1 Efficient Frontier

Let a portfolio of N assets be $\boldsymbol{\pi}$, whose expected return is $\boldsymbol{\mu}$ and the co-variance is $\boldsymbol{\Sigma}$.

1.1 Efficient Frontier with 3 Assets

According to the paper, the expected return of the portfolio, $E = \sum_{i=1}^{N} \pi_i \mu_i = \boldsymbol{\pi}^t \boldsymbol{\mu}$. The risk is analogous to the variance of the returns, i.e. $V = \sum_{i=1}^{N} \sum_{j=1}^{N} \sigma_{ij} \pi_i \pi_j = \boldsymbol{\pi}^t \boldsymbol{\Sigma} \boldsymbol{\pi}$.

Given $\boldsymbol{\mu} = \boldsymbol{m}$ and $\boldsymbol{\Sigma} = \boldsymbol{C}$ for a 3 assets, we can generate 100 random portfolios, where each portfolio $\boldsymbol{\pi} = (\pi_1 \pi_2 \pi_3)^t$ s.t. $\mathbf{1}^t \boldsymbol{\pi} = 1$ by `y=randn(3,1); y=y/norm(y,1)`. Then we can calculate $E - V$ for each of the portfolios by `E=y'*m; V=sqrt(y'*C*y)`.

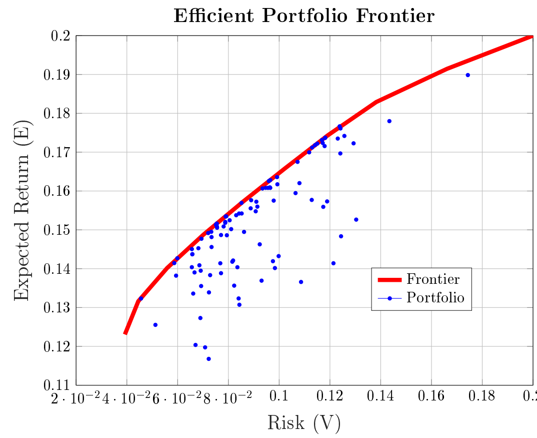


Figure 1: Efficient Portfolio

Finally I make the scatter plot and on the same figure I plot the efficient frontier using `estimateFrontier` function. As expected all the random portfolios were on the correct one side of the frontier.

1.2 Efficient Frontier with 2 Assets

To prepare three 2 asset porfolios, we remove the data points that are not necessary, i.e. that has the third asset. First I plotted random returns generated by the 2 asset mean and variance using `mvnrnd`. As can be noticed from Figure 2, asset 2 and 3 are almost uncorrelated, asset 1 and 2 are negatively correlated, asset 1 and 3 are positively correlated.

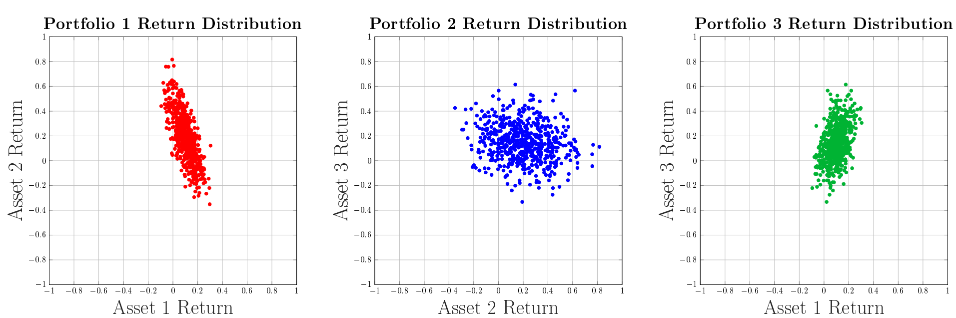


Figure 2: Distribution of 2 Asset Returns

As done previously with all three assets, I generate 100 random portfolios for each of the three 2 asset combinations and plot the $E - V$ scatters along with the efficient frontiers. Notice that in case of 2 asset portfolios, every portfolio construction is efficient and the frontier has a bend, i.e. risk increases for the lowest returns.

