Modified Grey Wolf Optimizer based Maximum Entropy Clustering Algorithm*

1st Jia Cai
School of Statistics and Mathematics
Big Data and Educational Statistics Application Laboratory
Guangdong University of Finance & Economics
Guangzhou, P. R. China
jiacai1999@gdufe.edu.cn

2nd Guanglong Xu
School of Statistics and Mathematics
Guangdong University of Finance & Economics
Guangzhou, P. R. China
guanglongxu2018@gmail.com

3rd Wenwen Ye
School of Statistics and Mathematics
Zhaoqing University
Zhaoqing, P. R. China
yww0314@126.com

***Abstract*—In this paper, we propose a new maximum entropy clustering algorithm by modified grey wolf optimizer (GWO), which modify the traditional GWO from twofold: First, nonlinear decay factor is constructed, which leads to flexible refined search; Second, proportional weights, which render the positions of distinct grey wolves, are adjusted adaptively according to the social hierarchy of them. Based on these two modifications, the rectified GWO could greatly improve both the accuracy and convergence rate. Experimental results on 12 benchmark functions demonstrate the effectiveness of those modifications. Furthermore, we utilize modified GWO to maximum entropy based fuzzy clustering problems. Experiments on 5 real datasets indicate the high performance and efficiency of the proposed approach.**

***Index Terms*—Grey wolf optimizer, Nonlinear decay factor, Maximum entropy, Fuzzy clustering**

I. INTRODUCTION

Clustering is one of the most popular unsupervised classification method, which partition the dataset into distinct clustering groups, such that data belonged to the same cluster are similar to each other, and data from different cluster are dissimilar. In simple words, clustering aims at separating groups with similar traits and assign them into clusters. In general, it can be classified into two categories. One is hard clustering, another one is soft (fuzzy) clustering. In hard clustering, each data point either belongs to a cluster completely or not. For instance, k-means [1] and partitioning around medoids (PAM or k-medoids) [2]. In soft clustering, instead of putting each data point into a separate cluster, a likelihood or probability of that data point in those clusters is assigned. However, several disadvantages of traditional maximum entropy based fuzzy clustering (MEC) exist in practice: (1). Many practical problems are high dimensional, and need to tune lots of parameters; (2). It is difficult to compute the partial derivative of nonconvex objective functions; (3). Traditional approaches are prone to get trapped in local optima.

Recently, swarm intelligence (SI) based algorithms [3] are popular. It is one branch of the population-based meta-heuristic optimization algorithms. Ant Colony Optimization (ACO) [4], Partical Swarm Optimization (PSO) [5], and Artificial Bee Colony (ABC) [6] are the most famous SI techniques. SI algorithms, which based on random initialization, have several advantages: (1). Only a few parameters need to be tuned; (2). Can find global optimum; (3). No requirement about the continuity of the search space, convexity and differentiability of the objective function. The common features of these SI algorithms are their ability to solve various practical problems without priori knowledge of the problem space, and can find good approximations of global optima.

Grey wolf optimizer (GWO), introduced by Seyedali Mirjalili [7], [8], is a recently proposed famous SI technique inspired by the leadership behavior and hunting mechanism of grey wolves. Compare to other SI algorithms, GWO has the merits of simple operation, high availability, fast convergence, and is beneficial to approximate the global optimal solution. The basic GWO has good stability and fast convergence, but is prone to stagnation when it approaches the optimum. We modify the traditional GWO, and make the following two modifications: (1). A novel nonlinear decay factor is introduced; (2). Proportional weights are constructed to determine the positions of the grey wolf packs adaptively.

Experiments are conducted on 12 benchmark functions with 7 evolutionary algorithms. The results on 12 benchmark functions show that the rectified GWO can effectively improve convergence speed and accuracy compared to the other evolutionary algorithms. Moreover, we also investigate the applications of the modified GWO to MEC problems. Experimental

The work described in this paper is supported partially by National Natural Science Foundation of China (11871167, 11671171), Science and Technology Program of Guangzhou (201707010228), Special Support Plan for High-Level Talents of Guangdong Province (2019TQ05X571), Project of Collaborative Innovation Development Center of Pearl River Delta Science & Technology Finance Industry (19XT01), Foundation of Guangdong Educational Committee (2019KZDZX1023), National Social Science Foundation (19AJY027).

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

results on 5 real datasets demonstrate the effectiveness of the proposed clustering algorithm.

The paper is structured as follows. In Section II, we review related work. Section III devotes to a brief review of GWO, investigates the modified GWO approach and proposes a novel MEC algorithm via rectified GWO. Section IV presents the experimental results. Section V concludes the work.

II. Related work

MEC

In practice, it is difficult to determine the membership of the data. Thus, soft clustering is widely used. The most famous ones are fuzzy c-means (FCM) [9] and MEC [10]. Denote $X = \{x_j | x_j \in R^d, j = 1, 2, ..., n\}$ as a given data set, c as the number of clusters, m as the fuzzifier parameter, v_i as the center of the $i-$th cluster, MEC aims at minimizing

$$\begin{aligned} \mathrm{J} = \min_{U,V} & \Big\{ \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m \|x_j - v_i\|^2 + \gamma \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij} \ln u_{ij} \Big\} \\ s.t. \quad & u_{ij} \in [0,1], \quad \sum_{i=1}^{c} u_{ij} = 1, \\ & 1 \le i \le c, 1 \le j \le n \end{aligned} \tag{1}$$

where $\|x_j - v_i\|^2$ denotes the distance between the pattern x_j and the cluster prototype v_i; $U \in \mathbb{R}^{c \times n}$ is the membership degree matrix consisting of $u_{ij} (i = 1, \cdots, c; j = 1, \cdots, n)$; u_{ij} denotes the grade of membership of x_j to the cluster prototype v_i; $V \in \mathbb{R}^{d \times c}$ is the cluster prototype matrix composed of all cluster prototype $v_i (i = 1, \cdots, c)$; $\gamma > 0$ is the regularization parameter of the Shannon's entropy. Here we set $m = 1$ for simplicity. The case for $m > 1$ will be the future work. Using Lagrange multiplier methods, we can achieve the solution of the MEC problem:

$$v_i = \frac{\sum_{j=1}^{n} u_{ij} x_j}{\sum_{j=1}^{n} u_{ij}}, \qquad i = 1, 2, \cdots, c. \tag{2}$$

$$u_{ij} = \frac{\exp(-\frac{\|x_j - \nu_i\|^2}{\gamma})}{\sum_{k=1}^{c} \exp(-\frac{\|x_j - v_k\|^2}{\gamma})}, \ i = 1, 2, \cdots, c; j = 1, 2, \cdots, n. \tag{3}$$

