METAHEURISTIC APPROACH TO SOLVE PORTFOLIO SELECTION PROBLEM

Taylan Kabbani∗†
*Huawei Turkey R&D Center, Istanbul, Turkey
taylan.kabbani2@huawei.com
†Graduate School of Engineering and Science Ôzye˘gin University, Istanbul
taylan.kabbani@ozu.edu.tr

Abstract—In this paper, a heuristic method based on Tabu Search and TokenRing Search is being used in order to solve the Portfolio Optimization Problem. The seminal mean-variance model of Markowitz is being considered with the addition of cardinality and quantity constraints to better capture the dynamics of the trading procedure, the model becomes an NP-hard problem that can not be solved using an exact method. The combination of three different neighborhood relations is being explored with Tabu Search. In addition, a new constructive method for the initial solution is proposed. Finally, I show how the proposed techniques perform on public benchmarks.

Index Terms—Tabu Search, TokenRing, POP, NP-hard, Metaheuristics, Stock Market, Markowitz Model.

I. INTRODUCTION

Financial markets are at the heart of the modern economy and provide an avenue for selling and purchasing assets such as bonds, stocks, foreign exchange, and derivatives. The prime objective of any investor when investing in any of these markets is to minimize the risk involved in the trading process and maximize the profits generated, this objective can be met by diversifying the amount of money to be invested (Capital) among a set of available assets. Each one of the different ways to choose assets and diversify the capital between them is called a portfolio(or Securities) and the problem of finding this optimal portfolio that maximizes profits and minimizes risk is called Portfolio Optimization Problem (POP) or Portfolio Selection Problem. In his seminal work, Markowitz [1] proposed the Mean-Variance model to solve the portfolio optimization problem, which assumes that the risk of a particular portfolio can be minimized by minimizing the co-variance between the portfolio assets, i.e., minimizing the chance of loss by choosing assets that do not tend to move together (low co-variance), so if one asset’s price is going down the other asset will still have a chance to make up for the losses. Despite the pioneer model of Markowitz, it does not reflect the real-life trading procedure by lacking cardinality constraints that set a limit to the number of assets allowed in a portfolio as well as lacking any bounding constraints that set limits to the amount of money allowed to be invested in each asset. In this paper, an extension of Markowitz [1] model is being considered, in which cardinality constraints and quantity constraints are imposed to make it more adherent to real-world trading mechanisms. The addition of these constraints will turn the model from Quadratic Programming (QP) problem to a Mixed Integer Quadratic Programming (MIQP) problem, which is an NP-Hard problem that can not be solved using an exact method, therefore a Metaheuristic approach is needed.

The use of local search techniques for the POP has been proposed by Chang et al. (2000). [2] Here, I investigate the possible improvement of their Tabu Search (TS) technique by first, proposing an alternative of Initial Solution generation (Constructive heuristics) based on the Sharp Ration rather than randomly assigning the initial solution. Second, I introduce an additional move to diversify the search space and escape the local optima also instead of using a fixed step size, different step sizes are being considered, and finally, I investigate the application of the Token-Ring method with the Tabu search algorithm to implement Multi-Neighborhood Search. The proposed techniques are tested on the benchmarks proposed by Chang et al. (2000). [2] which represent real stock markets.

II. RELATED WORK

Chang et al. [2] used three different metaheuristic approaches (Genetic Algorithm, Simulated Annealing, and Tabu Search) to solve a multi-objective portfolio optimization problem with cardinality & quantity constraints. They reported that no individual heuristic was found to be dominating across five data sets. Schaerf et al. [3] explored the use of Tabu Search on single-objective constrained POP, and also proposed new algorithms that combine different neighborhood relations. They implemented several token-ring procedures in a way to diversify and intensify the search space and they concluded that TS is the most promising heuristic to solve POP. In an attempt of making the mean-variance model more realistic, Soleimani et al. [4] introduced sector capitalization as an additional constraint to reduce investment risk, they used GA and compared the results of a small instance problem to the results found by LINGO (which can generate global optima for small problems) and reported that the difference between LINGO’s global optimum and GA’s best objective is just 2.9%. By combining local search techniques (SD & FD) and exact mathematical programming (Quadratic programming solver) Gaspero et al. [5] proposed a hybrid local search algorithm, according to their results, the developed solver finds the optimal solution in several instances and is at least comparable

The project’s data and code are on Github

arXiv:2211.17193v1 [q-fin.PM] 10 Nov 2022
to other state-of-the-art methods from the others. Surveying the techniques used in literature ([2], [3], [4], [7], [9], [9]) to solve the constrained portfolio optimization problem reveals that Genetic Algorithm is the most used technique to solve the problem, where a couple of studies ([3], [4]) considered using Tabu search.