1.3 Use of `linprog` in NaiveMV

In order to calculate the efficient frontier, we need two extreme points - maximum return for a portfolio regardless of the risk and minimum risk regardless of the return. For the first case, we have $E = \max_{w}(\boldsymbol{\pi}^t \boldsymbol{\mu})$

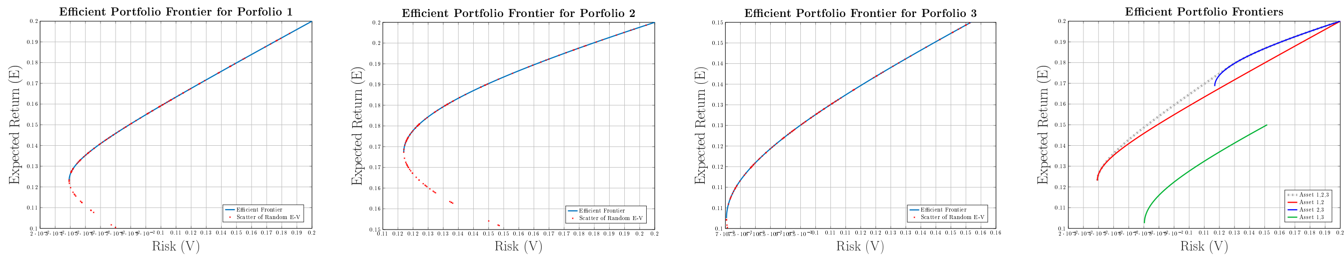


Figure 3: Efficient Frontier for 2 Asset Portfolios

s.t. $\mathbf{1}^t\boldsymbol{\pi} = 1$ which gives the portfolio that maximises the return regardless of the risk. We can then calculate $E - V$ for this portfolio, thus giving us the top corner of the $E - V$ graphs here. This is a linear equation of $\boldsymbol{\pi}$. Matlab's `linprog` function can solve linear equation can solve such equations of the following form -

$$\min_x (f^t x) s.t. \begin{cases} A.x \leq b \\ A_{eq}.x = b_{eq} \\ lb \leq x \leq ub \end{cases}$$

```
f=-ERet;A=[];b=[];Aeq=ones(1,N);beq=1;lb=0;ub=1
```

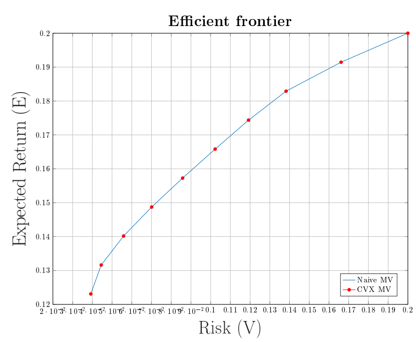
However, to calculate the portfolio that minimises the risk we need to solve a quadratic equation of $\boldsymbol{\pi}$, $\underset{w}{mim}(\boldsymbol{\pi}^t\boldsymbol{\Sigma}\boldsymbol{\pi})$ s.t. $\mathbf{1}^t\boldsymbol{\pi} = 1$. In this case we use the `quadprog` function. Finally, we choose N expected returns between the two extreme points and calculate the porfolio that minimises the risk while achieving the chosen expected returns. Thus the efficient portfolio is created.

1.4 Efficient Frontier : NaiveMV vs CVX

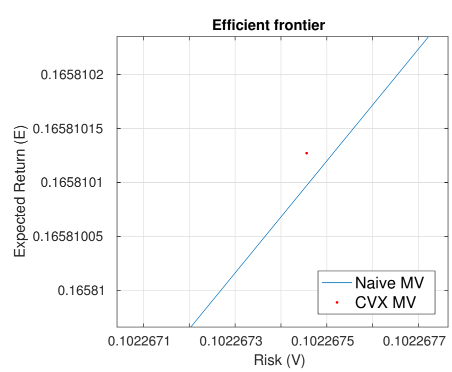
Using the CVX tool we can declarively perform the convex optimisations we performed earlier with `linprog` and `quadprog`, as follows -

```
cvx_begin quiet
        variable w(N,1)
        minimize( -ERet'*w )
        subject to
                ones(1,N)*w == 1;
                w >= zeros(N,1);
cvx_end
```

```
cvx_begin quiet
        variable w(N,1)
        minimize( 0.5*w'*ECov*w + zeros(N,1)'*w)
        subject to
                ones(1,N) * x == 1;
                x >= zeros(N,1);
cvx_end
```



(a) Similarity



(b) Difference

Figure 4: NaiveMV vs Using CVX

The results are extremely similar, since differences only show up in 10^{-7} scale. However, the CVX tool was noticeably slower.

