Adv. Math. Fin. App., 2024, **9**(2), P. 424-447

Advances in Mathematical Finance & Applications
www.amfa.iau-arak.ac.ir
Print ISSN: 2538-5569
Online ISSN: 2645-4610
Doi: 10.22034/amfa.2022.1950768.1688

Original Research

Portfolio Optimization Considering Cardinality Constraints and Based on Various Risk Factors Using the Differential Evolution Algorithm

Behnaz Ghadimi[a] , Mehrzad Minooei [b,]*, Gholamreza Zomorodian[c], Mirfeiz Fallah Shams[c]

[a]*Department of Financial Management, Central Tehran Branch, Islamic Azad University, Tehran, Iran.*
[b]*Department of Industrial Management, Central Tehran Branch, Islamic Azad University, Tehran, Iran.*
[c]*Department of Business Management, Central Tehran Branch, Islamic Azad University, Tehran, Iran.*

ARTICLE INFO

Article history:
Received 2022-01-26
Accepted 2022-07-21

Keywords:
Portfolio optimization,
Cardinality constraint,
Differential evolution algorithm, Value-at-risk,
Conditional value-at-risk

ABSTRACT

As the main achievement of the modern portfolio theory, portfolio diversification based on risk and return has attracted the attention of many researchers. The Markowitz mean-variance problem is a convex quadratic problem turned into a mixed-integer quadratic programming problem when incorporating cardinality constraints. Due to the high number of stocks in a market, this problem becomes an NP-hard problem. In this paper, a metaheuristic approach is proposed to solve the portfolio optimization problem with cardinality constraints using the differential evolution algorithm, while it is also intended to improve the solutions generated by the algorithm developed. In addition, variance, value-at-risk, and conditional value-at-risk are assessed as risk measures. Candidate models are solved for 50 top stocks introduced by the Tehran Stock Exchange by considering the cardinality constraints of not more than five stocks within the portfolio and 24 trading periods. Finally, the obtained results are compared with the results of genetic algorithm. The results show that the proposed method has reached the optimal solution in a shorter time.

1 Introduction

Diversification of investment options is an effective method introduced in modern portfolio theory to reduce risk. Markowitz proposed a quantitative approach to create attractiveness and investment diversity in different firm stocks [1]. He introduced the mean-variance model and used variance as a criterion to assess economic risk [2]. In this approach, the correlation among several stocks was considered besides the characteristics of a stock. Although the use of variance as a risk measure has been a cornerstone in the history of the portfolio theory, some alternative measures such as variance with skewness [3], semi-variance [4], mean absolute deviation [5], Max-Min model [6], value-at-risk [7], and conditional value-at-risk [8] have been proposed as portfolio optimization models in the finance

* Corresponding author. Tel.: +989128124145
E-mail address: mminoei2@gmail.com

literature. Each of these measures can show different efficiencies from itself with respect to the target market.

On the other hand, Markowitz's traditional portfolio optimization model fails to represent the complexities of real-world problems faced by decision-makers. These complexities appear as constraints such as boundary constraints, cardinality constraints, transaction costs, and transaction lots. Unconstrained portfolio optimization is an ordinary convex quadratic programming problem that can effectively be solved by exact methods such as linear and quadratic programming. Many researchers presented an extensive study on various issues of portfolio optimization, such as historical evolution and the use of exact methods. Nevertheless, as demonstrated by [9], this problem is converted into a mixed-integer quadratic programming problem with high dimensions and consequently an NP-hard problem when practical constraints such as cardinality constraints are added to the problem. Therefore, researchers paid particular attention to developing approximate methods such as metaheuristic algorithms. Metaheuristic methods aim to resolve shortcomings of classic optimization methods and ensure better results by comprehensive random search [10]. This research focuses on using a metaheuristic difference evolutionary algorithm to solve a realistic version of the portfolio optimization problem by assessing a set of stocks presented in the Tehran Stock Exchange. The research is intended to respond to the question of whether the optimal portfolio is obtained by solving the model using information predicted by metaheuristic models considering constraints on the number of assets within the portfolio, the number of desirable assets to be included in the portfolio, and floor and ceiling constraints for each asset. In addition, variance measures, value-at-risk, and conditional value-at-risk are assessed as risk factors for the portfolio optimization model.

The rest of the paper is organized as follows. Section 2 reviews the literature on portfolio optimization problems based on risk measures and solution approaches. In section 3, the model formulation and solution approach are described. Section 4 presents the methodology along with data and the analysis process. The results and analyses are presented in Section 5. Finally, Section 6 concludes the paper.

2 Literature Review

2.1 Risk Measure in the Portfolio Optimization Model

The purpose of solving portfolio optimization is to determine the weights for a set of assets within the portfolio to meet a specific return level for the investor besides minimizing the portfolio's risk. In the Markowitz mean-variance model, the risk of a portfolio is measured by the variance of the stock price. Despite its wide applications, this approach may lead to an inadequate forecast of portfolios regarding the asymmetric return distribution because this model assumes that the expected return has an asymmetric multivariate normal distribution. Therefore, Markowitz proposed a model based on semi-variance appropriate to deal with asymmetric distribution of return. Researchers suggested another approach by considering skewness in the mean-variance model in order to be able to describe the characteristics of the return distribution. Positive skewness of the portfolio return reduces negative aspects of risks and is desirable for investors. The mean-variance model includes a quadratic objective function with linear constraints. Konno and Yamazaki proposed the mean absolute deviation of stock return in all periods as an alternative risk measure to overcome problems caused by the quadratic structure of the mean-variance model. Their model calculates the covariance matrix with a low computational burden, and it can easily be updated when new data are added. In addition, risk modeling based on absolute deviation converts the model to a linear parametric programming model and makes portfolio optimization simpler. Nevertheless, according to the analysis conducted by [11], although the estimation error of both mean-variance model and mean-absolute deviation is high when dealing

with small samples and aggressive investors, the mean-variance model presents a lower estimation error for small samples and conservative investors. Value-at-risk, introduced by Jorion [12], is a risk measure describing the potential maximum loss (minimum return) for a portfolio based on profit and loss distribution in a target horizon. Despite positive features such as considering the return distribution, this measure suffers from undesirable mathematical properties such as lack of additivity and convexity. To resolve this problem, Rockefeller and Uryasev [8] introduced conditional value-at-risk to reduce excessive loss risks based on value-at-risk, which indicated an asymmetric risk measure. This method benefits from additivity that allows measuring portfolio risk and convexity of the conditional value-at-risk that makes the optimization easier. In this paper, mean-variance, mean-VaR[1], and mean-CVaR[2] models are assessed as basic models. Table 1 represents portfolio optimization models based on risk measures proposed by researchers.

Table 1: Portfolio Optimization Models Based on Risk Measures

Model	Author	Year	Structure
Mean-variance	Markowitz	1952	Quadratic
Mean-variance with skewness	Samuelson	1958	Quadratic
Mean-semi-variance	Markowitz	1959	Quadratic
Mean-absolute deviation	Konno and Yamazaki	1991	Linear
Mean-value-at-risk	Jorion	1997	Linear
Mean-conditional value-at-risk	Rockefeller and Uryasev	2000	Linear

2.2 Portfolio Optimization Problem

A literature review on the portfolio optimization problem shows approximate and exact efforts made to solve this problem. An unconstrained portfolio optimization problem is a quadratic programming problem that can be solved effectively by exact methods such as linear and quadratic methods. Nevertheless, as Bienstock [9] has shown, this problem is converted into a mixed-integer quadratic programming problem when practical constraints such as cardinality constraints are added to the problem. Therefore, researchers are developing approximate methods such as machine learning and metaheuristic algorithms. As an evolution of computational learning theory, among the artificial intelligence methods, machine learning algorithms such as artificial neural networks [13] and k-means clustering [14] have been less frequently applied in the literature, compared with metaheuristic algorithms which are based on either single-solution or population-based methods. Heuristic algorithms based on single-solution methods, such as simulated annealing algorithm [15], tabu-search algorithms [16], and greedy randomized adaptive search procedure (GRASP) [17], basically try to improve a single solution during the implementation. Although working on a single solution cannot use several search points, the algorithm is extremely fast because a solution improves numerously. On the other hand, population-based heuristic algorithms, which work on several solutions during the algorithm iterations, have been more frequently applied in portfolio optimization. Population-based algorithms can be categorized into two groups, namely evolutionary algorithms and swarm intelligence algorithms. Inspired by natural evolution, evolutionary algorithms evolve a population of chromosomes over time using selection, crossover, and mutation strategies to achieve better solutions. Various evolutionary algorithms have extensively been used to solve portfolio optimization problems. Although the concept of evolutionary algorithms was introduced in the mid-1970s, the first application of this approach to solve assets portfolio optimization problems was related to Chang et al. [18] at the beginning of this century. Ehrgott et al. [19] proposed a model for the mean-variance portfolio optimization problem

