ARPM PORTFOLIO CONSTRUCTION AND OPTIMIZATON

Author: Guo Yu, Rutgers University

1 Introduction:

For most of the mean-variance portfolio construction, only the historical distributions are utilized. And through the Monte-Carlo simulation for holdings, minimal variance and sharp-ratio holdings can be allocated. However, the past distribution cannot say everything about the future, a thoughtful and comprehensive future P&L distribution or ex-ante P&L distribution are needed. We need to use "future" to speak about future, even though the "future" is acquired from the past. This paper walks through all the step in the ARPM portfolio construction, and optimization, including my personal analysis on missing data and classifier comparison.

2 Data analysis:

The assets utilized in this paper are equities, foreign-exchange, and bond. In the following section this paper will go thorough missing data, risk driver, invariants test, normality test and flexible probability.

2.a Missing data

To start with any kind of portfolio construction, covariance matrix are need, but due to different trading period different assets have different length, therefore a method is need for filling all the missing data. In this paper, it utilize the cubic spline to interpret all the missing point. And to comparing with using mean to interpret the missing data. Paper use Score = $\sum\left|P_{expected} - P_{compited}\right|/P_{expected}$ and take Score = 10*MAX(10 - score,0) for each assets and calculate the mean. Here is the result

	Mean	Cubic spline
Mean socre	40.23	74.24

Clearly the cubic spline is the better method, here is the correlation heatmap after filling all the missing data

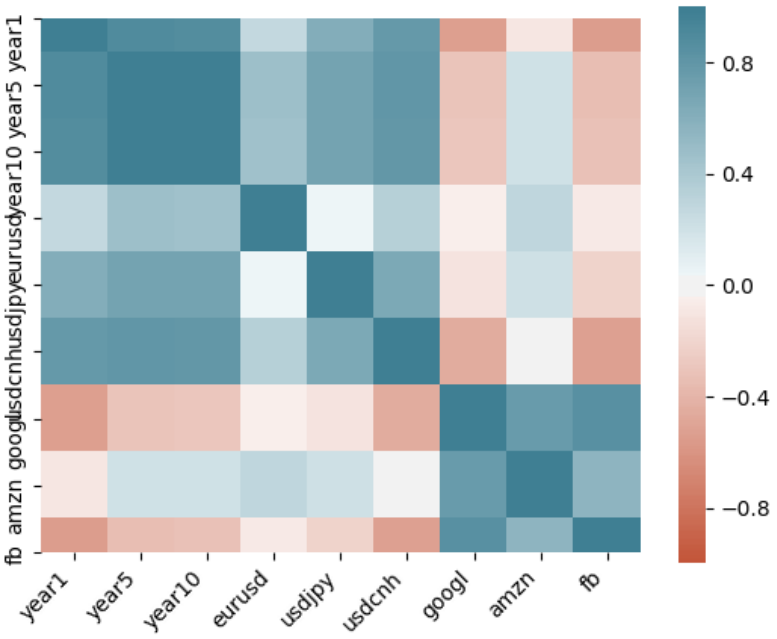
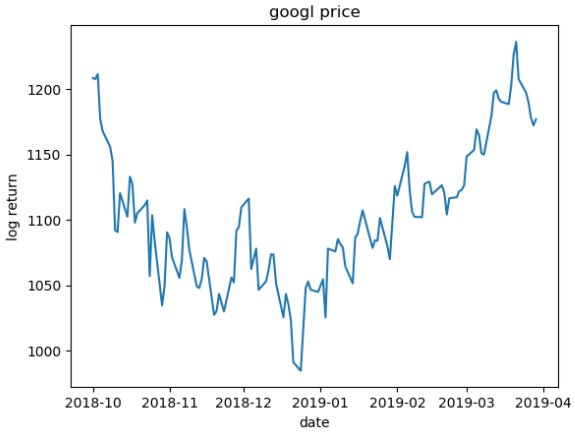
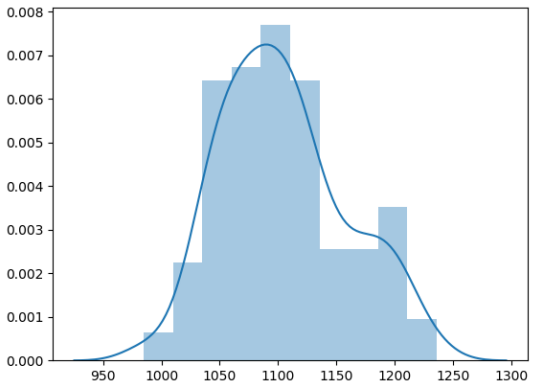


Fig: Assets correlation heatmap

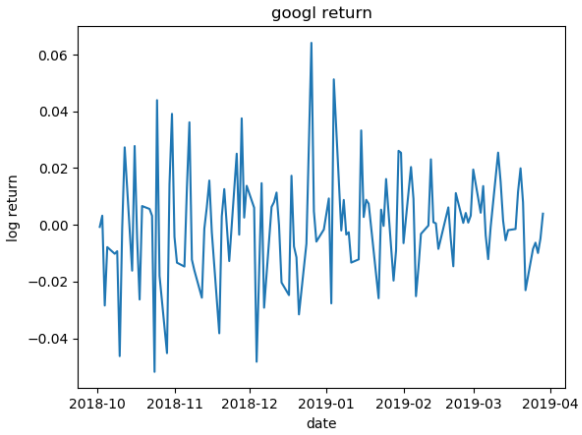
2.b risk drivers

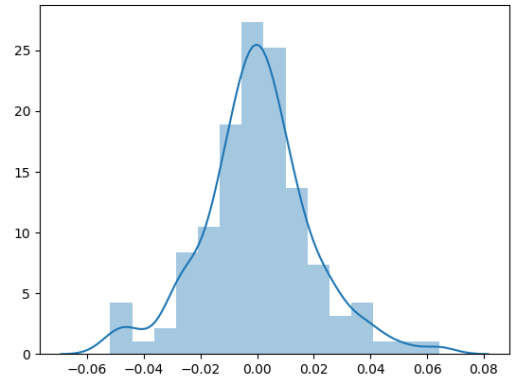
For the equity this paper has 5 stock, which are AAPL, AMZN, FB, GOOGL and NFLX. This paper chooses time period 10/01/2018 to 3/31/2019. And collected data from Bloomberg. Here is an example for GOOGL stock price and its distribution.

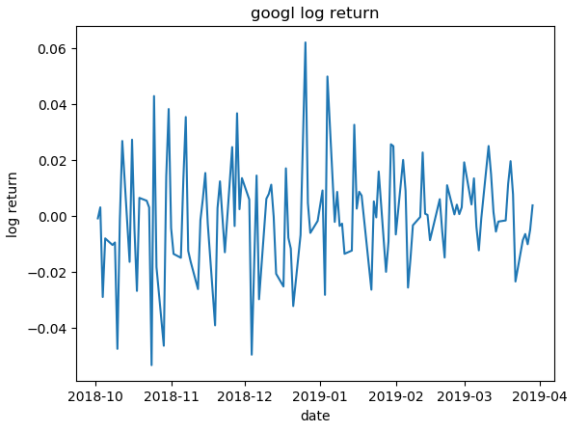


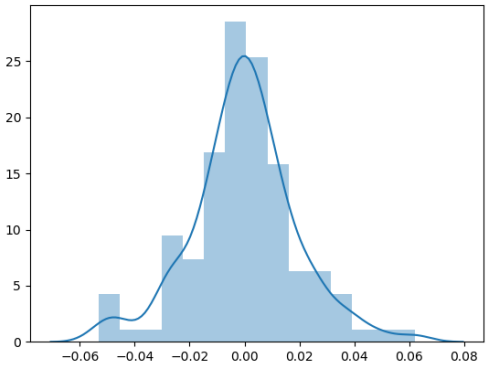


[1]Over the short horizon the risk driver for the equity we use the stock return, while over the long horizon we use the log-return as the risk driver. Typically, short horizon is within a day, and long horizon is otherwise. The reason we did not use the price as the risk driver is that we wish our risk driver are random walk, and the stock price did not perform like random walk. Here are example of stock return and log returns.









As we can tell from the distribution the stock return and log return are more random walk alike. Since our data size is relatively small, the return and the log return does not make a huge difference, but we still can tell they are different from their distribution.

For the FX, I choose three exchange rate, USDJPY, USDCNH and ERUUSD. For foreign exchange rate, since it's already an rate we can directly take it as an risk driver. Under the same period of time, 10/01/2018 to 3/31/2019, I collect data from Bloomberg. Here is an example of USDJPY and its opposite rate JPYUSD.

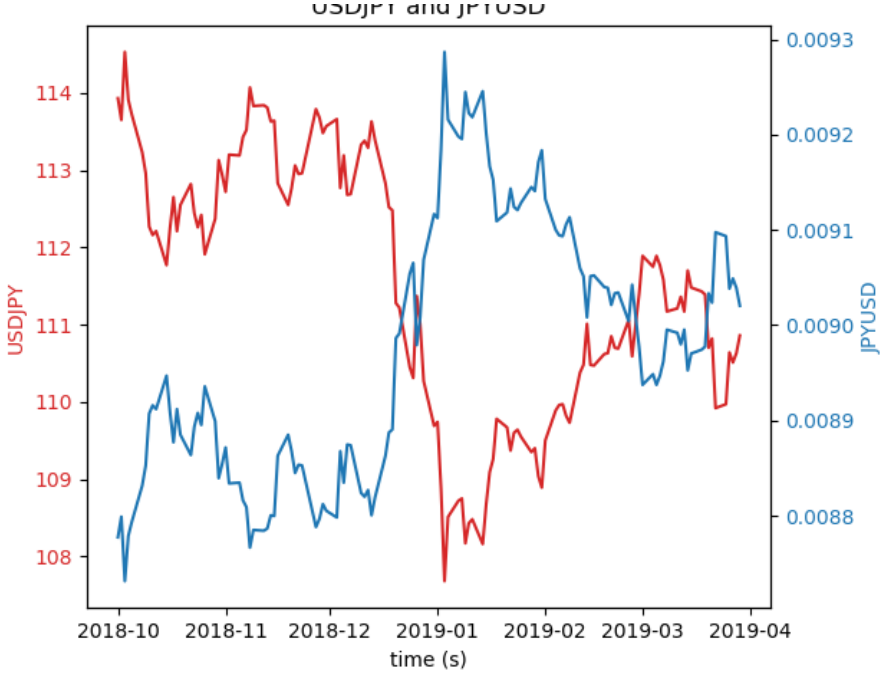


Fig: usdjpy and jpyusd

There is one problem here is that the graph is not really symmetric, as the range on the different exchange rate various. And because intuitively two opposite exchange rates should be symmetric. So I take the Ln exchange rate as the risk driver and here is the result.