In the literature, lots of references focus on the studies of MEC method. Qian et al. [11] devised the specific cluster prototypes and fuzzy memberships jointly leveraged framework for cross-domain MEC. Zhi et al. [12] proposed a fuzzy linear discriminant analysis based maximum entropy fuzzy clustering algorithm, which can efficiently partition the datasets. Li et al. [13] applied the maximum entropy fuzzy clustering to the problem of real-time target tracking. Zhou et al. [14] investigated multiple kernel fuzzy c-means clustering algorithm. Wang et al. [15] studied a new robust maximum entropy clustering algorithm, which can effectively label outliers and has better robustness than MEC. Reference [16] investigated a novel transfer learning based MEC algorithm. Kuo et al. [17] proposed three metaheuristic-based kernel intuitionistic FCM algorithms.

Meta-heuristic optimization algorithms

Meta-heuristic optimization algorithms can be classified into three main classes: evolutionary algorithm (EA), physics-based methods and SI algorithms. Genetic Algorithms [18] are the most popular branch of EA, the others are Differential Evolution [19], Evolutionary Programming [20], [21], Evolution Strategy [22], Biogeography-Based Optimizer [23] et al. Physics-based techniques mimic physical rules, such as Big-Bang Big-Crunch [24], Gravitational Search Algorithm [25], Charged System Search [26], Central Force Optimization [27], Artificial Chemical Reaction Optimization Algorithm [28], and Black Hole algorithm [29]. The inspirations of SI techniques originate mostly from natural colonies, flock, herds, and schools. They are widely used in fuzzy clustering problems. SI algorithms such as PSO [5], Glowworm Swarm Optimization (GSO) [30], ABC [6], Spider Monkey algorithm [31], ACO [4], Kidney-Inspired optimization [32], Squirrel Search Algorithm [33], Firework Algorithms [34], and GWO algorithms have gain lots of attention.

Wu [35] proposed a new feature selection method, which combined GA with FCM. Xie et al. [36] used GWO and maximum entropy principle to extract the best feature of hyperspectral image(HSI), and then combined the feature subspace decomposition to reduce the redundant features of HSI. In order to reduce the high dimension of hyperspectral image, GWO algorithms have been widely used but not limited to dimension reduction [37], feature selection [38], car plate recognition [39] and neural networks [40]. Gupta and Deep [41] investigated the performance of GWO for large scale problems, and indicates that GWO is robust for high-dimensional problems. Moreover, Faris et al. [8] gave a detailed review of GWO related methods and applications.

III. The proposed approach

In this Section, we shall first review GWO method.

A. Brief review of GWO method

GWO simulates the leadership behavior and hunting procedure of grey wolves, which are classified into four main groups such as alpha, beta, delta and omega wolves. The alpha wolves are the leaders of the grey wolf pack, and responsible for working decisions about hunting, sleeping place, time to wake, and so on. The second level in the hierarchy of grey wolves is beta. The beta wolves are subordinate wolves that help the alpha in decision-making or other pack activities. The delta wolves control the omega wolves and provide the information to the alpha and beta wolves. The lowest ranking grey wolf is omega, which plays the role of scapegoat (Fig 1). The main phases of grey wolf hunting are as follows:

- Tracking, chasing and approaching the prey;
- Pursuing, encircling and harassing the prey until it stops moving;
- Attack towards the prey.

The mathematical model of grey wolves encircle prey during the hunt could be described as:

$$\vec{D} = | \vec{C} \cdot \vec{X}_p(t) - \vec{X}(t) | \tag{4}$$

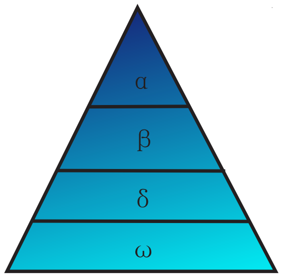
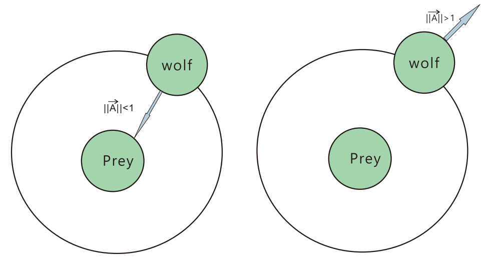


Fig. 1. Hierarchy of grey wolf (dominance decreases from top down).



(a) Attacking (b) Searching

Fig. 2. Attacking prey versus searching for prey.

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \tag{5}$$

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a} \tag{6}$$

$$\vec{C} = 2\vec{r}_2 \tag{7}$$

where we use the same notation as in [7], and $|\ |$, $\cdot$ denote component wise operations, t stands for the current iteration, $\vec{A}$ and $\vec{C}$ are coefficient vectors. $\vec{X}_p(t)$ is the position of the prey (optimum), and $\vec{X}$ denotes the position of a grey wolf, $\vec{r}_1$ and $\vec{r}_2$ are random vectors in $[0, 1]$, the components of $\vec{a}$ are linearly decreased from 2 to 0.