III. PORTFOLIO OPTIMIZATION PROBLEM

In this section, the portfolio selection (or optimization) problem is introduced in stages. The unconstrained Markowitz model is represented first in Section III-A. Subsequently, I introduce the specific constraints of the formulation considered in this study in Section III-B. Finally, the Efficient Frontier is explained in Section III-C.

A. Unconstrained Markowitz model

Given a set of n available assets, A = {a1, a2,..., an}, let \( \mu_i \) be the expected return (increase in asset price) for asset \( a_i \), and \( x_i \) the portion of the capital invested in asset \( a_i \), and each pair of assets \((a_i, a_j)\) has a real value of co-variance \( \sigma_{ij} \). Let \( \mu_p \) represent the desired return from the portfolio, thus the formulation of the unconstrained problem is given as:

\[
\min \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \sigma_{ij} \quad (1)
\]

\[
\text{S.t.} \quad \sum_{i=1}^{n} \mu_i x_i \geq \mu_p \quad (2)
\]

\[
\sum_{i=1}^{n} x_i = 1 \quad (3)
\]

\[
0 \leq x_i \leq 1 \quad i = 1, 2,..., n \quad (4)
\]

Eq. (1) minimizes the total variance (risk) associated with the portfolio, Eq. (2) ensures that the portfolio has an expected return of or bigger than \( \mu_p \). Eq. (3) ensures that 100% of the capital is being invested and short-selling is not feasible.

B. Constrained Markowitz model

While the previous single-objective model considers only minimal risk for a given expected return it’s desirable to also seek a maximum return for a given expected level of risk, this can be achieved by subsuming the expected return constraint into the objective function via a weighting approach which yields a profitable algorithmic approach. The objective function in Eq.(1) will become Eq.(5), where \( \lambda \) represents the risk aversion of the investor, when set to 0 the investor is taking high risk in order to maximize the profits, if 1, the investor is not willing to take any risk but profits will not be necessarily maximized (Eq.(10)).

The cardinality constraint will determine the number of assets k the portfolio must include. A binary variable \( z_i \) is introduced to denote whether an asset is being selected or not. Eq.(7) & Eq.(11). Whereas quantity constraint will set boundaries for the weights (proportions) of included assets by specifying a lower (\( \varepsilon \)) and upper (\( \delta \)) bounds allowed for the allocated proportions to each asset in the portfolio, Eq.(8).

\[
\min \lambda \left( \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \sigma_{ij} - (1 - \lambda) \sum_{i=1}^{n} \mu_i x_i \right) \quad (5)
\]

\[
\sum_{i=1}^{n} x_i = 1 \quad (6)
\]

\[
\sum_{i=1}^{n} z_i = K \quad (7)
\]

\[
\varepsilon z_i \leq x_i \leq \delta z_i \quad i = 1, 2,..., n \quad (8)
\]

\[
0 \leq x_i \leq 1 \quad i = 1, 2,..., n \quad (9)
\]

\[
0 \leq \lambda \leq 1 \quad (10)
\]

\[
z_i = \begin{cases} 1, & \text{if asset } i \text{ is held} \\ 0, & \text{otherwise} \end{cases} \quad (11)
\]

\[
z_i \in \{1, 0\}, \quad i = 1, 2,..., n \quad (12)
\]

Where:

* \( n \): number of stocks in the dataset.
* \( \lambda \): The risk aversion of the investor
* \( x_i \): Proportion of capital invested in asset \( i \)
* \( \sigma_{ij} \): The co-variance between asset \( i \) and \( j \)
* \( \mu_i \): The expected return of the asset \( i \)
* \( K \): the number of assets in the portfolio
* \( \varepsilon \): The minimum proportion invested in asset
* \( \delta \): The maximum proportion invested in asset

This formulation is a multi-objective mixed quadratic and integer programming problem for which exact methods do not exist.