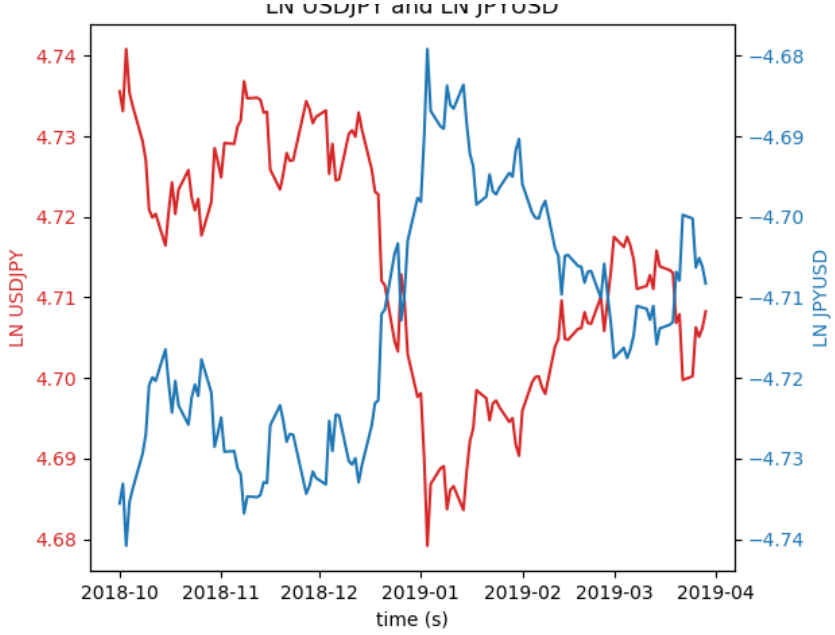


Fig:ln usdjpy and ln jpyusd

On the Ln exchange rate we observe that both exchange rate share the same range and truly symmetrical.

For the fixed income, I choose 6month, 1, 5, 10 and 30 year treasury rate. And I collect all the data from the FRED(Federal Reserve Economic Data)

And here is example of 1-year treasury rate yield.[1]

On the FRED, all the data is already the rolling data, therefore we do not need to perform the rolling value step.

2.c Invariant test[2]

Risk drivers are the variable that we use to determine the profit and loss. [1]The quest for invariants aims at identifying and fitting the most suitable next step model and extracts the realizations ϵ_t of the invariants ε_t from the realized time series x_t of the risk drivers X_t.

The mean idea of invariant is that with information available up to time t, invariants are totally responsible for the randomness of the next step risk driver.

There are four groups of phenomena observed empirically in the risk driver [2] efficiency, mean-reversion, long memory, vol. clustering. Market efficiency, or unpredictability of the risk driver, which is model by the random walk, where the invariants are the increments of the risk drivers Simple example of invariant is: in equity the risk driver is log value $X_t = lnV_t^{stock}$, and the invariant is the would be ΔlnV_t^{stock}(random walk).

Here are the all four phenomena

1. random walk

$$\varepsilon_t = X_{t+1} - X_t$$

2. mean reversion

$$\varepsilon_t = X_{t+1} - \tilde{b}X_t \qquad \text{AR(1)}$$

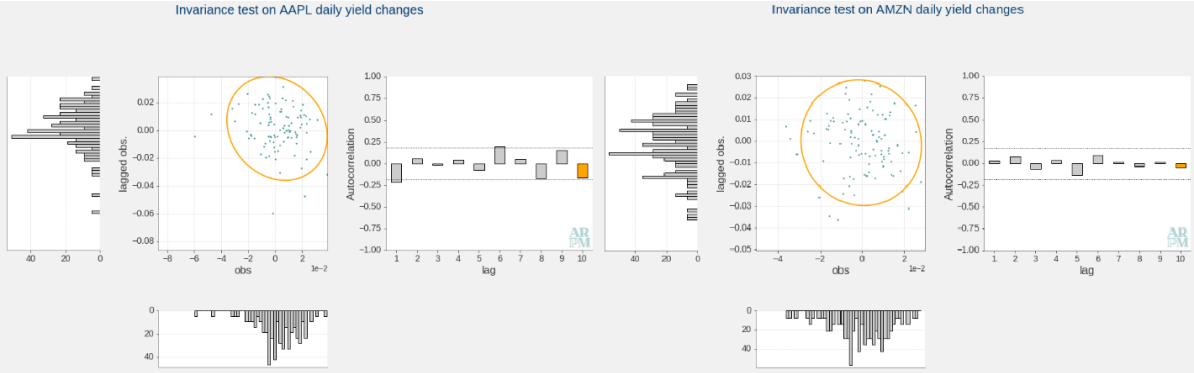
3. volatility clustering

$$\varepsilon_t = \frac{X_{t+1} - X_t - \mu}{\delta_{t+1}} \qquad \text{GARCH(1,1)}$$

4. long memory

Long memory is for the high-frequency, which is not applicable for this paper.

For the equity, we first test the random walk phenomenon, where the invariants are the increments of the risk drivers. For the equity our identified risk driver is log return, then the invariants are $\ln \Delta V_t^{stock}$, and we wish it has the random walk character which is uncorrelation between each lag invariants. We then perform ellipsoid test for five equities, under the time period of 2018/10/01-2019/03/31



We observed that not all equities completely pass the ellipsoid test, two equities that we can safely say that pass the ellipsoid test is AMZN and FB, while other equities are on the edge of passing or not passing, meaning their lag are slightly correlated.

Then I perform Kolmogoroov-Simronov test, to see how each equities invariants performed.

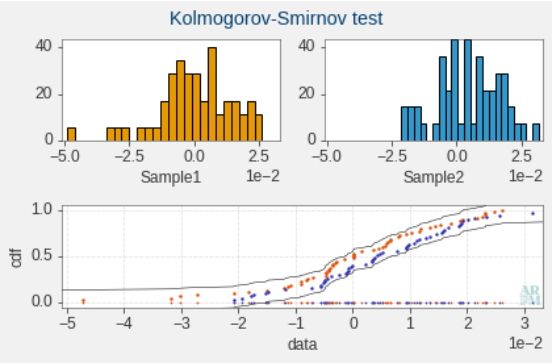


Figure a AAPL

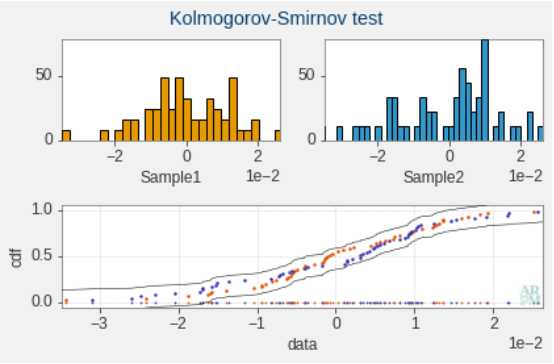
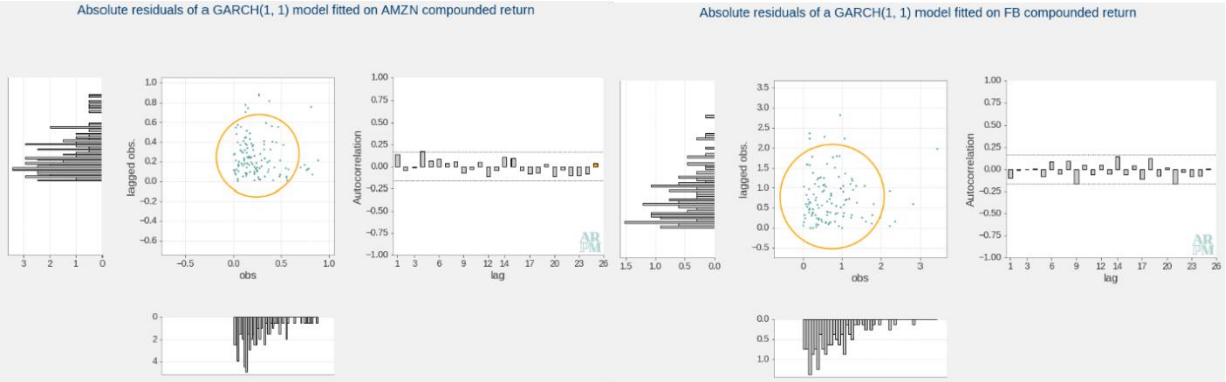


Figure AMZN

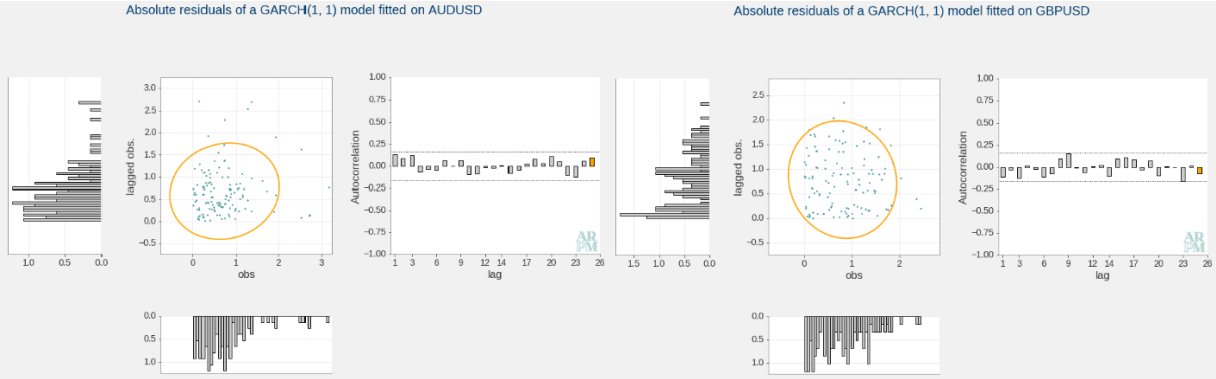
Now the Kolmogoroov-Simronov test gives all positive result for the invariants $\ln \Delta V_t^{stock}$. Kolmogoroov-Simronov test is a two-sided hypothesis test, and our result is that we do not reject null hypothesis, which is two sample come from the same distribution. Therefore, all invariants should be qualified.

However, our ellipsoid test disagrees. So we need further testing for the invariants. We perform autocorrelation ellipsoid test of compounded and absolute compounded return. We use GARCH(1,1) to model the fit the invariants. And we wish to observe the phenomenon of volatility clustering



Here we do see the volatility clustering phenomenon, and we can say that E_t is the invariant.

We observe that the FX also have volatility clustering, thus we perform the GARCH(1,1) to fit invariant, here is the result.



And we do see the phenomenon of volatility clustering.

For the long-term Bond we use AR (1) model to fit the invariant. The risk driver is the log-yield.

