Computer Science • 24(3) 2023 https://doi.org/10.7494/csci.2023.24.3.4962

Gyanaranjan Shial
Sabita Sahoo
Sibarama Panigrahi

A NATURE-INSPIRED HYBRID PARTITIONAL CLUSTERING METHOD BASED ON GREY WOLF OPTIMIZATION AND JAYA ALGORITHM

Abstract *This paper presents a hybrid meta-heuristic algorithm that uses the grey wolf optimization (GWO) and the JAYA algorithm for data clustering. The idea is to use the explorative capability of the JAYA algorithm in the exploitative phase of GWO to form compact clusters. Here, instead of using only one best and one worst solution for generating offspring, the three best wolves (alpha, beta and delta) and three worst wolves of the population are used. So, the best and worst wolves assist in moving towards the most feasible solutions and simultaneously it helps to avoid from worst solutions; this enhances the chances of trapping at local optimal solutions. The superiority of the proposed algorithm is compared with five promising algorithms; namely, the sine-cosine (SCA), GWO, JAYA, particle swarm optimization (PSO), and k-means algorithms. The performance of the proposed algorithm is evaluated for 23 benchmark mathematical problems using the Friedman and Nemenyi hypothesis tests. Additionally, the superiority and robustness of our proposed algorithm is tested for 15 data clustering problems by using both Duncan's multiple range test and the Nemenyi hypothesis test.*

Keywords grey wolf optimizer, JAYA algorithm, particle swarm optimization, sine-cosine algorithm, partitional clustering

Citation Computer Science 24(3) 2023: 361–405

1. Introduction

The field of machine learning that is used to organize unlabeled data items into similar groups and dissimilar objects into different groups is known as data clustering. In contrast to supervised machine-learning algorithms, clustering algorithms have no idea about data labels to extract meaningful information from a huge volume of data. Additionally, clustering algorithms can be classified into two categories; namely, partitional and hierarchical. Due to the increase in complexity with the increase in data volume, hierarchical clustering algorithms have not become more popular as compared to partitional clustering algorithms. Partitional clustering algorithms consider similarity measures to group similar data items into the same groups. Therefore, the inter-cluster and intra-cluster distance measures are used in order to maximize the similarity among the data items in a partitional clustering algorithm. Moreover, separate groups are formed based upon some criterion known as the fitness function. Once an appropriate fitness function is chosen, the clustering algorithm can be converted into an optimization problem that minimizes the intra-cluster distances (compactness) and maximizes the inter-cluster distances (separability). Additionally, a partitional clustering algorithm can handle a large volume of data, which leads to more applications toward the field of research for grouping patterns for example, medical data analysis (for classifying positive and negative symptoms uniquely [46, 47]), social network analysis (for identifying fake and real information or users), robotics (for classifying items or humans based on their body shape or activities), and market basket analysis (for classifying consumers according to their purchasing behaviors, etc.). In all of these applications, the nature of data items that are available as patterns are different from each other. Additionally, there has been no optimum optimization algorithm invented as of yet (as per the "No Free Lunch Theorem" [NFL]). Therefore, the user must select an appropriate algorithm based on the clustering problem in hand that resolves the above problem.

Most traditional clustering algorithms are computationally simple but generally get stuck in local optimal solutions due to their hill-climbing approaches. On the other hand, these clustering problems are multi-modal in nature; therefore, they have a higher chance make them get trapped in local optimal solutions. Additionally, nature-inspired algorithms have the capability of escaping from local optimal solutions with the help of their stochastic operators. This also ensures global search with the help of a set of random search agents with a few stochastic operators. These play vital roles in solving most multi-modal problems that are similar to the clustering task to convergence fast. It also helps in escaping from the local optimal solutions. Therefore, it enhances the probability of reaching near-optimal solutions. On the other hand it also reducea the time of search to obtain near optimal solutions. Most specifically, the stochastic operators and the algorithmic behavior of nature-inspired computing help to escape the algorithms from trapping at multiple local optimal solutions by maintaining a trade-off between the exploration and the exploitation.

The first nature-inspired algorithm was proposed in 1975 by Holland and his team; this is popularly known as the genetic algorithm (GA) [21]. The recent literature reports that most of the meta-heuristic-based optimization algorithms are able to solve real-life optimization problems and those algorithms are such as: grey wolf optimization (GWO) [34], teaching learning-based optimization (TLBO) [44], bacteria foraging optimization (BFO) [17], the whale optimization algorithm (WOA) [33], the JAYA algorithm [43], ant colony optimization (ACO) [10], the sine cosine algorithm (SCA) [32], simulated annealing (SA) [29], the spotted hyena optimization algorithm (SHOA) [9], ant lion optimization (ALO) [31], particle swarm optimizations (PSO) [28], chemical reaction optimization (CRO) [30], differential evolution (DE) [50], etc. In recent times, most researchers have received immense interest in applying such nature-inspired algorithms on clustering problems [46–49]. Similarly, Jafer and Sivakumar [25] published a review article on a nature-inspired-based meta-heuristic optimization algorithm on ant-based clustering. Hruschka et al. [23] published a review article on an evolutionary algorithm for data clustering. Hatallou et al. [19, 20] proposed a data-clustering algorithm that used the gravitational search algorithm (GSA). Hatallou et al. [18] applied the Big Bang-Big Crunch algorithm (BB-BC) to address clustering problems [20]. Similarly, Nanda and Panda [38] published a research review on nature-inspired algorithms and their applications toward data-clustering problems.

Most partitional clustering algorithms are heuristic in nature. Additionally, hybridizing two or more nature-inspired algorithms increases the performance to a certain level by maintaining a better trade-off between exploration and exploitation [49]. Aljarah et al. [2] presented a hybrid approach of GWO and a trajectory search (TS) for clustering. This approach balances the exploratory and exploitative behavior of the GWO algorithm. Compared to the previously proposed algorithms, this hybrid algorithm had a more substantial exploration capability for avoiding the local optimal stagnation problem of the GWO algorithm. However, this algorithm had a higher computational time as compared to GWO (with the side effect of TS). Niknam and Amiri proposed a hybrid algorithm by combining PSO, ACO, and k-means algorithm [39] to overcome the issues of a traditional k-means algorithm. Shelokar et al. [45] proposed an ant colony-based algorithm to address clustering problems. Niknam et al. [40] proposed a hybrid algorithm using k-means, simulated annealing (SA), and ACO algorithms for selecting near-optimal cluster centers for partitional clustering algorithms. Moreover, the GWO algorithm showed better results when compared to the other meta-heuristic-based algorithms that have been presented in the literature for solving multiple fields of research problems; e.g., image segmentation [36], conceptual-based text mining [51], the load dispatch problem [26], and feature subset selections [12]. Additionally, the recently proposed JAYA optimization algorithm has gone through several modifications due to its more accurate and reliable performance and simple mathematical logic. This showed its superior performance over other optimization algorithms for solving engineering optimization problems such as the team formation problem [11], the traveling salesman problem [16], and iden-

tifying the reliable parameters of photo voltaic models [56]. Motivated by this, we propose a hybrid GWO and JAYA algorithm that uses the benefits of both the algorithms that are complementary to each other. Therefore, it maintains a trade-off between exploration and exploitation for achieving near-optimal solutions for a set of clustering problems.

The key contributions of this paper are summarized as follows:

- formulated initial solutions of GWO using randomly chosen cluster centers in order to incorporate randomness into initial solution;
- incorporated three worst solutions into GWO algorithm along with existing three leader wolves into its search strategy;
- hybridized JAYA algorithm with GWO for improving performance of proposed algorithm and to maintain better trade-off between exploration and exploitation, where
  1) mean best and mean worst are decided based on gray wolf hierarchy of GWO algorithm using three best wolves and three worst wolves, respectively;
  2) obtained mean best wolf and mean worst wolf are mutated stochastically using JAYA algorithm;
- applied proposed algorithm for 23 mathematical benchmark functions and 15 benchmark data-clustering problems;
- superiority of proposed algorithm was established by comparing it with existing GWO, JAYA, PSO, SCA, and traditional k-means for data-clustering problems by considering four different performance metrics;
- applied statistical tests such as Friedman and Nemenyi hypothesis test and Duncan's multiple range test to draw decisive conclusions from mean simulation results.