[1] Mean – Value-at-risk
[2] Mean – Conditional value-at-risk

and designed three algorithms, namely genetic algorithm, tabu search, and simulated annealing algorithm, to solve the model. The genetic algorithm provided better results in terms of utility and standard deviation. Lin and Liu [20] presented a model for the mean-variance portfolio optimization problem with minimum transactions and used the genetic algorithm to solve the model. The results indicated that the genetic algorithm could provide near-optimal solutions in a reasonable and short time. Mishra et al. [21] suggested using multi-objective bacteria foraging optimization for mean-variance assets portfolio optimization problem with boundary and cardinality constraints and compared the proposed algorithm with three multi-objective evolutionary algorithms concluding the superiority of the proposed approach over others. Yin et al. [22] proposed particle swarm optimization for portfolio optimization based on a heterogeneous multiple population strategy using the Markowitz mean-variance portfolio selection model. Their computational results demonstrated that the proposed algorithm performed powerfully, particularly for large-scale problems. Macedo et al. [23] compared two multi-objective evolutionary algorithms in terms of high volume performance index and technical analysis-based strategies. Their results indicated that NSGA-II outperforms SPEA-II. Kalayci et al. [24] cardinality constrained portfolio optimization using an artificial bee colony algorithm. Their study demonstrated the high efficiency of the artificial bee colony algorithm compared with other algorithms. Kalayci et al. [25] proposed a hybrid algorithm combining ant colony and genetic algorithms for cardinality constrained portfolio optimization and assessed the model for five reference datasets. Yuen et al. [26]formulated the portfolio optimization problem using market index-tracking with cardinality constraints and solved the proposed model using a metaheuristic approach. Soroush et al. [27] evaluated the portfolio optimization problem using a teaching-learning-based optimization (TLBO) algorithm based on the mean-variance model and conditional value-at-risk model. The results indicated that TLBO performs better than other algorithms to find the efficient frontier and portfolio optimization. Tehrani et al. [2] assessed the portfolio optimization problem using the krill herd metaheuristic algorithm based on the mean-variance and mean-semi-variance models and expected downfalls. The results of their study indicated that the krill herd algorithm performs more satisfactorily in finding efficient frontier and portfolio optimization compared with conventional algorithms.

3 Portfolio Optimization with Cardinality Constraint

The general portfolio optimization model is defined as follows.

$$max \quad \frac{\sum_{i=1}^{N} w_i \mu_i - R_f}{risk_p} \tag{1}$$

$$subject\ to: \quad \sum_{i=1}^{N} w_i = 1 \tag{2}$$

$$\sum_{i=1}^{N} z_i = K \tag{3}$$

$$\varepsilon_i z_i \leq w_i \leq \delta_i z_i \qquad i = 1, \dots, N \tag{4}$$

$$z_i \in \{0,1\} \qquad i = 1, \dots, N \tag{5}$$

$$0 \leq w_i \leq 1 \qquad i = 1, \dots, N \tag{6}$$

$$0 \leq \varepsilon_i \leq \delta_i \leq 1 \qquad i = 1, \dots, N \tag{7}$$

The description of parameters and variables of Model 1-7 is as follows:

Parameters:

N : Number of available assets
μ_i : Expected return of asset i
K : Number of desirable assets kept in the portfolio
ε_i : Minimum weight of asset i in the portfolio
δ_i : Maximum weight of asset i in the portfolio
α : Probability or confidence level loss at the value-at-risk
R_f : Risk-free return

Variables:

w_i : The weight of asset i
z_i : A binary variable taking 1 if asset i is in the basket and 0 otherwise

Constraint (2) indicates the use of all the capital to invest in the available assets. Constraint (3) dictates that the maximum number of assets within the portfolio equals the desirable amount of K. Constraint (4) specifies the maximum and minimum allowable amount to invest in stock i. Constraint (5) determines the range of investment in stock i. The objective function (1) represents the maximization of the Sharpe ratio, i.e., portfolio return to portfolio risk. In this paper, three different portfolio risk factors $(risk_p)$ are used: Variance, Value-at-risk and conditional value-at-risk.

Variance: Equation (8), represents the calculation of $risk_p$ using the variance, which shows the same Markowitz mean-variance model.

$$risk_p = Variance = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{8}$$

Where σ_{ij} is the covariance between stock i and j

Value-at-risk and conditional value-at-risk: In continuous space, the conditional value-at-risk of an asset portfolio is calculated as Equation (9)

$$risk_p = CVaR_\alpha(x) = \frac{1}{1-\alpha}\int_{f(x,y)\geq VaR_\alpha(x)} f(x,y)p(y)dy \tag{9}$$

Where α is the confidence level, $f(x,y)$ represents the loss function for portfolio x and assets return y, $p(y)$ denotes the probability density function for return y, $VaR_\alpha(x)$ stands for the value-at-risk for portfolio x at the confidence level α and is calculated as $VaR_\alpha(x) = \min\{\xi : Pr[f(x,Y) \leq \xi] \geq \alpha\}$. Another way to calculate the conditional value-at-risk that simplifies the optimization process is to use the Equation (10).

$$CVaR_\alpha(x) = VaR_\alpha(x) + \frac{1}{1-\alpha}\int_{R^n} \max\{0, \big(f(x,y) - VaR_\alpha(x)\big)\}p(y)dy \tag{10}$$

The probability level is generally chosen in the range of 0.9 to 0.99 (here we have used the probability level of 0.95). In the proposed method by [28] , in order to describe the probability distribution of returns, first a limited sample of y_s $(s = 1, \dots, S)$ scenarios is selected. Each y_s is a n-dimensional vector that contains the returns of each of the n assets within the portfolio under scenario s. This instance of S scenario is stored as a scenario matrix with size $S \times n$. The risk proxy for optimization is then calculated for the x portfolio as an Equation (11).

$$CVaR_\alpha(x) = VaR_\alpha(x) + \frac{1}{(1-\alpha)S}\sum_{s=1}^{S} \max\{0, -y_s^T x - VaR_\alpha(x)\} \tag{11}$$

The value of $CVaR_\alpha(x)$ is calculated when $VaR_\alpha(x)$ is calculated. Loss function $f(x, y_s) = -y_s^T x$ is the portfolio loss, under scenario s. Under this definition, $VaR_\alpha(x)$ and $CVaR_\alpha(x)$ are estimates of $VaR_\alpha(x)$ and $CVaR_\alpha(x)$ based on certain scenarios.

3.1 Model Solution Based on Differential Evolution Algorithm

The differential evolution algorithm is an optimization algorithm, firstly introduced by Storn and Price [29]. They demonstrated that this algorithm was well able to optimize non-linear non-differentiable functions, and the algorithm was proven as a powerful and fast approach to optimization in continuous spaces.

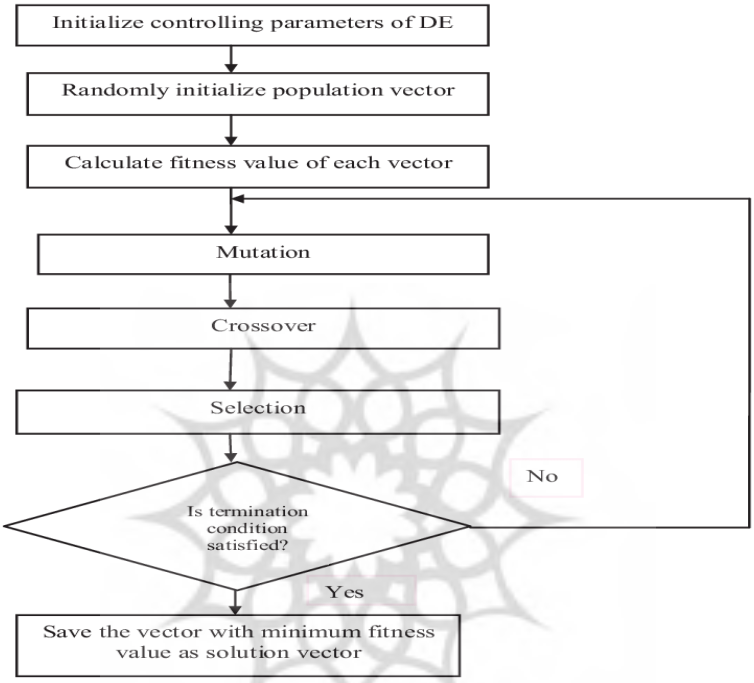


Fig.1: Flowchart of the Differential Evolution Algorithm [30]

The differential evolution algorithm was proposed to cover the main shortcoming of the genetic algorithm, i.e., the lack of a local search. The main difference between genetic algorithms and the differential evolution algorithm is their selection operators. In the selection operator of the genetic algorithm, the chance for a solution to be selected as a parent depends on its fitness value, but in the differential evolution algorithm, all solutions have an equal chance of being selected, i.e., their chance does not depend on their fitness. In this algorithm, after a new solution is generated by mutation and crossover operators, it is compared with the previous solution and replaced if it is a better solution. In contrast to other algorithms, in the differential evolution algorithm, the mutation operator is firstly performed, and then, the crossover operator applies to produce a new generation. No specific distribution is used to apply the mutation operator, and the mutation step length is determined in terms of distance between current members. Figure 1 represents the flowchart of the differential evolution algorithm.