C. Efficient Frontier

Due to the fact that the problem in this study is multi-objective optimization problem and usually has several different optimal solutions, it’s useful to approach the problem as Pareto Optimality Problem [10], that is by solving the QP in Section III-A for varying values of \( \mu_p \) (the desired return from the portfolio) we can trace a non-decreasing curve that represents the set of Pareto-optimal (non-dominated) portfolios, i.e., gives for each expected return the minimum associated risk. We call this curve The Unconstrained Efficient Frontier (UEF), Fig.1 shows the UEF for the smallest benchmark problem in [2], which has been computed taking 2000 different values of \( \mu_p \) and solving the unconstrained POP. In a similar way, the UEF for each data set in the benchmark is shown in Table III. Given that the constraint problem has never been solved exactly (Eqs.(5)-(12)), the quality of the heuristic method will be measured in average percentage loss w.r.t. the UEF, i.e., taking the sets of Pareto optimal portfolios obtained by solving the constrained POP using the proposed heuristic approach we can trace out the heuristic efficient frontier and compare it to the UEF (More is Section VI).
IV. PORTFOLIO OPTIMIZATION BY TABU SEARCH

For any local search method such as Tabu Search (TS), there are essential components we need to define, Initial Solution, Neighborhood Structures, and Search Space. In this section, these components will be described in addition to the TS algorithm parameters used in this study.

A. Search Space and Neighborhood Structures

In POP, the solution (portfolio) consists of two parts, the first part is the set of assets chosen from the pool of available ones, and the second part is the capital proportions allocated to each chosen asset. Hence, the solution will be represented as two sequences, \( L = [a_1, \ldots, a_n] \) and \( S = [x_1, \ldots, x_n] \) such that \( a_i \) is an asset in the solution (portfolio) and \( x_i \) is the capital fraction invested in asset \( a_i \), where the length of \( L \) satisfies Eq.(7) and \( S \) satisfies Eq.(6), Eq.(8) and Eq.(9).

As we are trying to optimize a continuous variable (capital fractions), the neighborhood generation is defined over the step of the move performed on the current solution. Given a step size \( q \) which is a real-value parameter we can define two moves to explore the search space, Increase_move and Decrease_move. The idea is to increase and decrease each of the asset weights in the solution in order to find the optimal ones. To illustrate the aforementioned neighborhood relation further suppose that the current solution is \( L = [2, 8] \) and \( S = [0.65, 0.35] \), the neighborhood solutions following the Increase_move with step size \( q = 0.8 \) would be \( S1 = [(1+0.8)\times0.65, 0.35] \) and \( S2 = [0.65, (1+0.8)\times0.35] \) and the same for the Decrease_move with replacing \( (1+0.8) \) by \( (1-0.8) \).

- **Increase_move (Solution, i, q):**
  - **Description:** Increases the weight of asset \( i \) in Solution. The number of neighbors generated equals to the number of assets \( k \) in the solution.
  - **Effects:** \( x_i = x_i \times (1+q) \)

- **Decrease_move (Solution, i, q):**
  - **Description:** Decreases the weight of asset \( i \) in the Solution. The number of neighbors generated equals to the number of assets \( k \) in the solution.
  - **Effects:** \( x_i = x_i \times (1-q) \), if \( x_i < \varepsilon \) replace asset \( i \) with a randomly chosen asset \( j \), not in the solution.

Notice in the example shown above, the \( S1 \) neighbor solution becomes \( S1 = [1.17, 0.35] \) which violates the constraints imposed. To ensure that all solutions in the searching space satisfy the imposed constraints (all weights sum up to one and satisfy the upper and lower bounds), Rescale (see Algorithm 1) algorithm is introduced [2].

In addition to the Increase_move and Decrease_move, a third move is added to diversify the search which is the Swap_move, this move swaps the asset with the lowest weight in the solution by an asset, not in the solution.