In order to mathematically simulate the hunting mechanism of grey wolves, [7] supposed that the alpha (best candidate solution, α), beta (β) and delta (δ) have better knowledge about the potential location of prey. Thus, the other search agents update their positions according to the positions of the alpha, beta and gamma, i.e.,

$$\begin{cases} \vec{D}_\alpha = \mid \vec{C}_1 \cdot \vec{X}_\alpha - \vec{X} \mid, \vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha \\ \vec{D}_\beta = \mid \vec{C}_2 \cdot \vec{X}_\beta - \vec{X} \mid, \vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot \vec{D}_\beta \\ \vec{D}_\delta = \mid \vec{C}_3 \cdot \vec{X}_\delta - \vec{X} \mid, \vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot \vec{D}_\delta \end{cases} \tag{8}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{9}$$

Repeating (4)-(8) constantly, we can find the fittest solution $\vec{X}_{\alpha*}$. This algorithm can be used to find the optimum of an objective function.

B. Modified grey wolf optimizer

In this section, we will introduce the modified GWO method, namely, NPGWO. We firstly propose a new nonlinear decay factor, which focuses on global search in the early stage and local search in the sequel stage; and then construct proportional weights for the position of the grey wolf packs.

1) A novel nonlinear decay factor: The grey wolves finish the hunt by attacking the prey when it stops moving. By decreasing the value of $\vec{a}$, we can mathematically model approaching the prey. That is, the fluctuation range of $\vec{A}$ is decreased by $\vec{a}$, which means when random values of $\vec{A}$ are in $[-1, 1]$, the next position of a search agent can be in any position between its current position and the position of the prey. Thus, as shown in Fig. 2, $\|\vec{A}\| < 1$ indicates the wolves attack the prey, and $\|\vec{A}\| > 1$ means wolves are searching a fitter prey [7]. Therefore, in practice, we prefer to conduct local search followed by global search. Linearly decreased $\vec{a}$ could not well embody this process. In the literature, several references focus on the modified a. Reference [42] studied a nonlinear transform of $\vec{a}$, i.e.,

$$a_1(t) = 2(1 - \frac{t^2}{t_{max}^2}) \tag{10}$$

where t denotes iteration counter, t_{max} is the maximum iteration number. While [43] used an exponential form of $\vec{a}$:

$$a_2(t) = a_{init}(1 - \sin(\frac{\pi}{\mu t_{max}} t)). \tag{11}$$

where a_{init} stands for the initial value of a, μ means nonlinear modulation index. In the sequel, we investigate a novel nonlinear decreased factor $\vec{a}$.

$$a_3(t) = \lambda a_1(t) + (1 - \lambda) a_2(t) \tag{12}$$

$$\lambda = \frac{t_{max} - t}{t_{max}} \tag{13}$$

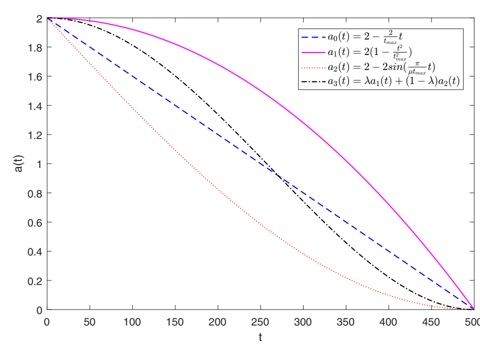


Fig. 3. Distinct forms of decay factor a.

Unlike traditional linearly decay rate of $\vec{a}$, Fig. 3 indicates that the decreasing rate of $\vec{a}$ depends on t, which is adaptive for flexible refined search.

2) Strategy of proportional weights: Grey wolves have the ability to recognize the location of prey and encircle them. The hunt is usually guided by the alpha. The beta and delta might participate in hunting occasionally. Thus, the position of the alpha, beta and delta may affect the hunting behavior and outcome. (9) does not render such differences, however. Hence, we propose a strategy of proportional weights to describe the importance of distinct grey wolves according to their social hierarchy, which is motivated by the discussion for Firework Algorithms [34]. Define

$$f_q(n) = \frac{1}{1 + e^{(n-1)/q}} \tag{14}$$

where q is the trajectory index affecting the decay behavior of the transfer function. We choose $n = 0, 1, 2$ to indicate the positions of the alpha, beta and delta, respectively,

$$w_n = \frac{f_q(n)}{\sum_{n=0}^{2} f_q(n)}. \tag{15}$$

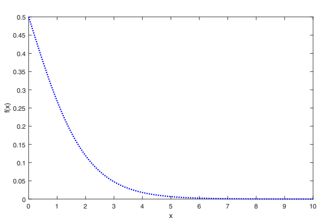
The positions of grey wolf packs could be described as

$$\overrightarrow{X}(t+1) = w_0\overrightarrow{X_1} + w_1\overrightarrow{X_2} + w_2\overrightarrow{X_3} \tag{16}$$

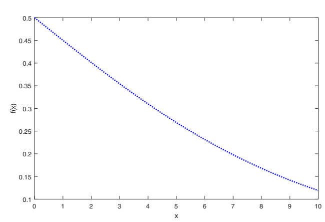
Large q will results in $\vec{a}$ decays like a line (see Fig. 4).

TABLE I
DISTINCT WEIGHTS ACCORDING TO DIFFERENT VALUES OF q.

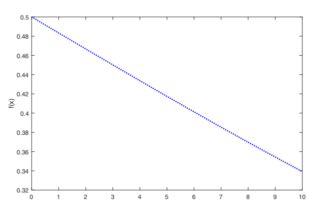
q	*1*	*5*	*10*	*15*	*20*	*25*
ω_0	0.5630	0.3700	0.3508	0.3448	0.3419	0.3401
ω_1	0.3028	0.3331	0.3333	0.3333	0.3333	0.3333
ω_2	0.1342	0.2969	0.3159	0.3219	0.3248	0.3265



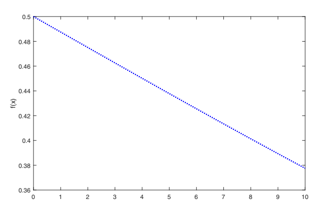
(a) q=1



(b) q=5



(c) q=15



(d) q=20

Fig. 4. Different Types of Transfer function.