To solve the portfolio optimization model by considering the cardinality constraints using the differential evolution algorithm, we first define the chromosomes that represent each of the problem solutions. The chromosomes of each solution are defined as follows.

w_1	w_2	…	w_N	z_1	z_1	…	z_N

Fig.2: Chromosomes for the Differential Evolution Algorithm for the Portfolio Optimization Problem

After generating a new solution (solutions of the initial population after applying mutation and cross-over operators), a feasible solution may be produced. Particularly, due to the presence of cardinality constraints and the necessity for the presence of a specific number of stocks within the portfolio, each new solution requires to be corrected to determine new weights of stocks. In this case, first, assets having the least weight exit the portfolio to improve the portfolio (adaptation with the predetermined desired number of stocks). Then, the remained maximum weight is allocated to the rest of the stocks within the portfolio until the number of stocks within the portfolio achieves a predetermined desired amount. Figure 3 describes the pseudocode for this procedure.

Inputs : $\mathrm{N}, \mathrm{K}, \mathrm{S}, \mathrm{W}$
Outputs : W^{r}

Parameters
N : Number of assets
K : Number of assets within the portfolio
S : A set of index assets
W : The set of weights entered into the correction process
$\mathrm{w}_{\mathrm{i}}^{\mathrm{r}}$: The weight of asset i entered into the correction process, $\mathrm{i} \in \mathrm{S}$
W^{r} : Set of weights after correction
z_{i} : Binary variable, $\mathrm{i} \in \mathrm{S}$

Process
Start

Until $\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{z}_{\mathrm{i}} = \mathrm{K}$, repeat the following process.
If $\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{z}_{\mathrm{i}} > \mathrm{K}$,
$\mathrm{w}_{\mathrm{i}} = 0$ and $\mathrm{z}_{\mathrm{i}} = 0$ for assets having the minimum w_{i}.
If $\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{z}_{\mathrm{i}} < \mathrm{K}$,
$\mathrm{z}_{\mathrm{i}} = 1$ for assets out of the portfolio.
$\mathrm{CS} = \sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{w}_{\mathrm{i}}$ $\mathrm{i} \in \mathrm{S}$ (Total weight of assets within the portfolio)
$\mathrm{FP} = 1 - \varepsilon \mathrm{K}$ (Maximum weight that can be assigned to assets within the portfolio)
$\mathrm{w}_{\mathrm{i}}^{\mathrm{r}} = \varepsilon \mathrm{z}_{\mathrm{i}} + \mathrm{w}_{\mathrm{i}} \mathrm{z}_{\mathrm{i}} \frac{\mathrm{FP}}{\mathrm{CS}}, \quad \forall \mathrm{i} \in \mathrm{W}, \forall \mathrm{i} \in \mathrm{W}^{\mathrm{r}}$

End

Fig.3: Pseudocode for the Correction of Generated Solutions

4 Methodology

In this research, the top 50 stocks in the Tehran Stock Exchange (introduced by the Tehran Stock Exchange) are considered as the candidate stocks to assess the model. K, the number of desired assets allowed to be held within the portfolio, is considered 5. Also, ε_i and δ_i are set to 0 and 1, respectively.

The model is assessed based on a moving time window of 24 months ending on 24 September 2021. The number of data in the sample for the model optimization is 250 days. After optimizing and determining the portfolio stocks and their corresponding weights, these stocks, along with their weights, are held for one month, and associated returns, risks, and Sharpe ratios are calculated. To evaluate the differential evolution algorithm, this algorithm is compared with the genetic algorithm. Also, Wilcoxon nonparametric test was used to evaluate the statistical differences between the results of different methods. We used MATLAB 2021 for coding the models and the execution system has a core i7 CPU (9^{th} generation) with 16 gb of RAM. Details of the code are provided in the Appendix B. The parameters used in DE algorithm, as suggested by [31], are presented as follows:

The size of population	:	10 times the number of assets
Lower bound of the scale factor (β_{min})	:	0.2
Upper bound of the scale factor (β_{max})	:	0.8
Crossover probability	:	0.8
Mutation factor	:	0.3
Stop condition	:	The objective function being unchanged for 100 iterations

5 Results

Figure 4 shows the return trend of the compared models over 24 periods. As seen, no model has dominance in all periods. A worth noting point is that the change in return for the conditional value-at-risk-based model is more than that for the variance-based model. The average return, standard deviation, and the Sharpe ratio for the 24 periods are compared in Figure 5. As seen, based on Figure 5, the most prioritized portfolio optimization models are mean-variance, mean-value-at-risk, and mean-conditional value-at-risk models, respectively (Appendix A represents the results for mean-variance, mean-value-at-risk, and mean-conditional value-at-risk separately). Although the figure shows evidence of differences between Sharp ratios, the differences do not appear to be noticeable. Therefore, Wilcoxon nonparametric test was used to evaluate the significance of this difference. ranksum tests the null hypothesis that data in two samples examined are samples from continuous distributions with equal medians, against the alternative that they are not. Table 2 presents the results of the Wilcoxon test [32] to evaluate the differences between the Sharp ratios of the models. As can be seen, given the p-value, the null hypothesis is not rejected and despite the difference between the return and risk of the models, there is no significant difference in the Sharp ratio between each model pair.

Fig 4: Comparison of the Return Trend for Models Studied

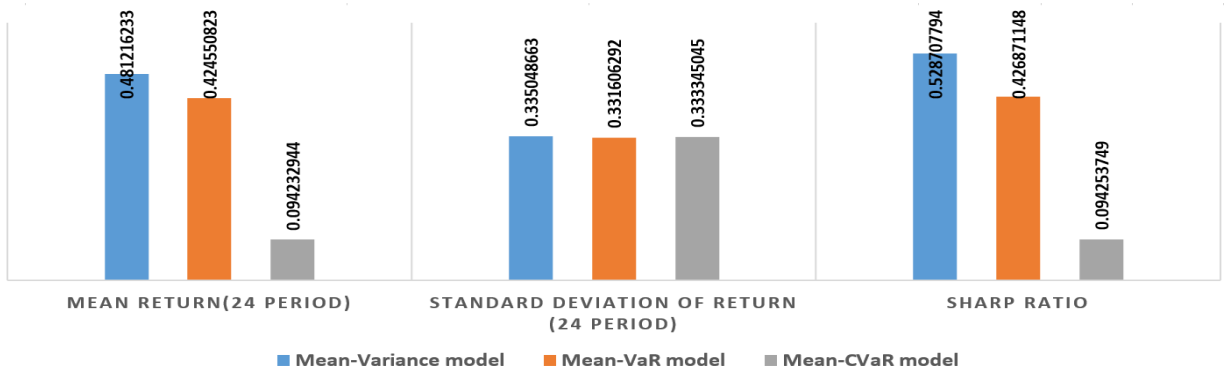


Fig5: Comparison of Average Return, Standard Deviation, and Sharpe Ratio for Studied Models

Table 2: Wilcoxon Test Results for Comparison between Portfolio Optimization Models

Models	Ranksum	Z stat	P-value	Result
Mean-variance & mean-value-at-risk	566	-0.44	0.65	There is no significant difference between two models
Mean-variance & mean-conditional-value-at-risk	575	-0.25	0.79	There is no significant difference between two models
Mean-value-at-risk & mean-conditional-value-at-risk	600	0.23	0.81	There is no significant difference between two models

Table 3 presents the results of the Wilcoxon test to evaluate the significant difference between the Sharp ratio results for the DE and GA algorithms. As can be seen, given the p-values, there is no significant difference between the results of the two algorithms in all portfolio optimization models. Table 4 shows the execution time of DE and GA algorithms to achieve the optimal solution. As can be seen, in general, the average solution time for DE algorithm was less than GA algorithm and it performed better in similar hardware conditions.

Table 3: Wilcoxon Test Results for Comparison Between Genetic Algorithm and Differential Evolution Algorithm

Models	**ranksum**	**Z stat**	**p-value**	**result**
Mean-variance	575	-0.258	0.797	There is no significant difference between two models
Mean-value-at-risk	580	-0.155	0.877	There is no significant difference between two models
Mean-conditional-value-at-risk	576	-0.237	0.813	There is no significant difference between two models

Table 4: Execution Time of Evolutionary Algorithms

Model	DE	GA
Mean-Variance	774.827	855.096
Mean-VaR	4390.979	7683.507
Mean-CVaR	5605.566	8718.386

6 Conclusion

In this paper, a new approach to solving the portfolio optimization model by considering the cardinality constraints using the differential evolution model is presented. In order to evaluate, the proposed

