Multi objective portfolio optimization for a private equity investment company under data insufficiency condition

Jamil Jalilian
Ph.D Student ,Department of Industrial Management, Science and Research Branch, Islamic Azad University, Tehran, Iran.
jamiljalilian@gmail.com

Reza Ehtesham Rasi
Department of Industrial Management, Qazvin Branch, Islamic Azad University, Qazvin, Iran. Nokhbegan Blvd, Qazvin, Iran, (Corresponding author)
ehteshamrasi@qiau.ac.ir

Mirfeiz Fallah Shams
Associate Professor, Iran, Tehran, Islamic Azad University Central Tehran Branch, Finance department.
mir.Fallah shams@iauctb.ac.ir

Submit: 00/00/2020 Accept: 00/00/2020

ABSTRACT

Selecting an appropriate portfolio making an optimal trade of between the return of assets and the associated risk of them has been always a fundamental challenge for different investors with different types of assets. The problem becomes more complex for an investor investing in private companies of which she doesn’t have enough data to evaluate its return and risk. Furthermore, this type of investment involves selecting more high risk assets which may not meet the risk attitude of the investor. In this study, a bi-objective portfolio optimization model has been developed to determine the best sets of portfolios for a private investing company. Due to the lack of data on private assets, a simulation based approach has been used to estimate the return of different assets as well as their correlations. A Covariance-Based Artificial Bee Colony is applied to solve the model. The results show that optimal portfolios consist both high-risk and low-risk assets.

Keywords:

portfolio, optimization, investment, simulation.

1. Introduction

Portfolio selection is one of the most common problems faced by many investors with different amounts of capital and is one of the most complex problems in the finance world (Qu & Suganthan, 2011). It can include from relatively small portfolios with few stocks, real states, etc. held by common individual investors to large ones with various types of assets managed by expert investors. The key issue in portfolio selection is to choose the best possible combination of assets and to determine their corresponding weights (Mishra, Panda, & Majhi, 2016) .The most well-known and common model presented for portfolio optimization is the one introduced by Markowitz in 1952. The model is based on two main goals of any investor in choosing a portfolio, the first one is to ensure a definite level of returns from the portfolio and the other one is to avoid the risks arise from the losses due to the market fluctuations. He proved that in an ideal world, the investor is seeking for the optimal portfolio, e.g. a portfolio minimizing the risk (keep the risk at a desired level) while maximizing the return of it (Markowitz, 1952). Optimal decision is made based on the existing trade-off between risk and return estimates of any asset. Optimal portfolio selection is considered a difficult problem due to two main reasons. First, investors have to deal with the risks inherent in the selected assets in the portfolio while maximizing the return of the investment in the assets (Mishra, Panda, & Majhi, 2014). Furthermore, there are various requirements that an investor should take into consideration in its investment decision which are not included in Markowitz model (Macedo, Godinho, & Alves, 2017). Therefore many practical constraints have been added to the basic Markowitz model during recent years to make the model more practical in real-world problems (Ponsich, Jaimes, & Coello, 2013). The most common types of these constraints are the lower and upper bound on the invested capital in each asset, the limitations on the number of the purchased assets, and lot size constraint imposed on the purchased stocks of a specific type, i.e. it is required to buy a security of any type in lots (Kumar & Mishra, 2017). In addition to these constraints, recent studies there has been a growing attention to other return and risk measures in portfolio optimization. These functions and constraints can make the main problem nonlinear, non-convex, with integer variables, which consequently convert it to a NP-Hard problem (Saborido, Ruiz, Bermúdez, Vercher, & Luque, 2016). Many studies have tried to model the problem in a form of Linear Problem (LP), Mixed-Integer Programming (MIP) and Mixed-Integer Non-Linear Programming (MINLP) (Mansini, Ogryczak, & Speranza, 2014) and Mixed-Integer Quadratic Programming (MIQP). These models have been proven to be generally NP-Hard problem (Mansini & Speranza, 1999). To tackle these problems meta-heuristic algorithms have been proven to have good performance to find near to optimal solutions (Talbi, 2009).

In this paper we have focused on using heuristic and meta-heuristic methods to solve the portfolio optimization problem with lack of data. The previous studies tackle the portfolio optimization problem for tradable assets such as stocks for which there are abundant amount of data to analyze and compute risks and returns. However, the conditions for non-tradable assets such as private equities are thoroughly different as the main issue in selecting these types of assets is the lack of reliable data to calculate their risks and returns. In this study we use a simulation-based approach to estimate the return of equity of private companies as well as their correlations. Furthermore, we developed our model based on a real case study problem in Iran.

2. Literature Review

Based on the approach of modeling, the related studies can be divided into two main groups. The first group tackle the problem as a single objective model by considering the risk of the portfolio as the only objective function which should be minimized with respect to a definite level of return, while the other group considers the problem as a multi-objective optimization problem (MOOP) aiming to find the best solutions that optimize different objective functions (Subbu, Bonissone, Eklund, Bollapragada, & Chalermkraivuth, 2005),12] use a novel risk measure as the single objective function for the problem and solve it by Particle Swarm Optimization (PSO) algorithm. The Sharpe ratio has been used as the objective function by (Fu, Chung, & Chung, 2013). They adopt a typical genetic algorithm (GA) to optimize the parameters of the technical analysis and a hierarchical genetic algorithm to find the optimal portfolio based on maximizing the Sharpe Ratio. (W.

Chen, 2015) applies an Artificial Bee Colony for a possibilistic portfolio selection in which the risk of portfolio should be minimized while the difference between the expected value of returns and transaction costs must be higher than a predetermined level. In his model, the risk of portfolio should be minimized. (Liao, Chen, Kuo, & Chou, 2015) Introduce a new risk assessment strategy, called fund standardization in order to reduce the complexity of the risk calculation in Markowitz model. They adopt a genetic algorithm to optimize portfolio selection based on maximizing the Sharpe Ratio. Besides return, (Mansini, Ogryczak, & Speranza, 2015)consider the transaction costs for investment process in which the investors incur commissions and other costs. Then the total return of the investment can be expressed as the difference between the returns and the costs.