The remainder of the paper is structured as follows: Section 2 presents a brief overview of preliminary works on clustering task and optimization algorithms; Section 3 presents the proposed methodology; Section 4 presents a performance evaluation on meta-heuristic algorithms using benchmark functions; Section 5 presents a performance evaluation on clustering benchmark problems; and finally, Section 6 concludes the paper.

2. Preliminaries

2.1. Partitional clustering

Among the many clustering algorithms, partitional clustering has achieved maximum attention by researchers because of its simplicity, lower computational complexity, and easy implementation procedures. Partitional clustering algorithms can handle a large volume of data; hence, it has become more popular nowadays in the field of image processing [54], data mining, pattern recognition [6], social network analysis, etc. However, the main limitation of the algorithm is that it is highly dependent on

initial cluster centers and has more chances of getting trapped in local optimal solutions. The problem of partitional clustering has been tackled in numerous research works (as presented in the literature [2, 3, 19, 20, 23, 25, 38–40, 42, 45]). Additionally, several research works have been carried out to overcome the above issues in partitional clustering algorithms. In recent years, partitional clustering algorithms that employ nature-inspired algorithms have gained tremendous interest among researchers [2, 19, 20, 23, 25, 38–40, 42, 45–49]. Basically, these algorithm divides a set of data points into separate groups based on some fitness values. To obtain the fitness values, several fitness measure are used. Additionally, in some cases the fitness measures aims at either minimizing the fitness values or maximizing it that, directly affects the formation of quality clusters. So, the problem of partitional clustering can be considered to be an optimization problem that is used to minimize intra-cluster distances and maximize inter-cluster distances. Among all optimization algorithms, population-based meta-heuristic algorithms play an important role in finding optimal cluster centers by maintaining a good trade-off between intra-cluster and inter-cluster distances that produces better-quality clusters. Although a single meta-heuristic can achieve better results in its own principles, it still cannot achieve the true aspect of exploration and exploitation in a given search space. To improve the performance and to maintain a proper balance between exploration and exploitation two or more algorithms are hybridized.

There are several partitional clustering algorithms, including k-means, k-medoid, and fuzzy c-means. Among these algorithms, the k-means algorithm is widely used and has gone through several modifications by researchers due to its simple mathematical model and its capability of handling a large volume of data. However, the algorithm has several drawbacks such as: (i) high dependency on initial solutions, (ii) inability to effectively produce solutions of multi-modal data, its need for prior information about the number of clusters that are present in a data set, (iii) ease of being trapped in local optimal solutions, (iv) inability to guarantee the exploration of an entire search space, and (v) constant production of spherical clusters, etc. [24]. Additionally, the k-medoid algorithm is more expensive than k-means, as it computes all pairwise distances. The time complexities of k-medoid is $\mathrm{O}(n^2 \cdot K \cdot i)$ and k-means is $\mathrm{O}(n \cdot K \cdot i)$, where n denotes the number of observations, K – the number of cluster centers, and i – the number of iterations to complete. Additionally, meta-heuristic-based partitional clustering algorithms handle such local convergence issues by using their stochastic operators, that ensures to adapt the global searches [7, 13, 48] for achieving near-optimal solutions. The traditional partitioning algorithms rely on some strict rules or mathematical logic for achieving optimal solutions, whereas meta-heuristic algorithms approach towards the optimal solutions by maintaining cooperation among population members with a few randomness [58]. Additionally, the performance of the hybrid algorithms has been proven to be better and more effective than single standalone algorithms as given in the literature [1, 4, 55]. Therefore, in this research, we have hybridized two meta-heuristic algorithms, i.e. GWO and JAYA, to achieve superior performance.

2.2. JAYA algorithm

Most meta-heuristic algorithms have some common parameters, such as (i) number of iterations, (ii) number of generations, and (iii) population sizes, etc. In addition, they have some special parameters, such as mutation factor (f), crossover probability (Cr) for DE [50, 55], inertia weight, an acceleration coefficient for PSO [53], a few random parameters for the SCA (in order to move the position of a search agent toward a near-optimal solution [32]), etc. The quality of any meta-heuristic or heuristic algorithm is highly dependent on its initial parameters and its stochastic behaviors. Moreover, the fine-tuning of these parameters acts as a major challenge for researchers in order to apply it in a real domain with unknown search-space information. However, the JAYA algorithm is one of the simplest algorithms that does not need any such special parameter; therefore, it is considered to be a parameter-free algorithm. The main motivation behind the JAYA algorithm is to move the omega wolves close toward the best solution while maintaining a sufficient gap from the worst solution. Initially, the algorithm was proposed by Rao [43] in 2016 and gained tremendous interest among several researchers due to its motivational characteristics. Additionally, the algorithm is more flexible, simple, and easy to implement. It also does not require any derivative information in its initial population in order to reach at near-optimal solution. Due to the simple concept and easy-to-implement steps of the algorithm, several other traditional and meta-heuristic algorithms have been hybridized with this algorithm in order to improve its performance with the hybrid approaches [15, 52]. However, the JAYA algorithm generally gets trapped in local optimal solutions for complex optimization problems due to the insubstantial consideration of population information in its search strategy [58]. Moreover, Zhang et al. [58] proposed an algorithm that was known as EJAYA that incorporates more exploration potential to the existing algorithm with the consideration of population information (population mean) to enhance the performance of the JAYA algorithm. Mathematically, the population updation process of the algorithm can be illustrated as shown in Eq. (1):

$$X_i^{(t+1)} = X_i^t + r_1 \cdot (X_{\text{best}}^t - |X_i^t|) - r_2 \cdot (X_{\text{worst}}^t - |X_i^t|) \tag{1}$$

where, the $X_i^{\text{t+1}}$ vector is the new probable search agent for search agent X_i^t, the X_{best}^t and X_{worst}^t vectors are the current best and current worst solutions, respectively, in the search space, and r_1 and r_2 are chosen randomly from the uniform distribution within interval [0-1]. Adding term $r_1 \cdot (X_{\text{best}}^t - |X_i^t|)$ will attract the solution to move toward the best candidate solution, and subtracting $r_2 \cdot (X_{\text{worst}}^t - |X_i^t|)$ from the whole term will establish a tendency to stay away from the worst candidate solution. Here, the new solution ($X_i^{\text{t+1}}$) will survive in the upcoming $(\text{t}+1)^{\text{th}}$ generation based on a greedy-selection strategy. In this case, subtracting the third term will bring most of the worst solutions toward the best solutions but not exactly replicate the candidate solutions that are similar to the best solution. This helps maintain high diversity among the population members. Additionally, this term allows us to improve the exploration and exploitation capabilities of the algorithm by allowing each candidate

solution to search in the neighborhood region of the best solution. The second term in Eq. (1) will allow the candidate solutions to move towards the best solution. However, the algorithm has certain limitations, i.e. stagnation to local optimal solutions due to the loss of diversity. It so happens due to the unique approach of the JAYA algorithm i.e. consideration of only two candidate solutions. To tackle the above issues of the JAYA algorithm (i.e., the inadequate consideration of population information during its course of iterations), Zhang et al. [58] proposed an enhanced JAYA algorithm that improves the performance by incorporating population information using the mean population that is represented by M (as in Eq. (2)). Additionally, they introduced two attract points (upper and lower) in order to use population information effectively, where the upper and lower attract points are calculated by using the best and worst candidate solutions, respectively. Mathematically, the upper and lower attract points are calculated from the population (as shown in Eq. (3) and Eq. (4)):

$$M = 1/N \sum_{\mathrm{i}=1}^{N} X_i \tag{2}$$

Here, X_i represents a candidate solution in the population, and N is the number of candidate solutions:

$$P_u = \lambda_3 \cdot X_{\mathrm{best}} + (1 - \lambda_3) \cdot M \tag{3}$$

where P_u is the upper local attract point, λ_3 is a random value within interval [0–1], X_{best} represents the best fit solution in the population, and M represents the mean of the population in the current iteration:

$$P_l = \lambda_4 \cdot X_{worst} + (1 - \lambda_4) \cdot M \tag{4}$$

where P_l is the lower local attract point, λ_4 is a random value within interval [0–1], X_{worst} represents the worst fit solution in the population, and M represents the mean of the population in the current iteration. Mathematically, term M is calculated as shown in Eq. (2).