- **Swap_move (Solution, j):**
  - **Description:** Swaps asset \( i \) which has the lowest weight in Solution by asset \( j \), not in Solution. The number of neighbors generated equals (number of assets available \( N \) - number of assets \( k \) in the solution).
  - **Effects:** Swap(\( i,j \)), \( i \in Solution, j \notin Solution \) keep the same weights

\[
\text{Algorithm 1: Rescale}
\]

**Input:** Solution

**Output:** Solution which satisfies cardinality and quantity constraints

1. \( \text{free}\text{-}proportion = 1 - (K \times \varepsilon) \)
2. \( \text{weight}\text{-}sum = \sum_{i \in \text{Solution}} x_i \)
3. **for** Each \( x_i \) in Solution **do**
   4. \( x_i = \varepsilon + (x_i/\text{weight}\text{-}sum) \times \text{free}\text{-}proportion \)
      // No weight is > \( \varepsilon \):
   5. \( R = \text{Assets in Solution with weights >} \delta \)
6. **if** \( R \text{ not empty} \) **then**
   7. \( L = \sum_{i \in \text{Solution}-R} x_i \)
   8. \( \text{free}\text{-}proportion = 1 - (K \times \varepsilon + \text{length}(R) \times \delta) \)
   9. **for** Each \( x_i \) in Solution **do**
      10. **if** \( x_i < \delta \) **then**
          11. \( x_i = \varepsilon + (x_i/L) \times \text{free}\text{-}proportion \)
      12. **else**
          13. \( x_i = \delta \)
14. **return** Solution

B. Initial Solution Construction

In the literature [2], the initial feasible solution is selected as the best among randomly generated portfolios with \( K \) assets. In this project, I propose a Greedy approach (similar to Fractional Knapsack Problem) based on Sharpe Ration [12].
which is basically the ratio of the expected return of an asset to its standard deviation. (See Algorithm 2). To illustrate the effectiveness of the proposed greedy algorithm to find an initial solution, I solve a small problem (toy problem) that had been solved using the exact method, the General Algebraic Modeling System (GAMS) in [13]. As the problem in [13] is an unconstrained Portfolio Optimization Problem (POP), the constraints are relaxed in our model to match with the one solved by [13]. Lambda = 1, Capital = 1000, k=3, epsilon = 0.0, delta = 1, T= 10000, where T is the number of iterations to generate random weights and evaluate them, so by increasing T we might get a better solution but with computation trade-off. The obtained results, i.e., proportions of the capital to be invested in each stock, are very close to the ones found by the exact method, which is a good indication of the efficiency of the proposed approach. Solution found by GAMS : {0.497 IBM, 0.0 WMT, 0.502 SEHI}. Solution found by the Greed algorithm : {0.442 IBM, 0.1036 WMT, 0.454 SEHI} (See Code & Notebook here)

Algorithm 2: Initial Solution

Input: Expected returns, Standard deviations, Number of iterations to perform(T)

Output: \{stock_1: x_1, stock_2: x_2, ..., stock_n: x_n\}

1. L = [] // Starting with empty set of assets
2. S=[] // Starting with empty set of fractions
3. T = Integer // Number of specified Iterations
4. Best_solution = {}
5. V = Infinite // Best value found for the objective function

6. for Each available asset do
7. Calculate Sharpe_ratio = Asset Return / Standard Deviation
8. Rank available assets based on their Sharpe_ratio in a descending order
9. Add the first K assets to L // Constraint in Eq.(7)
10. for r=1 to T do
11. S = {} // Current solution in the iteration
12. f = 0 // Current value for the objective function
13. for Each asset in L do
14. x_i = randomly assign a value between \(\varepsilon\) and \(\delta\)
15. Renormalize weights in S to add up to one // Constraint in Eq.(6)
16. if \(f < V \& length(S) = k\) then
17. Update the best solution
18. V = f
19. Best_solution = S
20. return L, S

C. Tabu Search Parameters

The parameters of TS are tuned using the Grid Search technique, i.e., testing different combinations of the parameters and choosing the best-performing one.

- **Tabu List:** For each neighborhood relation a tabu list is created. Tabu tenure is set to 3 throughout the whole algorithm (static) for Increase_move and Decrease_move, whereas for Swap_move the tabu tenure is 20 due to the fact that this move contributes to the stability of the search.