As it is observed, the nature of portfolio optimization is a multi-objective optimization problem (MOOP). An investor aims to reach optimum levels of different objective function, i.e. the return and the risk. Therefore, various researches have modeled the problem, based on Markowitz model, as a MOOP and used multi-objective optimization techniques to solve it. Particularly, due to the fact that the problem is NP-Hard, the focus of the majority of recent studies has been on applying multi-objective evolutionary algorithms and swarm intelligence to solve the problem (Anagnostopoulos & Mamanis, 2011) compare the performance of five multi-objective evolutionary algorithms, namely Niched Pareto genetic algorithm 2 (NPGA2), non-dominated sorting genetic algorithm II(NSGA-II), Pareto envelope-based selection algorithm (PESA), strength Pareto evolutionary algorithm 2 (SPEA2), and e-multi objective evolutionary algorithm (e-MOEA) for solving a bi-objective portfolio selection problem, i.e. minimization of the risk and maximizing the return with cardinality constraints. A fuzzy multi-objective genetic algorithm is applied by (Bermúdez, Segura, & Vercher, 2012) to tackle the optimal portfolio selection problem with uncertainties in the risk and return. They try to capture the uncertainty of the return and the risk through fuzzy logic and define the risk and the return as trapezoidal fuzzy numbers. Then they use a genetic algorithm to reach the fuzzy ranking strategy for selecting efficient portfolios with cardinality constraint. (A. H. Chen, Liang, & Liu, 2012) use an artificial bee colony algorithm to optimize the portfolio selection. Using a risk aversion parameter, they convert the bi-objective optimization model to a single objective one. The similar approach of conversion is used by (Deng, Lin, & Lo, 2012). They present an improved type of particle swarm optimization to tackle the portfolio optimization problem. (Vijayalakshmi Pai & Michel, 2012) solve a multi-objective constrained futures portfolio problem considering different types of assets. They introduce Herfindahl Index as a measure of portfolio diversification which should be minimized as well as the risk of portfolio and present a combination of multi-objective evolution strategy and multi-objective differential evolution to solve the problem. In a recent study conducted by (Kumar & Mishra, 2017) a novel multi-objective Artificial Bee Colony (ABC) named the Multi-objective Co-variance based ABC (M-CABC) is introduced to solve the multi-objective portfolio problem with cardinality constraints in which the risk and the return of the portfolio should be optimized.

Recently there has been a growing attention to other criteria as the objective function in the model. An invasive weed optimization (IWO) algorithm is used by (Pouya, Solimanpur, & Rezaee, 2016) to solve the multi-objective portfolio problem. Beside risk, the authors define P/E ratio and expert recommendation as other objective functions which should be optimized simultaneously. (Saborido et al., 2016) presents a novel multi-objective evolutionary algorithm to optimize the *Mean-Downside Risk-Skewness* (MDRS) model proposed for portfolio selection. Three objective functions have been taken into account in the model: the expected return, the down-side risk and the skewness of a given portfolio.

Artificial Bee Colony (ABC) algorithm is based on swarm intelligence simulates the intelligent foraging behavior of a honeybee swarm and in recent years was one of the most widely studied and deployed, algorithms. It was firstly developed by Karaboga in 2005 for numerical optimization (Karaboga, 2005) and has been proven to have a superior performance in comparison with other swarm intelligence algorithms (Karaboga & Akay, 2009; Karaboga & Basturk, 2008). For example (Karaboga, 2005) have comprehensively compared the performance of ABC algorithm with different single-based and population-based meta-heuristic algorithms, such as Simulated Annealing (SA), Differential Evolution (DE), PSO and GA on a large set of test problems. Furthermore, (Akbari, Hedayatzadeh, Ziarati, & Hassanizadeh, 2012) have shown that this algorithm has superior performance for unconstrained multi-objective problems. (W. Chen, 2015) proposed an ABC algorithm to solve the cardinality-constrained portfolio optimization problems and compared its performance with Tabu Search (TS), SA, and Variable Neighborhood Search (VNS) on three sets of data test. The results signified the superiority of ABC algorithm in terms of diversity, convergence, and effectiveness among all sets.

In this paper, we aim to use the Multi-objective Co-variance based ABC (M-CABC) presented by (Kumar & Mishra, 2017) to solve the portfolio selection for an investment company holding shares of different private IT companies offering software or hardware services in Iran. The algorithm has been proven to be more efficient to solve portfolio optimization problem compared with ordinary multi-objective ABC (Kumar & Mishra, 2017). Our basic model has some similarities to Markowitz model, however due to the fact that we deal with private companies we use return on equity (ROE) as one of our objective functions. We consider the cardinality constraints as well as specific requirements set by the company in investing in assets. Also, due to the lack of financial data of the companies, a simulation approach has been applied to calculate ROEs of them in the lack of information condition.

3. Mathematical model

In this section, we first introduce the basic Markowitz model for multi-objective portfolio optimization. Then the constraints which should be considered in portfolio selection procedure will be added to the model and, finally the other objective functions which are of our interest will be defined.

The Basic Markowitz Model

Suppose that there are n available securities and x_i is the proportion of the total available capital invested in i^{th} asset, then the problem can be defined as follows:

$$minimize\ f_1(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} x_i \sigma_{ij} x_j \quad (1)$$

$$maximize\ f_2(x) = \sum_{i=1}^{n} r_i x_i \quad (2)$$

In the above equations σ_{ij} is the element of covariance matrix between company i and company j and r_i is the return of asset i. These functions should be optimized by satisfying the following set of constraints

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

$$0 \le x_i \le 1, \forall i \in \{1,2,\dots,n\} \quad (4)$$

Equation (3) ensures that the sum of the fractions invested in securities is 1 meaning that all of money should be invested. The inequalities in (4) state that the fractions must take value between 0 and 1.

3.1. Set of Common Constraints

The majority of studies related to portfolio optimization of stocks used the following constraints in order to take the real world limitations into account in their model (Tapia & Coello, 2007) (Qu & Suganthan, 2011).

a) *Upper bound and lower bound on investments*

This constraint sets limitations on the minimum and the maximum level of investments in each company. That is, if x_i takes a value rather than 0, the following must hold:

$$\alpha \le x_i \le \beta,\ \forall x_i \ne 0 \quad (5)$$

Where α and β are the lower bound and the upper bound on the investment value, respectively.

b) *Cardinality Constraint*

This constraint limits the total number of purchased securities in the portfolio. There are two types of cardinality constraint. In the first one the number of selected assets should be equal to a predetermined value, K, while in the second one the number of purchased assets must be between a lower and upper bound.