C. Maximum entropy based fuzzy clustering algorithm via modified grey wolf optimizer (MEC-NPGWO)

Traditional maximum entropy based fuzzy clustering method finds the minimum of (1) by Lagrange multiplier method, which needs to find the optimal membership u_{ij} and cluster centers v_i $(i = 1, \cdots, c; j = 1, \cdots, n)$ by iteration. Here we use modified GWO method, which is easy to realize and can achieve satisfactory results. The basic idea is to unfold the membership degree matrix as depicted in Fig. 5. Each grey wolf consists of $s = cn$ features, and the search process of MEC-NPGWO begins with creating p random grey wolves based on (1), which have dimension s and whose values belong to $[0, 1]$. We sort them down with the biggest in front, the wolves corresponded to the first three minimum entropies are named α, β and δ wolf successively. The rest grey wolves are considered as ω wolves. Then, it update the positions of grey wolves according to (16), and finally output $\overrightarrow{X}_{\alpha^*}$ until maximum number of iterations or algorithm convergence is satisfied. Details could be found in Algorithm 1 and the work frame of it is addressed in Fig. 6.

Algorithm 1 MEC-NPGWO.

Input:
Maximum iteration counter t_{max}, number of grey wolves p, dimension s, modulation index μ, trajectory index q;

Output:
Membership degree matrix U $(\overrightarrow{X}_{\alpha^*})$;

1: Initialize the grey wolf population $\overrightarrow{X}_i = (X_{i,1}, \cdots, X_{i,s})$ $(i = 1, \cdots, p)$;
2: Evaluate the fitness value of each search agent; $\hat{J} = (J_1, \cdots, J_p)$;
3: **while** $t < t_{max}$ **do**
4: $[\hat{J}, index] = sort(\hat{J})$:
5: $X_\alpha = X(index(1))$, $X_\beta = X(index(2))$, $X_\delta = X(index(3))$;
6: Compute $a(t)$ via (12) and (13);
7: Calculate A, C, D_α, D_β and D_δ according to (4)-(8);
8: Compute w_0, w_1 and w_2 via (14) and (15);
9: Update the positions of grey wolf packs by (16);
10: Evaluate the fitness value of each search agent; $\hat{J} = (J_1, \cdots, J_p)$;
11: t=t+1;
12: **end while**
13: **return** $\overrightarrow{X}_{\alpha^*}$;

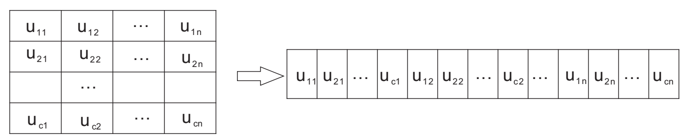


Fig. 5. Transformation of the membership degree matrix.

As demonstrated in [7], the proposed social hierarchy assists GWO to save the best solutions obtained so far over the course of iteration, and the hunting method allows candidate solutions to locate the probable positions of the prey. Exploration and exploitation are guaranteed by the adaptive values of a and $\vec{A}$, which allow GWO to smoothly transition between exploration and exploitation. Here, we modify the decay rate of a and the update rule of $\vec{X}$. Thus they can both solve optimization problems due to the same algorithmic ground. In fact, in the following experiments, we can see that those modifications are effective. However, theoretical analysis remains open.

IV. EXPERIMENTAL RESULTS

In the first part, we compare NPGWO with the other 7 SI algorithms to testify the convergence and stability of NPGWO, namely, Salp Swarm Algorithm (SSA) [44], Ant Lion Opti-

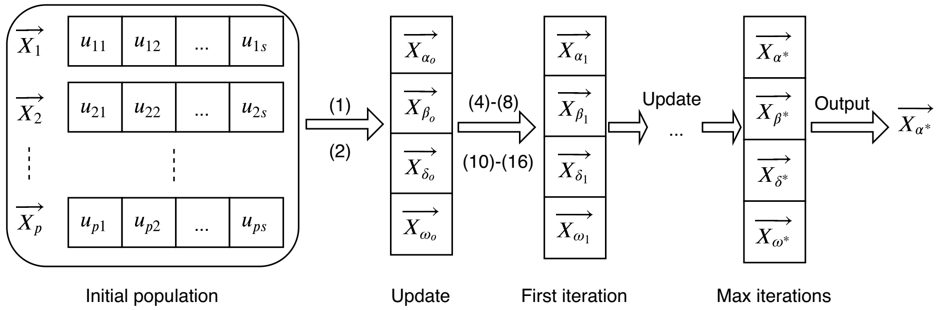


Fig. 6. Work frame of MEC-NPGWO algorithm.

mizer (ALO) [45], PSO, Grasshopper Optimisation Algorithm (GOA) [46], GWO, MGWO [42], and EEGWO [47]. In the second part, we compare the proposed MEC-NPGWO with the other 9 clustering algorithms on both synthetic datasets and real datasets.

A. Performance of NPGWO

To test the convergence of the proposed method, NPGWO algorithm is benchmarked on 12 benchmark functions with the other 7 SI algorithms. These benchmark functions are listed in Table II, where ‘Range’ is the boundary of the function’s search space, and $f_{\min}$ is the optimum.

The benchmark functions used are minimization functions and can be divided into three groups: unimodal ($F_1 - F_4$), mutimodal ($F_5 - F_7$), and fixed dimension multimodal ($F_8 - F_{12}$). They are the classical functions utilized by many researchers [20], [48]–[51].