Considering the above upper local attract point, lower local attract point, and mean population information, the exploitation of each candidate solution can be expressed as shown in Eq. (5):

$$V_i = X_i + \lambda_5(P_u - X_i) - \lambda_6(P_l - X_i) \tag{5}$$

where λ_5 and λ_6 are two random numbers, V_i is the newly obtained solution for updating the current candidate using greedy selection. The above approach uses population information in order to improve the exploitation capability. Furthermore, the strategy improves the solution by generating a next-generation population by using the population information in the current generation; this helps to overcome the local convergence issue. However, a higher number of control variables are used

in this algorithm, which contradicts the simplicity and parameter-free nature of the algorithm. Additionally, the performance of the JAYA algorithm is directly related to the problem of dimensionality; i.e., with increase in the number of problem variables, the performance decreases. Several machine-learning approaches handle such problems in order to yield more-efficient results by using dimensionality reduction, scaling, and transformation methods. However, due to such an inherent property of the JAYA algorithm, the GWO algorithm outperformed the performance of the JAYA algorithm. Additionally, the superior performance of GWO over JAYA is due to its lower suceptibility for the problem of dimensionality. Therefore, we have incorporated the classical JAYA algorithm and GWO algorithm in our proposed hybrid algorithm to enhance the strength of the GWO algorithm that improves the performance with the benefit of both algorithms that are complementary to each other.

2.3. Standard grey wolf optimization algorithm

2.3.1. Inspiration of GWO

GWO is a nature-inspired meta-heuristic algorithm that mimics the hunting behaviors of gray wolves. According to the food chain, these animals are ranked at the top of the food-chain hierarchy. Australian scholar Mirjalili [34] proposed the grey wolf optimization algorithm in 2014 by considering the behaviors of gray wolves during their encircling, hunting, and attacking processes. According to Mirjalili, grey wolves live together and generally follow a social hierarchy among themselves during the above activities. This algorithm mimics the process of mutual cooperation, the leadership hierarchy, and the teamwork of gray wolves during their hunting processes. The searching and hunting processes of the gray wolf are always guided by the leader wolves in the hierarchy. During the searching process, the hypothesized prey is surrounded by the leader wolves, whereas during the hunting process, the prey is encircled by the wolves.

2.3.2. Social hierarchy

Basically, the algorithm classifies the wolves in the pack into four categories during the searching and hunting phases: alpha (α), beta (β), delta (δ), and omega (ω). Here, the alpha wolf is considered to be the leader among the wolves in the pack and is the most dominant wolf in the hierarchy. The beta wolf stands the second-best fit solution after α and, similarly, the delta wolf stands third. Interestingly, all of the remaining wolves in a wolf pack are considered to be ω; these are guided by the three leader wolves during the entire process of searching and hunting. Moreover, the overall process of hunting is summarized into three basic steps: (1) searching for prey; (2) encircling the prey; and (3) attacking the prey.

2.3.3. Hunting mechanism

At the beginning when prey is found, the first iteration begins with the process of encircling by the gray wolves with the leadership of the leader wolves. Mathemati-

cally, the encircling process can be formulated by using the distance measures that are shown in Eq. (6). Subsequently, random parameter C (as in Eq. [7]) helps to stochastically emphasize or deemphasize the weight of the prey in order to overcome the local convergence issues. Initially, it is assumed that the leader wolves have better knowledge about the location of their prey ($X_p(t)$). Therefore, the simulation is carried out by considering the leader wolves' location as the most-probable location for the prey to be searched. The searching and encircling behavior of each omega wolf can be mathematically represented by using Eq. (8), Eq. (9), and Eq. (10) for the α, β, and δ wolves, respectively:

$$D = C.X_p(t) - X(t) \tag{6}$$

$$C = 2.r_1 \tag{7}$$

$$X_1 = X_\alpha(t) - A1.|C1.X_\alpha(t) - X(t)| \tag{8}$$

$$X_2 = X_\beta(t) - A2.|C2.X_\beta(t) - X(t)| \tag{9}$$

$$X_3 = X_\delta(t) - A3.|C3.X_\delta(t) - X(t)| \tag{10}$$

Here, X_1, X_2, and X_3 are calculated by considering the leader wolves' positions individually with respect to the α, β, and δ wolves, respectively. Supporting the above practice among the leader wolves, α will guide each single omega wolf to the next probable location for the grey wolf from its original position to X_1; similarly, β and δ will guide them to X_2 and X_3, respectively. Parameters $A1$, $A2$, and $A3$ are obtained by using Eqs. (11) and (12). Additionally, these three parameters vary in the interval of $[-2a$ to $+2a]$. The values of these three parameters change with a change in the random value of parameter $r2$ for each leader wolf separately (as shown in Eq. (11)):

$$A = 2 \cdot a \cdot r2 - a \tag{11}$$

$$a = 2 \cdot (1 - \frac{t}{T}) \tag{12}$$

In the initial phase, the value of a is maintained at a high level; this declines with a change in the iteration toward convergence. The high value helps to enhance an explorative search, which helps to improve the global search capability. Similarly, a low value enhances the exploitative search of the algorithm to search in the local region more deeply. Therefore, the algorithm allows for a more local search for $A > -1$ and $A < +1$; on the other hand, the algorithm makes a more global search when $A < -1$ and also when $A > +1$. Finally, the average of all of the predicted positions is obtained by using Eq. (13), which calculates the most-probable position for a single ω wolf:

$$X^{t+1} = \frac{X_1 + X_2 + X_3}{3} \tag{13}$$

The position updation is purely based on the the fitness function: if the fitness of X^{t+1} is better than X^t, then the newly obtained position will be considered for the next

iteration; otherwise, the current generation solution will remain unaltered for the next generation. Additionally, Gao and Zhao [14] proposed an improved version of these position updations by considering variable weights for each leader wolf to reduce the chance of getting trapped in local optimal solutions. Here, they considered an increasing order of weights for the α, β, and δ wolves (the highest weight is assigned to α as compared to β and δ). Similarly, the next-highest weight is assigned to β, and according to the leadership hierarchy, δ is assigned with the lowest weight among the leader wolves. However, the cumulative sum of the variable weights is equal to 1 in each iteration (as shown in Eq. (14)):

$$w_1 + w_2 + w_3 = 1 \tag{14}$$

where, w_1, w_2, and w_3 are assigned weights for the α, β, and δ leader wolves, respectively. Mathematically, the relationship among w_1, w_2, and w_3 is formulated as shown in Eq. (15). Subsequently, with a change in the iteration, the w_2 and w_3 weights are increased with a gradual decrease in the weight of α in order to encircle the prey. However, the weight updation always follows Eq. (15) to maintain the proper teamwork among the leader wolves while guiding a number of ω wolves:

$$w_1 \geq w_2 \geq w_3 \tag{15}$$

The two control parameters in Eq. (11) and Eq. (12) help control the movement direction while stochastically maintaining a sufficient gap from the best solution that has been found so far. Simultaneously, it also helps to approach the prey stochastically due to such stochastic behavior. Additionally, parameter a declines linearly in order to improve the exploitation capability while reducing the exploration potential of the algorithm. However, the maximum number of iterations that are required to converge the algorithm is not known in advance in most real-life problems. Therefore, Gao and Zhao [14] proposed an exponential declined equation in order to improve the convergence speed by incorporating a maximum number of permissible errors and an exponentially declined equation (as shown in Eq. (16)). Here, the controlling parameter is expected to decline very fast during the initial iterations; therefore, it converges to near-optimal values quickly as compared to the previous approach. Additionally, Eq. (16) also helps select a few wolves for global-searching purposes in order to avoid local optimal solutions [4]:

$$a = a_{\max} \cdot (e^{-\frac{t}{T}}) \tag{16}$$

where, a_{max} is the maximum value of a, T is the maximum number of iterations, and t denotes the current iteration number. Here, parameter T helps the algorithm to stop at some finite time (mostly based on the current computing facilities that are available).