In order to set this constraint in the primary model, first we define a binary variable, z_i according to equation (6).

$$z_i = \begin{cases} 1 & if\ x_i > 0 \\ 0 & if\ \ x_i = 0 \end{cases} \quad (6)$$

Now, the first type of cardinality constraint is defined as follows:

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

For the second type, we have to define Z_L and Z_U as the lower bound and the upper bound respectively. So the constraint will take the following form:

$$Z_L \leq \sum_{i=1}^{n} z_i \leq Z_U \quad (8)$$

c) *Round lot constraint*

According to this constraint, the securities must be purchased in lots. That is, the invested money for buying each type of security must be an integer multiple of the price of that security (Equation (9)).

$$x_i A = k y_i \;\&\; k \in Z^+ \quad (9)$$

3.2. Company-Specific Constraints

As mentioned in previous section, we aim to optimize the portfolio assets for a company seeking to invest in private equities, e.g. invest in its subsidiary companies having different levels of maturity, i.e. some of them are startup companies. In order to reach this goal, the company has set two specific requirements for its portfolio of assets. First of all, due to the high risk level associated with startup companies, the share of them in portfolio shouldn't exceed a predetermined value, 0.25. Let define the *j* as index of the startup companies having an age of less than 3 years, $j = \{i | i \in Startup\ Companies\}$. Therefore the constraint can be expressed according to the following equation:

$$\sum_j x_j \leq 0.25 \quad (10)$$

The second issue is using risk free rate bonds as a part of the company's portfolio to modify the liquidity risk and the credit risk of the portfolio. For this purpose we define *k* as risk free rate assets having a zero level of risk and a return less than 15%, e.g. Islamic Bonds, $k = \{i | risk_i = 0, return_i = 0.15\}$.

The corresponding constraint can be written as follows:

$$\sum_k x_k \geq 0.1 \quad (11)$$

3.3. Other Objective Functions

As it was mentioned earlier, in addition to the risk and the return of the portfolio, there has been a growing attention to other criteria such as P/E ratio and Sharpe Ratio as other objective functions which should be optimized, as well. Investors prefer to buy the securities with lower P/E ratio and higher Sharpe Ratio.

a) *Return on Equity*

ROE is used as a measure of profitability of a company and can be used as the return value in Markowitz model. It is calculated by multiplying Return on Asset in the Leverage, e.g. the ratio of assets to equity.

$$ROE = ROA \times Leverage \quad (12)$$

4. Simulation Method for ROE Calculation

With respect to the fact that there is not enough data on private equities a simulation approach has been applied to calculate the return and the elements of the risk. The simulation procedure is based on Latin Hypercube Sampling which is able to generate more viable samples when there is low amount of available data. LHS is a stratified Monte-Carlo (MC) simulation method which was developed by Conover in 1975 for the first time to improve the efficiency of the simple MC sampling (Iman, 2008).

Let X_i be a random variable having a distribution function $F(x)$. The stratification procedure of LHS is accomplished by dividing $F(x)$ into *n* disjoint intervals of equal length, where *n* is the number of computer runs to be made. Through the inverse function $F^{-1}(x)$, these *n* intervals divide the sample space of *x* into *n* intervals. The mapped intervals have the same probability, although they might not have equal length in the *x* space. Consequently, the *x* space is stratified into *n* non-overlapping intervals with equal probabilities. The next step in LHS scheme requires the random selection of a value within each of these intervals on the vertical axis. When these values are mapped through $F^{-1}(x)$, exactly one value will be selected from each of the intervals previously defined on the horizontal axis. This process serves to emulate the pairing of observations in a simple Monte Carlo process (McKay, Beckman, & Conover, 1979) .

5. Multi-objective Co-variance based Artificial Bee Colony Algorithm

Artificial Bee Colony (ABC) algorithm is based on swarm intelligence and in recent years was one of the most widely studied and deployed, algorithms. It was firstly developed by Karaboga in 2005 for numerical optimization (Karaboga, 2005) and has been proven to have a superior performance in comparison with other swarm intelligence algorithms (Karaboga & Akay, 2009; Karaboga & Basturk, 2008) It inspired by the intelligent foraging behavior of a honey bee swarm. In reality, honey bees live in crowded colonies and maintain a complex social organization.

5.1. The Basics of ABC Algorithm

Generally, three groups of bees are considered in the colony; they are employed bees, onlookers and scouts. The ABC algorithm uses these three types of bees which constantly improve the solution. The scout bees have the duty of exploring new locations in the search space. Therefore, the initial generation of all candidate solutions is discovered by scout bees i.e. initial population is created randomly. Thereafter, the nectar of food sources is utilized by joint cooperation of all three types of bees. The employed bees of every generation explore the search space and find the food sources of varied quality. The onlookers will exploit the search space in the proximity of the better food sources only. The employed bees whose food source has been exhausted are initialized randomly in the scout bee phase. These cycles of continual exploration and exploitation results in either of two situations (i) the final solution is searched out (ii) food sources are exhausted. The important parameters and steps of the algorithm is described as follows:

Parameter	Definition
$\overrightarrow{X_m}\{(x_{mi}, i = 1, \dots, d)\}$	candidate solution m^th
$\boldsymbol{D}$	number of dimensions in the problem
$\overrightarrow{Y_m}$	neighborhood of $\overrightarrow{X_m}$
x_{mi}	value of m^th variable on d^th dimension
$\lvert P \rvert$	population size
lb_i	lower bound for the i^th dimension
ub_i	upper bound for the i^th dimension
ϕ_{mi}	random number within a random range (-1, 1)

Algorithm 1. ABC General Scheme

Step1: Initialization phase.

//generate initial population

For each bee m and each dimension i

$x_{mi} = lb_i + random\,(0,1) * (ub_i - lb_i)$ (13)

Step2: Repeat step 2.1 through 2.4 until (termination condition)

2.1: Employed bee phase.

//explore whole search space

For each bee m and any random dimension i and random bee k

$$y_{mi} = x_{mi} + \phi_{mi}(x_{mi} - x_{ki}) \tag{14}$$

$$\boldsymbol{fitness}(\overrightarrow{X_m}) = \begin{Bmatrix} \frac{1}{1+f(\overrightarrow{X_m})} & if\ f(\overrightarrow{X_m}) \geq 0 \\ 1 + abs\left(f(\overrightarrow{X_m})\right) & if\ f(\overrightarrow{X_m}) < 0 \end{Bmatrix} \tag{15}$$

$$\overrightarrow{X_m} = better\ of\ (\overrightarrow{X_m}, \overrightarrow{Y_m}) \tag{16}$$

2,2: Onlooker bee phase.

//exploit food sources containing high nectar amount.

Choose an employed bee according to the probability p_m

$$\boldsymbol{p_m} = \frac{\boldsymbol{fit}(\overrightarrow{X_m})}{\sum_{m=1}^{|P|} \boldsymbol{fit}(\overrightarrow{X_m})} \tag{17}$$

And use equations (14)-(16) again for exploitation

2.3: Scout Bee Phase

//search new points instead of the points which cannot be further evolved

For each bee m, if its fitness function is not improving, use equation (13) to re-create it.

