ABSTRACT
Financial portfolio optimization is a difficult problem as it deals with many variables. Modern Portfolio Theory (MPT) is used for minimizing risk for a specific expected return. Many approaches are proposed to optimize portfolios. This paper proposes financial portfolio optimization using Monte Carlo and operation research. Results show an effective financial portfolio optimization.

Keywords
Financial Portfolio Optimization, Monte Carlo

1. INTRODUCTION
A financial portfolio is just a set of allocations in variety of securities. Portfolio optimization is a difficult problem as it deals with many variables. Portfolio management uses lower partial risk (downside risk) [1].

Harry Markowitz was awarded Nobel Prize in economy for his invention of Modern Portfolio Theory (MPT) [2]. MPT is used for minimizing risk for a specific expected return or equivalently, maximizing portfolio expected return if the risk amount is specified. MPT, also known as mean-variance method, explains the trade-off between mean and variance. MPT can be formulated as [3]:

\[
\min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}
\]

Subject to

\[
\sum_{i=1}^{N} w_i \mu_i = \mu^*
\]

and

\[
\sum_{i=1}^{N} w_i = 1
\]

Where, N is the total number of available portfolio assets, \( \mu_i \) is the expected return of portfolio asset \( i \), \( \sigma_{ij} \) is the covariance between portfolio assets \( i \) and \( j \), \( \mu^* \) is the total desired expected return, \( w_i \) is the proportion of the total capital invested in portfolio asset \( i \), Sharpe introduced a ratio of reward-to-variability [4]. It can be applied in portfolio optimization as shown in equation 4.

\[
S = \frac{R_p - R_f}{\sigma_p}
\]

Where \( R_p \) is expected portfolio return, \( R_f \) is risk free return, and \( \sigma_p \) is portfolio standard deviation.

Assuming \( R_F \) is zero,

\[
S = \frac{R_p}{\sigma_p}
\]

Many approaches are proposed to optimize portfolios, such as multi-objective evolutionary algorithms (MOEAs) [5], integer programming [6], shrinking the sample covariance matrix [7], stochastic control [8], parameter- dependent semi-martingales [9], mean–variance [10], and fuzzy logic [11]. MPT can be hybridized with artificial intelligence such as harmony search [3] and genetic algorithm [12]. MPT can be used for endowment [13] and non-profit organizations [14] too. Recently, MPT is adopted to reduce risks in energy planning [15] and environmental planning [16].

The rest of this paper is decomposed as follows. Section 2 is background and Section 3 is methodology. Section 4 is conclusion and Section 5 is future.

2. BACKGROUND
2.1 Monte Carlo Simulation
Monte Carlo simulation is used for finding the best Sharpe ratio. We assign a weight to each security in our portfolio in a random way, and then its mean and standard deviation of daily return are calculated. It is a technique used to understand the effect of risk and uncertainty and it is used in many fields such as physics, engineering, statistics [17] and finance [18]. There are methods that motivate improved simulation efficiency to conduct a deeper investigation into properties of a model [19]. Monte Carlo methods use deterministic point because of random point samples are wasted in clustering [20]. Monte Carlo methods are known as particle filters and they used to compute the high-dimensional [21].

2.2 Operation Research
Referring back to our Sharpe ratio; we want to actually maximize it, meaning we need an optimizer that will attempt to minimize the negative Sharpe ratio. The SciPy library is a common package that offers many efficient numerical routines for linear algebra and optimization. SciPy shall be used to calculate the optimal weight allocation.

The common method is Sequential Least Squares Program (SLSQP) [22]. Most financial practitioners often offer suboptimal solutions, so they use SLSQP because it converges to a local optimum near the seed. Non-negative least squares (NNLS) does not accept inequality constraints [23]. Competitive financial performance based on the SLSQP
method is achieved better in a portfolio [24].