Additionally, Yu et al. [57] proposed an opposition-based learning strategy to overcome the local convergence issues of the grey wolf optimization algorithm. According to this algorithm, an opposite vector that corresponds to a gray wolf search

agent is obtained by using a mathematical equation (as shown in Eq. (17)). The authors also proposed a modified transition parameter in order to optimize complex problems by modifying Eq. (12) into a nonlinear equation (as shown in Eq. (16)):

$$\widehat{X}_{pq} = U_{pq} + L_{pq} - X_{pq} \tag{17}$$

here, a candidate solution in the initial population is represented by X_p, where $X_p = X_{\mathrm{p1}}, X_{\mathrm{p2}}, X_{\mathrm{p3}}, ..., X_{\mathrm{pd}}$ has d dimensions. Considering the above assumptions, U_{pq} and L_{pq} are termed as the upper and lower bounds, respectivly, for a data point p with a dimension of q. Similarly, $\widehat{X}_{pq}$ is calculated as the opposite solution with respect to X_{pq}. Here, a random guess and the opposite solution can improve the chances of moving toward the near-optimal solution. The hybrid algorithm of the initial population selection selects a set of better candidate solutions from the initial random population and opposite solutions in order to speed up the convergence process (which saves computational overhead):

$$a = 2 - 2 \cdot \left(\frac{\mathrm{t}}{\mathrm{T}}\right)^{u} \tag{18}$$

where t is the current iteration, u is a nonlinear operator, and T is the maximum iteration. The above approach improves the search process by converging faster. However, the improved GWO algorithm is guided by the α, β, and δ wolves in order to achieve better performance. Therefore, the algorithm is likely to get trapped in local optimal solutions due to the only consideration of best three leader wolves. Additionally, the author also introduced a probability factor Jr to overcome the local convergence issue, which generates opposite solutions at each iteration based on the probability factor. Therefore, Jr is also considered to be a mutation factor that helps the algorithm to jump out of the local convergence issue. However, the algorithm maintains a large population (i.e., two times the initial population) due to the incorporation of opposite solutions.

Additionally, half of the iterations are devoted to exploration and the other half to exploitation in the classical GWO algorithm. Most generally, a higher exploration level can overcome the local convergence issue. Among the various possibilities of enhancing the chances of exploration, one such possibility is to use exponential functions instead of linear functions in order to decrease the value of a over the course of the iterations. Mittal et al. [35] proposed an exponential decay function that decreases the above parameter exponentially over the course of the iterations. Therefore, parameter A helps maintain the trade-offs between exploration and exploitation, with the devotion of most of the iterations toward exploration and a comparatively lower number of iterations toward exploitation (at a ratio of 70:30). The exponential decay function can be mathematically represented as shown in Eq. (19):

$$a = 2\left(1 - \frac{t^2}{T^2}\right) \tag{19}$$

Moreover, an improved grey wolf optimization algorithm (IGWO) was proposed to maintain a proper trade-off between exploration and exploitation and to address the global optimization problem [37]. The algorithm mostly benefits from dimension learning-based strategy (DLH) hunting. The approach uses the neighboring information of each wolf, which improves the diversity. The movement strategy of IGWO is to select the candidate solution from either the GWO search strategy or DLH search strategy based on the quality of the solutions that were obtained by each algorithm. The cooperation between these two algorithms helps IGWO improve the global and local search capabilities of the algorithm. The convergence analysis of I-GWO was found to have superior performance for unimodal problems with higher dimensions and was found to have faster convergence than the comparative algorithm. When compared to other algorithms, the IGWO also shows its competitive performance while solving the mathematical benchmark functions in most of the multi-modal problems. The reason behind the enhanced performance is the hybrid behavior of the GWO and DLH search strategies that are complementary to each other. Therefore, it maintains good trade-offs between exploration and exploitation in order to avoid local optima. However, the algorithm has been used to address only a single objective optimization problem and needs to be modified to address large-scale global optimization problems.

Similarly, Hau et al. [22] proposed a novel initial-wolf-position-initialization strategy by using improved chaotic tent mapping. Second, a Gaussian-distribution variation-based convergence factor is used to improve the search capability. Additionally, a dynamic-weight strategy is used to improve the speed of the convergence of the algorithm. The performance algorithm was tested on mobile-robot-path planning to verify the practical aspect of its superiority among other variants of GWO. The dynamic proportional-weighting strategy is mathematically illustrated in Eq. (20):

$$X^{t+1} = \frac{V_1 \cdot W_\alpha + V_2 \cdot W_\beta + V_3 \cdot W_\delta}{3} \tag{20}$$

where V_1, V_2, and V_3 are obtained from Eq. (21), Eq. (22), and Eq. (23), respectively, as shown below:

$$V_1 = \frac{|X_\alpha| + |X_\beta| + |X_\delta|}{|X_\alpha|} \tag{21}$$

$$V_2 = \frac{|X_\alpha| + |X_\beta| + |X_\delta|}{|X_\beta|} \tag{22}$$

$$V_3 = \frac{|X_\alpha| + |X_\beta| + |X_\delta|}{|X_\delta|} \tag{23}$$

Similar to the above mathematical equations, the W_α, W_β, and W_δ are also calculated using Eq. (24), Eq. (25), and Eq. (26), respectively where f_α, f_β, and f_δ denote the current adaptions of α, β, and δ, in their respective equation as given below:

$$W_\alpha = \frac{f_\alpha + f_\beta + f_\delta}{f_\alpha} \tag{24}$$

$$W_\beta = \frac{(f_\alpha + f_\beta + f_\delta)}{f_\beta} \tag{25}$$

$$W_\delta = \frac{(f_\alpha + f_\beta + f_\delta)}{f_\delta} \tag{26}$$

The test result on the above modification of the GWO algorithm shows that the algorithm has enough competence when compared with the test result of 15 benchmark functions with varieties of complexity and a wide variety of dimensions. When compared to eight other algorithms, the results confirmed its superiority in terms of solution accuracy and convergence speed.

Even though the test results confirm its superiority in most of the single-modal test functions, it still cannot reach its optimal value for maximum multi-modal problems. The results on mobile-robot-path planning show significant performance improvement in cost consumption and also in its convergence speed when compared with other algorithms. However, to incorporate more population information into the steps of the JAYA algorithm and to maintain the trade-off between the exploration and exploitation behavior of GWO the algorithm, we have hybridized both of the meta-heuristic algorithms.

3. Methodology for meta-heuristic based on data clustering

Data clustering is a multi-modal search problem that is also considered to be an NP-hard problem. The problems of data clustering have been solved by using several conventional techniques; however, these techniques have more chances to converge to local optimal solutions due to the multi-modal nature of the problems.

Therefore, to avoid local minima problems, most researchers have solved similar types of multi-modal problems such as time series forecasting [41], the FOPID-based damping controller [27], data-clustering problems [46, 47, 49], etc. by using nature-inspired meta-heuristic algorithms. Primarily, the aim of clustering is to minimize the intra-cluster distance and maximize the inter-cluster distance in order to form compact clusters. So, we considered Euclidian distance measures between data points (as shown in Eq. (27)) to access the similarities between them. The two-point Euclidian distance measure is worth mentioning here (as shown in Eq. (27)):

$$Distance(p_m - p_n) = \sqrt{\sum_{i=1}^{d} \left(p_m^i - p_n^i\right)^2} \tag{27}$$

where p_m^i and p_n^i are the i_{th}-dimension information of the p_m and p_n data points, respectively.

3.1. Problem formulations

Given a dataset (D) with n-data points ($p_1, p_2, p_3, \ldots, p_n$), the dimensions (features) of all data points are represented by:

$$\begin{vmatrix} p_1^1 & p_1^2 & \cdots & p_1^d \\ \vdots & & \ddots & \vdots \\ p_n^1 & \cdots & \cdots & p_n^d \end{vmatrix}$$

- The aim is to find the K number of non-overlapping compact groups from the given data set that satisfy the following mathematical illustrations (as shown in Eq. (28) and Eq. (29)):

$$\sum_{i=1}^{k}\sum_{j=1}^{k}(C_i \bigcap C_j) = \phi \tag{28}$$

where, $i \neq j$; and

$$\overset{k}{\underset{i=1}{\bigcup}} C_i = n \tag{29}$$

- In order to achieve the above objective, we considered sum squared errors as shown in Eq. (30):

$$\sum_{j=1}^{k}\sum_{i=1}^{n} M_{ij} \cdot \sqrt{\sum_{q=1}^{d}\left(p_i^q - {C_j}^q\right)^2} \tag{30}$$

where p_i^q represents the q^{th} dimension of the i^{th} data point and, similarly, M_{ij} represents the membership value for data point i with respect to cluster center j. C_j^q represents the q_{th} dimensional information of cluster center C.