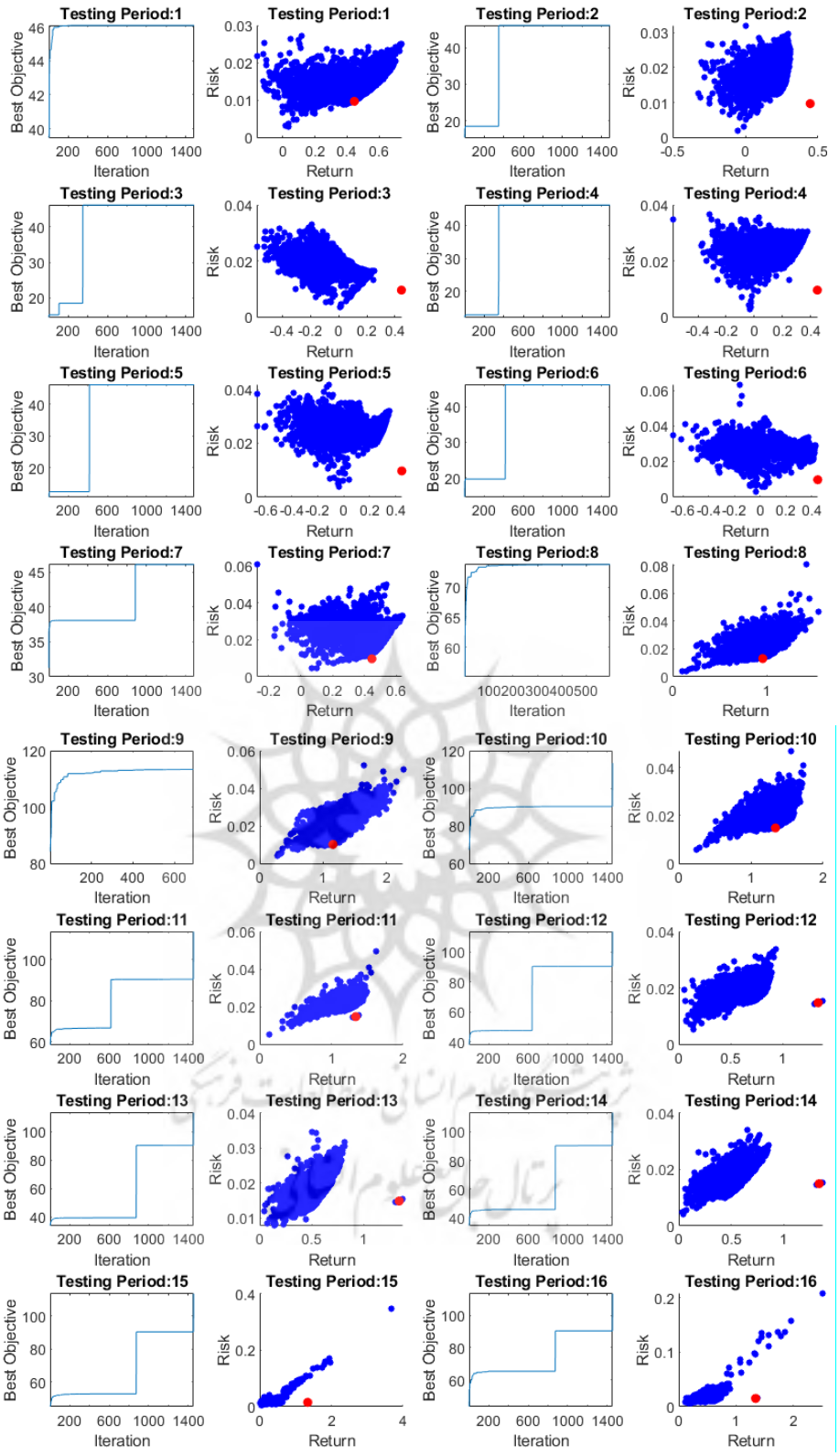
approach to solve models mean-variance, mean-VaR and mean-CVaR is considered. evaluation data included the top 50 stocks of the Tehran Stock Exchange. Also, the results obtained from the proposed approach are compared with the results obtained from the genetic algorithm. Based on the results, there is no significant difference in the optimal Sharp ratios in the three models of the portfolio optimized under evaluation. Also, no significant difference was observed in the optimal values of the objective function obtained from the GA and DE algorithms, but the average solution time for the DE algorithm is less than the GA model. The results of this research can be helpful in improving the investment process for small and large stockholders in the Iranian capital market. Future research may develop the comparison domain and solution approach. Other modeling methods for the assets return, such as autoregressive models, artificial intelligence-based models such as artificial neural networks, and in-depth networks, such as LSTM, can be used in future research. Furthermore, using other metaheuristic models such as ant colony and genetic algorithm and comparing these models can be an alternative direction for future research.

Appendix A

Results of mean-variance model

Table 5: Results of Return, Risk, and Sharpe Ratio for the Mean-Variance Model

Period	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Return	-0.08	0.14	-0.15	-0.77	-0.03	-0.37	-0.165	-0.246	-0.14	0.314	0.237	0.422
Risk	0.015	0.01	0.016	0.01	0.013	0.023	0.025	0.014	0.018	0.015	0.022	0.023
Sharpe ratio	-5.67	16.1	-9.51	-7.35	-2.207	-15.94	-6.557	-17.68	-7.89	20.62	10.84	17.97
Period	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
Return	0.189	0.13	0.023	0.003	0.098	0.014	0.225	0.229	-0.04	0.024	-0.315	-0.05
Risk	0.021	0.02	0.007	0.011	0.009	0.012	0.013	0.02	0.01	0.01	0.017	0.025
Sharpe ratio	9.036	7.69	3.355	0.24	10.47	1.138	17.28	11.303	-3.41	2.537	-18.37	-2.13


Testing Period:1
Best Objective
Iteration
Testing Period:1
Risk
Return
Testing Period:2
Best Objective
Iteration
Testing Period:2
Risk
Return
Testing Period:3
Best Objective
Iteration
Testing Period:3
Risk
Return
Testing Period:4
Best Objective
Iteration
Testing Period:4
Risk
Return
Testing Period:5
Best Objective
Iteration
Testing Period:5
Risk
Return
Testing Period:6
Best Objective
Iteration
Testing Period:6
Risk
Return
Testing Period:7
Best Objective
Iteration
Testing Period:7
Risk
Return
Testing Period:8
Best Objective
Iteration
Testing Period:8
Risk
Return
Testing Period:9
Best Objective
Iteration
Testing Period:9
Risk
Return
Testing Period:10
Best Objective
Iteration
Testing Period:10
Risk
Return
Testing Period:11
Best Objective
Iteration
Testing Period:11
Risk
Return
Testing Period:12
Best Objective
Iteration
Testing Period:12
Risk
Return
Testing Period:13
Best Objective
Iteration
Testing Period:13
Risk
Return
Testing Period:14
Best Objective
Iteration
Testing Period:14
Risk
Return
Testing Period:15
Best Objective
Iteration
Testing Period:15
Risk
Return
Testing Period:16
Best Objective
Iteration
Testing Period:16
Risk
Return