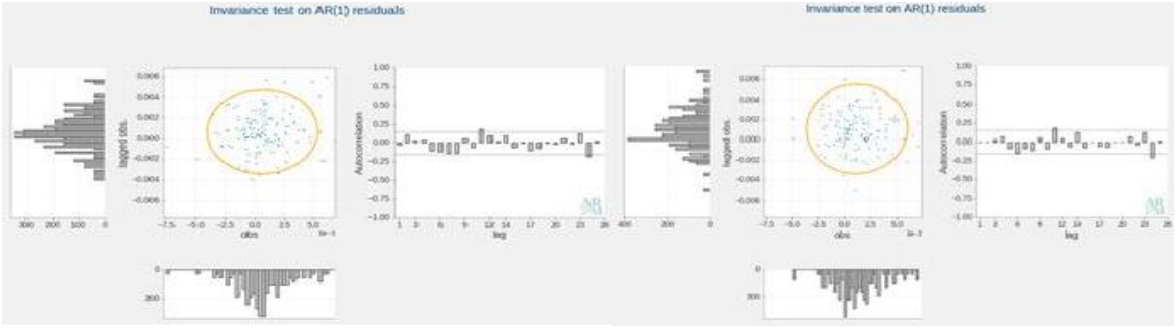


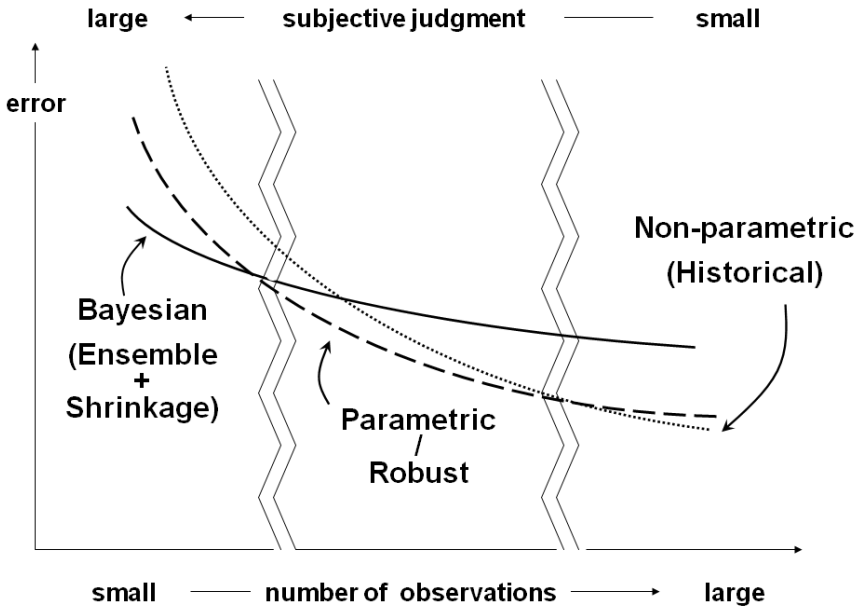
Figure 3 1-year *Figure 4 5-year*

2.d estimation and projection[3][4]

Under three different scenarios, we would pick one approach out of three, which are historical, analytical, and copula marginal. First scenario is that analytical results are required for the invariants. for instance, the GARCH residuals we used as invariants for the bonds. Under this circumstance, use need to use parametric/robust/shrinkage estimate (analytical approach). Second scenario is that no analytical results are needed, but the time serious of the risk driver are long comparing with the investment horizon. Under second circumstance, we use non-parametric estimate (historical approach). Third scenario is that no analytical results are needed, but the time serious of the risk driver are not long comparing with the investment horizon. Under the third circumstance, we use non-parametric/parametric/robust/shrinkage estimate (copula marginal approach).

After choosing the approach, we then determine the most suitable estimate technique. There are three major approaches for estimation: non-parametric, parametric/robust, and

Bayesian/shrinkage. ARPM provides a great visual verification of choosing most suitable approaches [1]



Roughly speaking, when we have abundant number of observation, we would choose non-parametric approach supported by law of large number. When the observations are not so abundant, we employ personal judgment to the distribution of invariants, and we achieve that by parametric approach. Lastly if observations are small, we rely on subjective judgement and robustness, Bayesian estimation.

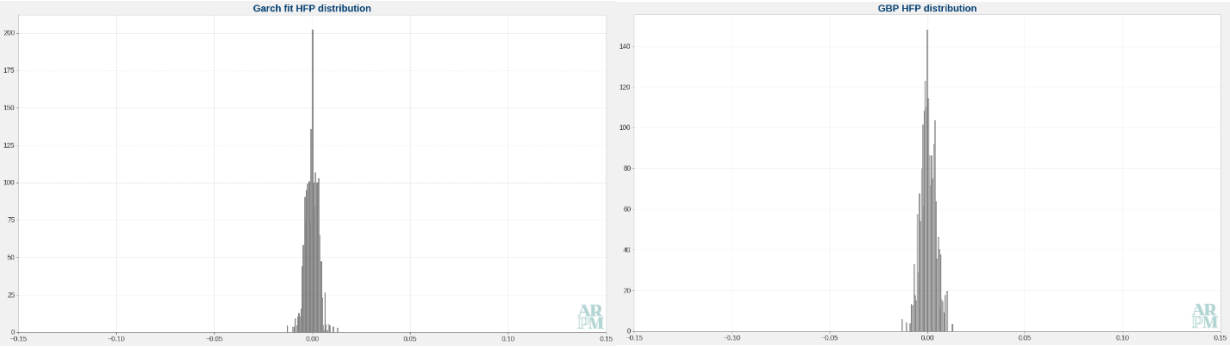
On this paper, it going to focusing flexible probability, the reasons are shown in the following part. Flexible probabilities allow us to give different relative weights to different period. There are exponential decay, crisp, and gaussian kernel state conditioning.

For the exponential decay we have equation$p_t|T_{HL} \equiv pe^{-\frac{\ln(2)}{T_{HL}}|t^*-t|}$

Where T_{HL} [2] is called the half life and represents the time required for decaying probabilities to fall to one half of the highest value in t^* end point. I present the result of exponential decay results for equities below.

For FX in my previous research and my time horizon, only the residual that fitted after GARCH(1,1) shows the volatility clustering phenomenon[1]. Therefore, let's investigate on the estimation of residual fitted by GARCH(1,1).

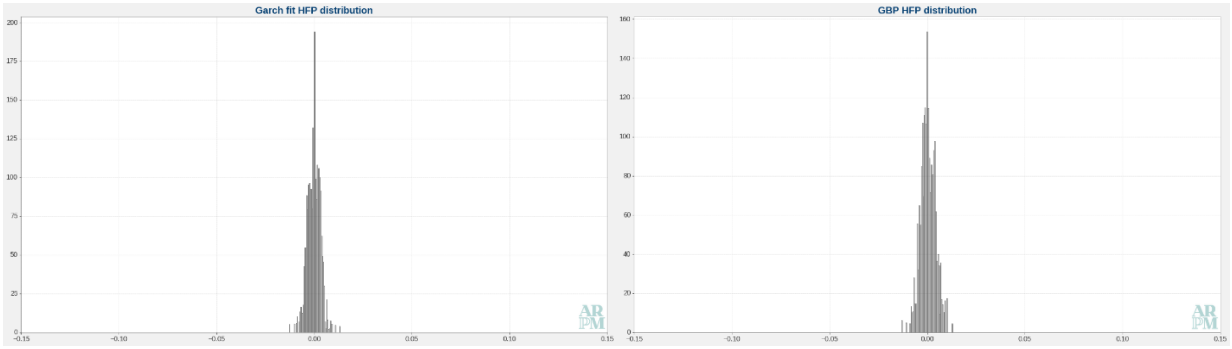
(a)Exponential decay



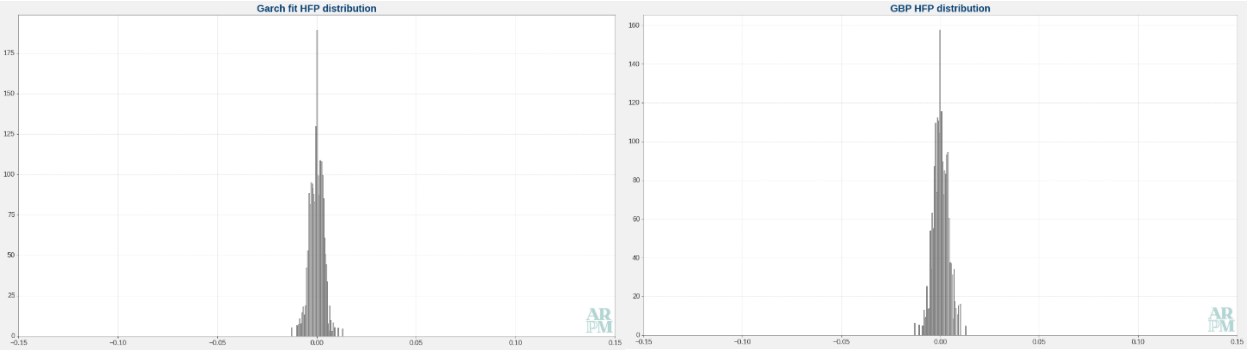
Here I showed when T = 100, the distribution of residual and the original log exchange rate GBPUSD under flexible probability of exponential decay. Figure on the left shows the residual distribution, figure on the right shows the log exchange rate. Interestingly, the GARCH fitted residual, which show the phenomenon of volatility clustering, does not give a more normal-like distribution.

Now, consider different half-life, and see if they provide a better residual distribution.

For T = 150



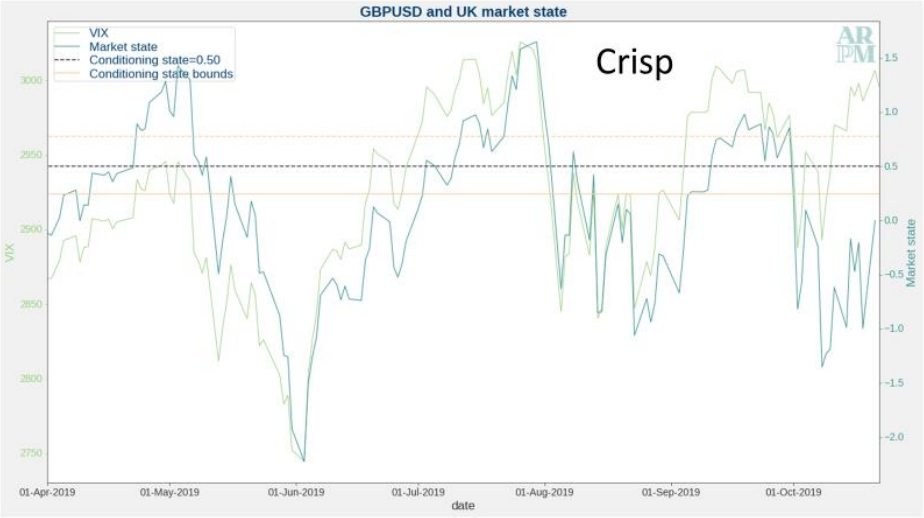
For T = 200

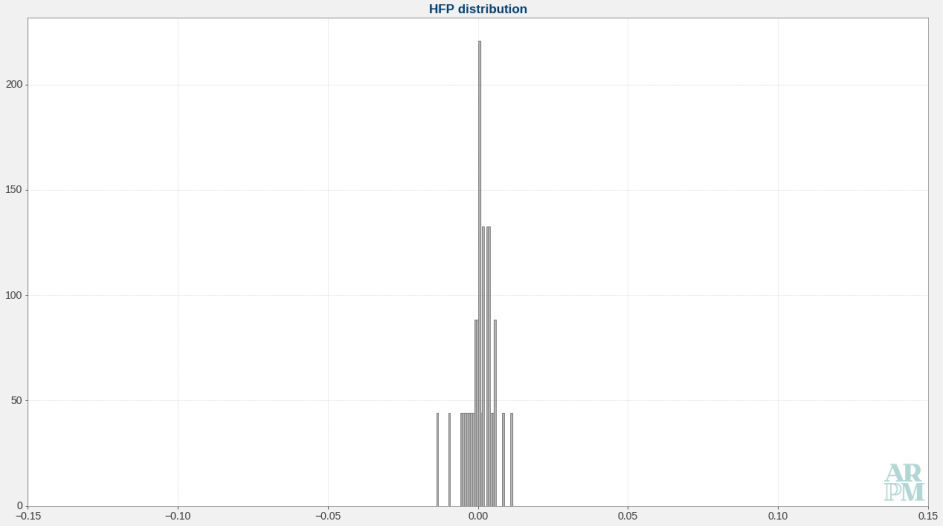


The result is that for longer half life, the residual distribution surely improves, however, it does not make it significantly better than the log exchange rate distribution.