3.2. Motivation

Given a data set that has D with n instances and d dimensional information in each, the aim is to form K distinct groups by comparing the data points by using a distance measure such that the similar points belong to the same cluster and the dissimilar points belong to different clusters. Finding the best initial cluster centers that can represent the centers of the K cluster is an NP-hard problem. Additionally, obtaining K near-optimal cluster centers from n d-dimensional data points is a combinatorial optimization problem with an exponentially increasing search space. Here, addressing the above problem using conventional algorithms may lead to local optimal solutions. To overcome this problem, most researchers used meta-heuristic algorithms. Even though meta-heuristic algorithms purely depend on the exploration and exploitation behavior of the algorithm, they still need certain parameters to be well-tuned in advance to achieve near-optimal solutions. To improve the performance of such algorithms, most researchers therefore prefer to use algorithms with only a few parameters. The JAYA algorithm is one such meta-heuristic algorithm that has only two control parameters to obtain a near-optimal solution. On the other hand, the GWO algorithm obtains near-optimal solutions by following the leader wolves and uses only a few control parameters to maintain a trade-off between exploration and exploitation. Additionally, GWO algorithm does not consider the worst candidate solutions to

achieve near optimal solutions whereas JAYA algorithm considers both best and worst candidate solutions. On the other hand, when both the complementary algorithms are hybridized, the GWO gets a better opportunity to avoid identical population information in the search process when all three leader wolves are approaching towards single local optimal solution. Additionally, the hybrid approach also helps the poor-performing omega wolves of the GWO algorithm to come close to prey in a faster way by avoiding all these local optimal solutions, which improves the exploration skills of GWO. Therefore, the GWO algorithm maintains a better trade-off between exploration and exploitation. With the internal strength of both the GWO and JAYA algorithms and their combined efforts, the GWO can maintain a trade-off between exploration and exploitation that helps it to achieve better performance.

When compared to the mutation operator of the existing GWO algorithm, the hybrid GWO approach with JAYA improves the mutation quality by considering both the best and worst solutions with the help of the leader-wolf strategy of the GWO algorithm. To achieve this, three leader wolves and the three lowest-quality omega wolves (and their respective combined efforts) are considered. Here, the proposed hybrid algorithm considers the leadership hierarchy of the GWO algorithm and the strategy of the JAYA algorithm. In the proposed algorithm, the three worst-performing wolves also take part in the search process to add some information that helps the other omega wolves to avoid getting trapped in local optimal solutions. Moreover, these worst-performing wolves generally do not participate in the encircling process but help the others in order to stay away from worst-quality solutions. The combined effort of the GWO and JAYA algorithms improves the performance by utilizing the exploration and exploitation skills of both algorithms. Additionally, this helps the GWO algorithm minimize the maximum interference of the leader wolves while updating the positions of any omega wolves. It also incorporates the consideration of maximum population information in the search space.

3.3. Proposed hybrid algorithm

The main steps of our proposed hybrid algorithm are presented in Figure 1; basically, it operates with three steps (initialization, iteration, and the final steps). In the first step, the parameters are initialized; e.g., the number of gray wolfs (n), the maximum number of iterations ($Tmax$), the number of variables to be optimized (d), the lower (l) and upper (u) bounds, and the initial population (X). In the iterative step, each omega wolf's position is updated with the help of three leader wolves, three worst-fit wolves, and JAYA operators. Then, a greedy-selection approach is followed to update the solution with the consideration of the fitness function. This iterative step is repeated for each omega wolf until the termination condition is satisfied. The final step selects the alpha wolf (α) as the best-fit solution for the optimization problem.

The proposed algorithm generates non-overlapping clusters using the hybrid grey wolf optimization and JAYA algorithm. Initially, the algorithm starts by a set of initial parameters such as the number of clusters (k), population size ($\mathrm{Max}_{\mathrm{pop}}$), and

maximum iterations ($\mathrm{Max}_{\mathrm{Iter}}$). Subsequently, a $\mathrm{Max}_{\mathrm{pop}}$ number of candidate solutions are selected randomly (one from each cluster) to form a candidate solution of a size of $k \times D$.

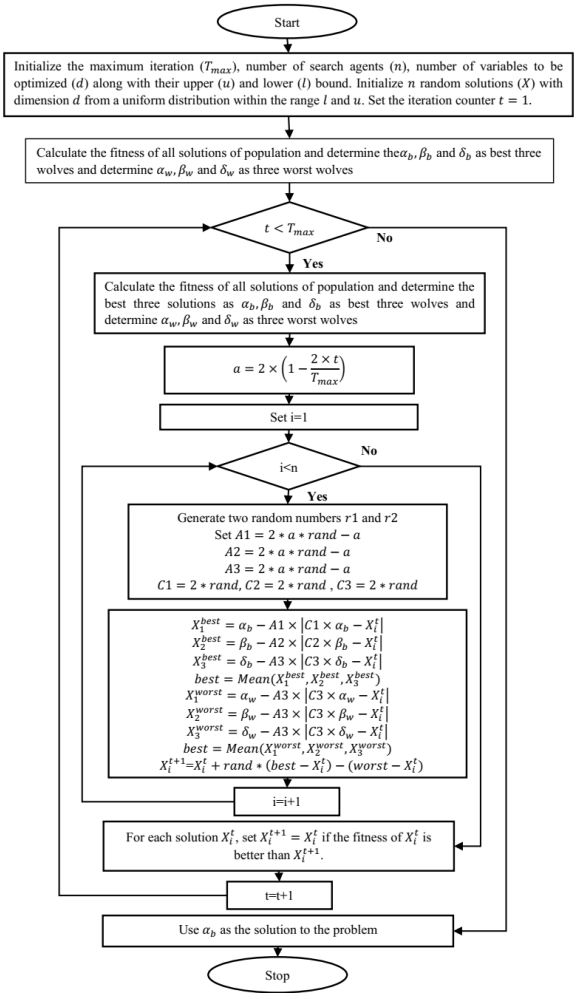

Figure 1. Flowchart of proposed hybrid algorithm

Thereafter, the fitness of each candidate solution is calculated in order to select the best candidate solution. Due to the random selection of data points in the initial population, the fittest candidate solutions in the whole population have greater chances to converge toward local optimal solutions. In order to search for better candidate solutions for data clustering, these candidate solutions are given as input to our hybrid clustering algorithm, which will produce a better population by using the strength of both the GWO and JAYA algorithms with their inherent exploration and exploitation capabilities.

Therefore, our algorithm is comprised of three steps: 1) population initialization; 2) fitness calculation; and 3) iterative steps for successive position updation, fitness calculation, and promising candidate selection for next generation. Algorithm 1 gives a brief idea about our proposed algorithm for data clustering.