2 Markowitz vs Naive $1/N$ Strategy on FTSE 100

2.1 Getting FTSE 100 Data

I downloaded the FTSE 100 index and the top 30 most traded companies' data over the last 3 years using a `bash` script. Then I used a `ruby` script to fill in the missing days (which may be weekends) with the previous day's data. As a result all data had the same number of rows. The returns will be calculated based on the adjusted close values. Figure **??** and **??** returns plots the data.

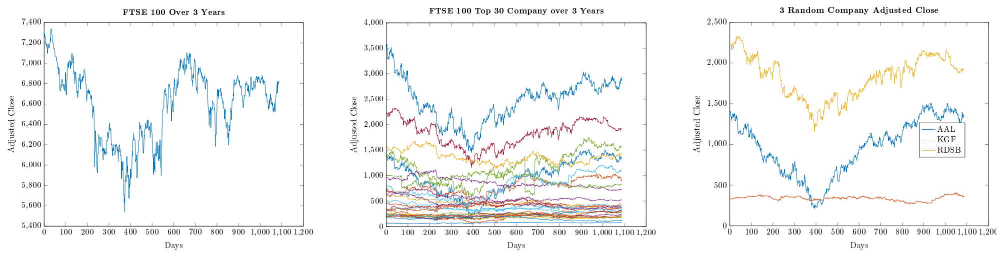


Figure 5: FTSE 100, Top 30 Most Traded Company and Selected 3 Company Adjusted Close over 3 Years

2.2 Returns and Efficient Portfolio of 3 Random Assets

I started by calculating the returns as percentage. I defined the return in two ways -

- $return(i+1) = (val(i+1) - \boldsymbol{val(i)})/val(i+1)$, return based on daily investment
- $return(i+1) = (val(i+1) - \boldsymbol{val(1)})/val(i+1)$, return based on first investment

I expected portfolio returns in first definition to be jittery in a small window since daily returns go up and down. Whereas the second definition would be smoother, although it implicitly imposes a correlation. Figure 6 and 8 build a portfolio out of **all** 30 assets, whereas Figure 7 and 9 select **3** random assets, which were controlled using `rng(1)`. The resultant assets were AAL, KGF and RDSB. Their adjusted closing values were plotted in Figure **??**.

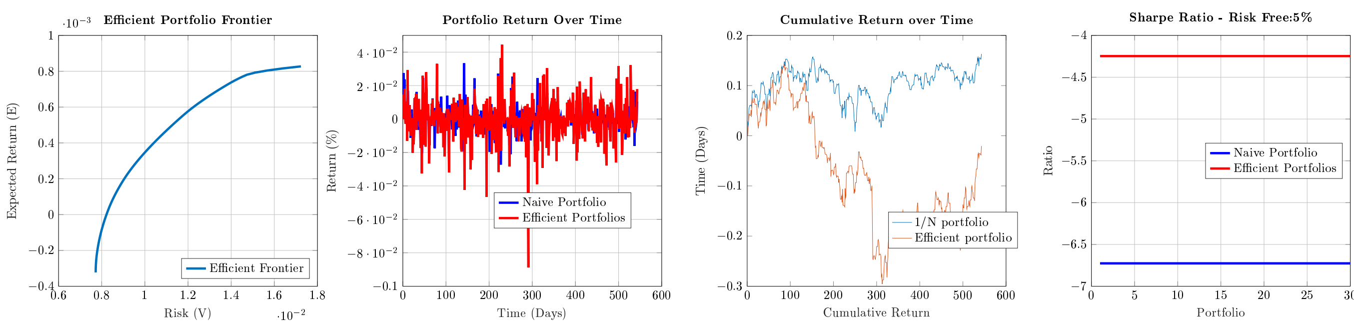


Figure 6: Portfolio of **all** 30 assets analysis where returns are based on **daily** change

As expected, the portfolio returns over time were moving between (± 0.02) if we calculated returns based on daily investment and were smoother compared to first investment. The efficient frontier in both cases were of expected shape.