For the crisp, we need different target for different exchange rate. For the GBPUSD, I choose the FTSE 100, which is British version of SP500, similarly ASX200 for AUDUSD, CSI300 for CNHUD, NIKKEI 225 for JPYUSD, and STOXX 600 for EURUSD. Here I present the result of GBPUSD's residual Crisp condition.

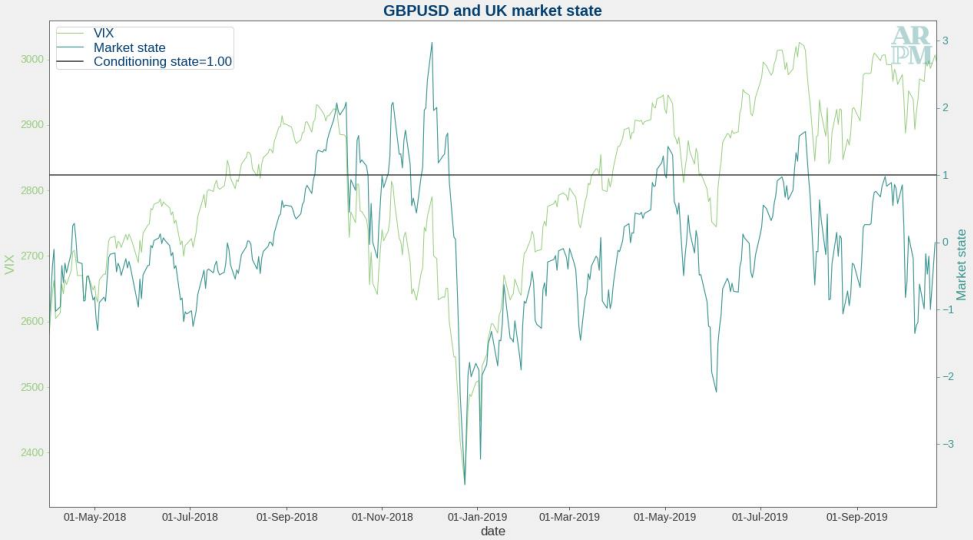
(b)crisp

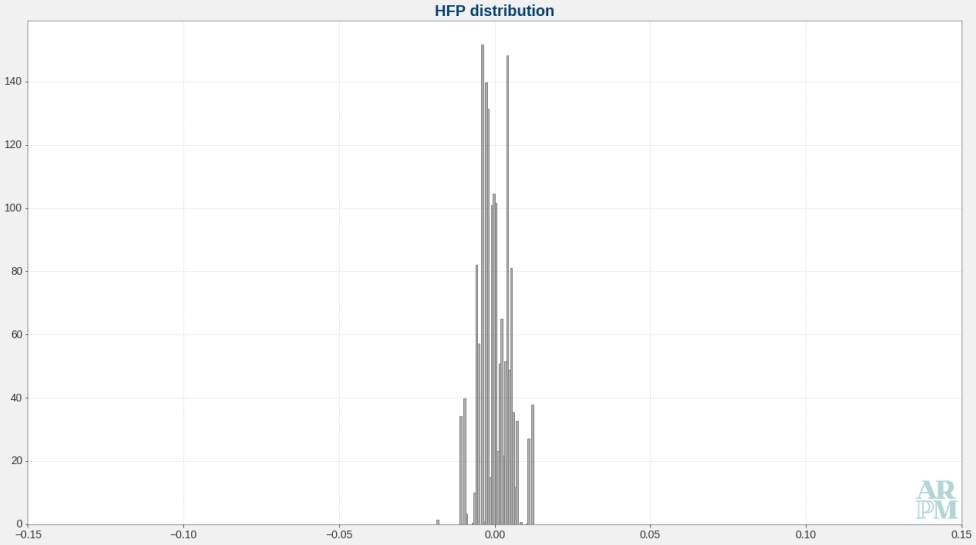




(c) Gaussian kernel state conditioning

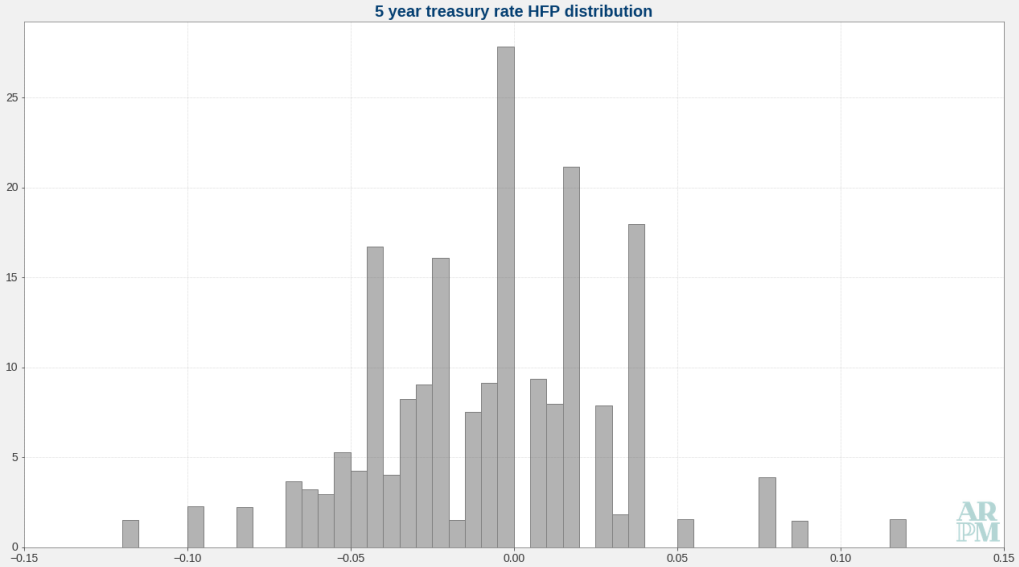
Using similar idea and same target for Gaussian kernel state conditioning. Here I show result of GBPUSD.





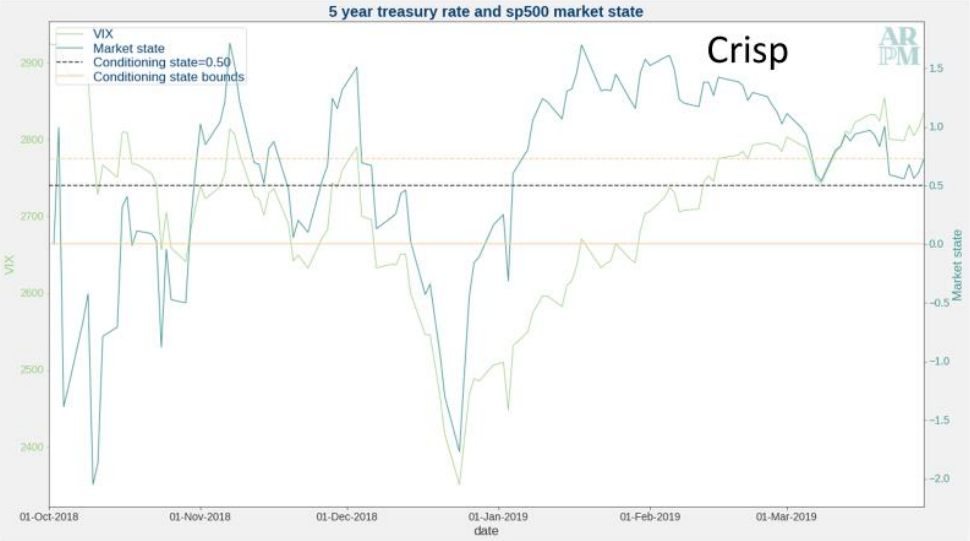
For bond, the AR(1), shows the volatility clustering. Therefore we do estimation after fitting AR(1). Here I presents results of 5-year treasury rate.

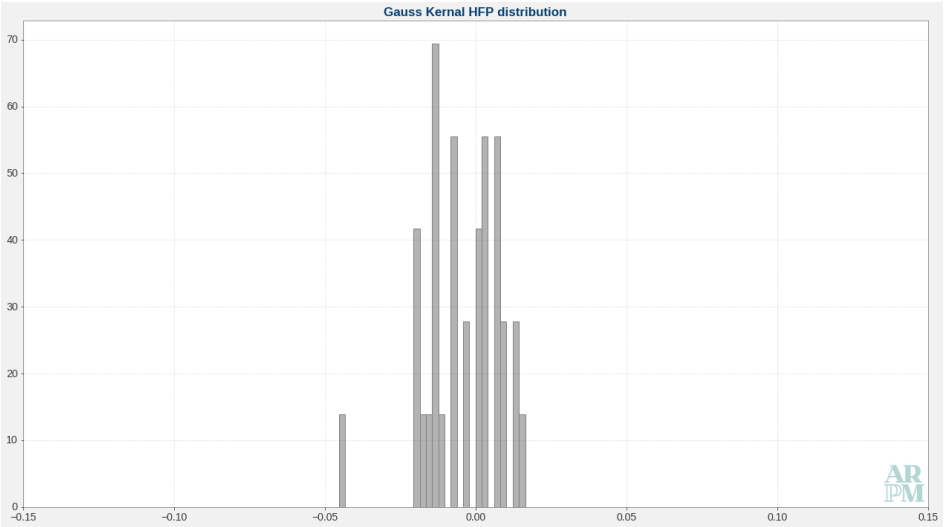
[a]Exponential decay.