Algorithm 1: Proposed hybrid algorithm for data clustering

Data: Input: Input Data Set (D), Number of Clusters (K)
Result: Output: Set of (K) Cluster Centers
Step 1: Initialization Step Initialize number of clusters (K), generation counter $gen = 0$, $t = 0$, $Max_{gen} = 100$, $i = 0$, $Max_{iter} = 1000$, $Max_{pop} = 20$. Randomly select data points from Data Set D and initialize initial wolf pack (R^{gen} with random cluster centers ($R^{gen} = R_1^{gen}$, R_2^{gen}, R_3^{gen}...R_n^{gen}) size $n * K$ where $R_j^{gen} = C_1^{gen}$,...,C_K^{gen}, where $C_r^{(gen,l)}$ is l^{th} feature of r^{th} wolf
Calculate fitness of each wolf of R^{gen}
Sort wolf pack (R_{gen}) with respect to fitness
Determine best-three solutions as α_b, β_b, and δ_b (best three wolves) and determine α_w, β_w, and δ_w as three worst wolves
while $gen < Max_{gen}$ **do**
  Step 2: Iteration Step
  while $t < Max_{iter}$ **do**
    Reinitialize i = 0
    Sort wolf pack with respect to fitness
    Calculate fitness of all solutions of population and determine best-three solutions as α_b, β_b, and δ_b (best three wolves) and determine α_w, β_w, and δ_w as three worst wolves
    Compute $a = 2 - \frac{(2*iter)}{Max_{iter}}$
    while $i < Max_{pop}$ **do**
      Generate two random numbers $r1$ and $r2$
      Set $A1 = 2 * a * rand - a$
      Set $A2 = 2 * a * rand - a$
      Set $A3 = 2 * a * rand - a$
      Set $C1 = 2 * rand$, $C2 = 2 * rand$, $C3 = 2 * rand$
      Compute $Y_1^{best} = \alpha_b - A1 \times |C1 \times \alpha_b - X_i^t|$
      Compute $Y_2^{best} = \beta_b - A2 \times |C2 \times \beta_b - X_i^t|$
      Compute $Y_3^{best} = \delta_b - A3 \times |C3 \times \delta_b - X_i^t|$
      Compute $best = Mean(Y_1^{best}, Y_2^{best}, Y_3^{best})$
      Set $A = 2 * a * rand - a$
      Compute $Y_1^{worst} = \alpha_w - A \times |C \times \alpha_w - X_i^t|$
      Compute $Y_2^{worst} = \beta_w - A \times |C \times \beta_w - X_i^t|$
      Compute $Y_3^{worst} = \delta_w - A \times |C \times \delta_w - X_i^t|$
      Compute $worst = Mean(Y_1^{worst}, Y_2^{worst}, Y_3^{worst})$
      Calculate $X_i^(t+1) = X_i^t + rand * (best - X_i^t) - (worst - X_i^t)$
      Update $i = i + 1$
      if $fitness(C_i^{gen+1}) < fitness(C_i^{gen})$ **then**
        Set $C_i^{gen+1} = C_i^{gen}$
    Update t = t+1
  Step 3: Final Step
  Use K seeds of α_b as near-optimal cluster centers
  Form K clusters
  $gen = gen + 1$

The initialization of control parameters (including the parameter setting and candidate-solution selection for the initial population) is done in the initial step of our proposed algorithm. In Step 2, the fitness for each candidate solution is calculated by considering the accuracy-performance metric. In Step 3, the hybrid approach of GWO and JAYA is used with the current population in order to improve the performance of the existing candidate solutions. In this step, the fitness is calculated and, accordingly, the promising candidate solutions are selected in order to update the candidate solution in the population in every iteration. Then, the hybrid approach tries to update the position of each candidate solution. The process of candidate selection, position updation, and fitness calculation are performed until maximum iteration is achieved. In order to simulate the behavior of GWO for optimization, the best-three-fittest solutions are termed alpha (α), beta (β), and delta (δ) wolves, and the remaining solutions are termed omega (ω) wolves. Considering the fitness of each wolf, these three wolves are expected to have better knowledge about the prey's location. Therefore, the omega (ω) wolves are guided by these three wolves (as represented mathematically in Eq. (13)). When facing more-complex or multi-modal problems, however, these leader wolves may lead toward local optimal solutions; this in turn gives the wrong guidance to the remaining wolves. However, these could have received better results from their own localities or from more-diversified locations by using a better search strategy. Therefore, we have hybridized the GWO algorithm with the JAYA algorithm to update the positions. Additionally, we introduce six new wolves; these includes the three leader wolves (Y_1^{best}, Y_2^{best}, and Y_3^{best}) and their mean effort as the best search agents for the JAYA algorithm. Similarly, the worst three ω wolves (Y_1^{worst}, Y_2^{worst}, and Y_3^{worst}) and their mean effort as the worst search agents for the JAYA algorithm. The best three wolves and their corresponding representation of each omega wolf (x_i) is calculated by Eq. (32), Eq. (34), and Eq. (36) for α_b, β_b, and δ_b, respectively. The mean result of these three leader wolves calculates the best solution for the JAYA algorithm. Similarly, the representation of the worst three omega wolves is obtained by using Eq. (39), Eq. (40), and Eq. (41). The mean result (as shown in Eq. (42)) for these calculated three worst omega wolf representations is termed as the worst solution for the JAYA algorithm. Finally, the most-probable solution (x_i^{t+1}) is obtained from Eq. (42). Subsequently, greedy selection is applied to select the best solution among the newly calculated solution and the solution that was obtained from the previous iteration:

$$A1 = 2 \cdot a \cdot r2 - a \tag{31}$$

$$Y_1^{best} = \alpha_b - A1 \cdot (C \cdot \alpha_b - x_i) \tag{32}$$

$$A2 = 2 \cdot a \cdot r2 - a \tag{33}$$

$$Y_2^{best} = \beta_b - A2 \cdot (C \cdot \beta_b - x_i) \tag{34}$$

$$A3 = 2 \cdot a \cdot r2 - a \tag{35}$$

$$Y_3^{best} = \delta_b - A3 \cdot (C \cdot \delta_b - x_i) \tag{36}$$

$$best = \frac{Y_1^{best} + Y_2^{best} + Y_3^{best}}{3} \tag{37}$$

$$A = 2 \cdot a \cdot r2 - a \tag{38}$$

$$Y_1^{worst} = \alpha_w - A \cdot (C \cdot \alpha_w - x_i) \tag{39}$$

$$Y_2^{worst} = \beta_w - A \cdot (C \cdot \beta_w - x_i) \tag{40}$$

$$Y_3^{worst} = \delta_w - A \cdot (C \cdot \delta_w - x_i) \tag{41}$$

$$worst = \frac{Y_1^{worst} + Y_2^{worst} + Y_3^{worst}}{3} \tag{42}$$

$$x_i^{t+1} = x_i + r_1(best - x_i) - r_2(worst - x_i) \tag{43}$$

where $r_1$and r_2 are two different random numbers within interval (0–1).

4. Performance evaluation of meta-heuristic algorithms using benchmark functions

To evaluate the performance of our proposed hybrid GWO and Jaya algorithm, we considered 23 mathematical benchmark functions. These function are $f1, f2, f3, ..., f23$ and can be classified as unimodal, flexible-dimensional multi-modal, fixed-dimensional multi-modal, etc. To evaluate the performance of our proposed hybrid algorithm, we compared it with the GWO [34], JAYA [43], PSO [28], and SCA-based meta-heuristic algorithms [32] using the above 23 mathematical functions. The parameter settings for the experimental work is presented in Table 1.

Table 1
Parameter settings

Number of search agents	50
Maximum number of Iterations	1000
Number of independent executions	30

The mean and standard deviation results are presented in Tables 2, 3, and 4 for the unimodal, multi-modal flexible-dimension, and multi-modal fixed-dimension benchmark functions, respectively.

Moreover, to effectively evaluate the performance of our proposed algorithm, we conducted a non-parametric-based Freidman and Nemenyi hypothesis test. The mean-rank results that were obtained from the test are presented in Figure 2, and the results are analyzed at p-value = 0.000 and critical distance = 1.3. One can see from the results that the proposed algorithm obtained the best mean rank (value = 63.79) while minimizing 23 mathematical benchmark functions.

Table 2
Mean and standard deviation of fitness values obtained in 30 independent simulations by meta-heuristic algorithms on unimodal functions

	Proposed Mean ± Std. Dev.	SCA Mean ± Std. Dev.	GWO Mean ± Std. Dev.	JAYA Mean ± Std. Dev.	PSO Mean ± Std. Dev.
f1	4.9E-108 ± 8.2E-108	2.8E-85 ± 1.5E-84	**3.6E-176 ± 0.0E+00**	2.9E-02 ± 5.9E-02	1.3E+00 ± 4.2E-01
f2	9.6E-54 ± 2.0E-53	3.4E-64 ± 1.8E-63	**2.4E-110 ± 1.3E-109**	4.4E-07 ± 2.1E-07	2.1E-03 ± 3.1E-03
f3	**2.7E-107 ± 1.0E-106**	2.7E+03 ± 2.3E+03	5.3E+00 ± 9.0E+00	6.2E-01 ± 3.9E-01	1.5E-11 ± 6.3E-11
f4	**7.2E-52 ± 1.1E-51**	1.5E+01 ± 1.8E+01	1.2E-41 ± 4.6E-41	**8.7E-06 ± 3.5E-06**	9.7E-03 ± 8.7E-03
f5	8.8E+00 ± 1.4E-01	8.9E+00 ± 1.7E-01	7.7E+00 ± 5.4E-01	9.1E-05 ± 4.4E-04	3.0E-01 ± 9.9E-01
f6	1.5E-06 ± 4.9E-07	8.0E-02 ± 7.6E-02	5.4E-02 ± 5.6E-02	4.7E-03 ± 2.1E-03	**6.3E-14 ± 3.4E-13**
f7	**2.0E-04 ± 1.1E-04**	3.4E-04 ± 4.3E-04	3.3E-04 ± 3.3E-04	3.9E-03 ± 2.4E-03	8.0E-04 ± 4.9E-04