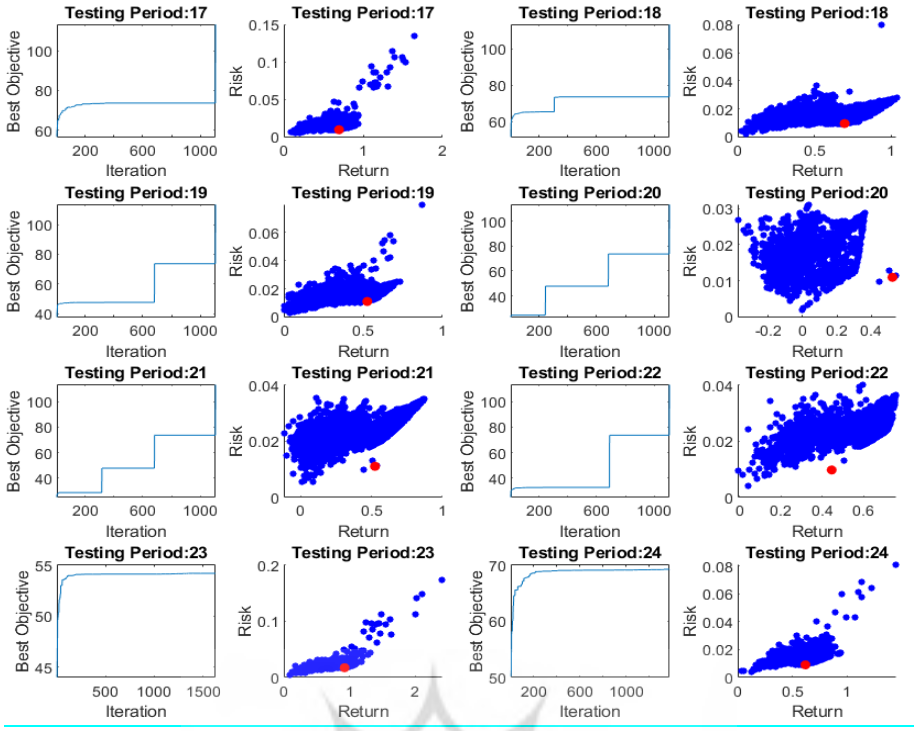
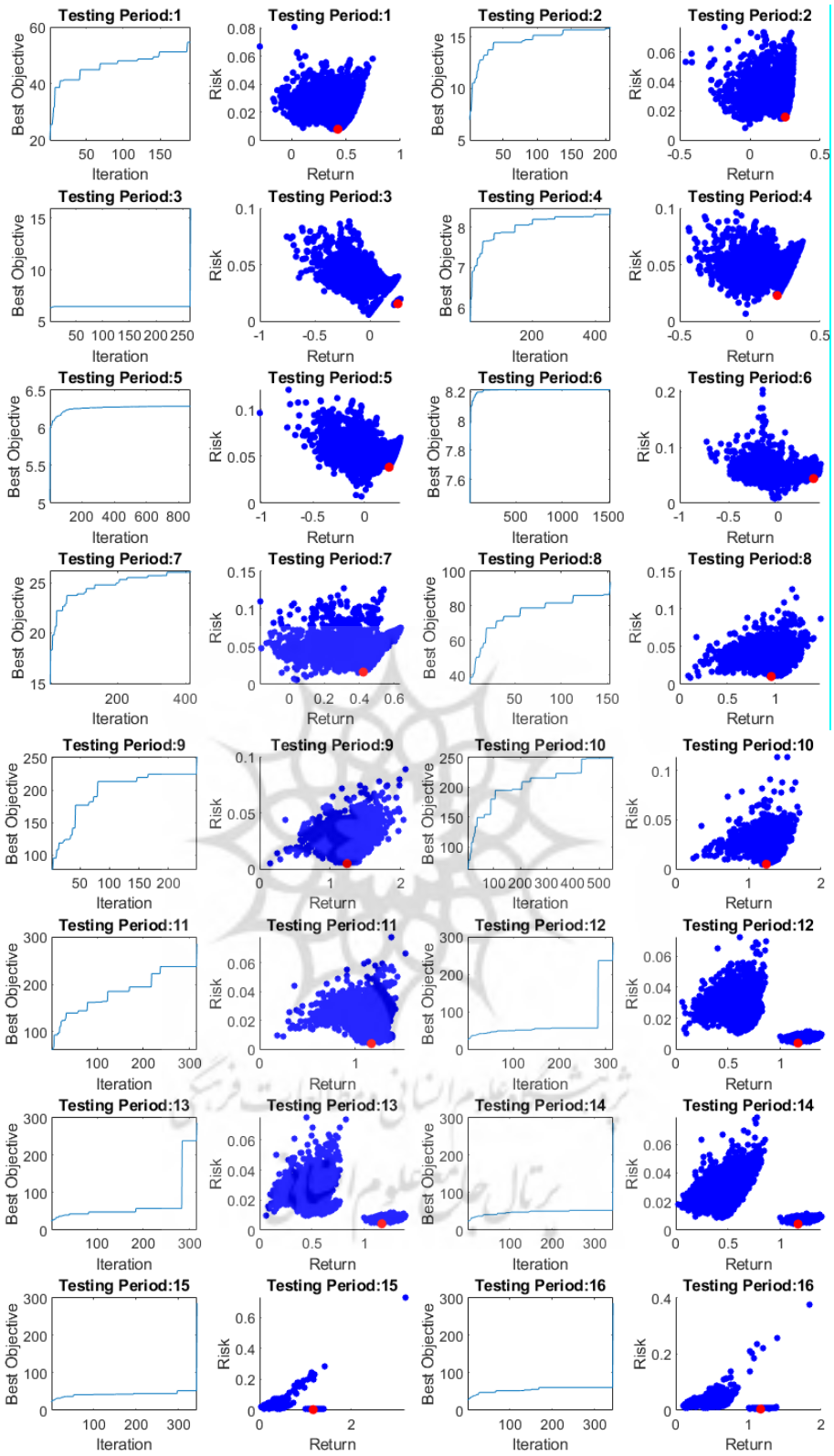


Fig 6: Changes in the Objective Function and Optimal Solutions in Different Iterations of the Differential Evolution Algorithm for the Mean-Variance Model

Results of mean-value-at-risk model

Table 6: Results of Return, Risk, and Sharpe Ratio for the Mean-Value-At-Risk Model

Period	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Return	-0.06	0.21	-0.16	-0.04	0.01	-0.317	-0.08	-0.316	-0.2	0.29	0.223	0.43
Risk	0.02	0.01	0.04	0.01	0.02	0.052	0.04	0.038	0.036	0.01	0.017	0.02
Sharpe ratio	-3.44	24.2	-4.5	-2.9	0.32	-6.13	-1.87	-8.252	-5.7	21.7	13.64	28.1
Period	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
Return	0.06	0.13	0.04	-0.04	0.1	-0.006	0.2	0.218	-0.07	0.01	-0.331	0.04
Risk	0.02	0.02	0.01	0.02	0.01	0.016	0.02	0.036	0.018	0.01	0.033	0.02
Sharpe ratio	3.29	7.82	5.61	-2.42	8.85	-0.349	11.5	6.118	-0.38	0.52	-10.16	1.61


Testing Period:1
Best Objective
Iteration
Testing Period:1
Risk
Return
Testing Period:2
Best Objective
Iteration
Testing Period:2
Risk
Return
Testing Period:3
Best Objective
Iteration
Testing Period:3
Risk
Return
Testing Period:4
Best Objective
Iteration
Testing Period:4
Risk
Return
Testing Period:5
Best Objective
Iteration
Testing Period:5
Risk
Return
Testing Period:6
Best Objective
Iteration
Testing Period:6
Risk
Return
Testing Period:7
Best Objective
Iteration
Testing Period:7
Risk
Return
Testing Period:8
Best Objective
Iteration
Testing Period:8
Risk
Return
Testing Period:9
Best Objective
Iteration
Testing Period:9
Risk
Return
Testing Period:10
Best Objective
Iteration
Testing Period:10
Risk
Return
Testing Period:11
Best Objective
Iteration
Testing Period:11
Risk
Return
Testing Period:12
Best Objective
Iteration
Testing Period:12
Risk
Return
Testing Period:13
Best Objective
Iteration
Testing Period:13
Risk
Return
Testing Period:14
Best Objective
Iteration
Testing Period:14
Risk
Return
Testing Period:15
Best Objective
Iteration
Testing Period:15
Risk
Return
Testing Period:16
Best Objective
Iteration
Testing Period:16
Risk
Return

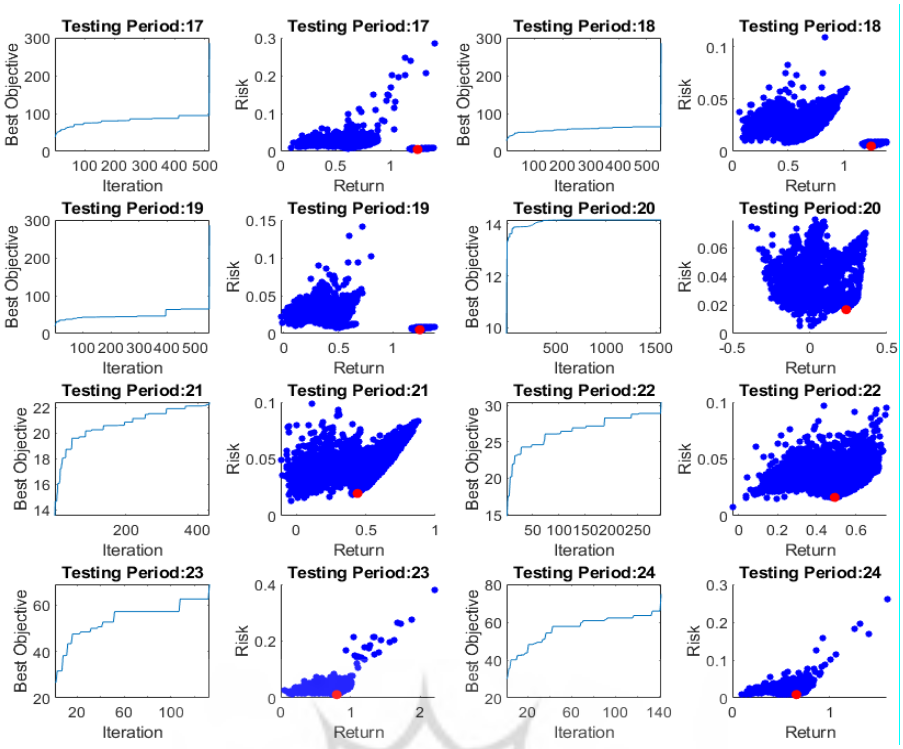
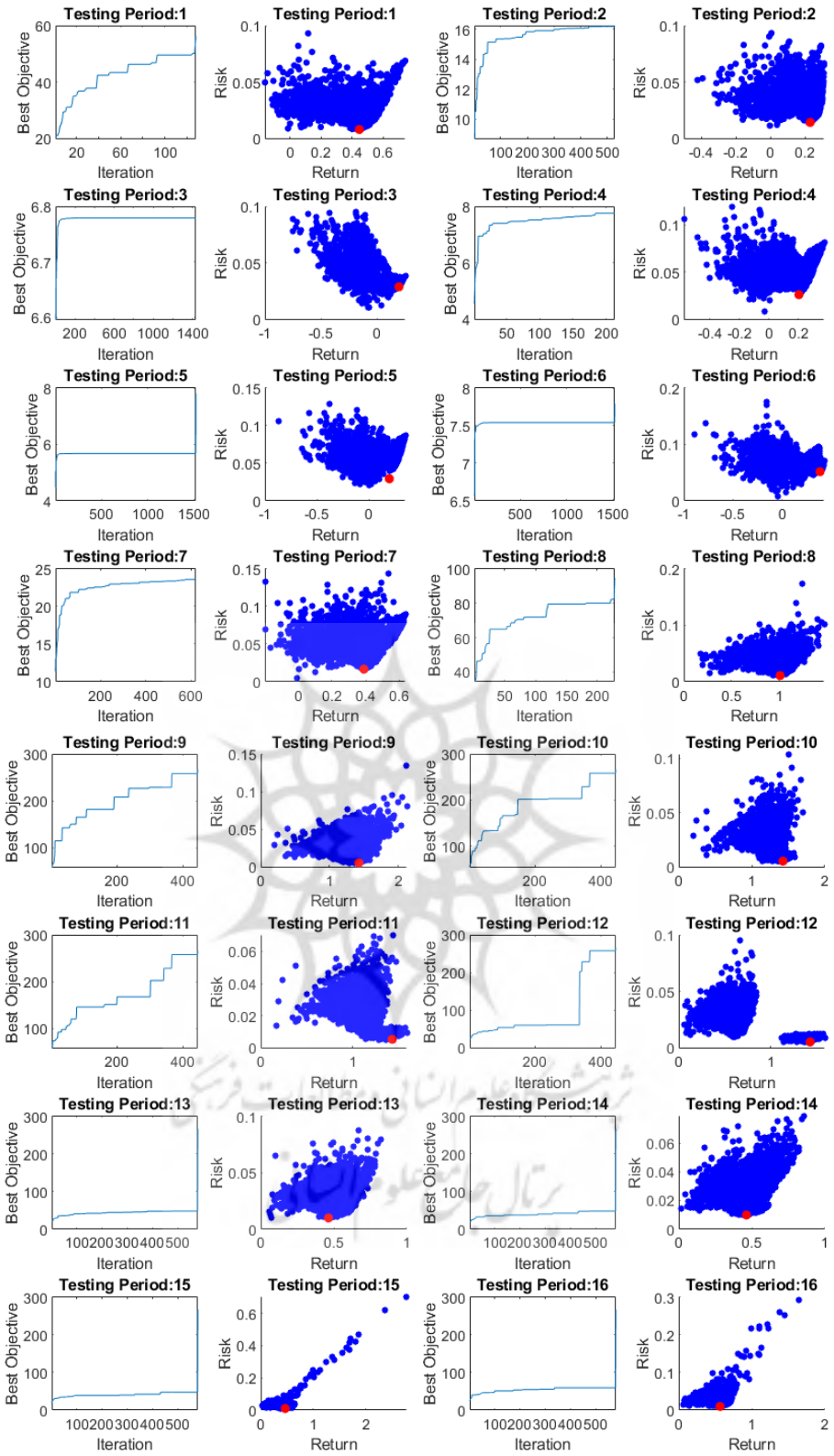