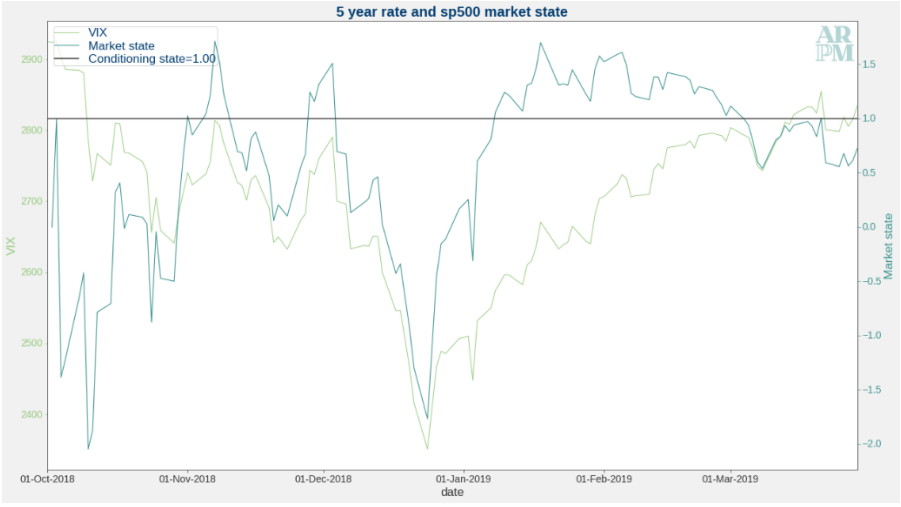
(b)Crisp

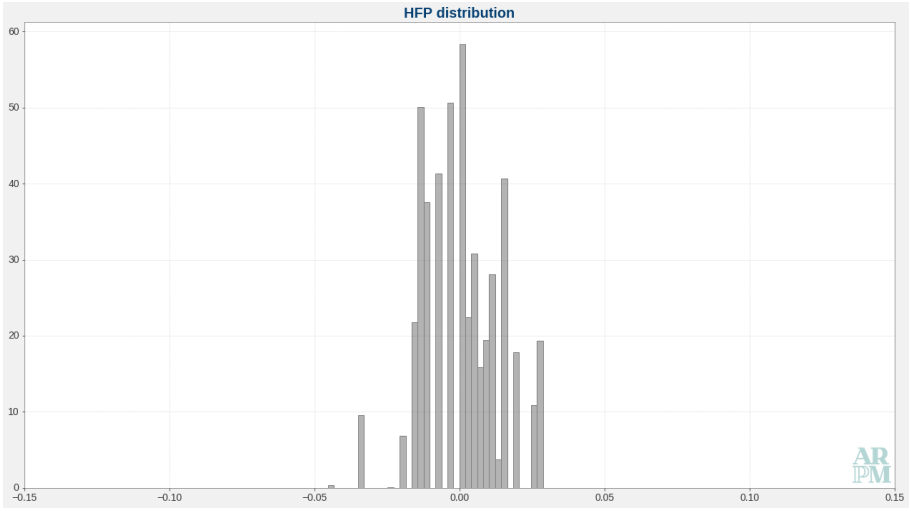
For the target, I choose SP500.





(c) Gaussian kernel state conditioning





From the result we can clearly tell that the exponential decay gives the best results. However, we cannot say that state conditioning is worse, because if we choose the target and target level wisely, we might have a completely different conclusion. For this paper, exponential decay is been relied on for simplicity.

3.Portfolio projection and attribution

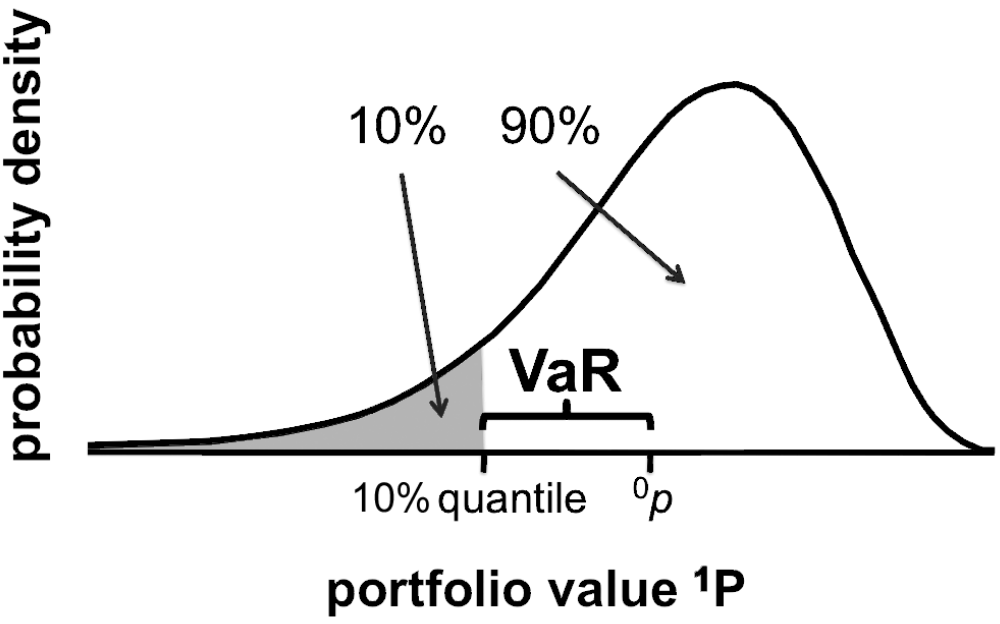
With all the estimation of the risk driver, one could easily acquire the projection, which is the P&L distribution from $t_{now} \rightarrow t_{hor}$ via historical bootstrapping[A]. Then, portfolio's P&L can be determined by the aggregation of all risk driver if allocation holdings h or weights are given. Therefore, started with static holdings, this paper will introduce few critical portfolio's attribution, which will latter be applied on portfolio optimization.

3.a mean and variance

First, mean and variance, known as the expectation $\mathbb{E}\{Y\} \equiv \int y f_Y(y) dy$ and risk$-\mathbb{V}\{Y\} \equiv -\int (y - \mathbb{E}\{y\})^2 f_Y(y) df$, are easiest to compute, but commonly used as they are the basics for the mean-variance portfolio and efficient frontier constructions. They are powerful, but one should have a clear understanding that none of them should heavily depended.

3.b VaR

Secondly, the VaR(value at risk), which measure the losses corresponds to a rather high threshold, the following graph[5] will help us better understood, the mean and how to compute it.



VaR represent from the worst scenario to the no-profit scenario, the biggest exposure or losses a portfolio might face. The variance measures risk from the perspective of how volatile a portfolio P&L distribution is while VaR measures risk from the actual losses perspective. Therefore, one can treat VaR as a satisfaction measure, and apply it as a constrain for portfolio optimization or in another words, weights selection.

3.c shortfall

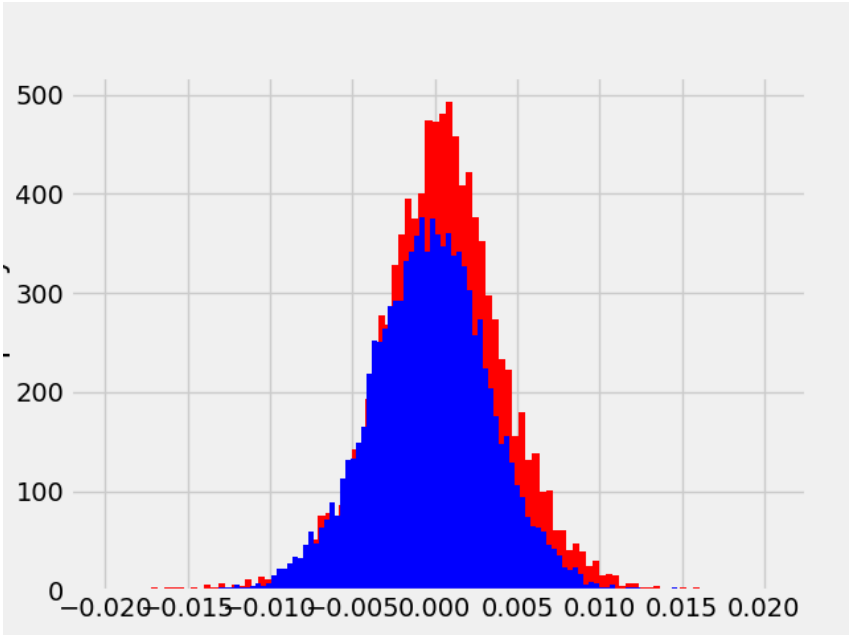
Thirdly, portfolio can also treat expected shortfall as satisfaction. Expected shortfall or conditional value at risk the is average losses above the value at risk, or in another word, the average losses among the worst-case scenario. Its calculation is $ES_c(h(.)) \equiv \mathbb{E}\{-\pi_{h(.)} | -\pi_{h(.)} \geq VaR_c(h(.))\}$. In the empirical test, this paper choose $c \equiv 99\%$, and applied it for the final selection of portfolio weights. One problems with the shortfall is that its calculation, required calculator to have full understanding of the distribution. Here is the formula this paper used to calculate the shortfall.$\mathbb{E}\{Y_h\} + sd\{Y_h\}[\Phi^{-1}(1-c) + \frac{1}{6}\left(\left(\Phi^{-1}(1-c)\right)^2 - 1\right) sk\{Y_h\}]$ [6]where $sk\{Y_h\}$ is the skewness of the distribution, and for Φ^{-1} is the probit function, which we really only have two choice here. First use normal distribution, second use t-distribution. From [B]appendix result from data analysis only few assets pass normality test, which tell us the possibility that portfolio P&L distribution is normal is very low. Thus, we must use t-distribution, with degree of freedom equal number of assets.

4.Portfolio construction and optimization

Quasi-optimization, quasi-optimization is a two-step portfolio optimization, firstly, with all the ex-ante P&L distribution, constructs a set of portfolio holdings at certain variance level. Computed all holdings portfolio ex-ante satisfaction, then rank them by the satisfaction. Finally choosing the weights of highest satisfaction. Important things remember that all the distribution we utilize are ex-ante distribution. Therefore, comparing with historical distribution, this quasi-optimization would provide a more solid and dynamic investment combination. For instance, one could either choose expected shortfall and sharp ratio as satisfaction. The expected shortfall may have smaller return but it upholds better chance to have a smaller draw back. The sharp ratio one may have better return but also suffer from high probability to encounter a greater draw back.

5.Empirical test

For the empirical test I choose the same data range as my data analysis. Filling the missing data, then start calculating the risk drivers, computing estimation, and projected via historical bootstrapping. Then start this test core part with investigating on how the weight influence the ex-ante distribution. Here is the graph of the exante P&L distribution of two potfolio with different holdings, blue one is drived from minimual variance, the red one is derived from performing quasi-optimal with satisfacttion is expected shortfall.