Save the best found solution until now

Step3: Return the saved best solution

5.2. Co-Variance Matrix for Direction Improvement

Deterministic optimization techniques such as Newton-Raphson method use the information of gradient of a function to iteratively find the optimum solution. For an n-dimension function, given we are in the k^th iteration, the new point, x_{k+1}, is calculated according to equation (18)

$$x_{k+1} = x_k - \frac{f(x_k)}{\nabla f(x_k)} \tag{18}$$

Therefore, the search direction $x_{k+1} - x_k$ can be defined as follows:

$$x_{k+1} - x_k = -\frac{\nabla f(x_k)}{\nabla^2 f(x_k)}$$

$$p_k = -\left(\nabla^2 f(x_k)\right)^{-1} . (\nabla f(x_k))$$

$$p_k = -{B_k}^{-1} . g_k$$

Where, p_k, is the search direction, and $\left(\nabla^2 f(x_k)\right)^{-1}$ or ${B_k}^{-1}$ is the symmetric Hessian Matrix in which $H_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j}$. The information of gradients (the first order derivatives) and Hessian (the second order derivative) are necessary for all of deterministic optimization techniques to converge to the local/global optima. However, calculation of gradient and Hessian Matrix is a complex task for high dimensional problems. Therefore, we use covariance matrices as a good way for approximating the gradient g and Hessian matrix H or B^{-1} . Co-variance is used to measure the similarity of movement between any two variables, and the co-variance matrix C can be considered in the same as the inverse of Hessian matrix. The calculation procedure is as follows:

Algorithm 2. Calculation of Co-variance Matrix C.

Input: Data set of N independent samples, each of size d, Input is of the form:

$$\begin{Bmatrix} x_{11} & x_{12} & x_{13} \dots x_{1d} \\ x_{21} & x_{22} & x_{23} \dots x_{2d} \\ \vdots & \vdots & \vdots \ddots \vdots \\ x_{N1} & x_{N2} & x_{N3} \dots x_{Nd} \end{Bmatrix}$$

Output: $C_{d\times d}$, the co-variance matrix

$D_{d\times d}$, a diagonal matrix with eigen values of C

$B_{d\times d}$, an orthogonal matrix with the property $B^TB=BB^T=I$

Step 1: For each dimension $i \in \{1, 2,\dots, d\}$ calculate mean $\bar{x}_i$ as $\bar{x}_i = \frac{1}{N}\sum_{j=1}^{N} x_{ij}$

Step 2: For each dimension i , $j \in \{1, 2,\dots, d\}$ $C_{i,j} = \frac{1}{N}\sum_{k=1}^{N}(x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j)$

Step 3: Eigen decompose C into B and D as $C=BD^2B^T$

There are two phases in every multi-objective algorithm to find the optimal solution: (1) to determine the non-dominated rank of one solution over another and (2) to generate offspring population from the best (non-dominated) parent solutions. In M-CABC the NSGA-2 algorithm is used to calculate the non-dominated rank of solutions and to generate the non-dominated fronts $F_1, F_2, F_3, \dots, F_n$. The procedure for M-CABC algorithm will be as follows:

Algorithm 3. Multi-objective Co-variance Based ABC (M-CABC)

Inputs: Number of Dimensions in Problem: d,

Maximum Number of Cycles: mx

Output: P^* (Pareto-optimal set)

Step 1: Initialization phase

1.1: *bee hive size*=d*10

1.2: *number of employed bees* (e) =*bee hive size*/2

1.3: *number of onlooker bees* (o) =*bee hive size*/2

1.4: ϕ= *random number between* −1 *and* 1

1.5: φ= *random number between* 0 *and* 1

1.6: *Identity matrix of size* $d\times d$

1.7: *B=normalized eigen vectors of C*

1.8: *D=diagonal matrix with square root of eigen values of C*

1.9: *scout bee limit* (*limit*) = d*10

1.10: *generation number (g) =0*

1.11: σ=0.5

1.12: *Generate initial population* P_g *of size e*

Step 2: Perform Non-dominated Sorting of P_g using NSGA-2 ranking procedure and generate non-dominated fronts $F_1, F_2, F_3, \dots, F_n$.

Step 3: Repeat step 3.1 through 3.7 until ($g < mx$)

3.1: Employed bee phase

3.2: Onlooker bee phase

3.3: Scout bee phase

3.4: g=g+1

3.5: hive= {*employed bees* ∪ *onlooker bees*}

3.6: Perform non-dominated sorting on hive using NSGA-2 algorithm and form fronts

3.7: P_g = Best e solutions from hive

Step 4: Return front F_1

The *scout bee limit* (*limit*) is ten times the number of dimension i.e. any bee in the bee hive will be initialized randomly if there is no improvement in the solution since last d×10 iterations. *Generation number* is a counter that holds the record of number of iterations executed so far. σ is the magnitude operator. It is a basic control parameter which is used for controlling the nearness and farness of child solution from parent.

Algorithm 4. Employed Bee Phase of M-CABC

Step1: For each i^{th} employed bee $X_i[x_1^i, x_2^i, \dots, x_k^i, \dots, x_n^i]: i \in \{1, \dots, e\}$ repeat step 1.1 through

step 1.7.

1.1: Select any random solution and any solution *BEST* [$b_1, b_2, \dots, b_k, \dots, b_d$] from front F_1.

1.2: Find a random dimension *dm* to change

1.3: $X_i^{old} = X_i$

1.4: $dfn = x_{dm}^i - x_{dm}^j$

1.5: $dfb = b_{dm} - x_{dm}^i$

1.6: $x_{dm}^i = x_{dm}^i + \phi(dfn) + \varphi(dfb)$

1.7: $X_i = pareto\ optimal\ selection\ (X_i^{old}, X_i)$

In employed bee phase we have used two distance measures, *dfn* (distance from any random employed bee in the search space) and *dfb* (distance of employed bee from any solution BEST lying in the best front i.e. the pareto optimal front F_1 The proper combination of the aforementioned two distance estimates i.e. dfn and dfb are used in the step 1.6 of Algorithm 4. This formula helps in getting an extremely balanced and guided exploration.

Algorithm 5. Pareto Optimal Selection

Input: Two solutions $A[a_1, a_2, \ldots, a_d]$, $B[b_1, b_2, \ldots, b_d]$

Output: $O*$

Step 1: If $\forall j \in \{1, \ldots, k\}\, f_j(\vec{A}) \le f_j(\vec{B})$&& ($\exists j \in \{1, \ldots, k\} f_j(\vec{A}) < f_j(\vec{B})$) //solution $f(\vec{A})$ is as good as $f(\vec{B})$ for all k objective functions and $f(\vec{A})$ is strictly better than $f(\vec{B})$ for at least one objective then return (A) else return (B).

Algorithm 6. Onlooker Bee Phase of M-CABC.

