IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 24, 2020, accepted March 16, 2020, date of publication March 20, 2020, date of current version April 1, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2982195

Optimal Portfolio Management for Engineering Problems Using Nonconvex Cardinality Constraint: A Computing Perspective

AMEER HAMZA KHAN[1], XINWEI CAO[2], VASILIOS N. KATSIKIS[3], PREDRAG STANIMIROVIĆ[4], IVONA BRAJEVIĆ[5], SHUAI LI[6], SEIFEDINE KADRY[7], AND YUNYOUNG NAM[8]

[1]Department of Computing, The Hong Kong Polytechnic University, Hong Kong
[2]School of Management, Shanghai University, Shanghai 200444, China
[3]Department of Economics, Division of Mathematics and Informatics, National and Kapodistrian University of Athens, 10559 Athens, Greece
[4]Department of Computer, Faculty of Sciences and Mathematics, University of Niš, 18000 Niš, Serbia
[5]Faculty of Applied Management, Economics and Finance, University Business Academy, 11000 Belgrade, Serbia
[6]Department of Electronics and Electrical Engineering, Swansea University, Swansea SA2 8PP, U.K.
[7]Department of Mathematics and Computer Science, Faculty of Science, Beirut Arab University, Beirut 11072809, Lebanon
[8]Department of Computer Science and Engineering, Soonchunhyang University, Asan 31538, South Korea

Corresponding authors: Xinwei Cao (xinweicao@shu.edu.cn) and Yunyoung Nam (ynam@sch.ac.kr)

This work was supported by the Soonchunhyang University Research Fund. The work of Predrag Stanimirović was supported by the Ministry of Education, Science and Technological Development, Serbia, under Grant 174013.

ABSTRACT The problem of portfolio management relates to the selection of optimal stocks, which results in a maximum return to the investor while minimizing the loss. Traditional approaches usually model the portfolio selection as a convex optimization problem and require the calculation of gradient. Note that gradient-based methods can stuck at local optimum for complex problems and the simplification of portfolio optimization to convex, and further solved using gradient-based methods, is at a high cost of solution accuracy. In this paper, we formulate a nonconvex model for the portfolio selection problem, which considers the transaction cost and cardinality constraint, thus better reflecting the decisive factor affecting the selection of portfolio in the real-world. Additionally, constraints are put into the objective function as penalty terms to enforce the restriction. Note that this reformulated problem cannot be readily solved by traditional methods based on gradient search due to its nonconvexity. Then, we apply the Beetle Antennae Search (BAS), a nature-inspired metaheuristic optimization algorithm capable of efficient global optimization, to solve the problem. We used a large real-world dataset containing historical stock prices to demonstrate the efficiency of the proposed algorithm in practical scenarios. Extensive experimental results are presented to further demonstrate the efficacy and scalability of the BAS algorithm. The comparative results are also performed using Particle Swarm Optimizer (PSO), Genetic Algorithm (GA), Pattern Search (PS), and gradient-based fmincon (interior-point search) as benchmarks. The comparison results show that the BAS algorithm is six times faster in the worst case (25 times in the best case) as compared to the rival algorithms while achieving the same level of performance.

INDEX TERMS Portfolio management, constrained optimization, nature-inspired algorithms, beetle search optimization.

I. INTRODUCTION

Portfolio selection is one the most important topic in finance [1]–[4]. It deals with finding an optimal choice of investments, which maximizes the profit for the portfolio holder while minimizing the financial risk. Although the financial parameters are stochastic and hard to predict with a high degree of accuracy [5]–[8], it is expected to find an optimal selection of stock options to maximize the chance of gaining monetary gain by leveraging the advances in the theory of mathematical modeling and optimization algorithms. Modern approaches on portfolio optimization rely on optimization-based algorithms to find the optimal proportion of each stock in the portfolio [9]–[11]. These approaches rely

The associate editor coordinating the review of this manuscript and approving it for publication was Mauro Gaggero.

on a fitness function, also called the objective function, which takes a proportion of each stock in the portfolio and outputs a fitness value. The goal is to find a specific proportion of each stock that maximizes the fitness function. The performance of the optimal solution depends on the formulation of the fitness function. For example, the fitness function can be formulated to maximize the expected profit regardless of the risk in investing in specific stocks. Similarly, it can also be formulated by considering both factors, i.e., maximize the profit, while minimizing the risk. A holistic fitness function accounting for all the relevant factors achieves better results.

A. BACKGROUND AND RELATED WORKS

Markowitz portfolio optimization method is one of the most well-known methods in modern portfolio theory [12]. Markowitz proposed a method based on the historical performance of stocks, i.e., estimate means and variances in return rate of the stocks to formulate the fitness function. In Markowitz's model, the risk of a stock is directly proportional to the variance of its return rate, while the expected return is equal to the mean of return rates. For the case of several stocks, covariances of stocks are also used in the formulation to evaluate the risk. Another classical approach is proposed by Elton *et al.* [13], which formulates a set of simple rules to find an optimal portfolio, and does not require solving an optimization problem. Easy availability of large-scale real-life datasets of stock market prices, advent of fast computing systems, e.g., general-purpose processors, digital signal processors, and an increase in memory capacity have allowed the researchers to study complex algorithms, which could not be efficiently implemented in real-time on traditional computers otherwise. With large processing and memory reservoirs at hands, the researchers were able to process large real-world datasets, allowing them to develop and train complex models to model the efficiency of a portfolio.

Several factors are being considered for formulating a holistic fitness function. Davis and Norman [14] introduced the concept of transaction cost [15], [16], i.e., the cost for buying the stocks. However, they do not consider the other constraints to reflect the real-world market factors affecting the stock prices. A later work by Chang *et al.* [17] introduced the concept of cardinality constraint [18], [19], i.e., the final portfolio can contain certain number of stocks. Machine learning-based approaches have also been studied for the portfolio selection problem [20]. Ledoit and Wolf [2] presented a new perspective on the estimation of covariances matrices by realizing that for a small dataset, using traditional approaches to estimate covariances matrices can lead to inaccurate results. However, they did not address the problem of solving the optimization problem. Baykasoğlu *et al.* [21] proposed a greedy randomized adaptive search procedure (GRASP) to solve the portfolio selection problem with cardinality constraints. The GRASP based approach decouples the original problems into two sub-problems: stock selection and proportion determination. The random search procedure only handles the first sub-problem, which effectively reduces the holistic nature of the optimization problem. However, in our work, the BAS optimization algorithm treats the optimization problem holistically and considers all the factors simultaneously while searching for the optimal portfolio. Other approaches use multiple criteria to evaluate the performance of a portfolio [4]. For example, Kalashnikov *et al.* [22] considers the multi-objective approach to address the problem; however, their work does not consider the problem of cardinality constraints as considers in this paper.

The recent trends in the field of metaheuristic optimization algorithms are specifically focused on nature-inspired algorithms. The process of biological evolution has given inspiration for a class of algorithms, called evolutionary algorithms (EAs) [23]. Similarly, the structure of the human genome has inspired the creation of genetic algorithms (GAs) [24]. Several algorithms have also been proposed based on the behavior of macroscopic organisms; for example, Ant Colony Optimization (ACO) [25], [26] is inspired by the working of an ant society. Similarly, the Cuckoo Search [27], Grey Wolf Optimizer (GWO) [28], Artificial Fish Swarm Algorithm [29], Honey Bee Algorithm (HBA) [30], [31], Invasive Weed Optimization (IWO) [32], and Firefly Algorithms (FAs) [33] are few of the recently proposed algorithms. The common feature of these algorithms is that they are inspired by the swarming behavior [34], [35] of insects, birds, and animals. Although algorithms based on swarming behavior have demonstrated excellent skills for searching optimal solution, however, they are also computationally extensive because each particle needs to evaluate the objective function individually. This increases the overall complexity multiplied by the number of particles in swarm. On the contrary, the BAS algorithm, as used in this paper, mimics the behavior of beetle, which are well-known for their skills for foraging the food individually, by just using their sense of smell. Therefore, the BAS algorithm only uses a single search particle, which contributes to the computational efficiency while achieving a comparable level of convergence performance.

B. AIM AND ORGANIZATION OF OUR WORK

Inspired by previously mentioned approaches, we have adopted a nonconvex formulation of the portfolio selection as a constrained optimization problem. The proposed problem formulation uses Markowitz's model to calculate the fitness of a portfolio; however, it introduces several other constraints absent from the original model. The goal of the proposed optimization problem is to minimize the loss, mathematically characterized as the variance of historical return values, and maximize the potential profit, mathematically characterized as expected value of the return. The constrained optimization is transformed into an unconstrained one by using the penalty term approach [36]–[40]. The unconstrained optimization problem has the advantage of numerical and computational efficiency. We propose a penalty function that penalizes the violation of optimization constraints by adding a factor to the objective function. The value of the penalty term is directly proportional to the magnitude of the violation of constraints.

An advantage of this approach is that the user can explicitly specify the importance of each constraint by adjusting the weight of the corresponding penalty terms.

To efficiently solve the reformulated optimization problem, we apply a nature-inspired metaheuristic algorithm; called Beetle Antennae Search (BAS) algorithm. We leverage the properties of metaheuristic algorithms in general, i.e., their well-known ability to efficiently solve complex nonlinear non-convex optimization problems [41]–[45]. Metaheuristic algorithms have found application is several practical situations [46]–[52]. The proposed algorithm is inspired by mathematical modeling of the food foraging behavior of beetles by Jiang and Li [53] and Zhang *et al.* [54]. The BAS algorithm have found practical applications in several real-world scenarios [55]–[67]. In this paper, we explore a new aspect of the application of beetle behavior and apply it to the portfolio selection problem. The highlights of this paper are as follow:

1) This paper adopts a nonlinear formulation of the portfolio optimization problem with various constraints, which comprehensively captures the nature of this problem. Note that traditional methods usually simplify this formulation to convex ones to reduce computational issues, but significantly degrades the quality of the solution.
2) BAS is applied to solve this constrained optimization. As verified by experiments, the solution is globally superior to others.
3) For some portfolio selection problems, e.g., real-time trading of stocks, it is critical to find the solution timely. As observed in extensive experiments, BAS presented in this paper is 6 times faster in the worst case than its rivals in portfolio optimization.