2.3 Comparison with Naive Strategy

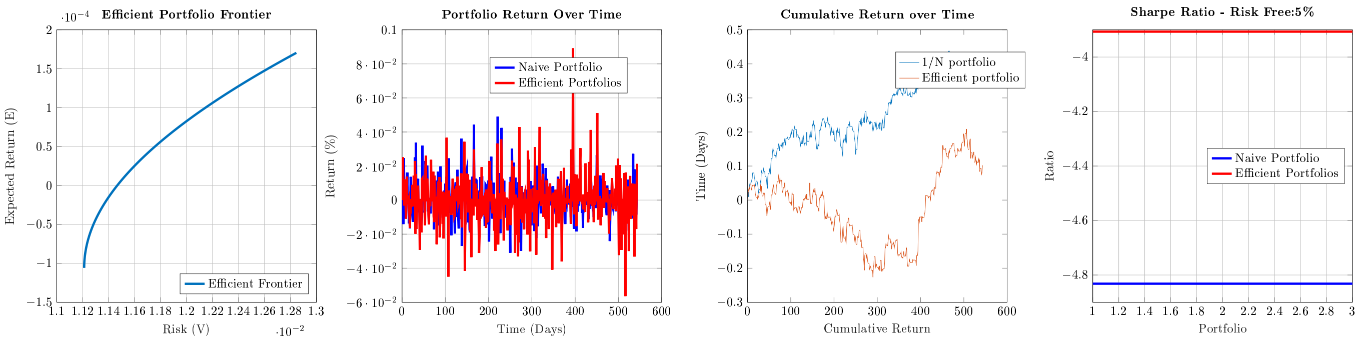


Figure 7: Portfolio of **3 random** assets analysis where returns are based on **daily** change

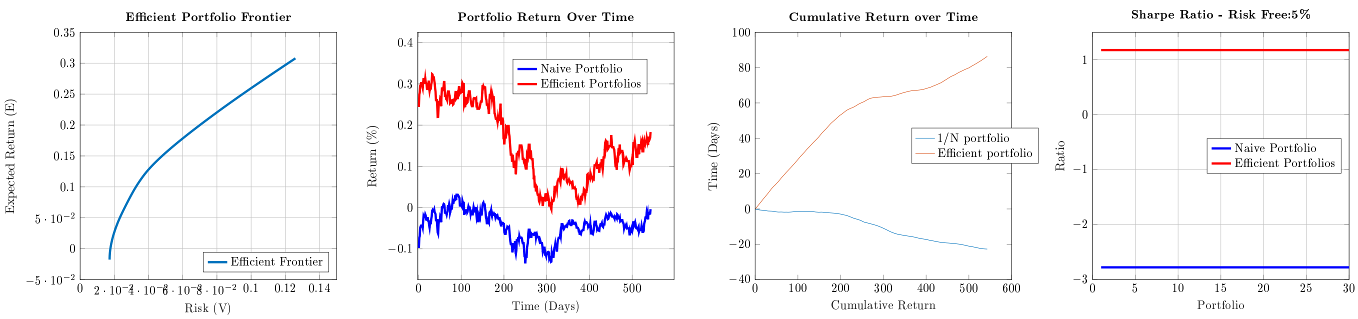


Figure 8: Portfolio of **all** 30 assets analysis where returns are based on **first return**

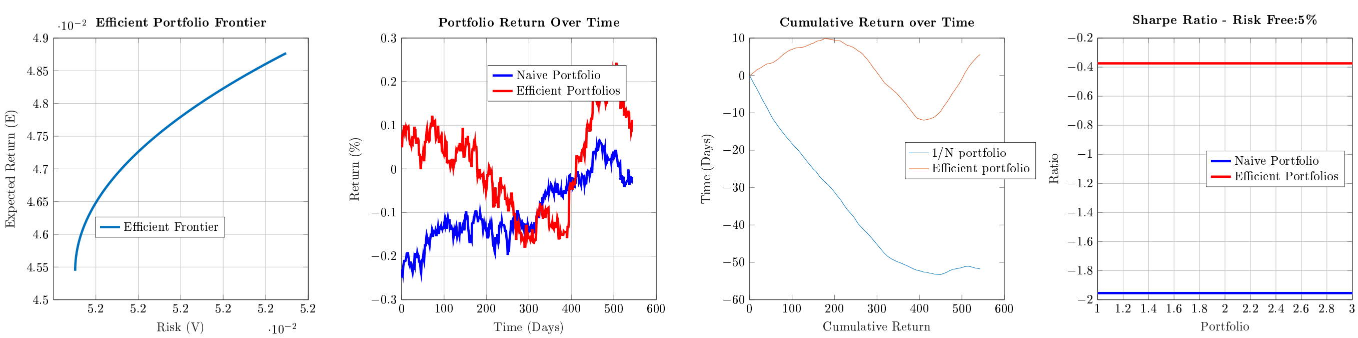


Figure 9: Portfolio of **3 random** assets analysis where returns are based on **first return**