- **Tabu Attribute:** Because TS aggressively selects the best admissible solution from the neighborhood space, moves are isolated and updated in short-term memory, which records the increase/decrease of the move value for each solution in the neighborhood. The tabu attribute of Increase_move and Decrease_move is the asset that has been increased/decreased. Swap_move is the asset that has been added by the move to the solution. For example, \{asset5: \{Increase: 0, Decrease: 3, Swap: 20\}\} which means that asset5 is tabu to be Decrease for the next three iterations but it’s not for Increasing move, and it’s tabu to be removed (swapped) if it has the lowest weight in solution for the 20 next iterations.

- **Aspiration Criteria:** If the current move is tabu but delivers a better solution than the best found one, its tabu statute is overridden.

- **Termination Condition:** When a solution that improves the Best (the incumbent) solution cannot be obtained in 200 consecutive iterations.

V. Token Ring Search

The token-ring search is a sequential solving strategy for combining different neighborhood functions. Given an initial state and a set of algorithms or what we call runners, the token-ring search makes circularly a run of each runner, always starting from the best solution found by the previous one. The effectiveness of the token-ring search for two runners has been stressed by several authors [11].

A. t1 Runner

Given a step size \(q\), the t1 runner uses the TS algorithm described in Section IV to search the space for the local optima solution. That is, given an initial solution \(S_0\), t1 will iterate to update the current and best solutions, by choosing a non-tabu move or Best-improving move from the Neighborhood Union of the three neighborhood relations applied on the current solution. Algorithm 3 shows the details of the t1-TS runner.

B. t2 Runner

In the t2 runner, the step size \(q\) is defined. Step Size \(q\) starts with a big value \((q = 5)\) and decreases by 0.2 for each run of the t1 runner until it reaches 0.2. By starting with big steps we guarantee diversification by making big moves and small steps provide intensification with small moves. Here, I define one run of the t2 runner as a complete iteration over all step sizes using the t1 runner. t2 runner stops when no improving solution is found by the current run (See Algorithm 4). t1_Ts always starts from the best solution found by the previous one.
Algorithm 3: t1_Ts

Input: Initial_Solution, q
Output: local optima solution found using step size q
1 Best_Solution = Initial_Solution
2 Current_Solution = Initial_Solution
   // Generate neighborhood solutions from the Current_Solution:
3 Neighbors_Solutions = []
4 for asset i in Current_Solution do
5    Add Increase_move (Current_Solution, i, q) &
6    Decrease_move ((Current_Solution, i, q) to
7    Neighbors_Solutions
8 for asset j not in Current_Solution do
9    Add Swap_move (Current_Solution, j) to
10   Neighbors_Solutions
11 Select admissible move from Neighbors_Solutions:
12 Select the non-tabu improving or least Improving
13 move. Override tabu statues if the move meet the
14 aspiration criteria.
15 Update Current_Solution & Best_Solution
16 Check Termination:
17 if No Best_Solution found in 200 consecutive iteration
18   return Best_Solution
19 else
20   Go to step 3

Algorithm 4: t2

Input: Initial_Solution
Output: Best Solution
1 Initialize: step_size_list = [q1 = 5, q2 = 4.8,...,qn = 0.2]
2 Initial_Solution = Initial Solution Algorithm
3 Next_Solution = t1_Ts(Initial_Solution, q0 = 5.2) // [1]
4 for step qi in step_size_list // One r2 run
5    do
6       Reset tabu time of all moves
7       Next_Solution = t1_Ts (Next_Solution, qi)
8    if Improved solution found by any step_size in 4 then
9       Go to 4 // another t2 run
10   else
11      // Final Solution
12      Stop & return Next_Solution

VI. COMPUTATIONAL EXPERIMENTS

A. Benchmarks and experimental setting

In this section, the proposed approach in this study is being used to solve the Constrained Model described in Section [III-B]. I have done all the computational experiments with five sets of benchmark data that have been already used in [2]. These data correspond to weekly prices from March 1992 to September 1997 and they come from the indices: Hang Seng in Hong Kong, DAX_100 in Germany, FTSE_100 in the UK, S&P_100 in the USA, and Nikke_225 in Japan. The number of assets varies within each data set: 31, 85, 89, 98, and 225 assets, respectively. Data Source