Setting: The NPGWO method is run 50 times on each benchmark function, and the average results are reported. The statistical results (average and standard deviation) are elucidated in Table III. For the proposed approach, we choose maximum iteration number $t_{max} = 500$ ($t_{max} = 3000$ for real datasets), $p = 30$, $\mu = 2$ and q from the integer sets $\{1, 2, \cdots, 20\}$ since large q will lead to almost equal weights, which does not embody the social hierarchy of grey wolves. For PSO method, the velocity of particle is set as $V_{max} = 5$, $c1 = c2 = 0.5$. We choose $c_{max} = 1$, $c_{min} = 4E - 4$ for GOA method. For PSO-KIFCM method, we choose the Gaussian kernel width σ as the minimum distance between data points. Experimental results on benchmark functions are given in Table III.

Results: Table III indicates that the modified GWO, namely, NPGWO is able to provide very competitive results. It outperforms all others except in F_9. In contrast to the unimodal functions, multimodal functions have many local optima, which makes them suitable for benchmarking the exploration ability of an algorithm. The results in Table III indicate that NPGWO is able to achieve more satisfactory results (good approximation of the optimum with smaller standard deviation) compared to well-known meta-heuristic optimization techniques. To further demonstrate the efficiency of the proposed algorithm, in the sequel, MEC-NPGWO is employed on both simulated datasets and real datasets.

B. Clustering results of MEC-NPGWO

In this Section, we compare the proposed algorithm with another extra 3 algorithms, namely, FCM, MEC, and PSO-KIFCM [17]. We first investigate the performance of the proposed algorithm on synthetic datasets, and then on real datasets, all the experiments are repeated 50 times, and we report the average results. For clustering results, we use three famous indices to testify the performance of clustering, namely, Jaccard Coefficient (JC), Fowlkes and Mallows Index (FMI), and Ran Index (RI). which take values from the interval $[0, 1]$. For these three indices, the bigger the better.

Simulated dataset: We construct three different types of synthetic datasets as the following:

- **Simulation 1**: there are two clusters. We choose 200 points equally from the interval $[-0.3, \frac{6}{5}\pi - 0.3]$ for x_1, and set $y_1 = sinx_1 + \frac{1}{10}\varepsilon_1$, $y_2 = sinx_1 + \frac{1}{10}\varepsilon_1 + 0.8$, where $\varepsilon_1 \sim N(0, 1)$.
- **Simulation 2**: there are two clusters. We choose 200 points equally from the interval $[-0.3, \frac{6}{5}\pi - 0.3]$ ($[\frac{\pi}{2} - 0.25, 2\pi - 0.25]$) for x_1 (x_2), and set $y_1 = sinx_1 + \frac{1}{10}\varepsilon_1$, $y_2 = cosx_2 + \frac{1}{10}\varepsilon_1 + 0.4$, where $\varepsilon_1 \sim N(0, 1)$.
- **Simulation 3**: there are three clusters. For the first cluster, We generate x_1, which is composed of 100 random numbers uniformly distributed from the interval $[0, 150]$, and generate vectors y_1 and z_1 in the same manner but from a different interval, namely, $[0, 120]$. For the second and third clusters, we generate x_2, y_2, z_2 and x_3, y_3, z_3 in the same manner, i.e., they are composed of 100 random numbers uniformly distributed from some interval. Specifically, the components of x_2, y_2, and z_2 are obtained from the interval $[60, 180]$. While the components of x_3, y_3, and z_3 are from the intervals $[120, 240]$, $[0, 130]$, and $[100, 240]$, respectively.

The visualization of the data could be found in Fig. 7.

Results Table IV suggests that MEC-NPGWO could obtain the largest JC, FMI, and RI (except on Simulation 2 dataset) values compared to others. The results on three simulated datasets demonstrate that MEC-NPGWO shows high performance in clustering problems. The next section shows the performance of it for real datasets.

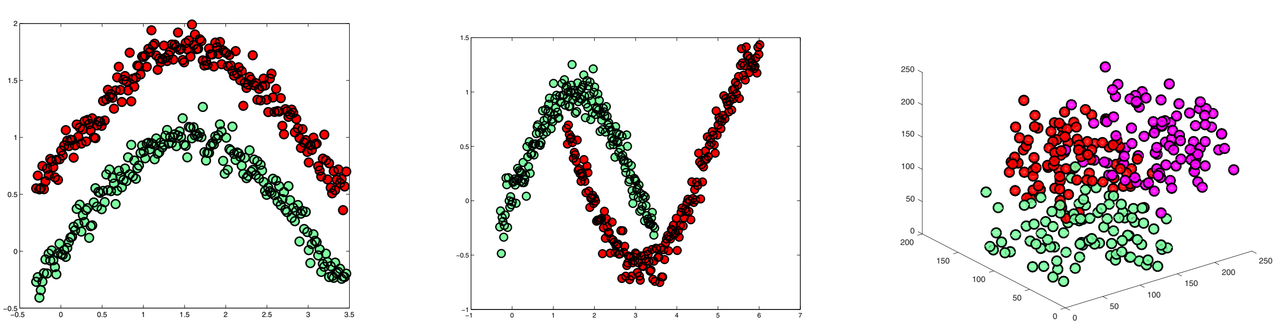
Real datasets: We select 5 datasets Iris, Wine, Breast,

TABLE II
BENCHMARK FUNCTIONS.

