsoft	0	0	0.00397414	0	0	0	0	0
Alphabet(GOOG	0	0	0	0.00468862	0	0	0	0
Booking	0	0	0	0	0.00784432	0	0	0
AT&T	0	0	0	0	0	0.00142147	0	0
Boeing	0	0	0	0	0	0	0.00036641	0
CVS	0	0	0	0	0	0	0	0.00218239

P2: weight for diagonal matrix with target return 5% (Short constrains<100%)

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS			
Volatility	7.92	4.37	6.30	6.85	8.86	3.77	1.91	4.67			
Porfolio	x1	x2	x3	x4	x5	x6	x7	x8			
	0.49	-1.00	0.13	0.30	0.29	-0.93	1.00	0.72	1.00	=	1.00
Rate of return (%)	5.00	<=	5.00								
Volatility (%)	8.53	<=									

P3: data for back testing from 2016.01-2016.12

回报率	时间	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
	2016/1/1	-0.13152	-0.00128	-0.00703	-0.02099	-0.1647	0.047951	-0.16917	-0.01207
	2016/2/1	-0.05874	0.029544	-0.07642	-0.06081	0.188024	0.024681	-0.01623	0.006005
	2016/3/1	0.074423	0.042919	0.085495	0.067615	0.018772	0.060081	0.074124	0.067511
	2016/4/1	0.111094	0.057543	-0.09705	-0.06972	0.042437	-0.00894	0.061919	-0.03114
	2016/5/1	0.095817	0.007014	0.062763	0.06163	-0.05904	0.008501	-0.06417	-0.0403
	2016/6/1	-0.00992	0.053022	-0.03453	-0.05929	-0.01259	0.103704	0.029489	-0.00736
	2016/7/1	0.060353	-0.0511	0.10768	0.110808	0.082024	0.001851	0.029183	-0.03154
	2016/8/1	0.01364	-0.02035	0.013761	-0.00226	0.0488	-0.05567	-0.0315	0.007334
	2016/9/1	0.088603	0.001607	0.002436	0.01335	0.038652	-0.0066	0.01769	-0.04722
	2016/10/1	-0.05672	-0.04537	0.040278	0.009327	0.001862	-0.09407	0.081145	-0.05495
	2016/11/1	-0.04969	0.047768	0.005674	-0.03378	0.019977	0.050014	0.057081	-0.08573
	2016/12/1	0.024102	0.033906	0.031198	0.018178	-0.02502	0.045302	0.034006	0.026271

P4: data back testing results

For the same process, we run the model for different matrixs:

i. Diagonal matrix

Porfolio	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
	0.49	-1.00	0.13	0.30	0.29	-0.93	1.00	0.72

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Mean returr	0.013454	0.012935	0.011188	0.002838	0.014933	0.014734	0.008631	-0.01693

Covariance matrix

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Amazon	0.005789	0	0	0	0	0	0	0
Exxon	0	0.0014	0	0	0	0	0	0
Microsoft	0	0	0.003697	0	0	0	0	0
Alphabet(G	0	0	0	0.003164	0	0	0	0
Booking	0	0	0	0	0.007021	0	0	0
AT&T	0	0	0	0	0	0.002852	0	0
Boeing	0	0	0	0	0	0	0.005042	0
CVS	0	0	0	0	0	0	0	0.001662

Rate of retu	-1.69	<=	
Volatility (%)	10.99	<=	

ii. Sample matrix

Porfolio	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
	0.32	-0.81	-0.75	1.00	0.24	-1.00	1.00	1.00

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Mean return	0.01345374	0.01293505	0.01118832	0.00283771	0.01493346	0.01473399	0.00863079	-0.0169324