All the results presented here have been computed as in [2], using the values K = 10, ε = 0.01 and δ = 1 for the problem formulation, and the values λλ = 0.02 for the implementation. So the method is implemented with 51 different values for the risk aversion parameter λ. As a result for each benchmark data we have 51 optimal solutions (Portfolios), and each has the best objective function value found by the proposed heuristic for the weighting parameter λ. From these obtained portfolios (solutions) we can trace back the Constrained Efficient Frontier (CEF) by calculating the Return and Variance of return (Risk) of each portfolio. Fig [2] shows the CEF for the smallest benchmark data, where 51 points are plotted, and each point represents a solution (Portfolio) (x=Risk,y = Return). Similarly, Table [III] shows the UEF for each data set in the benchmark.

![Fig. 2. Constrained Efficient Frontier corresponding to the smallest benchmark problem. (UEF of Hang Seng)](image)

B. Percentage Deviation Calculation

To measure and quantify the quality of the solutions obtained, hence the performance of the proposed method, the percentage deviation (distance) of each portfolio from the (linearly interpolated) Unconstrained Efficient Frontier (UEF) is computed as following:

Let the pair (v, r) represent the variance and mean the return of a point in the CEF (solution found by the proposed method). Let also (v̂, ŵ̂) represent the variance and mean the return of a point in the UEF. We can find v̂ and ŵ̂ which brackets v and r on the UEF as, v̂ = min[ŵ̂ / ŵ̂ ≥ v] and ŵ̂ = max[ŵ̂ / ŵ̂ ≤ v] and by simple geometry we can calculate ŵ̂ which is the value associated with the x-direction linear interpolated point with ŵ̂ = v on the UEF. ŵ̂ = r̂ + (r̂ - ŵ̂)[(r̂ - r̂) / (r̂ - r̂)]. From ŵ̂ we can calculate convenient percentage deviation error of the solution as ϕ = [100 × (r̂ - ŵ̂) / ŵ̂]. The percentage deviation error associated with y-direction is calculated in the same way but with keeping the Return ̃r̂ fixed instead of ŵ̂. The percentage deviation error measure of solution (v, r) is
TABLE I
COMPARISON OF THE PROPOSED METHOD PERFORMANCE WITH BENCHMARK TS ALGORITHM

| Index    | Assets | Median percentage error | Mean percentage error | Time (s) | TS&TokenRing | TS In Ref[2] |
|----------|--------|-------------------------|-----------------------|----------|--------------|--------------|
| Hang Seng| 31     | 1.812                   | 2.2656                | 1154.3   | 1.2181       | 1.1217       |
| DAX      | 85     | 4.21                    | 4.035                 | 2873     | 2.6380       | 3.3049       |
| FTSE     | 89     | 1.2406                  | 1.2959                | 2919     | 1.0841       | 1.6080       |
| S&P      | 98     | 2.3630                  | 2.5068                | 3107     | 1.2882       | 3.3092       |
| Nikkei   | 225    | 1.34635                 | 1.21220               | 5866.2   | 0.6093       | 0.8975       |

the minimum of the x-direction, and y-direction percentage deviations. Finally the Median percentage error and Mean percentage error of the 51 solutions found by the proposed method are reported and compared with reference [2] for each data set.

C. Results & Conclusion

The results for the proposed heuristic algorithm (Tabu Search with Token-Ring technique) with the number of assets in the portfolio \( K = 10 \), a minimum proportion of any asset that must be held (if any is held) \( \epsilon = 0.01 \) and maximum proportion that can be held of any asset (if any is held) \( \delta = 1 \) are shown in Table I.

The results show that the proposed method’s performance is very close to the benchmark approach. For the problems with assets = 98 and 89, the proposed method surpasses the benchmark results in the Mean percentage error, whereas for the rest of the data sets, it delivers sub-optimal results in comparison to the benchmark.

The major drawback of the proposed method in this study is the computation time which was significantly higher than the benchmark as shown in Table I. In conclusion, the proposed method of combining the Token Ring technique with three neighborhood relations failed to surpass the benchmark for some data instances and successfully outperformed it for other data instances. To better analyze the performance of the proposed method, the average error in the first objective (Variance return) and the average error in the second objective (Return) is being reported in Table II. Similarly, the average Return error is calculated. The analysis indicates that the studied method is performing well for the second objective (Return) and relatively poor with the first objective (Risk), that is for small values of lambda (giving more weight to the second objective) the method performs well, i.e., the method is performing well if we are considering portfolios with high return and small risk.