Step 1: For each front F_j : $1 \le j \le \lceil log_2 o \rceil$

repeat step 1.1 through 1.5

1.1: Find C, B and D for the front F_j

1.2: Find m, the mean of solutions on front F_j

1.3: $\lceil nof = e/2 \rceil$ // *number of onlooker bees assigned to this front*

1.4: *e=e-nof* // *remaining onlooker bees for subsequent fronts*

1.5: For each i^{th} onlooker bee Z_i : $i \in \{1, \ldots, nof\}$

1.5.1. Create a random vector $r_{d\times 1}: r_i \in (0,1)$

1.5.2. $Z_i = m + \sigma BDr$

Step 2: Reset *e*

Algorithm 7. Scout Bee Phase of M-CABC

Step 1: For each i^{th} bee in hive $S_i: i \in \{1, \ldots, (o+e)\}$

Repeat step 1.1

1.1. If the fitness S_i is not improving since limit number of iterations then re-initialize S_i randomly.

In traditional ABC algorithm, more number of onlooker bees are moved towards better solutions. In M-CABC half of the onlooker bees (*nof*) are guided to the current best front. From the remaining bees (*e-nof*) half of them are sent to the next available front and so on. In this way $\lceil log_2 o \rceil$ numbers of fronts are explored by onlooker bees. The formation of onlooker bees is done using the co-variance matrices in the formula $Z_i = m + \sigma BDr$. A solution, whose fitness is not increasing since last limit number of iterations, is reinitialized in the scout bee phase. The diagonal elements of matrix *d* represent the actual length of the axes of d dimensional distribution ellipsoid. In this way the concept of co-variance helps in coordinating a regulated exploitation of the mean solution *m* along a virtual ellipsoid.

5.3. Chromosome Representation

For an *n* security problem, each solution is represented by an *n* dimensional vector of real numbers. The constraints in equation 3-5 and equations 8-9 are considered in the optimization algorithm. Thus, a solution i.e. (a portfolio of the shares of the companies) is represented by the vector $P[p_1, p_2, \ldots, p_n]$ possessing the following properties:

(a) $\forall i \in \{1, \ldots, n\}\; p_i \in \{\Re^+ \cup 0\}$

(b) $\sum_{i=1}^{n} p_i = 1$

(c) $\forall i \in \{1, \ldots, n\}\; p_i = 0\ or\ \alpha \le p_i \le \beta$

(d) $p_i A = k \times price_i$

(e) Total number of non-zero p_i is limited in the range $[Z_L, Z_U]$

Where *A* is the total available capital to invest and $price_i$ is the price of i^{th} company's share. Thus we can define the money invested in i^{th} company, *I*, as follows:

$$I = \begin{Bmatrix} 0 & If\ p_i = 0 \\ p_i.A & If\ p_i > 0 \end{Bmatrix}$$

5.4. Generating Initial Population

The initial population is an $e \times n$ matrix, where *e* and *n* are the number of population and the dimension of the problem, respectively. The numbers are generated randomly in the range of [0,1). The procedure for generating initial population is described in Algorithm 8. The generated population of initial solutions may not satisfy all set of constraints. We use the procedure in Algorithm 9.

Algorithm 8. Initial Population Generation

Inputs: Number of securities: n

Population Size: e

Output:

Initial Population:

$$P=\begin{Bmatrix} p_1^1 & p_2^1 & p_3^1 \dots p_n^1 \\ p_1^2 & p_2^2 & p_3^2 \dots p_n^2 \\ \vdots & \vdots & \vdots \ddots \vdots \\ p_1^e & p_2^e & p_3^e \dots p_n^e \end{Bmatrix}$$

Step 1: For each population member p^j, $j \in \{1,2,\dots,e\}$ and for each dimension $i \in \{1,2,\dots,n\}$ repeat step 1.1

1.1: p_i^j = *random number* [0,1)

Step 2: For each portfolio P^j, $j \in \{1,2,\dots,e\}$ repeat step 2.1

2.1: Clean P^j

Algorithm 9. Clean Chromosome.

Inputs: Un-constrained portfolio: $P \{p_1, p_2, \dots, p_n\}$

Minimum number of securities in portfolio: Z_L

Maximum number of securities in portfolio: Z_U

Minimum investment in any security in portfolio: α

Maximum investment in any security in portfolio: β

Output: Cleaned portfolio *P*, satisfying all constraints.

Step 1: nz = *non-zero securities in P*

Step 2: If $nz < Z_L$ then Re-initialize $(Z_L - nz)$ number of zeroed securities to *random* number (0,1) and set $nz = Z_L$

Step 3: If $nz > Z_U$ then Re-initialize $(nz - Z_U)$ number of non-zero securities to 0 and set $nz = Z_U$

Step 4: For each dimension $i \in \{1,2,\dots,n\}$ if $p_i > \beta$ then $p_i = \beta$

Step 5: Set *vti*=0

Step 6: For each dimension $i \in \{1,2,\dots,n\}$ $vti = vti + p_i$

Step 7: $raf = 1 - (nz \times \alpha)$

Step 8: For each dimension $i \in \{1,2,\dots,n\}$ If $p_i \mathrel{!}= 0$ then $p_i = \alpha + (\frac{p_i}{vti} \times raf)$

6. Finding and Analysis

In this section, we will try to solve the defined problem by means of simulation technique combined with the meta-heuristic algorithm, M-CABC. Afterwards, we will compare M-CABC and NSGAII to investigate their performance in solving the problem based on predefined performance measures. The importance of finding closed-form solutions and the consequent search for simpler models, and when required, complex models provide stronger emphasis for computationally-intensive methods such as Monte Carlo simulations, numerical approximations to differential equations (ordinary and stochastic), population based approaches, metaheuristic of uncertain possibilities in a search space etc. provide just a gist of a variety of possibilities that computational sciences tend to offer. Addressing the availability of such high-valued computing techniques, and to overcome challenges faced by deterministic optimization methods, this work does focus towards having a look at stochastic-search based optimization routines towards optimal asset allocation strategies. The Excel software used to calculate the input data. To determine the portfolio for each year, genetic, artificial ant colony and colony of bee algorithms were analyzed using MATLAB 2018. For the analysis of algorithms, Minitab software was used. After performing the steps listed in the previous parts and collecting the required data from mature and startup companies for the years between 2005 and 2020, and solving Markowitz model through artificial bee colony, ant algorithm using MATLAB software for each selected year. Each basket was included in the basket of stocks and weighing per share and for each portfolio returns, risk and sharp measure calculated. The return and the risk of companies have been calculated by @risk simulation software. First the 5-year ROA and Leverage of companies has been calculated based on their financial statements in 5 years. Given that this amount of data is not enough for estimating ROA and Leverage, a simulation approach has been applied to estimate these two parameters.