Fig .7: Changes in the Objective Function and Optimal Solutions in Different Iterations of the Differential Evolution Algorithm for the Mean-Value-At-Risk Model

Results of mean-conditional value-at-risk model

Table 7: Results of Return, Risk, and Sharpe Ratio for the Mean-Conditional Value-At-Risk Model

Period	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Return	-0.1	0.19	-0.1	-0.1	-0.1	-0.41	-0.06	-0.3	-0.2	0.22	0.185	0.48
Risk	0.02	0.02	0.04	0.02	0.03	0.082	0.04	0.036	0.03	0.02	0.024	0.01
Sharpe ratio	-3.3	12.3	-3.2	-3.6	-0.23	-4.98	-1.49	-8.19	-7.1	11.4	7.791	35.2
Period	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
Return	0.08	0.15	0.03	-0.1	0.1	-0.02	0.18	0.2	-0.1	0.02	-0.27	0.01
Risk	0.02	0.01	0.01	0.02	0.02	0.021	0.02	0.042	0.02	0.02	0.026	0.03
Sharpe ratio	3.36	12.5	2.89	-3.2	5.12	-0.98	11.2	4.774	-0.3.1	1.55	-10.5	0.33


Testing Period:1
Best Objective
Iteration
Testing Period:1
Risk
Return
Testing Period:2
Best Objective
Iteration
Testing Period:2
Risk
Return
Testing Period:3
Best Objective
Iteration
Testing Period:3
Risk
Return
Testing Period:4
Best Objective
Iteration
Testing Period:4
Risk
Return
Testing Period:5
Best Objective
Iteration
Testing Period:5
Risk
Return
Testing Period:6
Best Objective
Iteration
Testing Period:6
Risk
Return
Testing Period:7
Best Objective
Iteration
Testing Period:7
Risk
Return
Testing Period:8
Best Objective
Iteration
Testing Period:8
Risk
Return
Testing Period:9
Best Objective
Iteration
Testing Period:9
Risk
Return
Testing Period:10
Best Objective
Iteration
Testing Period:10
Risk
Return
Testing Period:11
Best Objective
Iteration
Testing Period:11
Risk
Return
Testing Period:12
Best Objective
Iteration
Testing Period:12
Risk
Return
Testing Period:13
Best Objective
Iteration
Testing Period:13
Risk
Return
Testing Period:14
Best Objective
Iteration
Testing Period:14
Risk
Return
Testing Period:15
Best Objective
Iteration
Testing Period:15
Risk
Return
Testing Period:16
Best Objective
Iteration
Testing Period:16
Risk
Return

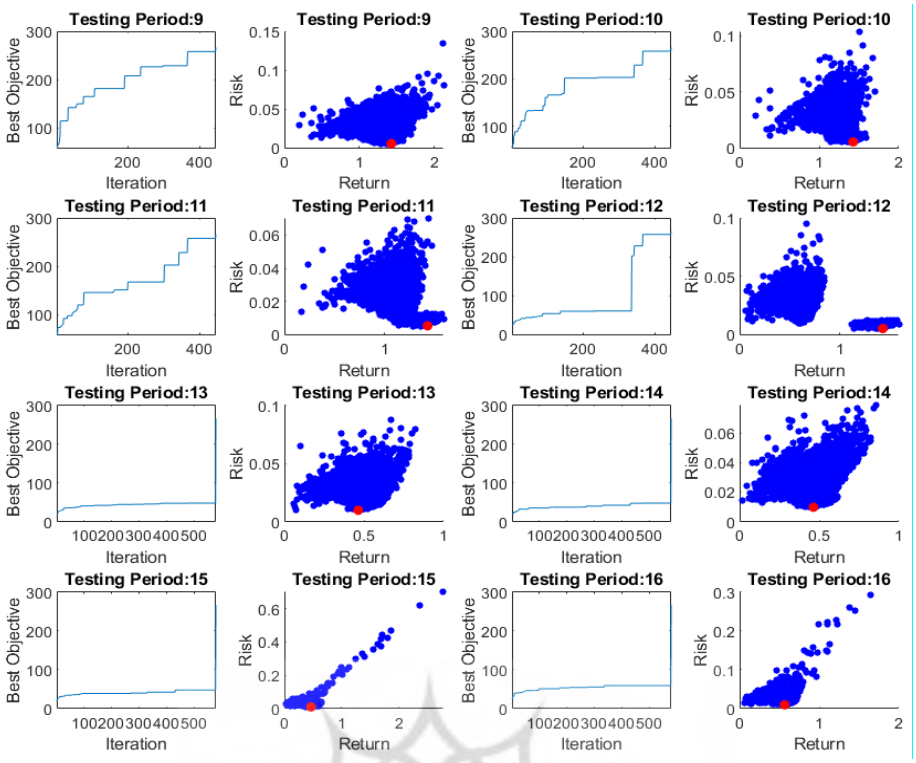


Fig.8: Changes in the Objective Function and Optimal Solutions in Different Iterations of the Differential Evolution Algorithm for the Mean-Conditional-Value-At-Risk Model

Appendix B
DE algorithm matlab code

```
clc;
clear;
close all;
%% Problem Definition
CostFunction=@(x) Sphere(x);    % Cost Function
nVar=20;             % Number of Decision Variables
VarSize=[1 nVar];   % Decision Variables Matrix Size
VarMin=-5;          % Lower Bound of Decision Variables
VarMax= 5;          % Upper Bound of Decision Variables
%% DE Parameters
MaxIt=1000;      % Maximum Number of Iterations
nPop=50;        % Population Size
beta_min=0.2;   % Lower Bound of Scaling Factor
beta_max=0.8;   % Upper Bound of Scaling Factor
pCR=0.2;        % Crossover Probability
%% Initialization
empty_individual.Position=[];
empty_individual.Cost=[];
BestSol.Cost=inf;
pop=repmat(empty_individual,nPop,1);
for i=1:nPop
    pop(i).Position=unifrnd(VarMin,VarMax,VarSize);
    pop(i).Cost=CostFunction(pop(i).Position);
    if pop(i).Cost<BestSol.Cost
        BestSol=pop(i);
    end
end
BestCost=zeros(MaxIt,1);
```

```
%% DE Main Loop
for it=1:MaxIt
    for i=1:nPop
        x=pop(i).Position;
        A=randperm(nPop);
        A(A==i)=[];
        a=A(1);
        b=A(2);
        c=A(3);
        % Mutation
        %beta=unifrnd(beta_min,beta_max);
        beta=unifrnd(beta_min,beta_max,VarSize);
        y=pop(a).Position+beta.*(pop(b).Position-pop(c).Position);
        y = max(y, VarMin);
        y = min(y, VarMax);
        % Crossover
        z=zeros(size(x));
        j0=randi([1 numel(x)]);
        for j=1:numel(x)
            if j==j0 || rand<=pCR
                z(j)=y(j);
            else
                z(j)=x(j);
            end
        end

        NewSol.Position=z;
        NewSol.Cost=CostFunction(NewSol.Position);
        if NewSol.Cost<pop(i).Cost
            pop(i)=NewSol;
            if pop(i).Cost<BestSol.Cost
               BestSol=pop(i);
            end
        end
    end
    % Update Best Cost
    BestCost(it)=BestSol.Cost;
    % Show Iteration Information
    disp(['Iteration ' num2str(it) ': Best Cost = ' num2str(BestCost(it))]);
end
%% Show Results
figure;
%plot(BestCost);
semilogy(BestCost);
```