The rest of this paper is organized as follows: Section II formulates a constrained optimization for the portfolio selection problem by considering transaction cost and cardinality constraint. Section III reformulates the constrained optimization problem into an unconstrained one and mathematically models the behavior of beetle as an optimization algorithm. Section IV presents the experimental methodology, convergence performance, and comparative results with Particle Swarm Optimizer (PSO), Genetic Algorithm (GA) and Pattern Search (PS). Section V concludes the paper

II. PROBLEM FORMULATION

In this section, we mathematically formulate the problem of portfolio selection. First, we briefly introduce the classical problem formulation by Markowitz [12] and then introduce cardinality constraint and transaction cost, which are used in modern portfolio selection. Additionally, we also present different variants of the problems and explain their advantages.

A. MARKOWITZ MODEL

Suppose we have a total of N available stocks options, with there names denoted by $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_N$. Let $\mu_1, \mu_2, \ldots, \mu_N$ be the mean return rate calculated from the past market price. Similarly, σ_{ij} where $i, j \in \{1, 2, \ldots, n\}$, denote the covariance between the return rate of stock $\mathcal{S}_i$ and $\mathcal{S}_j$; note that for $i = j$, σ_{ij} denotes the variance of the return rate of stock $\mathcal{S}_i$. Also the covariance between two stocks is symmetric, therefore, $\sigma_{ij} = \sigma_{ji}$. Let us define the following matrices for simplifying the mathematical notation.

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 & \mu_2 & \cdots & \mu_N \end{bmatrix} \tag{1}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1N} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{N1} & \sigma_{N2} & \cdots & \sigma_{NN} \end{bmatrix}, \tag{2}$$

where $\boldsymbol{\mu}$ is a row matrix of mean return rates, and $\boldsymbol{\Sigma}$ is the covariance matrix. Note that the covariance matrix is symmetric.

Suppose the total investment amount is $\mathbb{T}$, and we normalize it to 1 for the sake of simplicity. Later, we will denormalize the final solution to calculate the actual investment amount in each stock. If the normalized amount invested in each stock is denoted by $t_1, t_2, \ldots, t_N$, then their sum should not exceed the total normalized investment amount, i.e.,

$$\sum_{i=1}^{N} t_i = 1. \tag{3}$$

Let $\mathbf{t} = [t_1, t_2, \ldots, t_N]$ denote the normalized amount invested in each stock. Based on the above definitions, the expected return on the investment is given by

$$\bar{\mu}(\mathbf{t}) = \mathbf{t}\boldsymbol{\mu}^T, \tag{4}$$

where $\bar{\mu}(.)$ denotes the normalized expected return on the investment. The amount of total expected return is $\mathbb{T}\bar{\boldsymbol{\mu}} = \mathbb{T}\mathbf{t}\boldsymbol{\mu}^T$. Similarly, the risk on the invested amount is dependent on the covariance matrix of all the stocks and given by the following relation

$$\overline{\Sigma}(\mathbf{t}) = \mathbf{t}\boldsymbol{\Sigma}\mathbf{t}^T, \tag{5}$$

where $\overline{\Sigma}(.)$ denotes the variance of the entire portfolio.

In the original Markowitz mean-variance model, the objective is to minimize the risk, i.e., portfolio variance, while obtaining a required value of normalized expected return, say μ_{req}. It can be expressed as the following optimization problem

$$\begin{aligned} &\min_{\mathbf{t}} \ \overline{\Sigma}(\mathbf{t}) \\ &\text{Subject to: } \bar{\mu}(\mathbf{t}) = \mu_{req} \\ &\qquad \mathbf{1}\mathbf{t}^T = 1 \\ &\qquad 0 \le t_i \le 1, \quad i \in \{1, 2, \ldots, N\} \end{aligned} \tag{6}$$

where μ_{req} is the required value of normalized expected return; this value is assumed to be known in the Markowitz model. Symbol $\mathbf{1}$ is the second equality constraint is a row matrix of ones, i.e., $\mathbf{1} \in \{1\}^{1\times N}$. It is a compact form of (3). The last inequality constraint is arises from the fact that the normalized investment in each stock should be between zero (no investment) and one (all amount invested).

B. MODIFIED MODEL

Although the classical Markowitz model is useful for portfolio selection, however, modern approaches consider additional factors, e.g., transaction cost and cardinality constraint. Now we will incorporate these factors into the optimization problem formulated above.

1) TRANSACTION COST

The amount required to buy or sell the shares of a particular stock is called the transaction cost. A realistic portfolio selection model need to consider the transaction cost because of their significance in the financial market. The transaction cost for different stocks is different and plays an important factor in deciding the final return rate for the stock. Suppose the normalized transaction cost for investing t_i amount in stock $\mathcal{S}_i$ is given by a function $\phi_i(t_i)$. We can construct the following matrix with transaction costs of all N stocks

$$\boldsymbol{\phi}(\mathbf{t}) = \begin{bmatrix}\phi_1(t_1) & \phi_2(t_2) & \cdots & \phi_N(t_N)\end{bmatrix} \tag{7}$$

and the total transaction cost for investing in N stocks is given by,

$$\bar{\phi}(\boldsymbol{t}) = \sum_{i=1}^{N} \phi_i(t_i) \tag{8}$$

where $\phi(.)$ denotes the sum of all transaction costs.

Several different cost models have been used in literature [68]. For example, $\phi_i(.) = 0$, where $i \in \{1, 2, \ldots, N\}$, denotes a zero transaction cost model, i.e., the investor only need to pay for the shares and no additional fee is charged. However, a more realistic model is called a linear transaction cost model, in which the investor needs to pay a transaction fee proportional to the investment amount. Such a model can be mathematically defined as

$$\phi_i(t) = \alpha_i t, \quad i \in \{1, 2, \ldots, N\} \tag{9}$$

where α_i is the factor controlling the transaction cost for stock $\mathcal{S}_i$. Using (7), (8), and the linear cost model of (9) the total transaction cost can be compactly written as,

$$\phi(\mathbf{t}) = \boldsymbol{\alpha}\mathbf{t}^T \tag{10}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_N]$. Based on the above discussion, it can be seen that the total investment amount, is not only spent on buying the shares in stocks, but also need to pay for the transaction fee. Therefore, the second equality constraint of (6) can be modified as follow

$$(\mathbf{1} + \boldsymbol{\alpha})\mathbf{t}^T = 1, \tag{11}$$

we will later use this updated constraint in the formulation of the final optimization problem.

2) CARDINALITY CONSTRAINT

In the Markowitz model, it is assumed that the investor will invest some amount in each of the N stocks. However, in practical scenarios, it may be desirable to limit the number of stocks in the final portfolio. Such a limit is called cardinality constraint. Suppose the limit on the number of stocks in $k(< n)$, i.e., at maximum, the investor can hold a share of k stocks. To formulate it mathematically, consider binary variables $z_1, z_2, \ldots, z_k$ to denote which stock is present in the portfolio. These binary variables can either hold a value of zero or one. Let us denote $\mathbf{z} = [z_1, z_2, \ldots, z_N]$. According to a cardinality constraint

$$\sum_{i=1}^{N} z_i = k. \tag{12}$$

Additionally, we need to make sure that the stocks for which $z_i = 0$, should also have $t_i = 0$, i.e., no investment in those stocks. It can be ensured using modifying the third inequality constraint in (6) as follow

$$0 \leq t_i \leq z_i, \quad i \in \{1, 2, \ldots, N\}. \tag{13}$$

Note that if $z_i = 0$, the condition (13) becomes $t_i = 0$. We will update the constraint in the final optimization problem.

C. UNIFIED OPTIMIZATION PROBLEM

The classical Markowitz model in (6) can be updated by considering the transaction cost model (11) and cardinality constraint (12), (13). The updated optimization problem, can be written in the expanded form as

$$\begin{aligned}
&\min_{\mathbf{t}} \ \mathbf{t}\boldsymbol{\Sigma}\mathbf{t}^T \\
&\text{Subject to: } \mathbf{t}\boldsymbol{\mu}^T = \mu_{req} \\
&\qquad (\mathbf{1} + \boldsymbol{\alpha})\mathbf{t}^T = 1 \\
&\qquad 0 \leq t_i \leq z_i, \quad i \in \{1, 2, 3, \ldots, N\} \\
&\qquad \mathbf{1}\mathbf{z}^T = k \\
&\qquad \mathbf{z} \in \{0, 1\}^{1\times N}.
\end{aligned} \tag{14}$$

Note that $\{0, 1\}^{1\times N}$ denotes a binary matrix of dimension $1 \times N$. Although the above-constrained optimization problem is most commonly used in literature. However, some other works [68], incorporate the expected return into the objective function instead of an equality constraint. Such an approach eliminates the need to specify the required value of expected return μ_{req}. This approach is also intuitive since the goal of an investor is not to earn a predefined return but to maximize the final return. Therefore, by putting it into the objective function, we are informing the optimizer to achieve the maximum possible value of expected return while minimizing the risk. The user can explicitly control the trade-off between the risk and the expected return by changing the objective function. The modified problem can be written as

$$\begin{aligned}
&\min_{\mathbf{t}} \ \mathbf{t}\boldsymbol{\Sigma}\mathbf{t}^T - \Lambda\mathbf{t}\boldsymbol{\mu}^T \\
&\text{Subject to: } (\mathbf{1} + \boldsymbol{\alpha})\mathbf{t}^T = 1 \\
&\qquad 0 \leq t_i \leq z_i, \quad i \in \{1, 2, 3, \ldots, N\} \\
&\qquad \mathbf{1}\mathbf{z}^T = k \\
&\qquad \mathbf{z} \in \{0, 1\}^{1\times N}.
\end{aligned} \tag{15}$$

where the parameter Λ controls the tradeoff between the expected return and the risk, the second term is incorporated into the objective function with a negative sign so that when the optimizer minimizes the value of the objective function, the value of the second term $\mathbf{t}\boldsymbol{\mu}^T$ increases

III. OPTIMIZATION ALGORITHM

In this section, we will formulate the BAS optimization algorithm, by mathematically modeling the food foraging behavior of beetles. The BAS algorithm is designed for unconstrained optimization problem, first, we will reformulate the constrained optimization problem in Section II into an unconstrained optimization problem. Then we will derive the optimization algorithm.