Function	*d*	*Range*	$f_{\min}$
$F_1(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10,10]	0.0000
$F_2(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	[-100,100]	0.0000
$F_3(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30,30]	0.0000
$F_4(x) = \sum_{i=1}^{n} ix_i^4 + random[0,1]$	30	[-1.28,1.28]	0.0000
$F_5(x) = \sum_{i=1}^{n} [x_i^2 - 10cos(2\pi x_i) + 10]$	30	[-5.12,5.12]	0.0000
$F_6(x) = -20exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	30	[-32,32]	0.0000
$F_7(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})) + 1$	30	[-600,600]	0.0000
$F_8(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5,5]	-1.0316
$F_9(x) = -\sum_{i=1}^{4}(c_i exp(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2)$	6	[0,1]	-3.3200
$F_{10}(x) = -\sum_{i=1}^{5}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0,10]	-10.1532
$F_{11}(x) = -\sum_{i=1}^{7}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0,10]	-10.4028
$F_{12}(x) = -\sum_{i=1}^{10}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0,10]	-10.5363

TABLE III
RESULTS ON 12 BENCHMARK FUNCTIONS.

Function	*Metrics*	*SSA*	*ALO*	*PSO*	*GOA*	*GWO*	*MGWO*	*EEGWO*	*NPGWO*
F1	ave	2.2268	5.49E+01	3.14E-02	1.51E+01	0.0000	0.0000	0.0000	0.0000
	std	1.6967	5.44E+01	4.55E-02	1.36E+01	0.0000	0.0000	0.0000	0.0000
F2	ave	1.54E+03	4.15E+03	7.77E+01	3.48E+03	1.10E-05	2.04E-06	0.0000	0.0000
	std	1.06E+03	1.66E+03	2.86E+01	2.04E+03	1.82E-05	1.23E-05	0.0000	0.0000
F3	ave	1.44E+02	3.08E+02	8.58E+01	6.08E+03	2.71E+01	2.69E+01	2.87E+01	2.89E+01
	std	1.67E+02	4.03E+02	4.46E+01	9.79E+03	8.24E-01	6.58E-01	1.72E-01	3.60E-02
F4	ave	1.89E-01	2.43E-01	1.71E-01	4.11E-02	1.69E-03	1.42E-03	1.03E-04	5.90E-05
	std	7.30E-02	9.28E-02	5.37E-02	1.55E-02	1.09E-03	9.37E-04	8.75E-05	5.307E-05
F5	ave	5.30E+01	8.11E+01	5.50E+01	9.73E+01	2.7253	0.1055	0.0000	0.0000
	std	1.71E+01	2.78E+01	1.52E+01	3.55E+01	3.7644	0.7463	0.0000	0.0000
F6	ave	2.6978	4.8451	2.99E-01	5.4875	0.0000	0.0000	0.0000	0.0000
	std	1.0120	1.0120	4.98E-01	1.1694	0.0000	0.0000	0.0000	0.0000
F7	ave	1.62E-02	5.93E-02	8.75E-03	1.1306	2.57E-3	1.46E-03	0.0000	0.0000
	std	9.63E-03	2.51E-02	9.58E-03	1.31E-01	5.81E-03	4.71E-03	0.0000	0.0000
F8	ave	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316
	std	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0005	0.0000
F9	ave	-3.2318	-3.2844	-3.2707	-3.2784	-3.2441	-3.2661	-2.4240	-3.25
	std	5.93E-02	5.62E-02	5.95E-02	5.97E-02	1.11E-01	8.93E-03	3.12E-01	8.41E-02
F10	ave	-7.7474	-5.9121	-7.6612	-5.8262	-9.0641	-9.4904	-2.8010	-1.0153E+01
	std	3.2387	3.0667	3.1228	3.2414	2.2664	1.8224	0.6667	0.0000
F11	ave	-8.6116	-7.5113	-8.9210	-5.7634	-1.01E+01	-1.02E+01	-2.8388	-1.04E+01
	std	3.1338	3.1619	2.7454	3.6898	1.2563	1.0475	0.6891	0.0000
F12	ave	-9.5021	-7.4270	-9.3720	-5.5061	-1.0429E+01	-1.02E+01	-3.0784	-1.0536E+01
	std	2.6272	3.5333	2.5877	3.4176	7.57E-01	1.5446	0.7700	5.32E-07



(a) Simulation 1 (b) Simulation 2 (c) Simulation 3

Fig. 7. Visualization of the simulated datasets.

TABLE IV
CLUSTERING RESULTS ON SYNTHETIC DATASETS.

Data	*Metrics*	*SSA*	*ALO*	*PSO*	*GOA*	*FCM*	*MEC*	*GWO*	*PSO-KIFCM*	*MEC-NPGWO*
Simulation 1	JC	0.4538	0.3950	0.3317	0.3315	0.4676	0.3345	0.3354	0.4636	**0.4871**
	FMI	0.6243	0.4009	0.4982	0.4980	0.6373	0.5013	0.5024	0.6335	**0.6551**
	RI	0.6241	0.4988	0.4992	0.4992	0.6369	0.4998	0.5000	0.6343	**0.6559**
Simulation 2	JC	0.3436	0.4987	0.3313	0.3311	0.4063	0.3632	0.3632	0.4321	**0.5504**
	FMI	0.5115	0.7062	0.4977	0.4975	0.5778	0.5328	0.5328	0.6043	**0.7100**
	RI	0.5072	0.4987	0.4989	0.4987	0.5770	0.5339	0.5339	**0.7229**	0.7105
Simulation 3	JC	0.3364	0.3296	0.1989	0.1998	0.4481	0.6583	0.6655	0.5504	**0.6658**
	FMI	0.5035	0.5716	0.3318	0.3330	0.6265	0.7940	0.7992	0.7100	**0.7994**
	RI	0.6671	0.3356	0.5575	0.5532	0.7012	0.8635	0.8670	0.7105	**0.8671**

TABLE VI
CLUSTERING RESULTS ON REAL DATASETS.