GA algorithm matlab code

```
clc;
clear;
close all;
%% Problem Definition
global NFE;
NFE=0;
CostFunction=@(x) Sphere(x);     % Cost Function
nVar=5;             % Number of Decision Variables
VarSize=[1 nVar];   % Decision Variables Matrix Size
VarMin=-10;         % Lower Bound of Variables
VarMax= 10;         % Upper Bound of Variables
%% GA Parameters
MaxIt=200;      % Maximum Number of Iterations
nPop=100;        % Population Size
pc=0.8;                 % Crossover Percentage
nc=2*round(pc*nPop/2);  % Number of Offsprings (Parnets)
```

```
pm=0.3;                 % Mutation Percentage
nm=round(pm*nPop);      % Number of Mutants
gamma=0.05;
mu=0.02;          % Mutation Rate
ANSWER=questdlg('Choose selection method:','Genetic Algorith',...
    'Roulette Wheel','Tournament','Random','Roulette Wheel');
UseRouletteWheelSelection=strcmp(ANSWER,'Roulette Wheel');
UseTournamentSelection=strcmp(ANSWER,'Tournament');
UseRandomSelection=strcmp(ANSWER,'Random');
if UseRouletteWheelSelection
    beta=8;         % Selection Pressure
end
if UseTournamentSelection
    TournamentSize=3;   % Tournamnet Size
end
pause(0.1);
%% Initialization
empty_individual.Position=[];
empty_individual.Cost=[];
pop=repmat(empty_individual,nPop,1);
for i=1:nPop
    % Initialize Position
    pop(i).Position=unifrnd(VarMin,VarMax,VarSize);
    % Evaluation
    pop(i).Cost=CostFunction(pop(i).Position);
end
% Sort Population
Costs=[pop.Cost];
[Costs, SortOrder]=sort(Costs);
pop=pop(SortOrder);
% Store Best Solution
BestSol=pop(1);
% Array to Hold Best Cost Values
BestCost=zeros(MaxIt,1);
% Store Cost
WorstCost=pop(end).Cost;
% Array to Hold Number of Function Evaluations
nfe=zeros(MaxIt,1);
%% Main Loop
for it=1:MaxIt
    % Calculate Selection Probabilities
    P=exp(-beta*Costs/WorstCost);
    P=P/sum(P);
    % Crossover
    popc=repmat(empty_individual,nc/2,2);
    for k=1:nc/2
        % Select Parents Indices
        if UseRouletteWheelSelection
            i1=RouletteWheelSelection(P);
            i2=RouletteWheelSelection(P);
        end
        if UseTournamentSelection
            i1=TournamentSelection(pop,TournamentSize);
            i2=TournamentSelection(pop,TournamentSize);
        end
        if UseRandomSelection
            i1=randi([1 nPop]);
            i2=randi([1 nPop]);
        end
        % Select Parents
        p1=pop(i1);
        p2=pop(i2);
```

```
        % Apply Crossover
        [popc(k,1).Position popc(k,2).Position]=...
            Crossover(p1.Position,p2.Position,gamma,VarMin,VarMax);
        % Evaluate Offsprings
        popc(k,1).Cost=CostFunction(popc(k,1).Position);
        popc(k,2).Cost=CostFunction(popc(k,2).Position);
    end
    popc=popc(:);

    % Mutation
    popm=repmat(empty_individual,nm,1);
    for k=1:nm
        % Select Parent
        i=randi([1 nPop]);
        p=pop(i);
        % Apply Mutation
        popm(k).Position=Mutate(p.Position,mu,VarMin,VarMax);
        % Evaluate Mutant
        popm(k).Cost=CostFunction(popm(k).Position);
    end
    % Create Merged Population
    pop=[pop
         popc
         popm];
    % Sort Population
    Costs=[pop.Cost];
    [Costs, SortOrder]=sort(Costs);
    pop=pop(SortOrder);
    % Update Worst Cost
    WorstCost=max(WorstCost,pop(end).Cost);
    % Truncation
    pop=pop(1:nPop);
    Costs=Costs(1:nPop);
    % Store Best Solution Ever Found
    BestSol=pop(1);
    % Store Best Cost Ever Found
    BestCost(it)=BestSol.Cost;
    % Store NFE
    nfe(it)=NFE;
    % Show Iteration Information
    disp(['Iteration ' num2str(it) ': NFE = ' num2str(nfe(it)) ', Best Cost = '
num2str(BestCost(it))]);
end
%% Results
figure;
semilogy(nfe,BestCost,'LineWidth',2);
xlabel('NFE');
ylabel('Cost');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Functions

```
function [y1, y2]=Crossover(x1,x2,nAsset)
xx1=x1(1:nAsset);
xx2=x2(1:nAsset);
pSinglePoint=0.1;
pDoublePoint=0.2;
pUniform=1-pSinglePoint-pDoublePoint;
METHOD=RouletteWheelSelection([pSinglePoint pDoublePoint pUniform]);
switch METHOD
    case 1
        [yy1, yy2]=SinglePointCrossover(xx1,xx2);
```

```
    case 2
        [yy1, yy2]=DoublePointCrossover(xx1,xx2);

    case 3
        [yy1, yy2]=UniformCrossover(xx1,xx2);
end
xxx1=x1(nAsset+1:end);
xxx2=x1(nAsset+1:end);
[yyy1, yyy2]=SinglePointCrossover(xxx1,xxx2);
y1=[yy1,yyy1];
y2=[yy2,yyy2];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function y=Mutate(x,mu,nAsset,VarMin,VarMax)
x1=x(1:nAsset);
x2=x(nAsset+1:end);
nVar=numel(x1);
nmu=ceil(mu*nVar);
j=randsample(nVar,nmu);
sigma=0.1;
y1=x1;
y1(j)=x1(j)+sigma*randn(size(j))';
y1=max(y1,VarMin);
y1=min(y1,VarMax);
nVar=numel(x2);
nmu=ceil(mu*nVar);
j=randsample(nVar,nmu);
y2=x2;
y2(j)=1-x2(j);
y=[y1,y2];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function
[Z,PortRet,PortRisk]=ObjFunction(PortRet,PortRisk,simulatedRet,w,alpha,rf,nAsset,Variance,V
aR,CVaR,MAD,Mean,AR)
w=w(1:nAsset);
if Variance
    if isinf(PortRisk)
        PortRisk=100000000;
    end
    PortRisk=sqrt(w*PortRisk*w');
elseif VaR
    simulatedRet=simulatedRet*w';
    PortRisk=-prctile(simulatedRet,1-alpha);
elseif CVaR
    simulatedRet=simulatedRet*w';
    thr=prctile(simulatedRet,1-alpha);
    extRet=simulatedRet(simulatedRet<thr);
    PortRisk=-mean(extRet);
elseif MAD
    PortRisk=mean(w.*PortRisk);
end
if Mean
    PortRet=sum(PortRet*w')-rf;
elseif AR
    PortRet=sum(PortRet*w')-rf;
end
if isinf(PortRisk)
    PortRisk=100000000;
end
if isinf(PortRet)
```

```
    PortRet=0;
end
if PortRisk <= 0
    PortRisk = inf;
end
Z=PortRet/PortRisk;
End
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function repairedX=repairCons(x,eps,dlt,nAsset)
s=x(1:nAsset);
s=max(s,eps);
s=min(s,dlt);
z=x(nAsset+1:end);
idxOneAsset=find(z);
idxZroAsset=find(z==0);
s0=s(idxOneAsset);
eps0=eps(idxOneAsset);
dlt0=dlt(idxOneAsset);
L=sum(s0);
F=1-sum(eps0);
w=eps0+s0*(F/L);
badUpper=sum(w>dlt0);
while badUpper~=0
    R=find(w>dlt0); %bad
    QR=w<dlt0; %good
    L=sum(s0(QR));
    F=1-(sum(eps0(QR))+sum(dlt0(R)));
    w(QR)=eps0(QR)+s0(QR)*(F/L);
    w(R)=dlt0(R);
    badUpper=sum(w>dlt0);
end
x(idxOneAsset)=w;
x(idxZroAsset)=0;
repairedX=x;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function
[Z,PortRet,PortRisk]=TestFunction(PortRet,PortRisk,simulatedRet,w,alpha,rf,nAsset,Variance,
VaR,CVaR,MAD,Mean,AR)
w=w(1:nAsset);
if Variance
    if isinf(PortRisk)
        PortRisk=100000000;
    end
    PortRisk=sqrt(w*PortRisk*w');
elseif VaR
    simulatedRet=simulatedRet*w';
    PortRisk=-prctile(simulatedRet,1-alpha);
elseif CVaR
    simulatedRet=simulatedRet*w';
    thr=prctile(simulatedRet,1-alpha);
    extRet=simulatedRet(simulatedRet<thr);
    PortRisk=-mean(extRet);
elseif MAD
    PortRisk=mean(w.*PortRisk);
end