Table 3
Mean and standard deviation of fitness values obtained in 30 independent simulations by meta-heuristic algorithms on flexible-dimension multi-modal functions

	Proposed Mean ± Std. Dev.	SCA Mean ± Std. Dev.	GWO Mean ± Std. Dev.	JAYA Mean ± Std. Dev.	PSO Mean ± Std. Dev.
f8	-3.9E+01 ± 1.1E+00	-3.9E+01 ± 5.7E-01	-3.9E+01 ± 1.8E-01	**-3.9E+01 ± 7.1E-15**	**-3.9E+01 ± 1.5E-13**
f9	6.5E-01 ± 3.5E+00	**0.0E+00 ± 0.0E+00**	4.0E+00 ± 7.5E+00	2.4E+01 ± 6.7E+00	1.2E+01 ± 4.3E+00
f10	1.1E-15 ± 8.9E-16	**8.9E-16 ± 9.9E-32**	3.7E-15 ± 1.4E-15	6.5E-08 ± 3.3E-08	3.9E-02 ± 2.1E-01
f11	**0.0E+00 ± 0.0E+00**	**0.0E+00 ± 0.0E+00**	5.4E-03 ± 2.9E-02	2.8E-01 ± 1.1E-01	9.3E-02 ± 5.7E-02
f12	**1.8E-03 ± 4.8E-03**	2.5E+00 ± 7.6E+00	1.4E-02 ± 1.8E-02	4.8E-03 ± 2.5E-03	1.0E-02 ± 5.6E-02
f13	1.8E-01 ± 1.1E-01	9.0E-01 ± 2.9E+00	8.1E-02 ± 5.3E-02	**4.9E-16 ± 8.6E-16**	1.1E-03 ± 3.3E-03

Table 4

Mean and standard deviation of fitness values obtained in 30 independent simulations by meta-heuristic algorithms on fixed-dimension multi-modal functions

	Proposed Mean ± Std. Dev.	SCA Mean ± Std. Dev.	GWO Mean ± Std. Dev.	JAYA Mean ± Std. Dev.	PSO Mean ± Std. Dev.
f14	**1.0E+00 ± 4.6E-08**	**1.0E+00 ± 4.6E-07**	2.3E+00 ± 1.7E+00	1.0E+00± 5.1E-05	3.2E+00 ± 2.2E+00
f15	3.1E-03 ± 6.8E-03	7.3E-04 ± 5.0E-04	1.9E-03 ± 5.3E-03	3.9E-04± 2.8E-04	**3.1E-04 ± 2.1E-05**
f16	**-1.0E+00 ± 4.9E-08**	**-1.0E+00 ± 6.2E-06**	**-1.0E+00 ± 1.1E-08**	-1.0E+00 ± 1.2E-05	**-1.0E+00 ± 6.7E-16**
f17	4.0E-01 ± 7.1E-05	4.0E-01 ± 2.0E-03	4.3E-01 ± 1.6E-01	4.0E-01 ± 3.2E-04	**4.0E-01 ± 1.1E-16**
f18	**3.0E+00 ± 1.2E-07**	3.0E+00 ± 6.8E-02	3.9E+00 ± 4.8E+00	3.0E+00 ± 2.9E-04	**3.0E+00 ± 4.4E-16**
f19	**-3.9E+00 ± 5.3E-08**	-3.9E+00 ± 5.2E-04	-3.9E+00 ± 2.9E-05	**-3.9E+00 ± 1.3E-15**	**-3.9E+00 ± 1.3E-15**
f20	-3.3E+00 ± 6.1E-02	-3.2E+00 ± 1.3E-01	-3.3E+00 ± 7.8E-02	-3.3E+00 ± 5.9E-02	**-3.3E+00 ± 4.8E-02**
f21	-1.0E+01 ± 9.2E-01	-6.2E+00 ± 2.2E+00	-9.2E+00 ± 1.9E+00	**-1.0E+01 ± 1.8E-15**	-5.8E+00 ± 3.6E+00
f22	-9.8E+00 ± 1.7E+00	-5.6E+00 ± 2.0E+00	-8.2E+00 ± 2.8E+00	**-1.0E+01 ± 9.5E-01**	-7.3E+00 ± 3.7E+00
f23	-1.0E+01 ± 1.5E+00	-6.3E+00 ± 2.3E+00	-7.8E+00 ± 2.9E+00	**-1.1E+01 ± 8.9E-15**	-7.8E+00 ± 3.7E+00

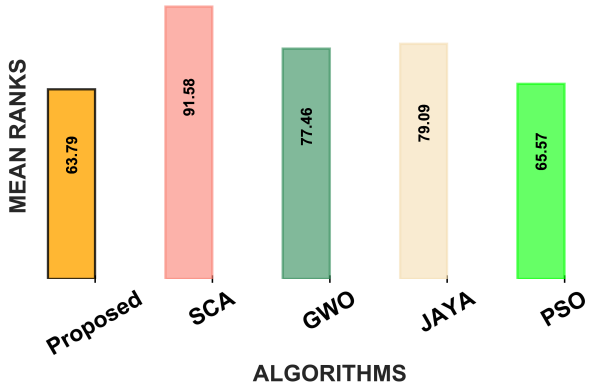

Figure 2. Mean Rank of meta-heuristic algorithms for 23 benchmark functions with p-value = 0.000 and critical distance = 1.3

4.1. Convergence analysis

To make a visual comparison among the meta-heuristic algorithms, we plotted the convergence curve for each benchmark function separately; these are shown in Figures 3, 4, and 5 for the unimodal, flexible-dimension multi-modal, and fixed-dimension multi-modal test functions, respectively.

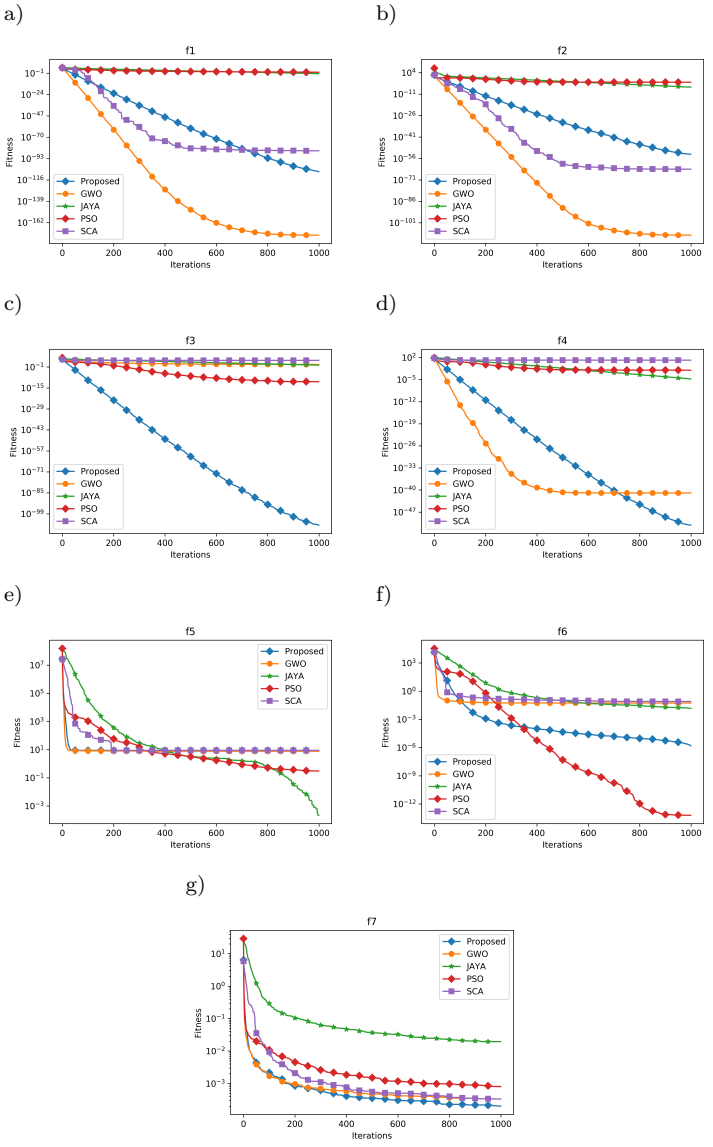

Figure 3. Unimodal mathematical benchmark functions: a) Sphere; b) Schwefel 2.22; c) Schwefel 1.2; d) Schwefel 2.21; e) Rosenbrock; f) Step; g) Quartic with noise;

Due to the change in the behaviors of the convergence curves in different simulations, we have considered the mean simulation results of 30 independent simulations on each meta-heuristic algorithm for each benchmark function separately.