Covariance matrix

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Amazon	0.00530675	0.00032296	0.00088605	0.00159012	0.00134364	-0.0003468	0.00200353	0.00031399
Exxon	0.00032296	0.00128318	-0.0011307	-0.0010872	4.2786E-05	0.00126865	0.00046931	0.00032745
Microsoft	0.00088605	-0.0011307	0.00338874	0.00296339	-0.0009995	-0.0003266	0.00032988	0.00026059
Alphabet(G	0.00159012	-0.0010872	0.00296339	0.00290072	-0.0003904	-0.000454	0.00018535	0.00032267
Booking	0.00134364	4.2786E-05	-0.0009995	-0.0003904	0.00643596	-0.0008405	0.00272333	0.00017018
AT&T	-0.0003468	0.00126865	-0.0003266	-0.000454	-0.0008405	0.00261399	-0.000431	0.00064234
Boeing	0.00200353	0.00046931	0.00032988	0.00018535	0.00272333	-0.000431	0.00462157	-0.0001086
CVS	0.00031399	0.00032745	0.00026059	0.00032267	0.00017018	0.00064234	-0.0001086	0.0015237

Rate of retu	-3.12	<=	
Volatility (%)	12.72	<=	

iii. Two-block covariance matrix

Porfolio	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
	0.57	-0.82	-1.00	1.00	0.06	-0.80	1.00	1.00

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Mean return	0.01345374	0.01293505	0.01118832	0.00283771	0.01493346	0.01473399	0.00863079	-0.0169324

Covariance matrix

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Amazon	0.00530675	0.00032296	0.00088605	0.00159012	0.00134364	-0.0003468	0.00200353	0.00031399
Exxon	0.00032296	0.00128318	-0.0011307	-0.0010872	4.2786E-05	0.00126865	0.00046931	0.00032745
Microsoft	0.00088605	-0.0011307	0.00338874	0.00296339	-0.0009995	-0.0003266	0.00032988	0.00026059
Alphabet(G	0.00159012	-0.0010872	0.00296339	0.00290072	-0.0003904	-0.000454	0.00018535	0.00032267
Booking	0.00134364	4.2786E-05	-0.0009995	-0.0003904	0.00643596	-0.0008405	0.00272333	0.00017018
AT&T	-0.0003468	0.00126865	-0.0003266	-0.000454	-0.0008405	0.00261399	-0.000431	0.00064234
Boeing	0.00200353	0.00046931	0.00032988	0.00018535	0.00272333	-0.000431	0.00462157	-0.0001086
CVS	0.00031399	0.00032745	0.00026059	0.00032267	0.00017018	0.00064234	-0.0001086	0.0015237

Rate of retu	-3.06	<=	
Volatility (%)	12.06	<=	

iv. Shrinkage convariance matrix

Porfolio	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
	0.70	-1.00	0.20	0.31	0.28	-0.51	0.02	1.00

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Mean return	0.01345374	0.01293505	0.01118832	0.00283771	0.01493346	0.01473399	0.00863079	-0.0169324

Covariance matrix

	Amazon	Exxon	Microsoft	Alphabet(G	Booking	AT&T	Boeing	CVS
Amazon	0.00530675	0.00032296	0.00088605	0.00159012	0.00134364	-0.0003468	0.00200353	0.00031399
Exxon	0.00032296	0.00128318	-0.0011307	-0.0010872	4.2786E-05	0.00126865	0.00046931	0.00032745
Microsoft	0.00088605	-0.0011307	0.00338874	0.00296339	-0.0009995	-0.0003266	0.00032988	0.00026059
Alphabet(G	0.00159012	-0.0010872	0.00296339	0.00290072	-0.0003904	-0.000454	0.00018535	0.00032267
Booking	0.00134364	4.2786E-05	-0.0009995	-0.0003904	0.00643596	-0.0008405	0.00272333	0.00017018
AT&T	-0.0003468	0.00126865	-0.0003266	-0.000454	-0.0008405	0.00261399	-0.000431	0.00064234
Boeing	0.00200353	0.00046931	0.00032988	0.00018535	0.00272333	-0.000431	0.00462157	-0.0001086
CVS	0.00031399	0.00032745	0.00026059	0.00032267	0.00017018	0.00064234	-0.0001086	0.0015237