A. UNCONSTRAINED OPTIMIZATION PROBLEM

Observing the constraints of the optimization problem (15) shows that satisfying third and fourth constraints are a combinatorial problem, which in itself is a computationally expensive task. Additionally, each constraint requires a different solving technique, which makes it complicated to incorporate additional constraints into an existing solution. Therefore to avoid the combinatorial problem and unify all constraints into a single framework, we first need to convert the constrained optimization problem (15) into an unconstrained optimization problem. In this paper, we will use the penalty-term approach to incorporate the equality and inequality constraints into the objective function. Using the penalty-term approach, the constrained optimization problem (15) can be written as following unconstrained optimization problem

$$\min_{\mathbf{t},\mathbf{z}} \mathbf{t}\boldsymbol{\Sigma}\mathbf{t}^T - \Lambda\mathbf{t}\boldsymbol{\mu}^T + \mathcal{P}(\mathbf{t},\mathbf{z}) \tag{16}$$

where $\mathcal{P}(.)$ is the penalty function for the constraints in (15). The penalty function is composed of four terms corresponding to the constraints in the optimization problem

$$\mathcal{P}(\mathbf{t},\mathbf{z}) = \beta_1\mathcal{P}_1(\mathbf{t}) + \beta_2\mathcal{P}_2(\mathbf{t},\mathbf{z}) + \beta_3\mathcal{P}_3(\mathbf{z}) + \beta_4\mathcal{P}_4(\mathbf{z}). \tag{17}$$

where $\mathcal{P}_i(.)$, $i \in \{1,2,3,4\}$ denotes the penalty terms and β_i denotes the weight of each term in the objective function. The purpose of the penalty term is to add a positive value in the objective function when the corresponding constraint is violated. Since the optimizer is trying to minimize the objective function, therefore these terms penalize the violation in constraints by adding additional value. If the constraints are not violated, then these terms become zero.

First, we define the penalty term for the equality constraints, i.e., constraints corresponding to lines 2 and 4 of (15)

$$\mathcal{P}_1(\mathbf{t}) = ((\mathbf{1}+\boldsymbol{\alpha})\mathbf{t}^T - 1)^2, \tag{18}$$
$$\mathcal{P}_3(\mathbf{z}) = (\mathbf{1}\mathbf{z}^T - k)^2. \tag{19}$$

These functions define a non-negative term which add a positive value into the objective function of (16), thus penalizing the violation of constraints. Next, the penalty term for the cardinality constraint, i.e., constraint corresponding to line 5 of the optimization problem (15) can be defined as

$$\mathcal{P}_4(\mathbf{z}) = \mathbf{z}(\mathbf{1}-\mathbf{z})^T(\mathbf{1}-\mathbf{z})\mathbf{z}^T \tag{20}$$

which is equivalent to $\mathcal{P}_4(\mathbf{z}) = \sum_{i=1}^{N} z_i^2(1-z_i)^2$. This function produces a small positive value when $z_i \in \{0,1\}$, otherwise it add a large penalty to the objective function (16). Now, for the inequality constraint, i.e., constraint corresponding to line 3 of the optimization problem (15), define the following penalty term

$$\mathcal{P}_2(\mathbf{t},\mathbf{z}) = \sum_{i=1}^{N} \mathcal{Q}(t_i, z_i). \tag{21}$$

where the function $\mathcal{Q}(.)$ is defined as

$$\mathcal{Q}(t,z) = \begin{cases} -t, & \text{if } t < 0 \\ 0, & \text{if } 0 \le t \le z \\ t-1 & \text{if } z < t. \end{cases} \tag{22}$$

It can be seen that such a definition of $\mathcal{P}_2$ makes sure that a positive number is added to the objective function (16) whenever an inequality constraint is violated.

Based on the definitions of penalty function (17), and replacing the value of individual penalty-term from (18), (21), (19), and (20), the final form of the penalty function can be written as

$$\begin{aligned}\mathcal{P}(\mathbf{t},\mathbf{z}) = \beta_1((\mathbf{1}+\boldsymbol{\alpha})\mathbf{t}^T - 1)^2 + \beta_2 \sum_{i=1}^{N} \mathcal{Q}(t_i, z_i) \\ + \beta_3(\mathbf{1}\mathbf{z}^T - k)^2 + \beta_4\mathbf{z}(\mathbf{1}-\mathbf{z})^T(\mathbf{1}-\mathbf{z})\mathbf{z}^T.\end{aligned} \tag{23}$$

This can be replaced in (16) to get the complete unconstrained optimization problem. Now we formulate the optimization algorithm to solve this unconstrained optimization problem.

B. BAS OPTIMIZATION ALGORITHM

The food foraging behavior of beetle inspires the BAS optimization algorithm. The behavior of beetle is inspiring because of their excellent ability to search for food in a previously unknown environment by just using its olfactory sense. The beetle senses the smell of food, and its goal is to search for the region with the maximum smell. Therefore, the behavior of the beetle can be characterized as an optimization algorithm. Beetle has a pair of antennae, which is used to measure the difference in intensity of smell in different directions while searching way toward food. At each step, beetle senses the intensity of smell at both antennae location and, based on the difference, move toward a direction where the intensity of smell is increasing. The mathematical modeling of this behavior leads to an efficient global optimization algorithm. The behavior is visualized in Fig. 1

Suppose we want to solve following unconstrained optimization problem

$$\max_{\mathbf{x}} f(\mathbf{x}), \tag{24}$$

where $f : \mathbb{R}^n \to \mathbb{R}$. The optimization variable $\mathbf{x}$ is an n-dimensional vector. In beetle's analogy, $\mathbb{R}^n$ is the space in

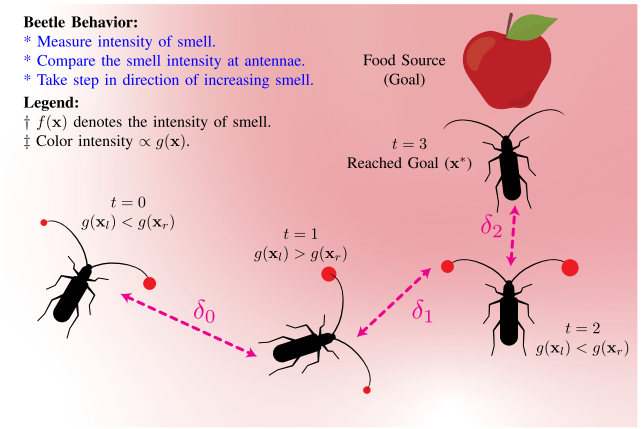

FIGURE 1. A visual illustration of the beetle's food foraging behavior. Each step depicts a single iteration of the BAS algorithm.

which the beetle is searching and $f(.)$ represents the intensity of smell at each point. The goal is to find a vector $\mathbf{x}^*$ such that the intensity of smell $f(\mathbf{x}^*)$ is maximized.

Suppose that beetle starts from a random location $\mathbf{x}_0$. For the sake of generality, consider it reaches a point $\mathbf{x}_m$ at step m. Generate a random direction vector $\vec{\mathbf{d}} \in \mathbb{R}^n$ with normally distributed elements. The random vector is generated to represent the direction of the beetle's antennae. Consider both antennae of beetle are pointing in opposite directions; therefore we can calculate the location of antennae's endpoint using the following relations

$$\mathbf{x}_{mL} = \mathbf{x}_m + \lambda_m \vec{\mathbf{d}}, \quad \mathbf{x}_{mR} = \mathbf{x}_m - \lambda_m \vec{\mathbf{d}}, \tag{25}$$

where we used variables $\mathbf{x}_{mL}$ and $\mathbf{x}_{mR}$ to denote the position of left antennae and right antennae, respectively. λ_m denotes the length of antennae.

The next step is to calculate the value of the objective function at both these antennae locations, mimicking the sense of smell of both antennae. Evaluate the objective function in (24) using both vectors; $\mathbf{x}_{mL}$ and $\mathbf{x}_{mR}$, as follow

$$f_{mL} = f(\mathbf{x}_{mL}), \quad f_{mR} = f(\mathbf{x}_{mR}), \tag{26}$$

where f_{mL} and f_{mR} denotes the value of objective function at left and right antenna location respectively. By comparing both values, we make the next step according to the following update-rule

$$\mathbf{x}'_{m+1} = \mathbf{x}_m + \delta_m(\lambda_m)\,\mathrm{sign}(f_{mL} - f_{mR})\vec{d}, \tag{27}$$

where $\mathbf{x}'_{m+1}$ is the updated location of the beetle, $\delta_k(.)$ is the parameter controlling the Euclidean length of actual step-size. The step-size is a function of antennae length λ_k for the sake of generality. Their relation is described later. The function sign(.) is used in the above update-rule to make sure that the updated location is in a direction toward which the value of the objective function is increasing.

The objective function is then evaluated at updated location $\mathbf{x}'_{m+1}$ to sense the intensity of smell at the new location

$$f'_{m+1} = f(\mathbf{x}'_{m+1}), \tag{28}$$

the value f'_{m+1} is then compared to the value f_m from the last location. If there is an improvement, i.e., the new value f'_{m+1} is higher, then the beetle remains at the new location $\mathbf{x}'_{m+1}$; otherwise it returns to its previous location. The location variable is updated as

$$\mathbf{x}_{m+1} = \begin{cases} \mathbf{x}'_{m+1} & \text{if } f'_{m+1} > f_m \\ \mathbf{x}_m & \text{if } f'_{m+1} \le f_m. \end{cases} \tag{29}$$

Similarly, the variable holding the value of the objective function is updated

$$f_{m+1} = \begin{cases} f'_{m+1} & \text{if } f'_{m+1} > f_m \\ f_m & \text{if } f'_{m+1} \le f_m. \end{cases} \tag{30}$$

The above steps are performed repeatedly until an optimal solution is reached. The steps of the algorithm are systematically presented in 1. Fig. 2 shows the schematic diagram of the BAS algorithm to visually depict the functionality and connection between different components of the algorithm.

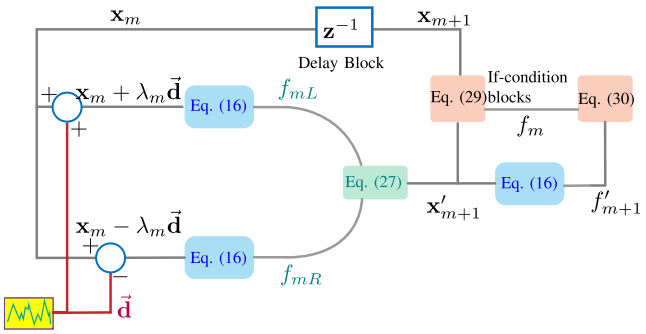

FIGURE 2. Schematic diagram of the BAS algorithm visually depicting the interconnections between different components of the Algorithm 1.