The simulation generates 10000 different scenarios for each of these parameters using Latin Hypercube sampling (LHS) method. In this method the given cumulative probability distribution used for generating random numbers is divided to equal intervals from which the random numbers are selected and generated (the number of intervals are equal to the number of scenarios). The simulation process leads to generation of two sets of distributions for ROA and Leverage, respectively (see Figure 1). The mean of these distributions are used in Equation (12) to estimate the ROE of the companies. As it was mentioned before, ROE has been used to calculate the return of portfolio.

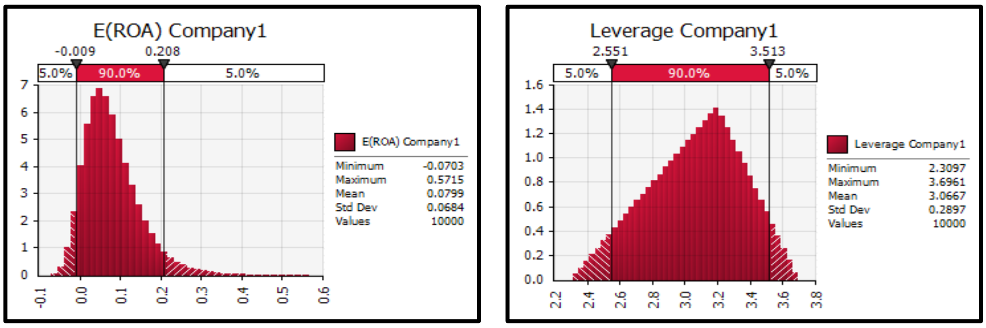

Figure 1-The distribution of ROA and Leverage for a given company based on 10000 scenario generation using @Risk simulation software

In order to calculate the risk of portfolio the correlation matrix for the companies should be defined. The @Risk simulation software was used to calculate the elements of this matrix, σ_{ij}. Finally, the optimal portfolio regarding two defined objective functions and can be obtained by means of multi-objective Co-variance based Artificial Bee Colony algorithm.

Table 1- The results of simulation of ROA and LEVERAGE of the companies. ROE is calculated through multiplying ROA by LEVERAGE

ASSET NAME	ROA	LEVERAGE	ROE
COMPANY 1(M)*	0.079	3.69	0.294
COMPANY 2(M)	0.130	2.64	0.344
COMPANY 3(M)	0.128	2.60	0.333
COMPANY 4(M)	0.160	1.86	0.299
COMPANY 5(S)*	0.164	2.84	0.468
COMPANY 6(S)	0.405	2.72	0.471
COMPANY 7(S)	0.189	2.92	0.351
ISLAMIC BOND	-	-	0.15 (rate of return)
	* (M): Mature Company	(S): Startup Company	

The minimum and maximum level for the number of selected assets in the portfolio, Z_L andZ_U, has been set as 4 and 6, respectively. The lower and upper bound for investing in each asset class has been determined separately. The lower bound for mature companies (Company 1 to Company 4) and startup companies (Companies 5 to 7) are set 0.1 and 0.05, respectively and the upper bounds to invest in them are 0.5 and 0.25. As mentioned in mathematical model section, the sum of high risk companies (startup companies) weights shouldn't exceed 0.25 of the entire portfolio. Also the Islamic Bond should always be selected in the portfolio for which the lower bound is considered as 0.1 and the upper bound is 1.

Table 2- The lower bound and upper bound for investing in each asset class

ASSET NAME	LOWER BOUND	UPPER BOUND
COMPANY 1	0.1	0.5
COMPANY 2	0.1	0.5
COMPANY 3	0.1	0.5
COMPANY 4	0.1	0.5
COMPANY 5	0.05	0.25

ASSET NAME	LOWER BOUND	UPPER BOUND
COMPANY 6	0.05	0.25
COMPANY 7	0.05	0.25
ISLAMIC BOND	0.1	1

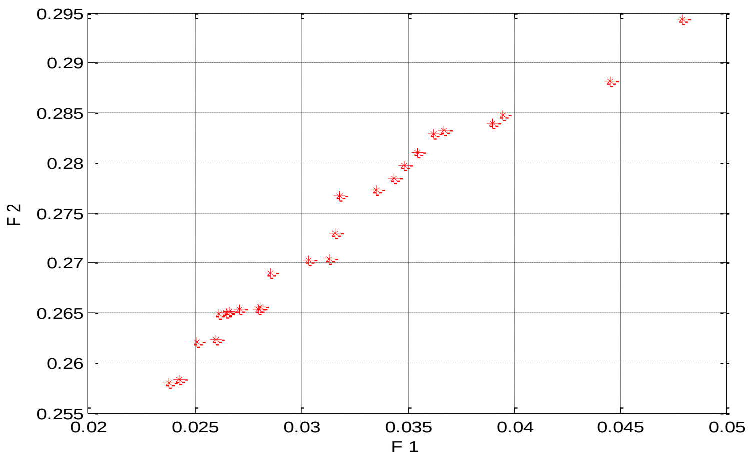

Figure 2. The Pareto Optimal solutions for the bi-objective optimization problem

According to Figure 2, it is observed that there are 25 non-dominated solutions for the problem. The F1 and F2 axis indicate the risk and the return of portfolio, respectively. Figure (3) shows the participation percentage of each asset class in Pareto Optimal portfolio. Below table one of these 25 optimal solutions is shown with its respective risk and return. As it is observed in this solution, the number of selected assets is 6 indicating that we have 6 different types of assets in our selected portfolio having a return rate of 28%.

Solution	Risk	Return
[0.149 0.201 0.201 0.201 0.086 0.00 0.00 0.160]	0.035	0.28

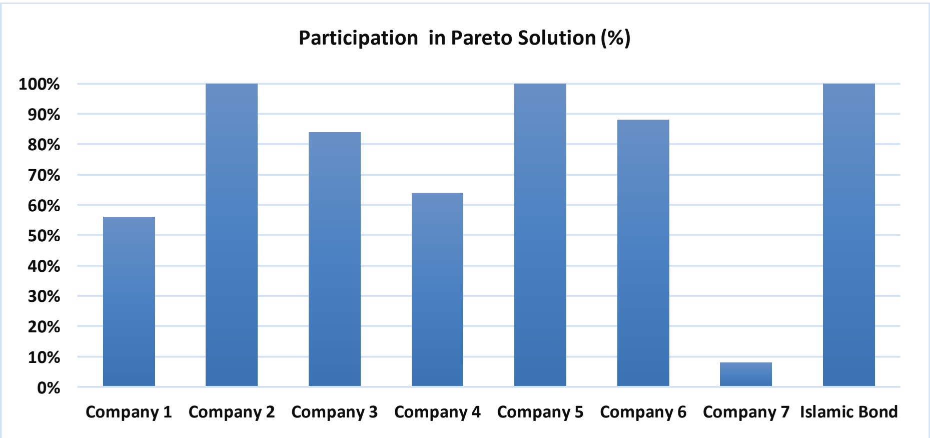

Figure 3- The percentage of asset participation in pareto optimal solutions

As it is observed, an optimal portfolio from both risk and return's point of view is formed by a combination of mature and startup companies. That is, one company from each type (mature and startup) should be selected to reach the optimal solutions. As it is observed Company 2 as a mature company along with company 5 as a startup one are selected in all optimal portfolios. On the other hand, Company 7 is the least desirable asset to invest in due to its high level of risk and moderate return.