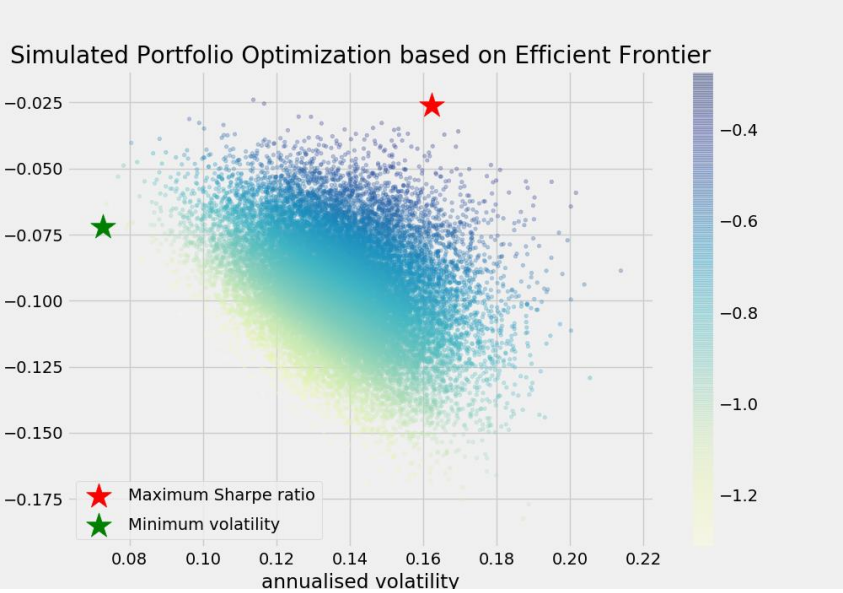
Blue:mini-variance,Red:smallest short fall

Fig: P&L distribution

From the distribution we can esasily tell that the quasi-portfolio which is using the expected shortfall have the better result, as the center is slightly shfited to the left. Here are the table of statistic of the four quasi-portfolio with different satisfaction.

	Minimal variance	sharp ratio	Expected shortfall	static
Annual return	-0.071	-0.0258	0.013	-0.081
Annual volatility	0.07	0.16	0.11	0.14

Then, here is the graph of the efficient frontier.[7]



From this graph we can tell that the P&L distribution are not so friendly for a long only strategy. But let's see

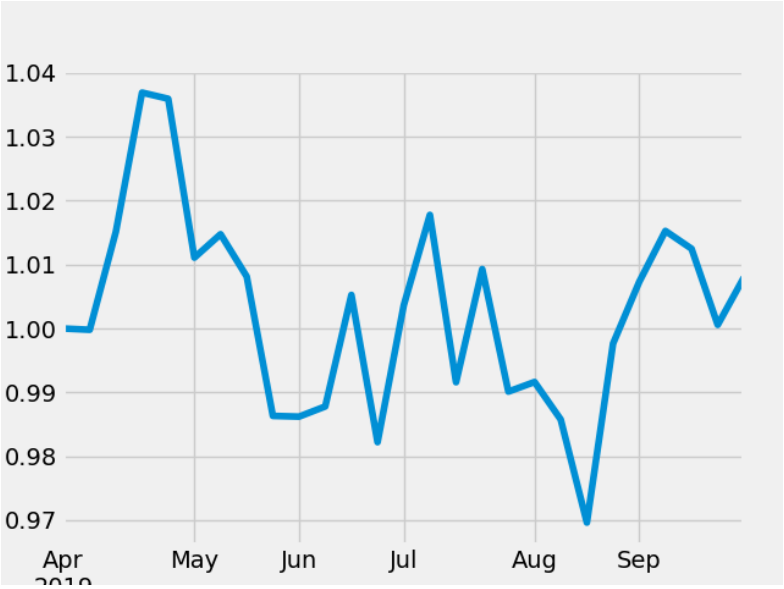


Figure. Sat:Expected short fall return

Fig: static holding return

Fig:sat:max sharp-ratio return

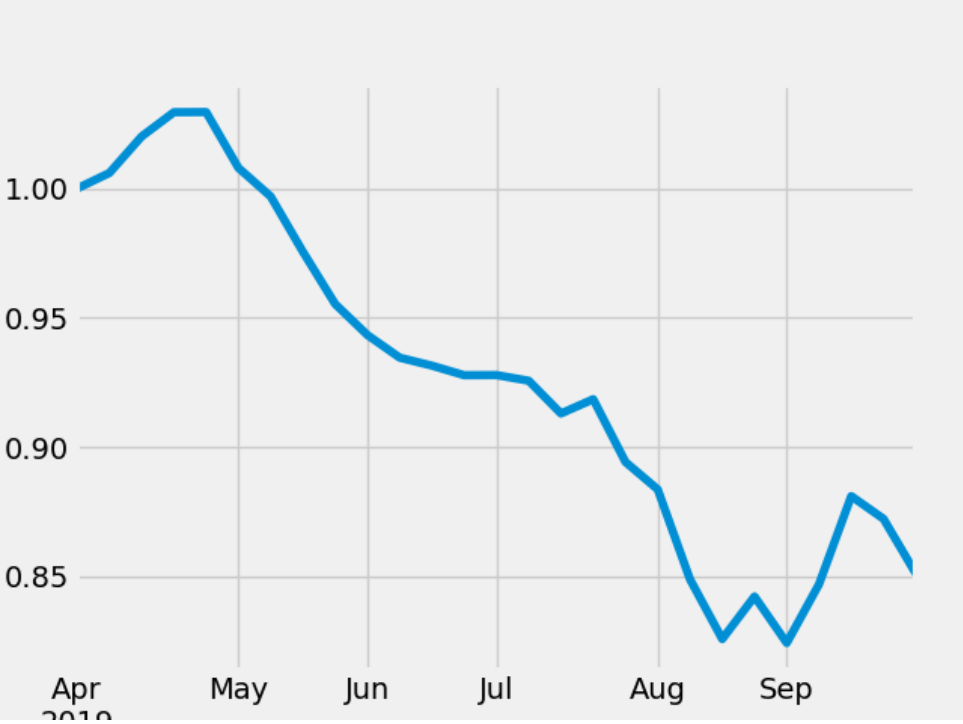


Fig: sat: minimum variance return

6 Conclusion:

Through the empirical test, it proved what this paper had been demonstrating, that using ex-ante distribution to allocate the portfolio will provide a better result, even in a long-unfriendly market. The quasi-optimzation had shown that under this market situation and comparing with other satisfaction expected shortfall is the better one. However, both ex-ante projection and expected shortfall can not be the optimal solution, as the return are not satisfiable. We could improve the portfolio further, but it required more data, if we can have factor data, then SVM[e] classifiers could boost the return drastically.

7 Referrence

[1]ARPM https://www.arpm.co/lab/risk-drivers-identification.html

[2]ARPMhttps://www.arpm.co/lab/quest-for-invariance.html

[3]ARPMhttps://www.arpm.co/lab/lbl-estimation.html

[4]ARPMhttps://www.arpm.co/lab/lbl-projection.html

[5]https://www.glynholton.com/blog/risk-measurement/var_measure/ Glyn Holton | Oct 11, 2018

[6]https://medium.com/quaintitative/expected-shortfall-in-python-d049914e1e85 shorfall

[7] Efficient Frontier Portfolio Optimization in Python Ricky Kim https://towardsdatascience.com/efficient-frontier-portfolio-optimisation-in-python-e7844051e7f

[8]bootstrapping projection https://www.arpm.co/lab/s_projection_stock_bootstrap.html

[9]normality: test A Gentle Introduction to Normality Tests in Python: **Jason Brownlee** https://machinelearningmastery.com/a-gentle-introduction-to-normality-tests-in-python/ August 8, 2019

[10] https://www.arpm.co/lab/EBEvalNumericalExample.html#x1470-3374000S.7.2

[11] https://www.arpm.co/lab/s_projection_stock_bootstrap.html

8 Appendix

[A]Normality test[9]

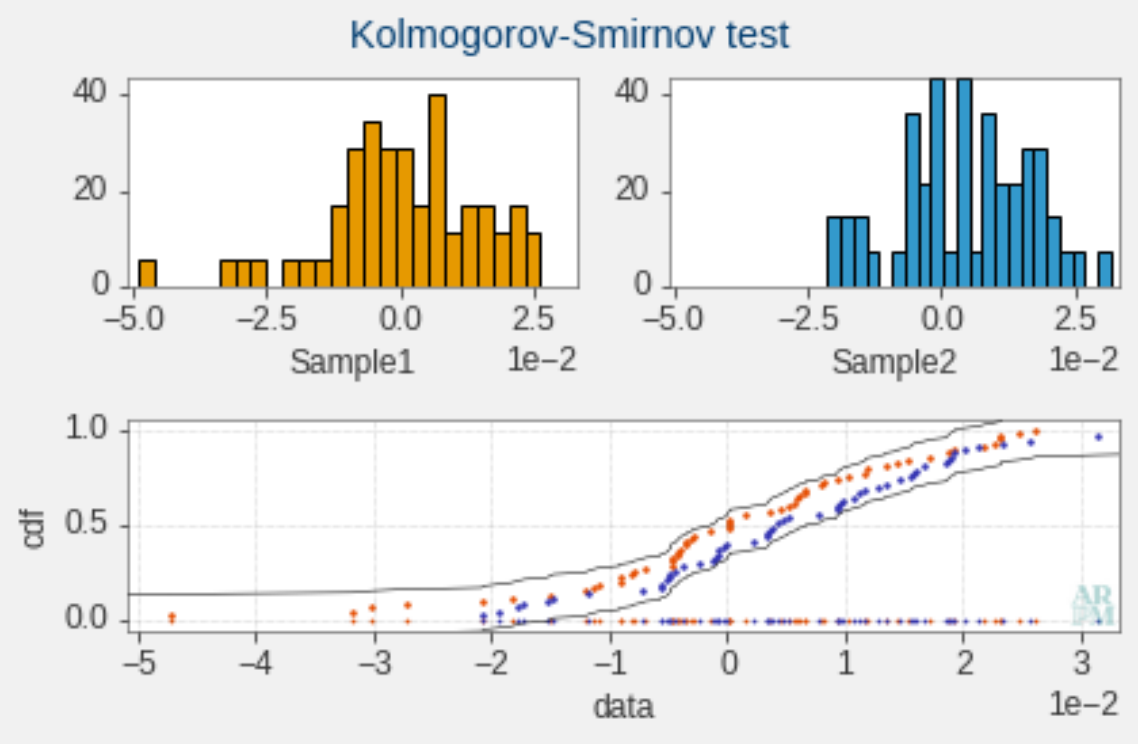
There are many statistical tests that we can apply to test whether a sample looks like normal distribution. In this test, I use three commonly used normality test. Shapiro-Wilk[a], D'Agostino's K^2[b], and Anderson-Darling test. For the Anderson-Darling, the test is a modified version of a more sophisticated nonparametric goodness-of-fit statistical test called the Kolmogorov-Smirnov test. Since we already did K-S test in week nine, I would use K-S test directly. To begin with, our null hypothesis H_0 is that sample is drawn from a normal distribution, alpha is 5%.

Here are Shapiro-Wilk and D'Agostino's K^2 test results of equities.