The value of parameters λ_m and $\delta_m(.)$ affect the convergence rate of the BAS algorithm. By empirical observations, we found that the following rules provide a reasonable convergence rate

$$\lambda_m = c_1 \sqrt{n} \mathrm{e}^{-\alpha m}, \quad \delta_m(\lambda_m) = c_2 \lambda_m, \tag{31}$$

where m denotes the current step number, α controls the speed of decay. c_1 and c_2 are constant values and generally their default values are chosen as $c_1 \in (0, 2)$ and $c_2 = 1$. However, for a specific implementation, their values can be further tuned. The rule starts the algorithm with a large value of antennae length and slowly converges to a small value according to the exponential decay law.

Remark 1: Note that although the above algorithm is formulated for a maximization problem, it can equivalently be applied to the minimization problem

$$\min_{\mathbf{x}} f(\mathbf{x}), \tag{32}$$

by modifying the update-rule in (29) as follow

$$\mathbf{x}'_{m+1} = \mathbf{x}_m - \delta_m(\lambda_m)\mathrm{sign}(f_{mL} - f_{mR})\vec{d}, \tag{33}$$

and reversing the inequality conditions in (27) and (30).

TABLE 1. NASDAQ symbols for the selected companies.

Companies 01-13	GOOGL	AMZN	FB	T	AAXN	VZ	F	GM	OKE	TERP	CTRE	IRBT	NFLX
Companies 14-26	HAYN	PM	A	EPR	CARB	BBRG	GOGO	LEN	HBIO	ORM	TD	AHL	OCSL
Companies 27-39	HAS	CNP	WFC_W	ETJ	PAA	TMQ	AZO	CG	BDC	HBNC	VTVT	KYE	ETY
Companies 40-50	ORC	SNMX	AEUA	PZZA	WYY	SHLD	RIOT	GES	JWN	WSO	PEGA		

Remark 2: The unconstrained optimization (16) can be written in the same form as (32) by defining $\mathbf{x} = [\mathbf{t} \quad \mathbf{z}]$, i.e., augmenting both row matrices to form a single large matrix. In this case, the dimension of optimization problem becomes $n = 2N$, i.e., $\mathbf{x} \in \mathbb{R}^{2N}$.

Algorithm 1 BAS Algorithm - Portfolio Optimization

Input: Mean of return rate for all stocks $\boldsymbol{\mu}$, covariance matrix of return rate $\boldsymbol{\Sigma}$, normalized transaction cost $\boldsymbol{\alpha}$, number of stocks in final portfolio k, weight parameters: Λ, β_1, β_2, β_3, and β_4.

Output: An optimal selection of stocks $\mathbf{z}^*$ and optimal investment amount in each stock $\mathbf{t}^*$.

Construct the objective function in (16) according to method given in Remark 2.

$\mathbf{x}_0 \leftarrow$ Initial location

$m \leftarrow 0$ $k_{stop} \leftarrow$ maximum number of steps allowed

while $m < m_{stop}$ **do**

- Generate a normalized random vectors, $\vec{\mathbf{d}} \in \mathbb{R}^n$.
- Calculate both antennae's locations using using (25).
- Evaluate the objective function at both vectors.
- Use the calculated values to determine the new location according to the update-rule (27).
- Calculate the value of objective function at new location as given in (28).
- Update the location according to (29) and (30).
- $m \leftarrow m + 1$

end

C. COMPUTATIONAL COMPLEXITY

Here the complexity analysis of the BAS algorithm is presented. Steps listed inside while loop of Algorithm 1 are used to calculate the number of mathematical operations. The first step of the algorithm, requires the generation of n normally distributed random variables, which requires a total of $b_1\, n$ floating-point operations, where b_1 denotes the operations required to generate a single random variable. The second step, i.e., calculation of antennae's end-point locations, requires a $2n$ additions and $2n$ multiplication operations, making up a total of $4n$ floating-point operations in this step. The third step requires the evaluation of the objective function twice. A careful analysis of the objective function tells that it requires approximately $n^3 + 7n$ floating-point operations to evaluate it once. The most complex part in evaluation of the objective function is computation of $\mathbf{t}\boldsymbol{\Sigma}\mathbf{t}^T$, which contributed the n^3 term. Since we need to evaluate the objective function twice, it requires $2n^3 + 14n$ floating-point operations. The fourth step, i.e., updating the location according to the rule (27), requires $2n + 2$ floating-point operations. The fifth step, requires the evaluation of the objective function again, therefore requiring approximately n^3+7n floating-point operations. The last step, i.e., updating the location according to (29) and (30) requires 2 comparisons. Summation of floating-point operations for all the steps make a total of: $(b_1\, n+4n+(2n^3+14n)+2n+2+n^3+7n+2) = 3n^3+(27+b_1)n+3\alpha_2+2$. It implies that the overall computational complexity of the BAS algorithm is $O(n^3)$, i.e., polynomial with respect to the dimensionality of optimization variable.

IV. RESULTS AND DISCUSSION

In this section, we will present the experimental methodology and the optimization results using the BAS algorithm. First, we will describe the processing of the dataset to obtain the mean return rate and the covariance matrix from the historical prices of stocks. Then we will describe the type of experiments and then present their results along with the discussion.

A. EXPERIMENTAL METHODOLOGY

To evaluate the viability of the BAS algorithm in practical scenarios, we used the real-world dataset [69] to calculate the parameters for the optimization problems. The dataset [69] contains the stock prices, up to 2017, for major companies in the NASDAQ stock exchange. The dataset contains stock prices for 7192 companies. Out of those, we selected 50 companies to perform our experiments. We used the stock price data for the year 2017 to calculate the mean return rate and covariance matrix for the selected companies. The name of the companies used in our experiments are listed in Table 1. The dimension of mean return rate matrix $\boldsymbol{\mu}$ is 1×50 and covariance matrix $\boldsymbol{\Sigma}$ is 50×50. Therefore, we have visualized them as a grid of rectangular pixels and used colormap to show their values. Fig. 4(a) and Fig. 4(b) visualize the matrices $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ respectively. To obtain a value of $\boldsymbol{\alpha}$, i.e., the transaction cost vector for purchasing the shares, we generated a uniformly distributed random vector with a dimension of 1×50, such that $\mathbf{1}\boldsymbol{\alpha}^T = 0.1$. All the experiments were conducted with MATLAB running on macOS Cataline with 2.2 GHz Quad-Core Intel Core i7 and 16 GB of RAM. The MATLAB was chosen because it provided the environment to quickly prototype and test the performance of the algorithm. The real-world implementation of the algorithm will usually require a more specialized embedded processor programming using a low-level programming language, e.g., C or C++. It must be noted that the since the optimal value of the optimization variables, for the problem of portfolio selection,

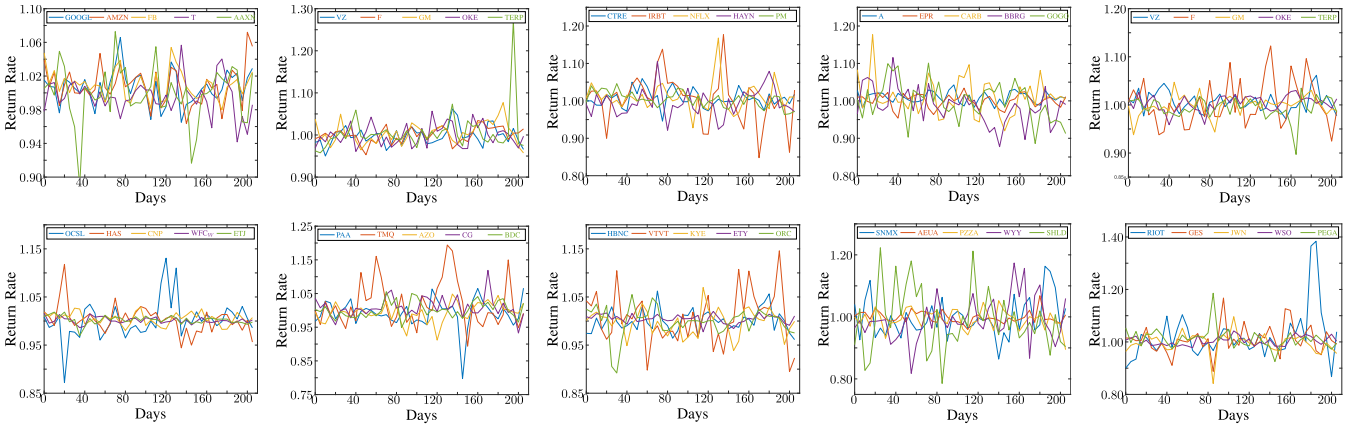

FIGURE 3. The return rate of the stocks of the companies listed in Table 1. Each figure corresponds to five companies. This return rate is used in the calculation of mean and covariance matrices visualized in Fig. 4. The values of return rate are concentrated above one, i.e., company is giving profit. The chaotic behaviour of these curves is expected from a real-world dataset.

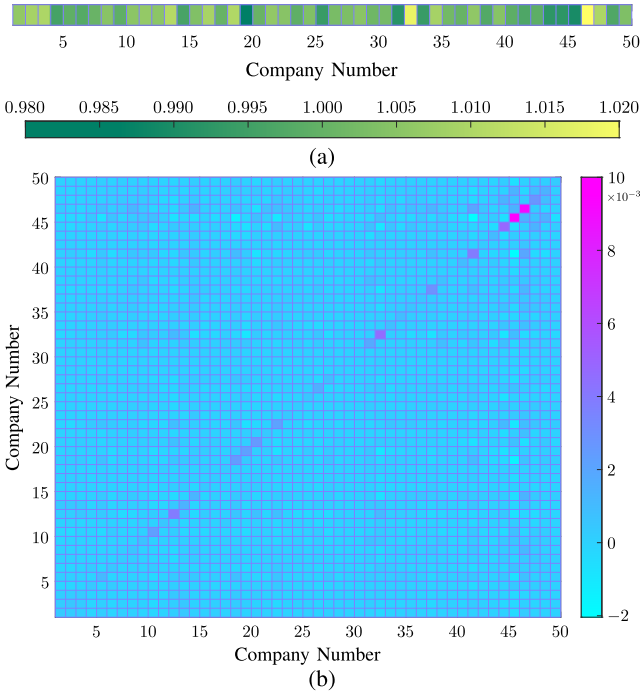

FIGURE 4. Visualization of mean return rate and covariance matrix as grid of rectangular pixels with values shown using colormap. (a) Visualization for 1 × 50 mean return rate matrix μ. (b) Visualization for 50 × 50 covariance matrix. Most of the values are concentrated near the middle (one for mean and zero for covariance) with only few outliers, which can be expected from a real-world dataset.

lies in the range [0, 1]; therefore, the algorithm was initialized with uniformly distributed random values, i.e., $\mathbf{x}_0 \in [0, 1]^n$, where n is the dimensionality of the optimization variable as explained in Remark 2.