Rate of retu	-2.05	<=	
Volatility (%)	10.44	<=	

As mentioned above, the returns for four matrix are -1.69%, -3.12%, -3.06%, . The main problem is from the shortage constrains. Mainly, we short the stock with good performance and buy the stocks with least performance.
In conclusion, this project has flaws especially when the short constrains were implemented. Therefore, this models have some points to improve:

1. *pick larger size of stock pools*
2. *set short constrains with no negative or less than one*
3. *for counting return, we need to sort stocks with dividends and ones without, so we can better count the log-return.*
4. *Use python or C++ to improve the efficiency.*

4. Conclusion

The two projects are based on the same methodology, which is mainly implement portfolio theory and data, simulation to test the portfolio.
To carry out the Markowitz portfolio strategy, first we can use python or C++, VBA to improve the efficiency of allocating, and secondly we need to learn from papers to improve the accuracy of the models.

5. Appendix

For the first project, the python codes are as follows:

```
import pandas_datareader.data as web
import datetime
import fix_yahoo_finance as yf
yf.pdr_override()


start=datetime.datetime(2017, 1, 1)
end=datetime.datetime.today()
apple=web.get_data_yahoo('AAPL',start,end)
microsoft=web.get_data_yahoo('MSFT',start,end)
google=web.get_data_yahoo('GOOG',start,end)
facebook=web.get_data_yahoo('FB',start,end)
netflix=web.get_data_yahoo('NFLX',start,end)
import matplotlib
import matplotlib.pyplot as plt
import pandas as pd
```

```
stocks = pd.DataFrame({"AAPL":apple["Adj Close"],"MSFT": microsoft["Adj Close"], "GOOG": google["Adj Close"],"FB": facebook["Adj Close"],"NFLX": netflix["Adj Close"]})

stock_return = stocks.apply(lambda x: x / x[0])

print stock_return.head()

stock_return.plot(grid =True,title="stock price graph")

import numpy as np

import statsmodels.api as sm #统计运算

import scipy.stats as scs #科学计算


stock_change = stocks.apply(lambda x: np.log(x) - np.log(x.shift(1))) # shift moves dates back by 1.

stock_change.plot(grid = True,title="Log-return of stocks").axhline(y = 0, color = "black", lw = 2)

print stock_change.mean()*252

print stock_change.cov()*252


#5.monte carlo simulation

port_returns = []

port_variance = []

for p in range(4000):

  weights = np.random.random(5)

  weights /=np.sum(weights)

  port_returns.append(np.sum(stock_change.mean()*252*weights))

  port_variance.append(np.sqrt(np.dot(weights.T, np.dot(stock_change.cov()*252, weights))))

port_returns = np.array(port_returns)

port_variance = np.array(port_variance)


risk_free = 0.04


plt.figure(figsize = (8,4))
```

```
plt.scatter(port_variance, port_returns, c=(port_returns-
risk_free)/port_variance, marker = 'o')
plt.grid(True)
plt.title("Portfolio Simulation")
plt.xlabel('excepted volatility')
plt.ylabel('expected return')
plt.colorbar(label = 'Sharpe ratio')


#6.挑选 Sharpe ratio
def pick(weights):
   weights =np.array(weights)
   port_returns = np.sum(stock_change.mean()*weights)*252
   port_sd =
np.sqrt(np.dot(weights.T,np.dot(stock_change.cov()*252,weights)))
   return np.array([port_returns,port_sd,port_returns/port_sd])
#optimization problem
#first, max sharpe
import scipy.optimize as sco
def max_sharpe(weights):
   return -pick(weights)[2]
cons = ({'type':'eq', 'fun':lambda x: np.sum(x)-1})
bnds = tuple((0,1) for x in range(5))
opts = sco.minimize(max_sharpe, 5*[1./5,], method = 'SLSQP', bounds = bnds,
constraints = cons)
#print opts
print opts['x'].round(5)
print pick(opts['x']).round(5)


#min variance
def min_variance(weights):
   return pick(weights)[1]


optv = sco.minimize(min_variance, 5*[1./5,],method = 'SLSQP', bounds = bnds,
constraints = cons)
```

```
print optv['x'].round(5)
print pick(optv['x']).round(5)
```