PortRet=sum(PortRet*w')-rf;

if isinf(PortRisk)
    PortRisk=100000000;
end
```

```
if isinf(PortRet)
    PortRet=0;
end
Z=PortRet/PortRisk;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [y1 y2]=DoublePointCrossover(x1,x2)
    nVar=numel(x1);
    cc=randsample(nVar-1,2);
    c1=min(cc);
    c2=max(cc);
    y1=[x1(1:c1) x2(c1+1:c2) x1(c2+1:end)];
    y2=[x2(1:c1) x1(c1+1:c2) x2(c2+1:end)];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [y1 y2]=SinglePointCrossover(x1,x2)
    nVar=numel(x1);
    c=randi([1 nVar-1]);
    y1=[x1(1:c) x2(c+1:end)];
    y2=[x2(1:c) x1(c+1:end)];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [y1 y2]=UniformCrossover(x1,x2)
    alpha=randi([0 1],size(x1));
    y1=alpha.*x1+(1-alpha).*x2;
    y2=alpha.*x2+(1-alpha).*x1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function i=RouletteWheelSelection(P)
    r=rand;
    c=cumsum(P);
    i=find(r<=c,1,'first');
end
```

References

[1] Markowitz, H., Portfolio selection, *The journal of finance*, 1952; **7**(1):77-91.doi: 10.1111/j.1540-6261.1952.tb01525.x

[2] Tehrani, R., Fallah Tafti, S., Asefi, S., Portfolio optimization using krill herd metaheuristic algorithm considering different measures of risk in tehran stock exchange, *Financial research journal*, 2018; **20**(4):409-426. doi: 10.22059/FRJ.2019.244004.1006538

[3]Samuelson, P.A., The fundamental approximation theorem of portfolio analysis in terms of means, variances and higher moments, *in Stochastic optimization models in finance*. 1975; Elsevier: 215-220. doi: 10.1016/B978-0-12-780850-5.50023-X

[4]Markowitz, H., Portfolio selection: efficient diversification of investments, *Cowies Foundation Monograph*, 1959;16.

[5] Konno, H., Yamazaki, H., Mean-absolute deviation portfolio optimization model and its applications to Tokyo stock market, *Management science*, 1991; 37(5):519-531. doi: 10.1287/mnsc.37.5.519

[6] Young, M.R., A minimax portfolio selection rule with linear programming solution, *Management science*, 1998; 44(5): 673-683. doi: 10.1287/mnsc.44.5.673

[7] Jorion, P., Value at risk: the new benchmark for controlling market risk, 1997; Irwin Professional Pub.

[8] Rockafellar. R.T., Uryasev, S., Optimization of conditional value-at-risk, *Journal of risk*, 2000; **2**: 21-42.

[9] Bienstock, D., Computational study of a family of mixed-integer quadratic programming problems, in *International Conference on Integer Programming and Combinatorial Optimization*. 1995;Springer. doi: 10.1007/3-540-59408-6_43

[10] Zamani, M., et al., An Expert System For Stocks Price Forecasting And Portfolio Optimization Using Fuzzy Neural Network, *Fuzzy Modeling And Genetic Algorithm*. 2015.

[11] Simaan, Y., Estimation risk in portfolio selection: the mean variance model versus the mean absolute deviation model, *Management science*, 1997; 43(10):1437-1446.

[12] Jorion, P., Value at risk: the new benchmark for managing financial risk, 2, 2001; McGraw-Hill New York.

[13] Nti, I.K., Adekoya, A.F., and Weyori, B.A., A systematic review of fundamental and technical analysis of stock market predictions, *Artificial Intelligence Review*, 2020; **53**(4): 3007-3057. doi: 10.1007/s10462-019-09754-z

[14] Marti, G., Nielsen, F., Bińkowski, M., Donnat, P., A review of two decades of correlations, hierarchies, networks and clustering in financial markets, *Progress in Information Geometry*, 2021; 245-274. doi: 10.48550/arXiv.1703.00485

[15] Lukovac, V., Pamučar, D., Popović M.,, Đorović, B.,Portfolio model for analyzing human resources: An approach based on neuro-fuzzy modeling and the simulated annealing algorithm, *Expert Systems with Applications*, 2017; **90**: 318-331. doi: 10.1016/j.eswa.2017.08.034

[16] Schaerf, A., Local search techniques for constrained portfolio selection problems, *Computational Economics*, 2002; **20**(3): 177-190. doi: 10.1023/A:1020920706534

[17] Baykasoğlu, A., Yunusoglu, M.G., and Özsoydan F.B., *A GRASP based solution approach to solve cardinality constrained portfolio optimization problems*, Computers & Industrial Engineering, 2015; 90: 339-351. doi: 10.1016/j.cie.2015.10.009

[18] Chang, T.-J., Meade, N., Beasley, J.E., Sharaiha, Y.M., Heuristics for cardinality constrained portfolio optimisation, *Computers & Operations Research*, 2000; 27(13): 1271-1302. doi: 10.1016/S0305-0548(99)00074-X

[19] Ehrgott, M., Klamroth, K., Schwehm, C., An MCDM approach to portfolio optimization, *European Journal of Operational Research*, 2004; 155(3):752-770. doi: 10.1016/S0377-2217(02)00881-0

[20] Lin, C.-C. and Y.-T. Liu, Genetic algorithms for portfolio selection problems with minimum transaction lots, *European Journal of Operational Research*, 2008; 185(1): 393-404. doi: 10.1016/j.ejor.2006.12.024

[21] Mishra, S.K., Panda, G., and Majhi, R., Constrained portfolio asset selection using multiobjective bacteria foraging optimization, *Operational Research*, 2014; 14(1): 113-145. doi: 10.1007/s12351-013-0138-1

[22] Yin, X., Ni, Q., Zhai. Y., A novel PSO for portfolio optimization based on heterogeneous multiple population strategy, *in 2015 IEEE Congress on Evolutionary Computation (CEC)*, 2015; IEEE. doi: 10.1109/CEC.2015.7257025

[23] Macedo, L.L., Godinho, P., Alves, M.J., Mean-semivariance portfolio optimization with multiobjective evolutionary algorithms and technical analysis rules, *Expert Systems with Applications*, 2017; 79: 33-43. doi: 10.1016/j.eswa.2017.02.033

[24] Kalayci, C.B., Ertenlice, O., Akyer, H., Aygoren, H., An artificial bee colony algorithm with feasibility enforcement and infeasibility toleration procedures for cardinality constrained portfolio optimization, *Expert Systems with Applications*, 2017, **85**, P. 61-75. Doi: 10.1016/j.eswa.2017.05.018

[25] Kalayci, C.B., Polat, O., Akbay, M.A., An efficient hybrid metaheuristic algorithm for cardinality constrained portfolio optimization. *Swarm and Evolutionary Computation*, 2020, **54**, P.100662.

[26] Yuen, M.-C., , Ng, S-C. Leung, Che, M.F., H., *Metaheuristics for Index-Tracking with Cardinality Constraints*, in *2021 11th International Conference on Information Science and Technology (ICIST)*. 2021. IEEE. doi: 10.1109/ICIST52614.2021.9440584

[27] Asoroosh, A., Atrchi, R., Ramtinnia, S., Portfolio Optimization Using Teaching-Learning Based Optimization (TLBO) Algorithm in Tehran Stock Exchange (TSE), *Financial Research Journal*, 2017; 19(2): 263-280. doi: 10.22059/JFR.2017.234738.1006462

[28] Cornuejols, G., Tütüncü, R., Optimization methods in finance, *Cambridge University Press.* 2006; 5.

[29] Storn, R., Price, K., Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *Journal of global optimization*, 1997; 11(4): 341-359. doi: 10.1023/A:1008202821328

[30] Dash, R., Rautray, R., Dash, R., Utility of a Shuffled Differential Evolution algorithm in designing of a Pi-Sigma Neural Network based predictor model, *Applied Computing and Informatics*, 2020. doi: 10.1016/j.aci.2019.04.001/full/html

[31] Deng, W., Shang, S., Cai, X., Zhao, H., Song, Y., Xu, J., An improved differential evolution algorithm and its application in optimization problem, *Soft Computing*, 2021; 25(7): 5277-5298. doi: 10.1007/s00500-020-05527-x

[32] Conover, W.J., Practical nonparametric statistics, *john wiley & sons*. 1999; 350.