Additionally, to evaluate the comparative performance of the BAS algorithm, we also conducted the experiments using PSO [70], GA [71], PS [72], and fmincon [73]. Among these, fmincon is a gradient-based optimizer available in MATLAB [73], while other are are well-tested nature-inspired optimization algorithms and have the focus of several studies [24], [72]. All major optimization libraries have a robust implementation of these algorithms and, therefore, can be readily used to benchmark the performance of a given optimization algorithm. We implemented the BAS algorithm in MATLAB [73]. Since MATLAB also provides an implementation of PSO, GA, PS, and fmincon, this enabled us to conduct a fair benchmark and comparative analysis of the BAS algorithm. To present fair and accurate comparison results, we manually tuned the hyperparameters of these metaheuristic optimization algorithms until we reach a satisfactory level of performance. The execution time present for each algorithm is estimated using code profiling tools provided by MATLAB, which provide accurate timing information and take care of any biases caused by OS scheduler.

Additionally, to evaluate the scalability of the BAS algorithm, we considered the different number of companies to be included in the portfolio. In this regard, we considered four different cases. In the first case, we chose a total of 5 companies out of the 50 companies initial selected, i.e., $N = 5$. We used the mean return rate and covariance matrices for those 5 companies to formulate an objective function according to (16). The cardinality constraint was used to enforce that only 3 companies can be present in the final portfolio, i.e., $k = 3$. For the second case, we chose 10 companies, i.e., $N = 10$ and $k = 5$ was used to formulate the objective function. For the third case, the number of companies was 20 and $k = 10$. Similarly, for fourth case, $N = 50$ and $k = 20$ was chosen. Note that, $\Lambda = 1$, $\beta_1 = 5$, $\beta_2 = 1$, $\beta_3 = 2$, and $\beta_4 = 10$ were used in all the experiments.

B. COMPARATIVE RESULTS AND DISCUSSION

First, we will discuss the convergence behavior of the BAS algorithm for the four experimental scenarios, as described in the previous subsection. Then we will compare the optimal results obtained using the BAS algorithm with results from

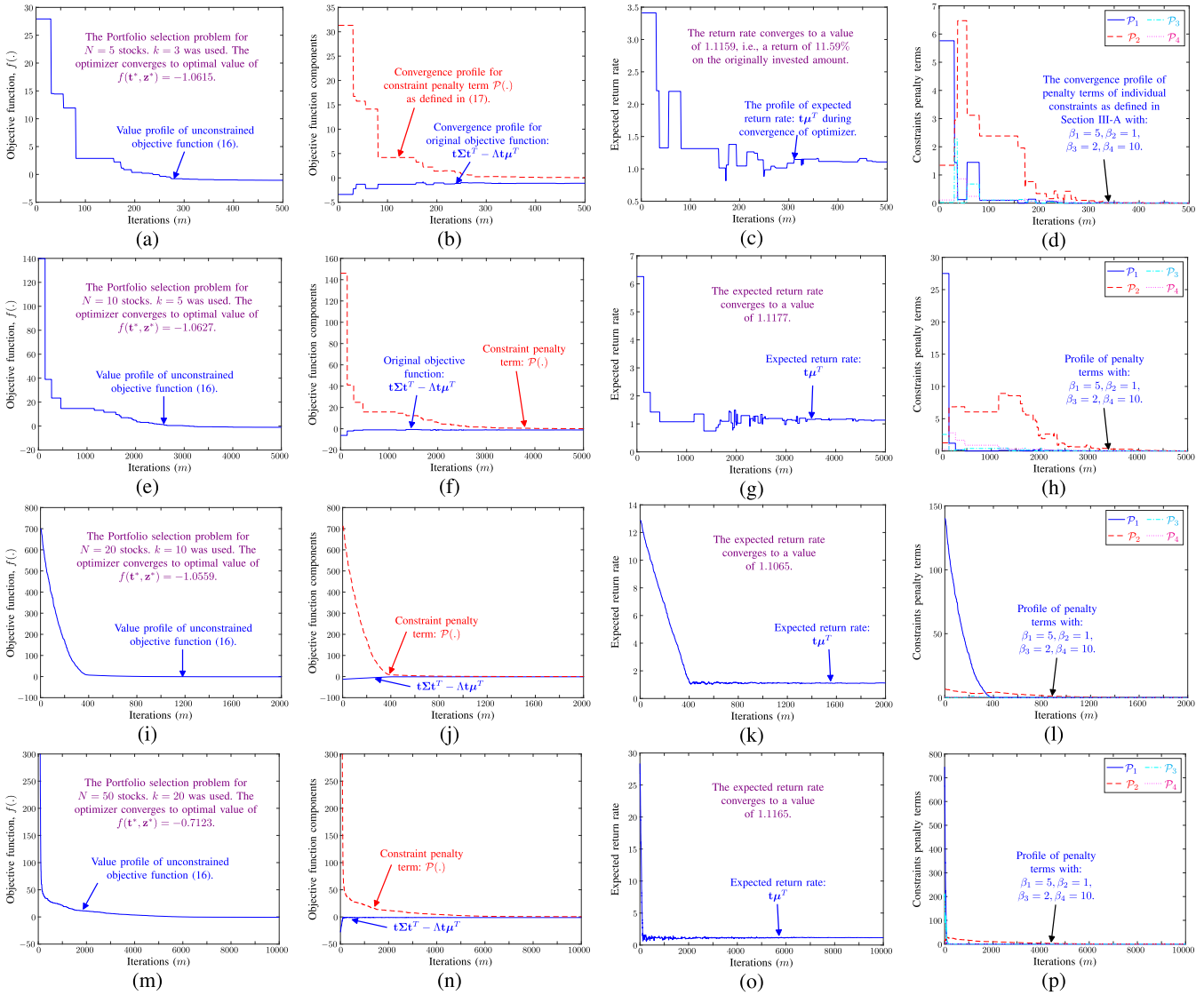

FIGURE 5. **Convergence performance of the BAORNN algorithm for different experimental scenarios along with profile of important metrices. (a)-(d) show the results for the case when 5 stocks were considered and $k = 3$, i.e., only 3 companies could be present in final portfolio. (a) shows the convergence of the unconstrained objective function. (b) shows the value of two components of the unconstrained objective function. (c) shows the evolution of return rate with number iterations. (d) shows the profile of individual constraint penalty term as defined in Section III-A. (e)-(h) show the similar results for the case of 10 companies and $k = 5$. (i)-(l) show the results for 20 companies and $k = 10$. (m)-(p) show the results for 50 companies and $k = 20$.**

PSO and GA. The comparison includes several metrics: optimal value, optimal return rate and variance, execution time, number of iterations, and number of evaluations of objective functions.

Fig. 5 shows the convergence performance of the BAS algorithm for the four different cases. Each row of figures show results for a single case. Fig. 5(a)-(d) shows the results for the first case, i.e., $N = 5, k = 3$. Fig. 5(a) shows the decay in the value of unconstrained objective function along with the number of iterations of the optimization algorithm. It can be seen that the value of objective function monotonically decreases until it reaches the minimum optimal value. Additionally, it can be seen that in the beginning, the value of objective function decreases rapidly in large steps, however, as the number of iterations increase, the reduction in value of the objective function becomes small; it can be described in term of the rule (31), which used to calculate the length of beetle's antennae. In the beginning, the length is λ_0; however, with the number of iterations increase, the length decreases by an exponential decay factor. The smaller antennae length implies that the algorithm is only able to take small steps; however, such a decaying behavior is necessary to avoid overshooting near-optimal points. Fig. 5(b) shows different two components of the unconstrained objective function. It can be seen in (16), that the objective function have two components; first component is from Markowitz's model and second is the combined penalty term for the optimization constraints $\mathcal{P}(.)$ as defined

TABLE 2. Summary of convergence profiles of the BAS algorithm.

	5 Companies	10 Companies	20 Companies	50 Companies
$f(\mathbf{t}^*, \mathbf{z}^*)$	-1.062	-1.062	-1.056	-0.712
$\mathbf{t}^*\boldsymbol{\mu}^T$	1.116	1.117	1.106	1.117
$\mathbf{t}^*\boldsymbol{\Sigma}\mathbf{t}^{*T}$	0.002	0.002	0.0004	0.0002
$\mathcal{P}(\mathbf{t}^*, \mathbf{z}^*)$	0.0505	0.0503	0.0502	0.7025
$\mathcal{P}_1(\mathbf{t}^*)$	0.0101	0.0101	0.0100	0.0135
$\mathcal{P}_2(\mathbf{t}^*, \mathbf{z}^*)$	0	0	0	0.4503
$\mathcal{P}_3(\mathbf{z}^*)$	2×10^{-8}	3×10^{-7}	6×10^{-7}	0.0087
$\mathcal{P}_4(\mathbf{z}^*)$	4×10^{-7}	2×10^{-6}	2×10^{-6}	0.0167
BAS:				
c_1	1.5	1.65	0.01	0.0612
α	0.99	0.999	0.9992	0.9995
c_2	1	1	1	1
PSO:				
Swarm Size	25	50	80	100
Neighborhood size	0.25	0.25	0.25	0.25
Max. Iterations	1000	2000	4000	10000
GA:				
Population Size	25	50	80	100
Max. Generations	500	1000	2000	5000
PS:				
Max. Evaluations	10000	20000	40000	100000

in (17). We have shown the convergence of both of these components to demonstrate that although the overall value of unconstrained objective function decreases, the value of individual components might not follow the same pattern. For example, in Fig. 5(b), the value of the Markowitz's model term increases with the number of iterations; this can be explained by considering that the initial point produced a lower value of the first term, but it violated the constraints. Therefore, such a point cannot be taken as solution of the optimization problem. As the states of optimizer evolved with iterations, the individual components also reached an optimal solution. The violation of constraints at the initial point can be seen in Fig. 5(d). Fig. 5(c) shows the profile of expected return rate $\mathbf{t}\boldsymbol{\mu}^T$. It can be seen that initially, the value of expected return rate is quite high but achieving it is unrealistic because the optimization variables violated the constraints at the initial point. However, the expected return rate reaches a realistic optimal value of 11.59%. Finally, Fig. 5(d) shows the value of individual penalty term for optimization constraints. These penalty terms are defined in Section III-A. The profile of these penalty terms shows that in the beginning, the optimization variable violated all the constraints, however, as the optimizer minimized the value of objective function these penalty terms also converged to an optimal minimum value.