Data	*Metrics*	*SSA*	*ALO*	*PSO*	*GOA*	*FCM*	*MEC*	*GWO*	*PSO-KIFCM*	*MEC-NPGWO*
Iris	JC	0.3491	0.3274	0.1937	0.2083	0.5653	0.5935	0.5858	0.5717	**0.6217**
	FMI	0.5175	0.2696	0.3246	0.3448	0.7440	0.7449	0.7388	0.7358	**0.7667**
	RI	0.6814	0.3334	0.5553	0.5668	0.7598	0.8322	0.8278	0.7892	**0.8464**
Wine	JC	0.2920	0.3340	0.2067	0.2072	0.4999	**0.8725**	0.8004	0.5011	**0.8725**
	FMI	0.4519	0.5813	0.3426	0.3433	0.6792	**0.9319**	0.8891	0.6829	**0.9319**
	RI	0.6340	0.3340	0.5593	0.5597	0.7211	**0.9543**	0.9256	0.7156	**0.9543**
Breast	JC	0.1130	0.1300	0.0847	0.0912	0.2558	0.2649	0.2577	0.2295	**0.2832**
	FMI	0.2031	0.2304	0.1562	0.1671	**0.4830**	0.4193	0.4103	0.3949	0.4415
	RI	0.7405	0.7355	0.7145	0.7296	0.5813	0.8011	0.7966	0.6950	**0.8126**
Zoo	JC	0.1416	0.2331	0.0890	0.1148	0.4544	0.4737	0.4669	0.4737	**0.6166**
	FMI	0.2560	0.4828	0.1681	0.2113	0.6741	0.6588	0.6529	0.6432	**0.7696**
	RI	0.7190	0.2331	0.6838	0.6976	0.7202	0.8634	0.8614	0.8279	**0.9024**
Hayes-Roth	JC	0.2394	0.2586	0.2052	0.2140	0.2046	0.2338	0.2444	**0.2747**	0.2711
	FMI	0.3864	0.3988	0.3408	0.3527	0.3399	0.3791	0.3930	**0.4311**	0.4266
	RI	0.5715	0.3586	0.5462	0.5453	0.5425	0.5664	0.5733	**0.6013**	0.5866

ZOO, and $\mathrm{Hayes-Roth}$ from the UCI repository [1]. The clustering results on real datasets are given in Table VI.

TABLE V
DATA STRUCTURES: NUMBER OF DATA (N), DATA DIMENSION (D), AND NUMBER OF CLASSES (C).

Data	*n*	*d*	*c*
Iris	150	4	3
Wine	178	13	3
Breast	106	9	6
Zoo	101	16	7
Hayes-Roth	160	4	3

Results: Table VI indicates that MEC-NPGWO algorithm achieves the maximum value on almost all the datasets, except that FMI value achieved by MEC-NPGWO is slightly less than it obtained via the FCM method on Breast dataset, and the results achieved are less than those obtained by PSO-KIFCM on Hayes-roth dataset, which demonstrates the superior performance of it again. This study shows that the two modifications employed in MEC-NPGWO has merit among the current well known heuristics such as SSA, ALO, GOA, and GWO.

[1] http://archive.ics.uci.edu/ml/index.php.

V. CONCLUSION

In this paper, we first modify the conventional GWO method by introducing nonlinear decay factor and constructing adaptive proportional weights. Experiments on 12 benchmark functions demonstrate the effectiveness and efficiency of these modifications. We use modified GWO to optimize the membership degree matrix of MEC problems. Experiments on both synthetic datasets and real datasets indicate high performance of this novel algorithm, which give insights to a novel understanding of the MEC related problems. Therefore the proposed approach could be applied in but not limited to global optimization of non-differentiable complex functions, non-convex and nonlinear constrained optimization problem and combinatorial optimization problem [8].

REFERENCES

[1] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[2] H. S. Park and C. H. Jun, "A simple and fast algorithm for k-medoids clustering," *Expert Systems with Applications*, vol. 36, no. 2-part-P2, pp. 3336–3341.

[3] G. Beni, *Swarm Intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2019, pp. 1–28.

[4] M. Dorigo and G. Di Caro, "Ant colony optimization: a new meta-heuristic," in *Proceedings of The 1999 Congress on Evolutionary Computation*, vol. 2. IEEE, 1999, pp. 1470–1477.

[5] R. Eberhart and J. Kennedy, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4. Citeseer, 1995, pp. 1942–1948.

[6] X. S. Yang, "Engineering optimizations via nature-inspired virtual bee algorithms," in *International Work-Conference on the Interplay Between Natural and Artificial Computation*. Springer, 2005, pp. 317–323.

[7] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[8] H. Faris, I. Aljarah, M. A. Al-Betar, and S. Mirjalili, "Grey wolf optimizer: a review of recent variants and applications," *Neural computing and applications*, vol. 30, no. 2, pp. 413–435, 2018.

[9] J. C. Bezdek, R. Ehrlich, and W. Full, "Fcm: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2-3, pp. 191–203, 1984.

[10] N. B. Karayiannis, "Meca: Maximum entropy clustering algorithm," in *Proceedings of 1994 IEEE 3rd International Fuzzy Systems Conference*. IEEE, 1994, pp. 630–635.

[11] P. Qian, Y. Jiang, Z. Deng, L. Hu, S. Sun, S. Wang, and R. F. Muzic, "Cluster prototypes and fuzzy memberships jointly leveraged cross-domain maximum entropy clustering," *IEEE Transactions on Cybernetics*, vol. 46, no. 1, pp. 181–193, 2015.

[12] X. B. Zhi, J. L. Fan, and F. Zhao, "Fuzzy linear discriminant analysis-guided maximum entropy fuzzy clustering algorithm," *Pattern Recognition*, vol. 46, no. 6, pp. 1604–1615, 2013.

[13] L. Li, H. Ji, and X. Gao, "Maximum entropy fuzzy clustering with application to real-time target tracking," *Signal Processing*, vol. 86, no. 11, pp. 3432–3447, 2006.

[14] J. Zhou, C. P. Chen, and L. Chen, "Maximum-entropy-based multiple kernel fuzzy c-means clustering algorithm," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2014, pp. 1198–1203.