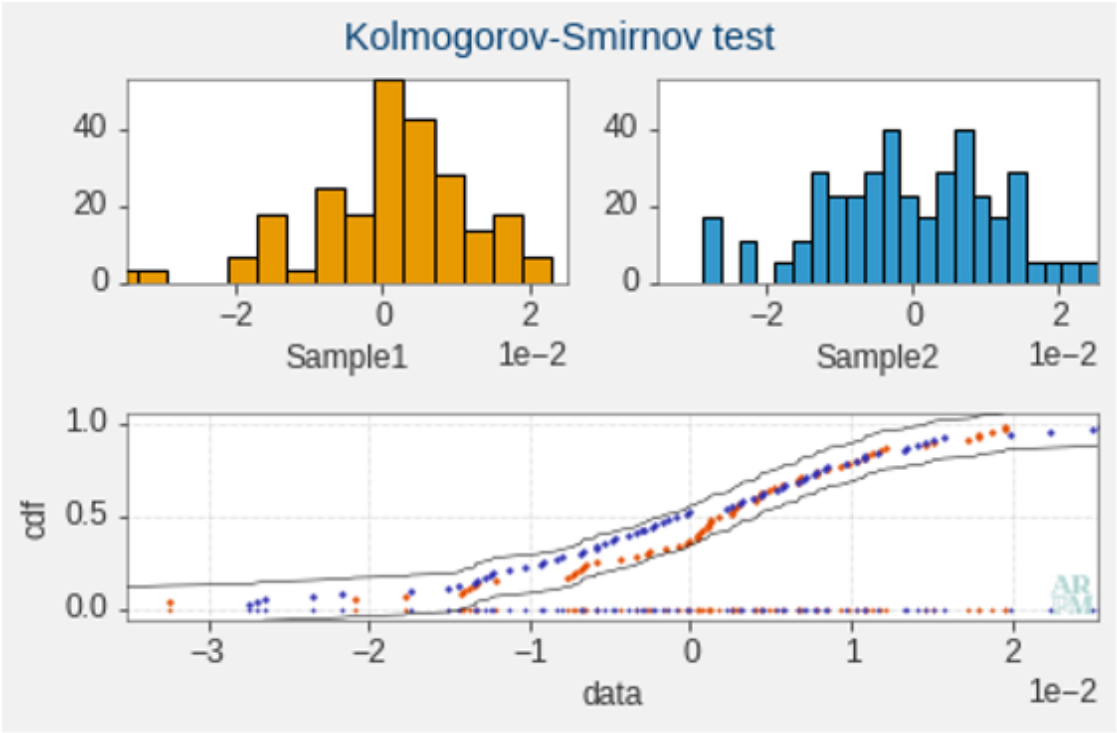
	Shapiro-Wilk	**D'Agostino's K^2**
AAPL	Statistics=0.972, p=0.013 Sample does not look Gaussian(reject H_0)	Statistics=17382, p=0.000 Sample does not look Gaussian(reject H_0)
AMZN	Statistics=0.977, p=0.030 Sample does not look Gaussian(reject H_0)	Statistics=3.579, p=0.167 Sample does not look Gaussian(fail to reject H_0)
GOOGL	Statistics=0.980, p=0.070 Sample does not look Gaussian(fail to reject H_0)	Statistics=4.281, p=0.118 Sample does not look Gaussian(fail to reject H_0)
FB	Statistics=0.950, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=17.086, p=0.000 Sample does not look Gaussian(reject H_0)

NFLX	Statistics=0.949, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=32.740, p=0.000 Sample does not look Gaussian(reject H_0)

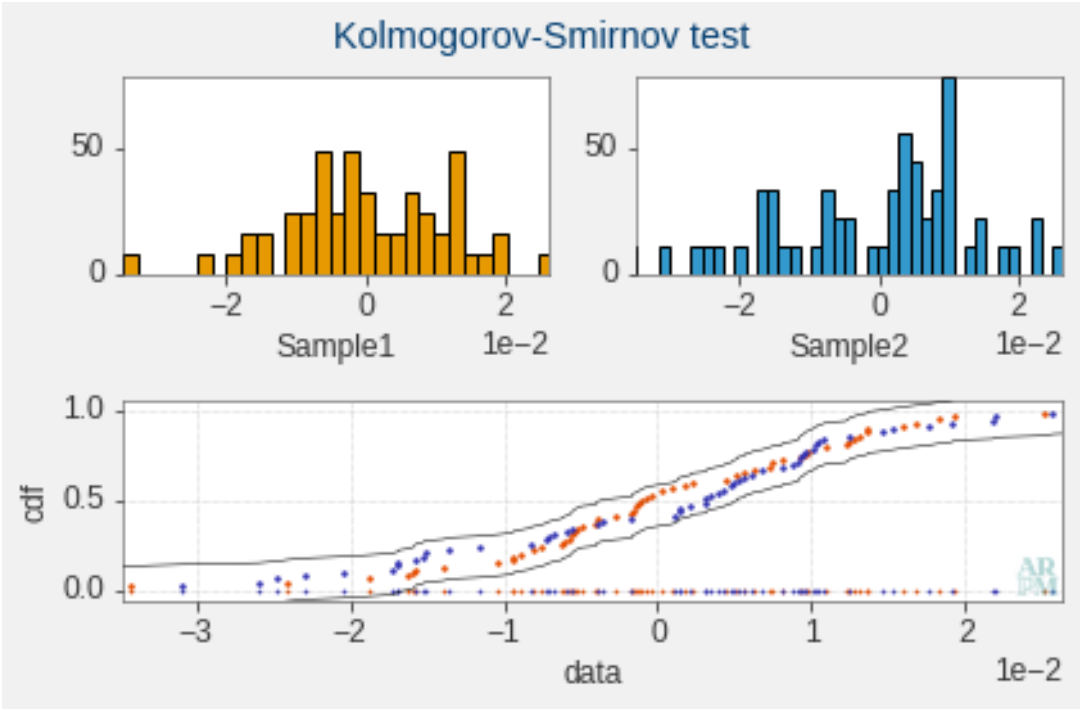
From the result table we could tell that only GOOGL certainly look normal, as it pass both normality test. For the AMZN only D'Agostino's K^2 test give positive results, and the rest of the equities are far from normal. Then we take a look at K-S test.