Similar trends can be observed for the second, third, and fourth experimental cases. Fig. 5(e)-(h) shows the results for the case of 10 companies and cardinality constraint of $k = 5$. It can be seen that for this case, the optimizer took 5000 iterations to converge to an optimal point. Results for the case of 20 companies and 50 companies are summarized in Fig. 5(i)-(l) and Fig. 5(m)-5(p) respectively. These experimental results prove the scalability of the BAS algorithm on the real-world dataset and demonstrate its efficacy in practical scenarios. Value of different parameters at optimal point are summarized in Table 2.

Next, we conducted experiments to perform the comparison of the BAS algorithm with PSO, GA, PS, and fmincon. The comparison results are summarized in Table 3. The table presents six different metrics to measure the performance of an algorithm; the optimal value of the unconstrained objective function, the value of expected return rate and variance at the optimal point, the total execution time as measured in MATLAB, the number of iterations of respective algorithm, and the total number of evaluations of the objective function. It can be seen that in almost all the cases, BAS and PSO demonstrate almost identical performance in terms of the final optimal point, i.e., the value of the objective function and the expected return rate. The performance of the GA algorithm deteriorates as the number of companies increases. However, the computational efficiency of the BAS algorithm is much higher as compared to the PSO and GA algorithms. In terms of execution time, it can be seen that even in the best performance scenario, the BAS algorithm is at least six-folds (in case of 10 companies) faster than PSO and GA. In other scenarios, the numerical efficiency of the BAS is even higher. Remember that all the experiments were performed in MATLAB, which provided a native implementation of PSO and GA; therefore, the comparison of execution time is fair. The authors even tuned the initial parameters of PSO and GA to enhance their performance. The comparison of the number of iterations is not directly important because each optimization algorithm has a different formulation, therefore a more holistic comparison is provided by the number of evaluations of the objective function; which directly corresponds to the computational complexity of the optimization algorithm. It can be seen from Table 3 that BAS clearly outperforms PSO and GA in term of the number of evaluations of the objective function.

In order to prove the robustness and consistency of the proposed algorithm, statistical results were also generated by

TABLE 3. Comparison of BAS with PSO, GA, PS, and fmincon for different number of companies in the portfolio.

	5 Companies (k=3)					10 Companies (k=5)				
	BAS	PSO	GA	PS	fmincon#	BAS	PSO	GA	PS	fmincon
Optimal Value	-1.062	-1.062	-1.05	-1.059	-0.1275	-1.062	-1.060	-0.992	-1.05	-0.984
Expected Return**	1.116	1.116	1.102	1.114	1.112	1.117	1.111	1.104	1.104	1.110
Expected Variance***	0.002	0.004	0.001	0.003	0.0061	0.004	0.002	0.003	0.001	0.004
Execution Time†	0.015	0.422	0.647	0.36	0.387	0.105	0.727	0.786	0.950	0.708
Iterations	1000	408	651	1001	104	5000	635	630	1476	94
Func. Evaluations‡	3000	40900	60250	14650	1618	15000	63600	119910	40000	2383
	20 Companies (k=10)					**50 Companies (k=20)**				
	BAS	PSO	GA	PS	fmincon#	BAS	PSO	GA	PS	fmincon
Optimal Value	-1.056	-1.06	-0.987	1.8772	-0.984	-0.712	-0.878	0.647	49.62	-0.852
Expected Return*	1.106	1.105	1.104	1.05	1.106	1.117	1.106	1.102	0.990	1.10
Expected Variance**	0.0004	0.004	0.0002	0.009	0.0004	0.0005	0.0002	0.0009	0.370	0.0003
Execution Time†	0.117	1.069	1.256	1.443	1.040	0.203	3.245	5.010	3.591	2.098
Iterations	2000	917	937	1410	66	10000	2598	2174	1336	28
Func. Evaluations‡	6000	91800	178240	80000	3000	30000	259900	413270	200000	3031

* Value of expected return rate $\mathbf{t}^* \boldsymbol{\mu}^T$.
** Value of expected variance $\mathbf{t}^* \boldsymbol{\Sigma} \mathbf{t}^{*T}$.
† Execution time in seconds as measured in MATLAB.
‡ Total number of function evaluations.
# Gradient-based optimization algorithm.

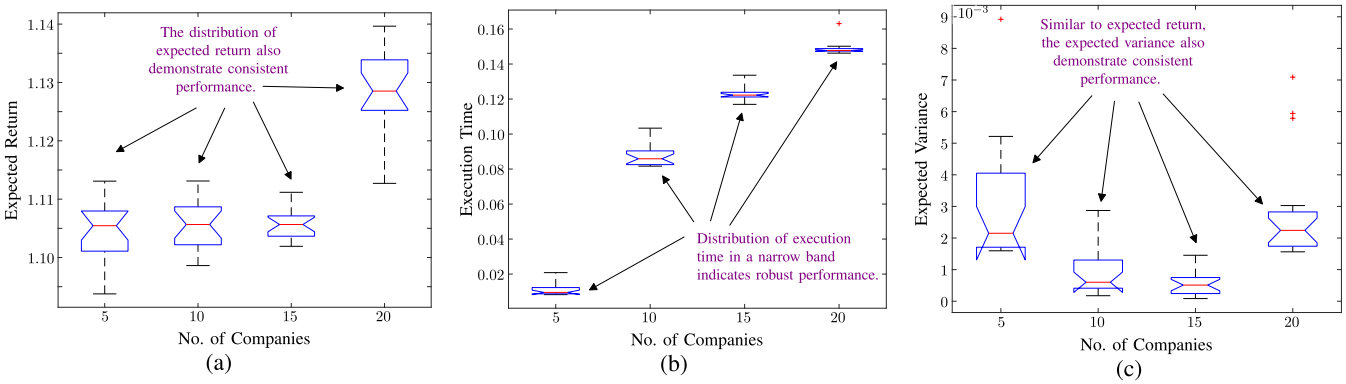

FIGURE 6. The box and whisker plot showing the distribution of performance parameters under varying initial conditions. Each experiment was repeated 20 times, and the statistics were recorded in the form of the whisker graph. (a) shows the results for execution time for different number of companies. Similarly, (b) and (c) shows the statistics for expected return and variance respectively.

TABLE 4. Standard deviation for the performance metrices using BAS algorithm.

	5 Companies	10 Companies	20 Companies	50 Companies
Execution Time	0.001	0.0067	0.0015	0.003
Expected Return	0.01	0.0062	0.0028	0.0214
Expected Variance	0.0038	0.0007	0.0004	0.006

repeating each experiment 20 times with different initial conditions. The same initialization strategy was used as described in Section IV-A. The value of performance metaheuristic, i.e., execution time, expected return, and expected variance, were recorded and presented in the form of the whisker graph, as shown in Fig. 6. The graph shows the distribution of performance metrics around median value and their overall variation under different initial conditions. Fig. 6(a) shows the results for execution time. It can be seen that the values are tightly distributed in a narrow band, which shows the consistency in the performance of the proposed algorithm and demonstrate its robustness to the initial conditions. Similar results can be observed for the expected return and expected variance in Fig. 6(b) and 6(c) respectively. The value of standard deviation for each parameter are also presented in the Table 4. The value also demonstrates that there is little influence on the performance of the algorithm if initial conditions are changed

V. CONCLUSION AND FUTURE WORK

In this paper, we applied a nature-inspired metaheuristic optimizer, called BAS, to the financial problem of portfolio selection. The BAS algorithm mimics the food foraging behavior of a beetle, and therefore have a remarkable ability to search for an optimal point in a computationally efficient manner. To apply the optimization algorithm on the portfolio selection

problem, we first formulated a constrained optimization problem inspired by the classical Markowitz model and enhanced it to include the cardinality constraint and transaction cost. Next, it is reformulated as an unconstrained optimization problem using the penalty term approach. The parameters of the optimization problem are calculated using the historical stock price dataset. The portfolio selection problem is then solved by considering a specific number of available stocks to evaluate the scalability of the BAS algorithm. Additionally, the performance is also compared to PSO, GA, and PS, which are well-studied metaheuristic optimization algorithms. The experimental results show that the BAS algorithm is, in the worst case 6 times faster as compared to the other optimization algorithm and about 25 times faster in the best case with the comparable accuracy, which proves efficacy and computational efficiency of the BAS algorithm.

The following topics for the extension of the proposed study are potential future research directions. Investigating and exploring the effect of different portfolio selection models, such as the Sharpe ratio, on the performance of the proposed algorithm. Additionally, exploring hardware-level implementation options, such as GPU and bare-metal processing, to deploy the solution in a commercial environment will also act as a potential future research direction.