[15] S. Wang, F. C. Korris, Z. Deng, D. Hu, and X. Wu, "Robust maximum entropy clustering algorithm with its labeling for outliers," *Soft Computing*, vol. 10, no. 7, pp. 555–563, 2006.

[16] S. Sun, Y. Jiang, and P. Qian, "Transfer learning based maximum entropy clustering," in *2014 4th IEEE International Conference on Information Science and Technology*. IEEE, 2014, pp. 829–832.

[17] R. Kuo, T. Lin, F. E. Zulvia, and C. Tsai, "A hybrid metaheuristic and kernel intuitionistic fuzzy c-means algorithm for cluster analysis," *Applied Soft Computing*, vol. 67, pp. 299–308, 2018.

[18] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992.

[19] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[20] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[21] D. B. Fogel, *Artificial intelligence through simulated evolution*. Wiley-IEEE Press, 1998.

[22] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[23] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[24] O. K. Erol and I. Eksin, "A new optimization method: big bang–big crunch," *Advances in Engineering Software*, vol. 37, no. 2, pp. 106–111, 2006.

[25] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "Gsa: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[26] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: charged system search," *Acta Mechanica*, vol. 213, no. 3-4, pp. 267–289, 2010.

[27] R. Formato, "Central force optimization: a new metaheuristic with applications in applied electromagnetics. prog electromagn res 77: 425–491," 2007.

[28] B. Alatas, "Acroa: artificial chemical reaction optimization algorithm for global optimization," *Expert Systems with Applications*, vol. 38, no. 10, pp. 13 170–13 180, 2011.

[29] A. Hatamlou, "Black hole: A new heuristic optimization approach for data clustering," *Information Sciences*, vol. 222, pp. 175–184, 2013.

[30] K. Krishnanand and D. Ghose, "Glowworm swarm optimisation: a new method for optimising multi-modal functions," *International Journal of Computational Intelligence Studies*, vol. 1, no. 1, pp. 93–119, 2009.

[31] A. A. Al Azza, A. A. Al-Jodah, and F. J. Harackiewicz, "Spider monkey optimization: A novel technique for antenna optimization," *IEEE Antennas and Wireless Propagation Letters*, vol. 15, pp. 1016–1019, 2015.

[32] N. S. Jaddi, J. Alvankarian, and S. Abdullah, "Kidney-inspired algorithm for optimization problems," *Communications in Nonlinear Science and Numerical Simulation*, vol. 42, pp. 358–369, 2017.

[33] M. Jain, V. Singh, and A. Rani, "A novel nature-inspired algorithm for optimization: Squirrel search algorithm," *Swarm and Evolutionary Computation*, vol. 44, pp. 148–175, 2019.

[34] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *International Conference in Swarm Intelligence*. Springer, 2010, pp. 355–364.

[35] J. Wu, "Unsupervised intrusion feature selection based on genetic algorithm and fcm," in *Information Engineering and Applications*. Springer, 2012, pp. 1005–1012.

[36] F. Xie, C. Lei, F. Li, D. Huang, and J. Yang, "Unsupervised hyperspectral feature selection based on fuzzy c-means and grey wolf optimizer," *International Journal of Remote Sensing*, vol. 40, no. 9, pp. 3344–3367, 2019.

[37] E. Emary, W. Yamany, A. E. Hassanien, and V. Snasel, "Multi-objective gray-wolf optimization for attribute reduction," *Procedia Computer Science*, vol. 65, pp. 623–632, 2015.

[38] E. Emary, H. M. Zawbaa, and A. E. Hassanien, "Binary grey wolf optimization approaches for feature selection," *Neurocomputing*, vol. 172, pp. 371–381, 2016.

[39] J. Zhao and X. Wang, "Vehicle-logo recognition based on modified hu invariant moments and svm," *Multimedia Tools and Applications*, vol. 78, no. 1, pp. 75–97, 2019.

[40] S. Amirsadri, S. J. Mousavirad, and H. Ebrahimpour-Komleh, "A levy flight-based grey wolf optimizer combined with back-propagation algorithm for neural network training," *Neural Computing and Applications*, vol. 30, no. 12, pp. 3707–3720, 2018.

[41] S. Gupta and K. Deep, "Performance of grey wolf optimizer on large scale problems," in *AIP conference proceedings*, vol. 1802, no. 1. AIP Publishing, 2017, p. 020005.

[42] N. Mittal, U. Singh, and B. S. Sohi, "Modified grey wolf optimizer for global engineering optimization," *Applied Computational Intelligence and Soft Computing*, vol. 2016, p. 8, 2016.

[43] J. C. Yang and W. Long, "Improved grey wolf optimization algorithm for constrained mechanical design problems," in *Applied Mechanics and Materials*, vol. 851. Trans Tech Publ, 2016, pp. 553–558.

[44] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Advances in Engineering Software*, vol. 114, pp. 163–191, 2017.

[45] S. Mirjalili, "The ant lion optimizer," *Advances in Engineering Software*, vol. 83, pp. 80–98, 2015.

[46] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: theory and application," *Advances in Engineering Software*, vol. 105, pp. 30–47, 2017.

[47] W. Long, J. Jiao, X. Liang, and M. Tang, "An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization," *Engineering Applications of Artificial Intelligence*, vol. 68, pp. 63–80, 2018.

[48] J. G. Digalakis and K. G. Margaritis, "On benchmarking functions for genetic algorithms," *International Journal of Computer Mathematics*, vol. 77, no. 4, pp. 481–506, 2001.

[49] M. Molga and C. Smutnicki, "Test functions for optimization needs," *Test Functions for Optimization Needs*, vol. 101, 2005.

[50] S. Mirjalili and A. Lewis, "S-shaped versus v-shaped transfer functions for binary particle swarm optimization," *Swarm and Evolutionary Computation*, vol. 9, pp. 1–14, 2013.

[51] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Computing and Applications*, vol. 25, no. 3-4, pp. 663–681, 2014.