6.1. Comparison of the Performance of M-CABC with NSGAII

In order to evaluate the performance of M-CABC algorithm to solve our model, a comparison has been made between it and another multi-objective meta-heuristic algorithm, NSGAII. Since the actual optimal solutions (the optimal pare to frontier) of our problem have not been determined in advance, we analyze their performance based on 3 measures: running time, the number of reached pareto solutions, and the value of individual objective functions, instead of usual performance measure such as variance or other distance measures requiring exact solutions to be compared with those reached by the algorithms.

Table 3) indicates the parameters set for performance comparison. Population numbers and the number of generations for both algorithms have been set equal in order to make the comparison as fair as possible. The other parameters of NSGAII have been determined based on the study of (Anagnostopoulos & Mamanis, 2011) and those of M-CABC was tuned according to (Kumar & Mishra, 2017).

As it is observed M-CABC is able to reach solutions much faster than NSGAII; however the latter is capable of reaching more number of Pareto optimal solutions, i.e. 170 against 15. Nevertheless, we have tried to compare the performance of these algorithms based on their ability to reach the best value of objective functions. To do so, the algorithms were run 10 times and were compared for their return and risk functions, respectively.

Table 3- Parameters of M-CABC & NSGAII used to compare their performance

Parameter	M-CABC	NSGAII
Number of Population of each generation, N_{pop}	200	200
Number of generations (iterations)	100	100
$P_{crossover}$	-	0.9
$P_{mutation}$	-	$1/N_{pop}$
Number of Employed bees	50	-
Number of Onlooker bees	50	-

Algorithm	Average time to reach solution (s)	Average number of pareto solutions
M-CABC	14.60	15.08
NSGAII	84.52	170.54

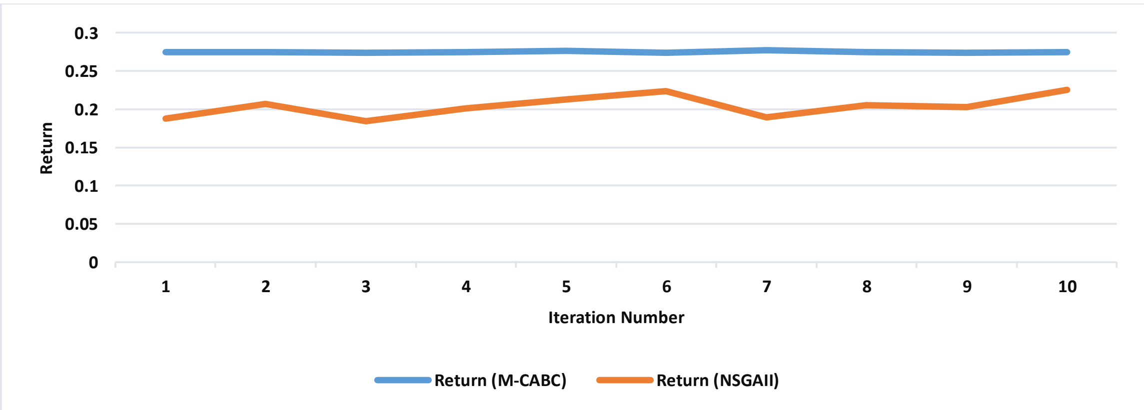

Figure 4-Comparison of M-CABC algorithm with NSGAII based on their achieved returns in 10 runs

Figure 4) indicates the results of comparison between performances of two algorithms based on their values of return function. As it can be observed M-CABC algorithm averagely, is able to reach portfolios with better rates of return.

The similar results can be observed in Figure 5) indicating the average rates of risk of portfolios in Pareto frontier achieved by these two algorithms. M-CABC also possesses a better performance in this aspect and obtains portfolios associating with lower levels of risk.

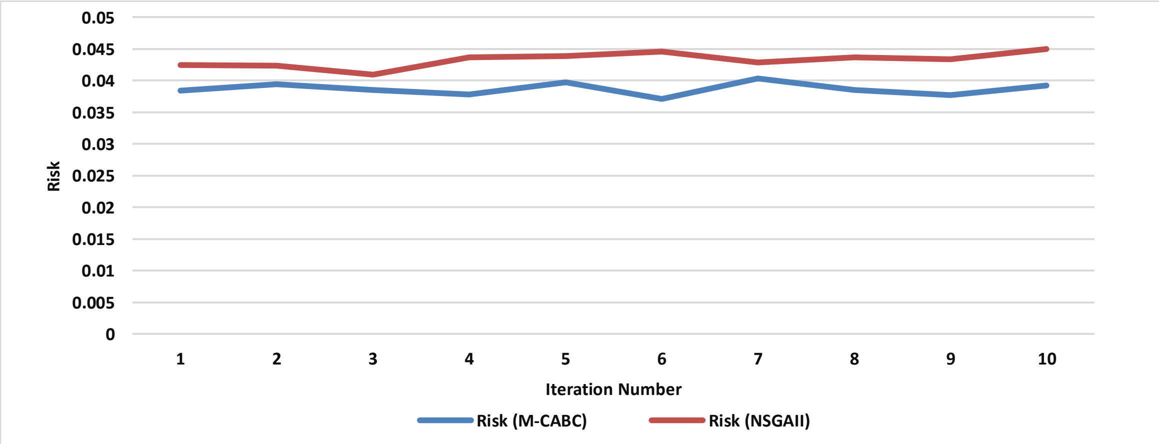

Figure 5- Comparison of M-CABC algorithm with NSGAII based on their achieved risks in 10 runs

7. Conclusion and Further Research

In this paper a portfolio optimization problem has been introduced in which we were aimed to determine the best investment portfolio for a private investment company so as to minimize the risk of portfolio and simultaneously maximize the expected return of it. Therefore a bi-objective optimization model with different sets of constraints was developed. The constraints include common sets of constraints in portfolio optimization literature such as the number of assets limits as well as specific ones defined based on special considerations of the company. The first set of special constraints set limitation on the use of startup companies (high risk companies) in the portfolio. Another set of special constraints used to modify overall risk of the portfolio is about considering a lower limit on the use of zero risk assets in the portfolio. Due to the lack of data on private companies, a simulation based approach has been applied to estimate the return of companies and also the correlation among them. Then the estimation of return on equity of the companies has been used as their return. As the model is considered a NP-Hard problem, we use the Covariance-Based Artificial Bee Colony algorithm. The result of algorithm indicates the optimal portfolios contain both mature and startup companies due to the low risk of the former and high return of the latter. Also, a comparison has been made in order to pit the performance of the algorithm against another multi-objective meta-heuristic algorithm, NSGAII according to 3 performance index inclusing running time, the number of reached pareto-optimal solutions, and the values of objective functions. While M-CABC can perform faster relative to NSGAII, the latter find more pareto-optimal solutions, however the quality of the solutions reached by M-CABC is much higher than those of NSGAII.