REFERENCES

[1] C. R. Harvey, J. C. Liechty, M. W. Liechty, and P. Müller, "Portfolio selection with higher moments," *Quant. Finance*, vol. 10, no. 5, pp. 469–485, 2010.
[2] O. Ledoit and M. Wolf, "Nonlinear shrinkage of the covariance matrix for portfolio selection: Markowitz meets goldilocks," *Rev. Financial Stud.*, vol. 30, no. 12, pp. 4349–4388, Dec. 2017.
[3] Y. Zhu, "Uncertain optimal control with application to a portfolio selection model," *Cybern. Syst.*, vol. 41, no. 7, pp. 535–547, Sep. 2010.
[4] B. Aouni, M. Doumpos, B. Pérez-Gladish, and R. E. Steuer, "On the increasing importance of multiple criteria decision aid methods for portfolio selection," *J. Oper. Res. Soc.*, vol. 69, no. 10, pp. 1525–1542, Oct. 2018.
[5] S. Dutta, M. P. Biswal, S. Acharya, and R. Mishra, "Fuzzy stochastic price scenario based portfolio selection and its application to BSE using genetic algorithm," *Appl. Soft Comput.*, vol. 62, pp. 867–891, Jan. 2018.
[6] R. Mishra, S. Acharya, and S. Dutta, "Genetic algorithm approach for solving multi-objective fuzzy stochastic programming problem," *Int. J. Math. Oper. Res.*, vol. 11, no. 1, pp. 1–28, 2017.
[7] M. Masmoudi and F. B. Abdelaziz, "Portfolio selection problem: A review of deterministic and stochastic multiple objective programming models," *Ann. Oper. Res.*, vol. 267, nos. 1–2, pp. 335–352, Aug. 2018.
[8] F. B. Abdelaziz and M. Masmoudi, "A multiple objective stochastic portfolio selection problem with random beta," *Int. Trans. Oper. Res.*, vol. 21, no. 6, pp. 919–933, Nov. 2014.
[9] Y.-H. Chou, S.-Y. Kuo, and Y.-T. Lo, "Portfolio optimization based on funds standardization and genetic algorithm," *IEEE Access*, vol. 5, pp. 21885–21900, 2017.
[10] M.-E. Wu and W.-H. Chung, "A novel approach of option portfolio construction using the Kelly criterion," *IEEE Access*, vol. 6, pp. 53044–53052, 2018.
[11] C.-H. Chen, Y.-H. Chen, J. C.-W. Lin, and M.-E. Wu, "An effective approach for obtaining a group trading strategy portfolio using grouping genetic algorithm," *IEEE Access*, vol. 7, pp. 7313–7325, 2019.
[12] H. Markowitz, "Portfolio selection," *J. Finance*, vol. 7, no. 1, pp. 77–91, 1952.
[13] E. J. Elton, M. J. Gruber, and M. W. Padberg, "Simple criteria for optimal portfolio selection," *J. Finance*, vol. 31, no. 5, pp. 1341–1357, 1976.
[14] M. H. Davis and A. R. Norman, "Portfolio selection with transaction costs," *Math. Oper. Res.*, vol. 15, no. 4, pp. 676–713, 1990.
[15] W. Chen and W.-G. Zhang, "The admissible portfolio selection problem with transaction costs and an improved PSO algorithm," *Phys. A, Stat. Mech. Appl.*, vol. 389, no. 10, pp. 2070–2076, May 2010.
[16] W.-G. Zhang, Y.-J. Liu, and W.-J. Xu, "A possibilistic mean-semivariance-entropy model for multi-period portfolio selection with transaction costs," *Eur. J. Oper. Res.*, vol. 222, no. 2, pp. 341–349, Oct. 2012.
[17] T.-J. Chang, N. Meade, J. E. Beasley, and Y. M. Sharaiha, "Heuristics for cardinality constrained portfolio optimisation," *Comput. Oper. Res.*, vol. 27, no. 13, pp. 1271–1302, Nov. 2000.
[18] J. Gao, D. Li, X. Cui, and S. Wang, "Time cardinality constrained mean–variance dynamic portfolio selection and market timing: A stochastic control approach," *Automatica*, vol. 54, pp. 91–99, Apr. 2015.
[19] S. Ito, D. Hatano, S. Hanna, A. Yabe, T. Fukunaga, N. Kakimura, and K.-I. Kawarabayashi, "Regret bounds for online portfolio selection with a cardinality constraint," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 10588–10597.
[20] G.-Y. Ban, N. El Karoui, and A. E. Lim, "Machine learning and portfolio optimization," *Manage. Sci.*, vol. 64, no. 3, pp. 1136–1154, 2016.
[21] A. Baykasoğlu, M. G. Yunusoglu, and F. B. Özsoydan, "A GRASP based solution approach to solve cardinality constrained portfolio optimization problems," *Comput. Ind. Eng.*, vol. 90, pp. 339–351, Dec. 2015.
[22] V. Kalashnikov, F. Benita, F. López-Ramos, and A. Hernández-Luna, "Bi-objective project portfolio selection in lean six sigma," *Int. J. Prod. Econ.*, vol. 186, pp. 81–88, Apr. 2017.
[23] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 773–791, Oct. 2016.
[24] F. Petroski Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," 2017, *arXiv:1712.06567*. [Online]. Available: http://arxiv.org/abs/1712.06567
[25] M. Dorigo and T. Stützle, "Ant colony optimization: Overview and recent advances," in *Handbook of Metaheuristics*. Cham, Switzerland: Springer, 2019, pp. 311–351.
[26] A. Baykasoglu, T. Dereli, and I. Sabuncu, "An ant colony algorithm for solving budget constrained and unconstrained dynamic facility layout problems," *Omega*, vol. 34, no. 4, pp. 385–396, Aug. 2006.
[27] J. Zhao, S. Liu, M. Zhou, X. Guo, and L. Qi, "Modified cuckoo search algorithm to solve economic power dispatch optimization problems," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 4, pp. 794–806, Jul. 2018.
[28] S. Gupta and K. Deep, "A novel random walk grey wolf optimizer," *Swarm Evol. Comput.*, vol. 44, pp. 101–112, Feb. 2019.
[29] Z. Zhang, K. Wang, L. Zhu, and Y. Wang, "A Pareto improved artificial fish swarm algorithm for solving a multi-objective fuzzy disassembly line balancing problem," *Expert Syst. Appl.*, vol. 86, pp. 165–176, Nov. 2017.
[30] M. Ahmad, A. Hameed, F. Ullah, I. Wahid, S. U. Rehman, and H. A. Khattak, "A bio-inspired clustering in mobile adhoc networks for Internet of Things based on honey bee and genetic algorithm," *J. Ambient Intell. Hum. Comput.*, pp. 1–15, vol. 9, Nov. 2018.
[31] A. Baykasoglu, L. Ozbakir, and P. Tapkan, "Artificial bee colony algorithm and its application to generalized assignment problem," in *Swarm Intelligence: Focus on Ant and Particle Swarm Optimization*, vol. 1. Rijeka, Croatia: InTech Publisher, 2007.
[32] R. Liu, R. Wang, M. He, and X. Wang, "Improved artificial weed colonization based multi-objective optimization algorithm," in *Intelligent Computing, Networked Control, and Their Engineering Applications*. Singapore: Springer, 2017, pp. 181–190.
[33] L. Tighzert, C. Fonlupt, and B. Mendil, "A set of new compact firefly algorithms," *Swarm Evol. Comput.*, vol. 40, pp. 92–115, Jun. 2018.
[34] M. G. Hinchey, R. Sterritt, and C. Rouff, "Swarms and swarm intelligence," *Computer*, vol. 40, no. 4, pp. 111–113, Apr. 2007.
[35] X. Feng, Y. Wang, H. Yu, and F. Luo, "A novel intelligence algorithm based on the social group optimization behaviors," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 1, pp. 65–76, Jan. 2018.
[36] A. Munjiza and K. R. F. Andrews, "Penalty function method for combined finite-discrete element systems comprising large number of separate bodies," *Int. J. for Numer. Methods Eng.*, vol. 49, no. 11, pp. 1377–1396, Dec. 2000.
[37] J. Liu, K. L. Teo, X. Wang, and C. Wu, "An exact penalty function-based differential search algorithm for constrained global optimization," *Soft Comput.*, vol. 20, no. 4, pp. 1305–1313, Apr. 2016.