3. METHODOLOGY
The proposed algorithm is shown in Fig. 1.

```python
data_source = "http://finance.yahoo.com"
start_date = 2015-03-02
end_date = 2020-02-28
assets = ['aapl', 'amzn', 'fb', 'goog']
working_days_per_year = 252

dataset = data_acquisition(data_source, start_date, end_date, assets)

mean_of_log_returns = mean( log (dataset) ) × working_days_per_year

covariance = cov( log (dataset) ) × working_days_per_year

random_weights = create_random_weights()
norms = Rebalance_to_sum_to_1 (random_weights)

Sharpe_Ratio = get_expected_return(norms) ÷ get_expected_volatility(norms)

optimal_point = argmax (Sharpe_Ratio)
efficient_frontier = Plot (Sharpe_Ratio)
```

First step is data acquisition from http://finance.yahoo.com. The selected companies to invest in are Amazon (AMZN), Apple (AAPL), Facebook (FB), and Google (GOOG). The stock is for the five years from 2015-03-02 to 2020-02-28, as shown in figure 2. The first five rows are shown in table 1.

<table>
<thead>
<tr>
<th>Date</th>
<th>aapl</th>
<th>amzn</th>
<th>fb</th>
<th>goog</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015-03-02</td>
<td>118.796295</td>
<td>385.660004</td>
<td>79.750000</td>
<td>569.775696</td>
</tr>
<tr>
<td>2015-03-03</td>
<td>119.044762</td>
<td>384.609985</td>
<td>79.599998</td>
<td>572.069397</td>
</tr>
<tr>
<td>2015-03-04</td>
<td>118.290138</td>
<td>382.720001</td>
<td>80.900002</td>
<td>571.800110</td>
</tr>
<tr>
<td>2015-03-05</td>
<td>116.330099</td>
<td>387.829987</td>
<td>81.209999</td>
<td>573.754761</td>
</tr>
<tr>
<td>2015-03-06</td>
<td>116.504845</td>
<td>380.089996</td>
<td>80.010002</td>
<td>566.130676</td>
</tr>
</tbody>
</table>

The mean of the percentage change are 0.000787 for aapl, 0.001425 for amzn, 0.000863 for fb, and 0.000794 for goog. The correlation matrix is shown in Table 2.

<table>
<thead>
<tr>
<th></th>
<th>aapl</th>
<th>amzn</th>
<th>fb</th>
<th>goog</th>
</tr>
</thead>
<tbody>
<tr>
<td>aapl</td>
<td>1.000000</td>
<td>0.508584</td>
<td>0.470717</td>
<td>0.547329</td>
</tr>
<tr>
<td>amzn</td>
<td>0.508584</td>
<td>1.000000</td>
<td>0.581652</td>
<td>0.632800</td>
</tr>
<tr>
<td>fb</td>
<td>0.470717</td>
<td>0.581652</td>
<td>1.000000</td>
<td>0.609763</td>
</tr>
<tr>
<td>goog</td>
<td>0.547329</td>
<td>0.632800</td>
<td>0.609763</td>
<td>1.000000</td>
</tr>
</tbody>
</table>

Clearly, the correlation between any asset and itself is 1. The highest correlation is between amzn and goog. The lowest correlation is between aapl and fb.

This paper will now switch over to using log returns instead of arithmetic returns, for many of our use cases they are almost the same, but most technical analyses require de-trending/normalizing the time series and using log returns is a nice way to do that.

Log returns are convenient to work with in many of the algorithms we will encounter. As shown in Table 3.

<table>
<thead>
<tr>
<th>Date</th>
<th>aapl</th>
<th>amzn</th>
<th>fb</th>
<th>goog</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015-03-03</td>
<td>0.002089</td>
<td>-0.002726</td>
<td>-0.001883</td>
<td>0.004018</td>
</tr>
<tr>
<td>2015-03-04</td>
<td>-0.006359</td>
<td>-0.004926</td>
<td>0.016200</td>
<td>-0.000471</td>
</tr>
<tr>
<td>2015-03-05</td>
<td>-0.016709</td>
<td>0.013263</td>
<td>0.003825</td>
<td>0.003413</td>
</tr>
<tr>
<td>2015-03-06</td>
<td>0.001502</td>
<td>-0.020159</td>
<td>-0.014887</td>
<td>-0.013377</td>
</tr>
</tbody>
</table>
Fig. 3 shows the histogram of log returns. The transpose of log returns based on statistical measures as shown in TABLE 4.