The future works can be conducted on determining optimal portfolios with higher degree of variety. Also, in this study we have assumed that the investor has a pre-determined amount of capital to invest. It is possible to relax the assumption this assumption and allow the investor to use debts in its investment process. This will increase the options of the investor and consequently the complexity of the solution space of the problem since the weights can take negative values. Furthermore, we can consider the amount of capital as a function of the return of portfolio.

References

1) Akbari, R., Hedayatzadeh, R., Ziarati, K., & Hassanizadeh, B. (2012). A multi-objective artificial bee colony algorithm. Swarm and Evolutionary Computation, 2, 39-52.
2) Anagnostopoulos, K. P., & Mamanis, G. (2011). The mean–variance cardinality constrained portfolio optimization problem: An experimental evaluation of five multiobjective evolutionary algorithms. Expert Systems with Applications, 38(11), 14208-14217.
3) Bermúdez, J. D., Segura, J. V., & Vercher, E. (2012). A multi-objective genetic algorithm for cardinality constrained fuzzy portfolio selection. Fuzzy Sets and Systems, 188(1), 16-26.
4) Chen, A. H., Liang, Y.-C., & Liu, C.-C. (2012). An artificial bee colony algorithm for the cardinality-constrained portfolio optimization problems. Paper presented at the Evolutionary Computation (CEC), 2012 IEEE Congress on.
5) Chen, W. (2015). Artificial bee colony algorithm for constrained possibilistic portfolio optimization problem. Physica A: Statistical Mechanics and its Applications, 429, 125-139.
6) Deng, G.-F., Lin, W.-T., & Lo, C.-C. (2012). Markowitz-based portfolio selection with cardinality constraints using improved particle swarm optimization. Expert Systems with Applications, 39(4), 4558-4566.
7) Fu, T.-c., Chung, C.-p., & Chung, F.-L. (2013). Adopting genetic algorithms for technical analysis and portfolio management. Computers & Mathematics with Applications, 66(10), 1743-1757.
8) Iman, R. L. (2008). Latin hypercube sampling: Wiley Online Library.
9) Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Retrieved from
10) Karaboga, D., & Akay, B. (2009). A comparative study of artificial bee colony algorithm. Applied Mathematics and Computation, 214(1), 108-132.
11) Karaboga, D., & Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. Applied soft computing, 8(1), 687-697.
12) Kumar, D., & Mishra, K. (2017). Portfolio optimization using novel co-variance guided Artificial Bee Colony algorithm. Swarm and Evolutionary Computation, 33, 119-130.
13) Liao, B.-Y., Chen, H.-W., Kuo, S.-Y., & Chou, Y.-H. (2015). Portfolio Optimization Based on Novel Risk Assessment Strategy with Genetic Algorithm. Paper presented at the Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on.
14) Macedo, L. L., Godinho, P., & Alves, M. J. (2017). Mean-semivariance portfolio optimization with multiobjective evolutionary algorithms and technical analysis rules. Expert Systems with Applications, 79, 33-43.
15) Mansini, R., Ogryczak, W., & Speranza, M. G. (2014). Twenty years of linear programming based portfolio optimization. European Journal of Operational Research, 234(2), 518-535.
16) Mansini, R., Ogryczak, W., & Speranza, M. G. (2015). Portfolio Optimization with Transaction Costs Linear and Mixed Integer Programming for Portfolio Optimization (pp. 47-62): Springer.
17) Mansini, R., & Speranza, M. G. (1999). Heuristic algorithms for the portfolio selection problem with minimum transaction lots. European Journal of Operational Research, 114(2), 219-233.
18) Markowitz, H. (1952). Portfolio selection. The journal of finance, 7(1), 77-91.
19) McKay, M. D., Beckman, R. J., & Conover, W. J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics, 21(2), 239-245.
20) Mishra, S. K., Panda, G., & Majhi, B. (2016). Prediction based mean-variance model for constrained portfolio assets selection using multiobjective evolutionary algorithms. Swarm and Evolutionary Computation, 28, 117-130.
21) Mishra, S. K., Panda, G., & Majhi, R. (2014). A comparative performance assessment of a set of multiobjective algorithms for constrained portfolio assets selection. Swarm and Evolutionary Computation, 16, 38-51.
22) Ponsich, A., Jaimes, A. L., & Coello, C. A. C. (2013). A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications. IEEE Transactions on Evolutionary Computation, 17(3), 321-344.
23) Pouya, A. R., Solimanpur, M., & Rezaee, M. J. (2016). Solving multi-objective portfolio optimization problem using invasive weed

optimization. Swarm and Evolutionary Computation, 28, 42-57.

24) Qu, B. Y., & Suganthan, P. N. (2011). Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods. Engineering Optimization, 43(4), 403-416.
25) Saborido, R., Ruiz, A. B., Bermúdez, J. D., Vercher, E., & Luque, M. (2016). Evolutionary multi-objective optimization algorithms for fuzzy portfolio selection. Applied soft computing, 39, 48-63.
26) Subbu, R., Bonissone, P. P., Eklund, N., Bollapragada, S., & Chalermkraivuth, K. (2005). Multiobjective financial portfolio design: A hybrid evolutionary approach. Paper presented at the Evolutionary Computation, 2005. The 2005 IEEE Congress on.
27) Talbi, E.-G. (2009). Metaheuristics: from design to implementation (Vol. 74): John Wiley & Sons.
28) Tapia, M. G. C., & Coello, C. A. C. (2007). Applications of multi-objective evolutionary algorithms in economics and finance: A survey. Paper presented at the Evolutionary Computation, 2007. CEC 2007. IEEE Congress on.
29) Vijayalakshmi Pai, G., & Michel, T. (2012). Metaheuristic optimization of marginal risk constrained long-short portfolios. Journal of Artificial Intelligence and Soft Computing Research, 2.