TABLE II
COMPARISON BETWEEN THE ERRORS OF THE FIRST OBJECTIVE AND SECOND OBJECTIVE

| Index     | Assets | First Obj (Risk) error | Second Obj (Return) error |
|-----------|--------|------------------------|---------------------------|
| Hang Seng | 31     | 7.0405                 | 2.4751                    |
| DAX       | 85     | 49.7685                | 1.4008                    |
| FTSE      | 89     | 6.0545                 | 1.8224                    |
| S&P       | 98     | 20.2107                | 3.5328                    |
| Nikkei    | 225    | 9.221                  | 2.2761                    |
TABLE III
THE UNCONSTRAINED EFFICIENT FRONTIER (UEF) AND THE
CONSTRAINED EFFICIENT FRONTIER (CEF) OF EACH DATA SET. THE
RETURN (SECOND OBJ) IS REPRESENTED ON THE Y-AXIS, WHEREAS THE
RISK (FIRST OBJ) IS REPRESENTED ON THE X-AXIS

Fig. 3. UEF of DAX  
Fig. 4. CEF of DAX

Fig. 5. UEF of FTSE  
Fig. 6. CEF of FTSE

Fig. 7. UEF of S&P  
Fig. 8. CEF of S&P

Fig. 9. UEF of Nikkei  
Fig. 10. CEF of Nikkei

REFERENCES
[1] Markowitz, H., 1952. Portfolio Selection. The Journal of Finance, 7(1), p.77.
[2] Chang T-J, Meade N, Beasley JE, Sharaiha YM. Heuristics for cardinality-constrained portfolio optimization. Comput Oper Res 2000;27(13):1271–302.
[3] Schaffer A. Local search techniques for constrained portfolio selection problems. Comput Econ 2002;20(3):177–90. 0104017
[4] Soleimani H, Golmakani HR, Salimi MH. Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm. Expert Syst Appl 2009;36(3):5058–63.
[5] Gaspero LD, Tollo GD, Roli A, Schaefer A. Hybrid metaheuristics for constrained portfolio selection problems. Quant Finance 2011;11(10):1473–87.
[6] J.-S. Chen, J.-L. Hou, S.-M. Wu, and Y.-W. Chang-Chien, “Constructing investment strategy portfolios by combination genetic algorithms,” Expert Systems with Applications, vol. 36, no. 2, Part 2, pp. 3824 – 3828, 2009.
[7] H. Dastkhan, N. S. Ghaneh, and H. Golmakani, “A linguistic-based portfolio selection model using weighted max-min operator and hybrid genetic algorithm,” Expert Systems with Applications, vol. 38, no. 9, pp. 11735 – 11743, 2011.
[8] T.-J. Chang, S.-C. Yang, and K.-J. Chang. “Portfolio optimization problems in different risk measures using genetic algorithm,” Expert Systems with Applications, vol. 36, no. 7, pp. 10529 – 10537, 2009.
[9] Aldaihani, M. M., &amp; Deehani, T. M. (2010). Mathematical models and a tabu search for the portfolio management problem in the Kuwait stock exchange. International Journal of Operational Research, 7(4), 445. doi:10.1504/IJOR.2010.032421
[10] Sawaragi Y, Nakayama H, Tanino T. Theory of multiobjective optimization. In: Bellman R, editor. Mathematics in science and engineering, vol. 176. New York: Academic Press Inc.; 1985.
[11] F. Glover and M. Laguna. Tabu search. Kluwer Academic Publishers, 1997.
[12] Investopedia. 2020. How To Use The Sharpe Ratio To Analyze Portfolio Risk And Return. https://www.investopedia.com/terms/s/sharperatio.asp
[13] Ahmed, S., n.d. The Optimization Process: An Example Of Portfolio Optimization. [online] Ww2.isye.gatech.edu. Available at: https://www2.isye.gatech.edu/ sahmed/isye6669/notes/portfolio. [Accessed 28 November 2020].