[38] L. Cheng, W. Liu, C. Yang, T. Huang, Z.-G. Hou, and M. Tan, "A neural-network-based controller for piezoelectric-actuated stick–slip devices," *IEEE Trans. Ind. Electron.*, vol. 65, no. 3, pp. 2598–2607, Mar. 2018.
[39] C. Yang, G. Peng, Y. Li, R. Cui, L. Cheng, and Z. Li, "Neural networks enhanced adaptive admittance control of optimized robot–environment interaction," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2568–2579, Jul. 2019.
[40] C. Yang, J. Luo, C. Liu, M. Li, and S.-L. Dai, "Haptics electromyography perception and learning enhanced intelligence for teleoperated robot," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1512–1521, Oct. 2019.
[41] J. Fang, L. Zhang, and H. Li, "Two-dimensional pattern-coupled sparse Bayesian learning via generalized approximate message passing," *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2920–2930, Jun. 2016.
[42] J. Fang and H. Li, "Distributed estimation of Gauss–Markov random fields with one-bit quantized data," *IEEE Signal Process. Lett.*, vol. 17, no. 5, pp. 449–452, May 2010.
[43] J. Fang, Y. Shen, F. Li, H. Li, and Z. Chen, "Support knowledge-aided sparse Bayesian learning for compressed sensing," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 3786–3790.
[44] S. Ling, H. Wang, and P. X. Liu, "Adaptive fuzzy dynamic surface control of flexible-joint robot systems with input saturation," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 1, pp. 97–107, Jan. 2019.
[45] H. Wang, P. X. Liu, X. Zhao, and X. Liu, "Adaptive fuzzy finite-time control of nonlinear systems with actuator faults," *IEEE Trans. Cybern.*, early access, May 8, 2019, doi: 10.1109/TCYB.2019.2902868.
[46] M. Shang, X. Luo, Z. Liu, J. Chen, Y. Yuan, and M. Zhou, "Randomized latent factor model for high-dimensional and sparse matrices from industrial applications," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 1, pp. 131–141, Jan. 2019.
[47] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1273–1284, May 2014.
[48] X. Luo, M. Zhou, S. Li, Z. You, Y. Xia, and Q. Zhu, "A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 579–592, Mar. 2016.
[49] H. Wang, P. Xiaoping Liu, X. Xie, X. Liu, T. Hayat, and F. E. Alsaadi, "Adaptive fuzzy asymptotical tracking control of nonlinear systems with unmodeled dynamics and quantized actuator," *Inf. Sci.*, Apr. 2018.
[50] Y. Zhang, Z. Qi, M. Yang, J. Guo, and H. Huang, "Step-width theoretics and numerics of four-point general DTZN model for future minimization using jury stability criterion," *Neurocomputing*, vol. 357, pp. 231–239, Sep. 2019.
[51] Y. Zhang, Z. Qi, B. Qiu, M. Yang, and M. Xiao, "Zeroing neural dynamics and models for various time-varying problems solving with ZLSF models as minimization-type and Euler-type special cases [research frontier]," *IEEE Comput. Intell. Mag.*, vol. 14, no. 3, pp. 52–60, Aug. 2019.
[52] Y. Shi and Y. Zhang, "Solving future equation systems using integral-type error function and using twice ZNN formula with disturbances suppressed," *J. Franklin Inst.*, vol. 356, no. 4, pp. 2130–2152, Mar. 2019.
[53] X. Jiang and S. Li, "BAS: Beetle antennae search algorithm for optimization problems," 2017, *arXiv:1710.10724*. [Online]. Available: http://arxiv.org/abs/1710.10724
[54] Y. Zhang, S. Li, and B. Xu, "Convergence analysis of beetle antennae search algorithm and its applications," 2019, *arXiv:1904.02397*. [Online]. Available: http://arxiv.org/abs/1904.02397
[55] A. H. Khan, S. Li, and X. Luo, "Obstacle avoidance and tracking control of redundant robotic manipulator: An RNN based metaheuristic approach," *IEEE Trans. Ind. Informat.*, early access, Sep. 17, 2019, doi: 10.1109/TII.2019.2941916.
[56] Z. Zhu, Z. Zhang, W. Man, X. Tong, J. Qiu, and F. Li, "A new beetle antennae search algorithm for multi-objective energy management in microgrid," in *Proc. 13th IEEE Conf. Ind. Electron. Appl. (ICIEA)*, May 2018, pp. 1599–1603.
[57] X. Yin and Y. Ma, "Aggregation service function chain mapping plan based on beetle antennae search algorithm," in *Proc. 2nd Int. Conf. Telecommun. Commun. Eng. (ICTCE)*, 2018, pp. 225–230.
[58] X. Lin, Y. Liu, and Y. Wang, "Design and research of DC motor speed control system based on improved BAS," in *Proc. Chin. Autom. Congr. (CAC)*, Nov. 2018, pp. 3701–3705.
[59] Y. Sun, J. Zhang, G. Li, Y. Wang, J. Sun, and C. Jiang, "Optimized neural network using beetle antennae search for predicting the unconfined compressive strength of jet grouting coalcretes," *Int. J. Numer. Anal. Methods Geomechanics*, vol. 43, no. 4, pp. 801–813, Mar. 2019.
[60] Q. Wu, X. Shen, Y. Jin, Z. Chen, S. Li, A. H. Khan, and D. Chen, "Intelligent beetle antennae search for UAV sensing and avoidance of obstacles," *Sensors*, vol. 19, no. 8, p. 1758, 2019.
[61] M.-J. Lin and Q.-H. Li, "A hybrid optimization method of beetle antennae search algorithm and particle swarm optimization," in *Proc. Int. Conf. Elect., Control, Automat. Robot.*, 2018, pp. 396–401.
[62] Q. Wu, H. Lin, Y. Jin, Z. Chen, S. Li, and D. Chen, "A new fallback beetle antennae search algorithm for path planning of mobile robots with collision-free capability," *Soft Comput.*, vol. 24, no. 3, pp. 2369–2380, Feb. 2020.
[63] Y. Fan, J. Shao, and G. Sun, "Optimized PID controller based on beetle antennae search algorithm for electro-hydraulic position servo control system," *Sensors*, vol. 19, no. 12, p. 2727, 2019.
[64] S. Xie, X. Chu, M. Zheng, and C. Liu, "Ship predictive collision avoidance method based on an improved beetle antennae search algorithm," *Ocean Eng.*, vol. 192, Nov. 2019, Art. no. 106542.
[65] J. Yang and Z. Peng, "Beetle-swarm evolution competitive algorithm for bridge sensor optimal placement in SHM," *IEEE Sensors J.*, early access, Aug. 13, 2019, doi: 10.1109/JSEN.2019.2934996.
[66] Q. Wu, Z. Ma, G. Xu, S. Li, and D. Chen, "A novel neural network classifier using beetle antennae search algorithm for pattern classification," *IEEE Access*, vol. 7, pp. 64686–64696, 2019.
[67] L. Wang, Q. Wu, F. Lin, S. Li, and D. Chen, "A new trajectory-planning beetle swarm optimization algorithm for trajectory planning of robot manipulators," *IEEE Access*, vol. 7, pp. 154331–154345, 2019.
[68] H. Zhu, Y. Wang, K. Wang, and Y. Chen, "Particle swarm optimization (PSO) for the constrained portfolio optimization problem," *Expert Syst. Appl.*, vol. 38, no. 8, pp. 10161–10169, Aug. 2011.
[69] B. Marjanovic. (2017). *Huge Stock Market Dataset, Version 3*. Accessed: Oct. 2019. [Online]. Available: https://www.kaggle.com/borismarjanovic/price-volume-data-for-all-us-stocks-etfs/
[70] R. C. Eberhart and Y. Shi, "Particle swarm optimization: Developments, applications and resources," in *Proc. Congr. Evol. Comput.*, vol. 1, May 2001, pp. 81–86.
[71] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.
[72] V. Torczon, "On the convergence of pattern search algorithms," *SIAM J. Optim.*, vol. 7, no. 1, pp. 1–25, Feb. 1997.
[73] *MATLAB: Global Optimization Toolbox 2018b*, Mathworks, Natick, MA, USA, 2018.

AMEER HAMZA KHAN received the B.S. degree in electrical engineering from the Pakistan Institute of Engineering and Applied Sciences, Islamabad, Pakistan, in 2015. He is currently pursuing the Ph.D. degree in optimal control of robotic systems with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong.

He was a Research Assistant with the Department of Computing, The Hong Kong Polytechnic University. His research interests include nonlinear optimization, metaheuristic algorithms, adaptive control, and machine learning.

XINWEI CAO received the bachelor's degree from Shandong University, China, in 2009, the master's degree from Tongji University, China, in 2012, and the Ph.D. degree from Fudan University, China, in 2017, all in management. She is currently an Assistant Professor (Lecturer) with Shanghai University, China. Her research interests include operations research, operational management, computational and quantitative finance, empirical asset pricing, and financial market.

VASILIOS N. KATSIKIS received the Diploma degree in mathematics from the National and Kapodistrian University of Athens, the M.Sc. degree in applied mathematics and the Ph.D. degree in mathematics from the National Technical University of Athens. He was with the Department of Economics, Division of Mathematics and Informatics, National and Kapodistrian University of Athens, as an Assistant Professor of mathematics and informatics. He has published more than 40 journal articles. His research interests include computational mathematics, matrix analysis, and linear algebra. He serves as a reviewer for many journals and congresses.

PREDRAG STANIMIROVIĆ received the Ph.D. degree in mathematics from the Faculty of Mathematics, University of Niš, Niš, Serbia. He is currently a Full Professor with the Departments of Computer Science, Faculty of Sciences and Mathematics, University of Niš, Niš. He has thirty four years of experience in scientific research in diverse fields of mathematics and computer science, which span multiple branches of numerical linear algebra, recurrent neural networks, linear algebra, symbolic computation, nonlinear optimization, and others. Within recent years, he has successfully published over 245 publications in scientific journals, including five research monographs, six text-books, and 75 peer-reviewed research articles published in conference proceedings and book chapters. His main research topics include numerical linear algebra, operations research, recurrent neural networks, and symbolic computation. He is the Editor-in-Chief of the Scientific Journal Facta Universitatis, Series: Mathematics and Informatics and section editor in Filomat and several other journals.

IVONA BRAJEVIĆ received the B.S. and M.S. degrees in mathematics from the Faculty of Mathematics, University of Belgrade, in 2006 and 2008, respectively, where she is currently pursuing the Ph.D. degree with the Computer Science Department, Faculty of Mathematics. She was a Teaching Assistant with the College of Business and Economy, Belgrade. She is currently an Assistant Professor with the University Business Academy, Serbia. She is the coauthor of seven scientific articles. Her current research interest includes nature inspired metaheuristics.

SHUAI LI received the B.E. degree in precision mechanical engineering from the Hefei University of Technology, Hefei, China, in 2005, the M.E. degree in automatic control engineering from the University of Science and Technology of China, Hefei, in 2008, and the Ph.D. degree in electrical and computer engineering from the Stevens Institute of Technology, Hoboken, NJ, USA, in 2014. He is currently leading the Robotic Lab, conducting research on robot manipulation and impedance control, multirobot coordination, distributed control, intelligent optimization and control, and legged robots. He is the Founding Editor-in-Chief of the *International Journal of Robotics and Control* and the General Co-Chair of 2018 International Conference on Advanced Robotics and Intelligent Control.

SEIFEDINE KADRY received the bachelor's degree in applied mathematics from Lebanese University, in 1999, the M.S. degree in computation from Reims University, France, and EPFL, Lausanne, in 2002, the Ph.D. degree from Blaise Pascal University, France, in 2007, and the HDR degree in engineering science from Rouen University, in 2017. He is currently working as an Associate Professor with Beirut Arab University, Lebanon. His current research interests include education using technology, smart cities, system prognostics, stochastic systems, and probability and reliability analysis. He is a Fellow of IET, ACSIT, and ABET Program Evaluator.

YUNYOUNG NAM received the B.S., M.S., and Ph.D. degrees in computer engineering from Ajou University, South Korea, in 2001, 2003, and 2007, respectively. He was a Senior Researcher with the Center of Excellence in Ubiquitous System, Stony Brook University, Stony Brook, NY, USA, from 2007 to 2010, where he was a Postdoctoral Researcher, from 2009 to 2013. He was a Research Professor with Ajou University, from 2010 to 2011. He was a Postdoctoral Fellow with the Worcester Polytechnic Institute, Worcester, MA, USA, from 2013 to 2014. He has been the Director of the ICT Convergence Rehabilitation Engineering Research Center, Soonchunhyang University, since 2017, where he is currently an Assistant Professor with the Department of Computer Science and Engineering. His research interests include multimedia database, ubiquitous computing, image processing, pattern recognition, context-awareness, conflict resolution, wearable computing, intelligent video surveillance, cloud computing, biomedical signal processing, rehabilitation, and healthcare systems.