aAMZNL



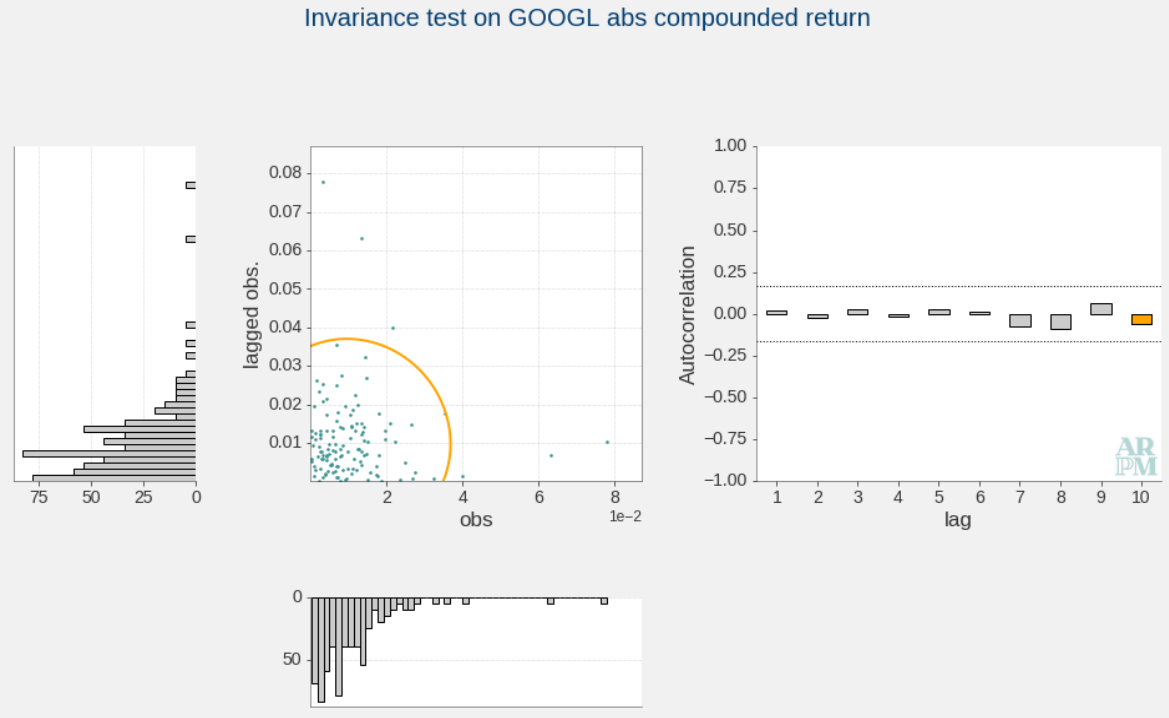
bAAPL

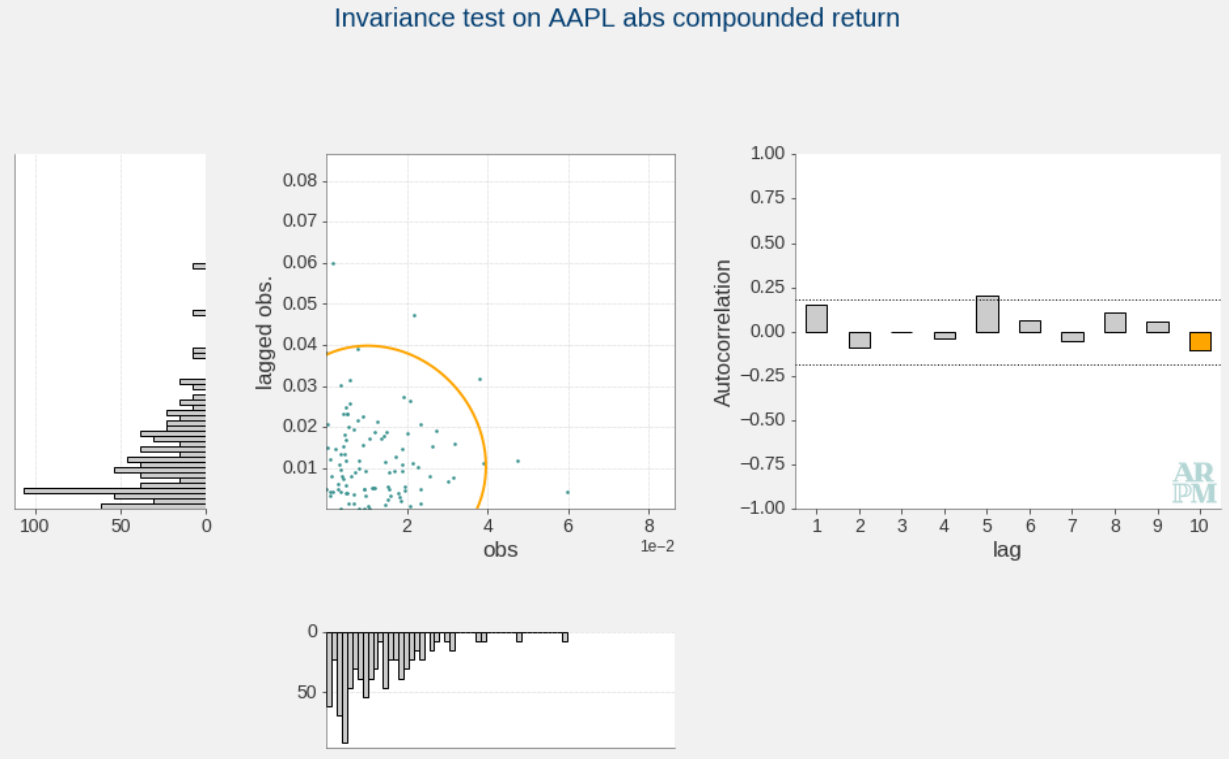


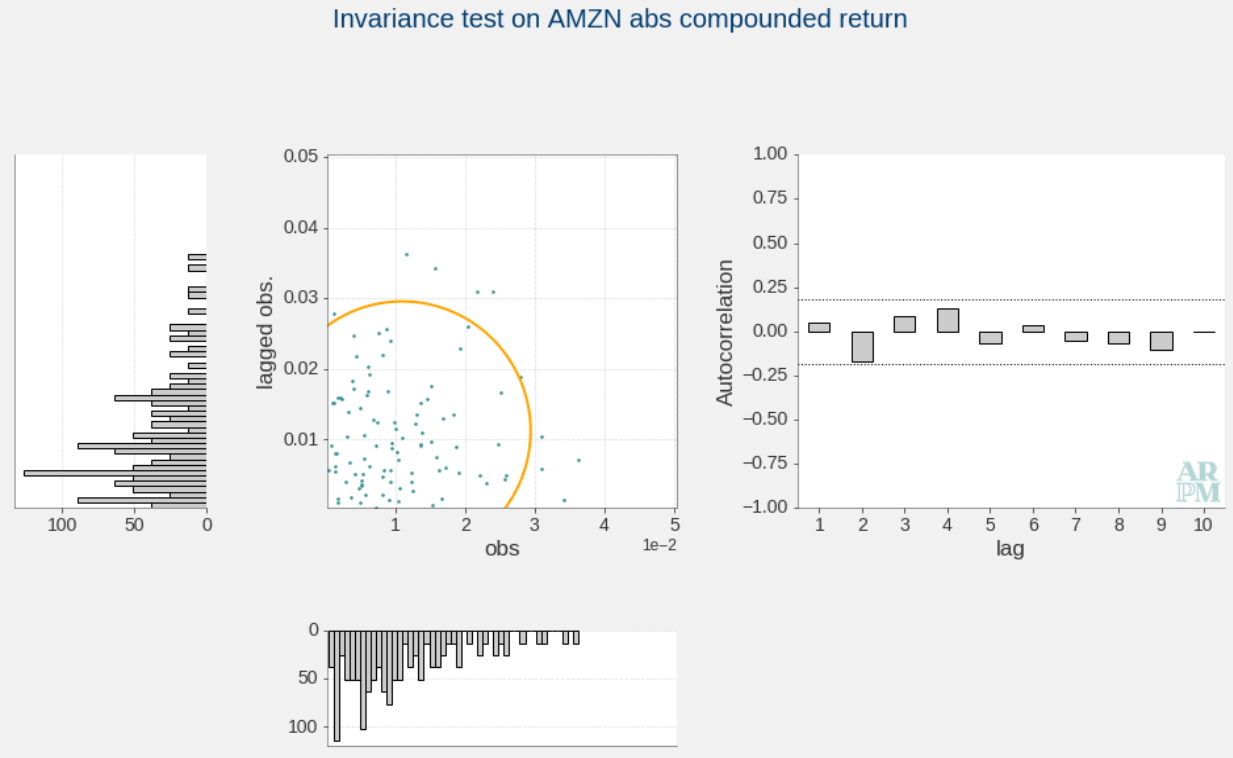
cGOOGL

It appears than K-St test do agree with the two normality test before as we see GOOGL has the more perfect fit.

Then we check the results of each equities' invariant test to see whether it agrees with the normality test.







We can tell invariants test do agree with the normality test reaults, as only GOOGL pass invariants test.

Here are the results for the FX and Bond

	Shapiro-Wilk	**D'Agostino's K^2**
USDJPY	Statistics=0.926, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=11.262, p=0.000 Sample does not look Gaussian(reject H_0)
UADUSD	Statistics=0.939, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=13.021 p=0.000 Sample does not look Gaussian(reject H_0)
GBPUSD	Statistics=0.960, p=0.016 Sample does not look Gaussian(reject H_0)	Statistics=34.000, p=0.000 Sample does not look Gaussian(reject H_0)
ERUUSD	Statistics=0.945, p=0.013 Sample does not look Gaussian(reject H_0)	Statistics=7.996, p=0.018 Sample does not look Gaussian(reject H_0)

CNYUSD	Statistics=0.921, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=37.000, p=0.000 Sample does not look Gaussian(reject H_0)
	Shapiro-Wilk	D'Agostino's K^2
6-month	Statistics=0.933, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=8.721, p=0.013 Sample does not look Gaussian(reject H_0)
1-year	Statistics=0.925, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=5.891 p=0.034 Sample does not look Gaussian(reject H_0)
5-year	Statistics=0.938, p=0.012 Sample does not look Gaussian(reject H_0)	Statistics=24.008, p=0.000 Sample does not look Gaussian(reject H_0)
10-year	Statistics=0.919, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=23.335, p=0.018 Sample does not look Gaussian(reject H_0)
30-year	Statistics=0.918, p=0.000 Sample does not look Gaussian(reject H_0)	Statistics=34.368, p=0.000 Sample does not look Gaussian(reject H_0)

[B]exponential decay[10]

```
def exp_decay_fp(t_, tau_hl, t_star=None):

    """


    if t_star is None:
        t_star = t_

    # compute probabilities
    p = np.exp(-(np.log(2) / tau_hl) * abs(t_star - np.arange(0, t_)))
    # rescale
    p = p / np.sum(p)
    return p
```

[c]bootstrapping [11]

```
def bootstrap_hfp(epsi, p=None, j_=1000):


    if len(epsi.shape) == 1:
        epsi = epsi.reshape(-1, 1)
```

```
    t_, i_ = epsi.shape

    if p is None:
        p = np.ones(t_) / t_

    # Step 1: Compute subintervals

    s = np.r_[np.array([0]), np.cumsum(p)]

    # Step 2: Draw scenarios from uniform distribution

    u = np.random.rand(j_)

    # Step 3: Compute invariant scenarios

    ind = np.digitize(u, bins=s) - 1
    epsi_j = epsi[ind, :]

    return epsi_j
```

[d] projection function estimation_projection

```
def estimation_projection(price1):
    epsi = np.diff(np.log(price1))
    tau_hl=30
    m_ = 10  # number of monitoring times
    j_ = 1000  # number of scenarios
    x = np.log(np.array(price1))
    t_ = len(epsi)
    t_star = t_
    p_exp = exp_decay_fp(t_, tau_hl, t_star)
    x_tnow_thor = simulate_rw_hfp(x[-1].reshape(1), epsi, p_exp, j_, m_).squeeze()
    testing = []
    for i in range(0,len(x_tnow_thor)):
        test = np.diff(x_tnow_thor[i])
        testing.append(test)
    flatten_list = [j for sub in testing for j in sub]
    return flatten_list
```

[e]SVM

$$\epsilon_i = \frac{1}{N}(\sum_j w_j \delta(C_i(x_j) \neq y_j)) \quad (1)$$

Then renew the weights:

$$D_{t+i}(i) = \begin{cases} \frac{D_t(i)}{Z_t} \times e^{-a_t} & if h_t(x_i) = y_i \\ \frac{D_t(i)}{Z_t} \times e^{a_t} & otherwise \end{cases} \quad (2)$$

$$Here, a_i = \frac{1}{2} \ln \frac{1-\varepsilon_i}{\varepsilon_i}$$

The final Adaboosr-SVM algorithms for T period:

$$H(x) = \sum_{t=1}^{T} a_t h_t(x) \quad (3)$$

- DATA

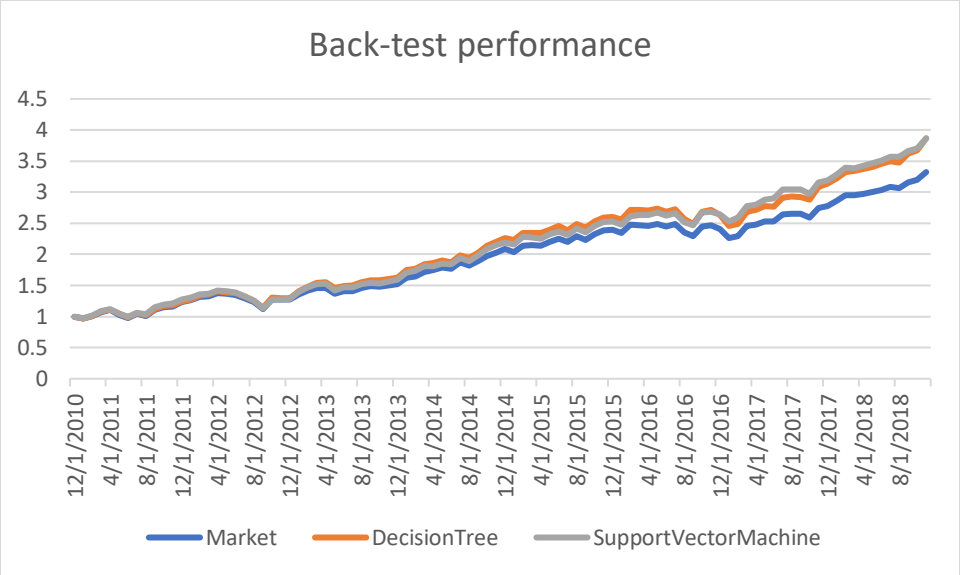
- 12/31/2010-11/30/2018 is our back-testing period, which means we make our first trade at 12/31/2010.

- At the end of each month, we use previous six years data (72 piecies monthly data) to train the model. Like 12/31/2010, the model we used is trained by 12/31/2004-11/30/2010.

- Use the model and data this month to predict one-month future return and sort the predicted return.

- Long the stock with top 30% predicted return, hold it for one month and close the position and repeat all the steps again at the end of next month.

A	B	C
Class	**Factor**	**Index Calculation**
Market Capital	Log_Mkt_Cap	Log(Market Capital)
Earnings Yield	PE	Current Stock Price/ Earnings per Share
	PB	Current Stock Price/ NBV per Share
Financial Qualit	ROE	Net Profit/Net Asset
	ROA	Net Profit/Total Asset
Momentum	one_month_momentum	Current Closing Price/ Closing Price One Month Ago - 1
	three_month_momentum	Current Closing Price/ Closing Price Three Month Ago - 1
	six_month_momentum	Current Closing Price/ Closing Price Six Month Ago - 1
	one_year_momentum	Current Closing Price/ Closing Price One Year Ago - 1
Volume	Log_Volume	log(trading volume)
Volatility	one_month_volatility	std(One Month Daily Stock Prices)
	two_month_volatility	std(TwoMonth Daily Stock Prices)
	three_month_volatility	std(Three Month Daily Stock Prices)



Ratio Statistic

	Total_Return	*Annually_Return*	*Sharp_Ratio*
Market	2.32304	0.161962	0.944821
DecisionTree	2.865643	0.184139	1.037326
SupportVectorMachine	2.866193	0.18416	1.05041

We can see the strategies constructed by the machine learning method both outperform the market. The annually returns outperform the market by about 13.67% and the sharp ratios beat the market by about 10% which is significantly positive.