![Fig 3: The histogram of log returns](image)

**TABLE 4. Statistical Measures**

<table>
<thead>
<tr>
<th></th>
<th>count</th>
<th>mean</th>
<th>std</th>
<th>min</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>aapl</td>
<td>1258</td>
<td>0.00066</td>
<td>0.015775</td>
<td>-0.1049</td>
<td>0.068052</td>
</tr>
<tr>
<td>amzn</td>
<td>1258</td>
<td>0.00126</td>
<td>0.018019</td>
<td>-0.0814</td>
<td>0.132178</td>
</tr>
<tr>
<td>fb</td>
<td>1258</td>
<td>0.00070</td>
<td>0.018045</td>
<td>-0.2102</td>
<td>0.144286</td>
</tr>
<tr>
<td>goog</td>
<td>1258</td>
<td>0.00067</td>
<td>0.015089</td>
<td>-0.0801</td>
<td>0.148872</td>
</tr>
</tbody>
</table>

The mean of log returns multiply in 252 days are 0.166941 for aapl, 0.317717 for amzn, 0.176489 for fb, and 0.171208 for goog. Note that we assume that 252 days per year, which are the working days only.

You will compute pairwise covariance of columns as shown in TABLE 5.

**TABLE 5. Covariance of Columns**

<table>
<thead>
<tr>
<th></th>
<th>aapl</th>
<th>amzn</th>
<th>fb</th>
<th>goog</th>
</tr>
</thead>
<tbody>
<tr>
<td>aapl</td>
<td>0.000249</td>
<td>0.000145</td>
<td>0.000134</td>
<td>0.000131</td>
</tr>
<tr>
<td>amzn</td>
<td>0.000145</td>
<td>0.000325</td>
<td>0.000189</td>
<td>0.000174</td>
</tr>
<tr>
<td>fb</td>
<td>0.000134</td>
<td>0.000189</td>
<td>0.000326</td>
<td>0.000166</td>
</tr>
<tr>
<td>goog</td>
<td>0.000131</td>
<td>0.000174</td>
<td>0.000166</td>
<td>0.000228</td>
</tr>
</tbody>
</table>

Multiplying covariance by days (252) is shown in TABLE 6.

**TABLE 6. Covariance Multiplied by 252**

<table>
<thead>
<tr>
<th></th>
<th>aapl</th>
<th>amzn</th>
<th>fb</th>
<th>goog</th>
</tr>
</thead>
<tbody>
<tr>
<td>aapl</td>
<td>0.062711</td>
<td>0.036644</td>
<td>0.033692</td>
<td>0.033101</td>
</tr>
<tr>
<td>amzn</td>
<td>0.036644</td>
<td>0.081818</td>
<td>0.047676</td>
<td>0.043821</td>
</tr>
<tr>
<td>fb</td>
<td>0.033692</td>
<td>0.047676</td>
<td>0.082060</td>
<td>0.041786</td>
</tr>
<tr>
<td>goog</td>
<td>0.033101</td>
<td>0.043821</td>
<td>0.041786</td>
<td>0.057371</td>
</tr>
</tbody>
</table>

We see clearly, the highest covariance between fb and itself. Now, Let’s get Single Run for Some Random Allocations. Stocks are ['aapl', 'amzn', 'fb', 'goog']. Creating Random Weights such as:

-0.51639863, 0.57066759, 0.02847423, 0.17152166
Rebalance to sum to 1.0:
0.40122278, 0.44338777, 0.02212343, 0.13326603

Expected Portfolio Return is 0.23457 and Expected Volatility is 0.22533

We get Sharpe Ratio by dividing expected return on expected volatility is 1.04101

Sharpe max is 1.111657517772319, Sharpe argmax is 2328 and all
Weights are 0.26188068, 0.20759516, 0.00110226 and 0.5294219.

You can plot all previous data as shown in Fig. 4 the red dot for a max Sharpe ratio.

**Fig 4: Maximum Sharpe ratio**

Fig. 5 show the curve adding the efficient frontier. Portfolios under this frontier are sub-optimal.

**Fig 5: Adding a Frontier line**
4. CONCLUSIONS
This paper proposed financial portfolio optimization using Monte Carlo and operation research. First step is data acquisition from data source. The mean of log returns and covariance are calculated and multiplied by the working days per year. Then random weights are generated and normalized to sum to 1. Sharpe ratio is calculated by dividing the expected return by the expected volatility. The optimal point is calculated using argmax function. Efficient frontier is plotted where portfolios under this frontier are sub-optimal. Results show an effective financial portfolio optimization.

5. FUTURE WORK
A possible future direction may be trying machine learning methods such as artificial neural networks [24].

6. REFERENCES