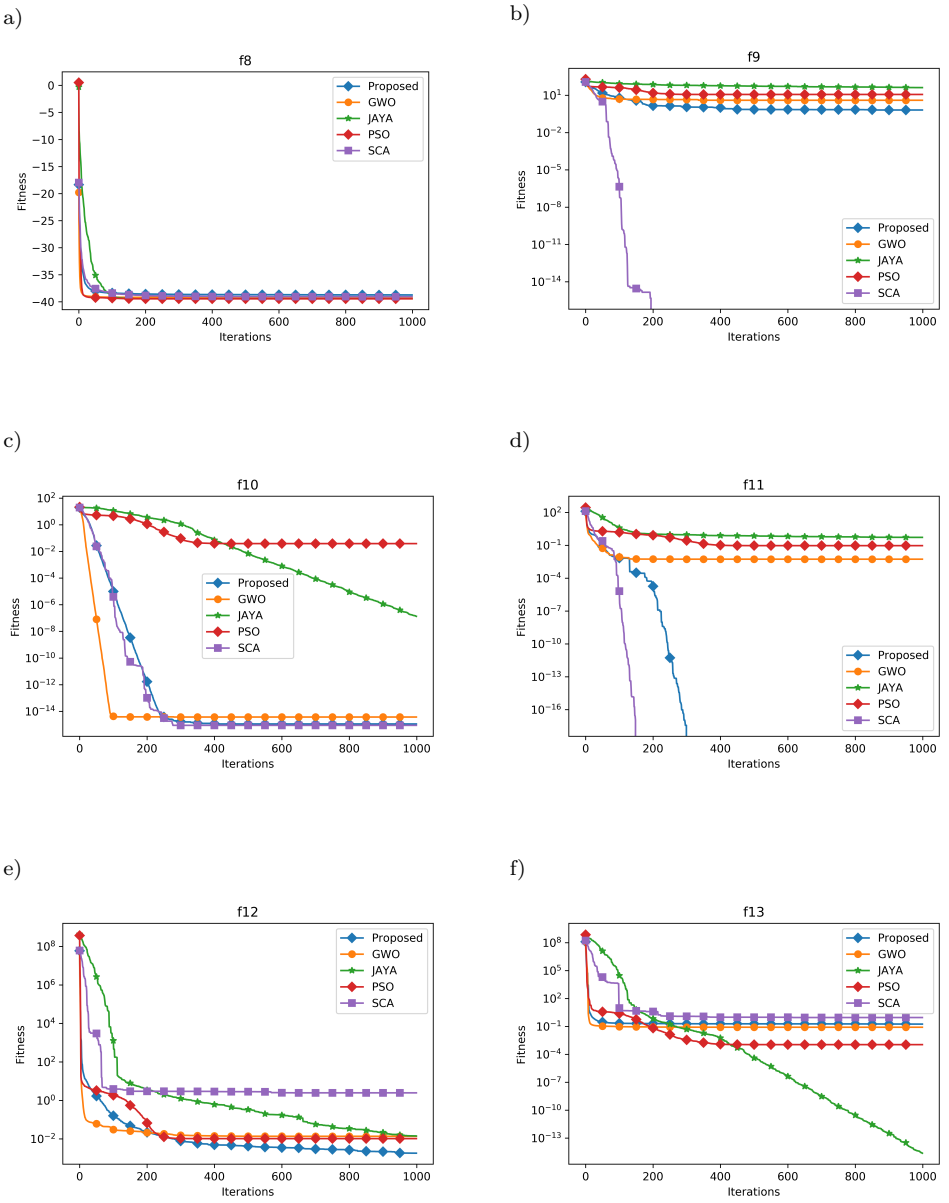

Figure 4. Flexible multi-modal benchmark functions: a) Schwefel 2.26; b) Rastrigin; c) Ackley; d) Griewank; e) Penalized 1; f) Penalized 2

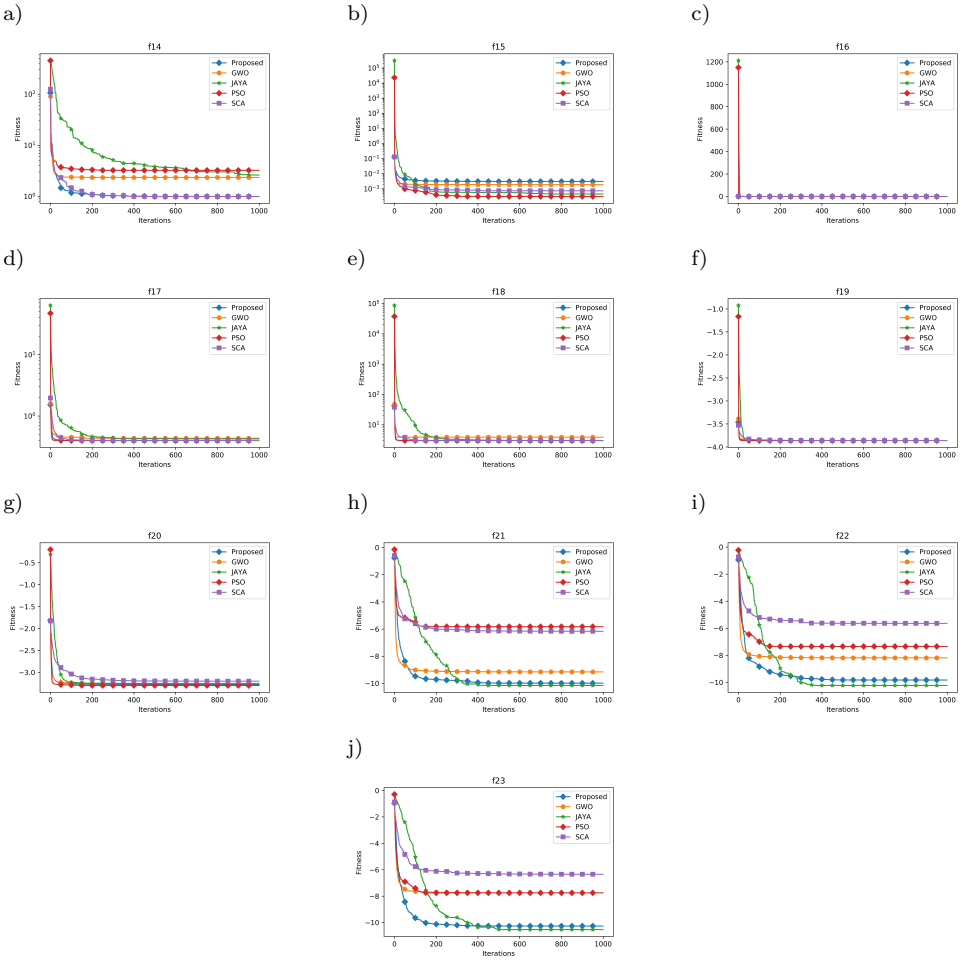

Figure 5. Fixed-dimension benchmark functions: a) Shekel's Foxholes; b) Kowalik; c) Six-hump camel back; d) Branin; e) Goldstein-Price; f) Hartman's family; g) Hartman's family; h) Shekel's family; i) Shekel's family; j) Shekel's family

5. Performance evaluation of benchmark clustering problems

5.1. Data set

In order to measure the performance of the proposed algorithm as compared to the GWO, JAYA, PSO, SCA, and k-means algorithms, we considered 15 benchmark data sets (as depicted in Table 5) from the UCI machine-learning repository.

Initially, the data sets were pre-processed by replacing the missing values with mean values of the same attribute. Similarly, those attributes that had more than two categories were simply removed from the data set. Some attributes that had

binary categorical information were replaced with either zero or one. Additionally, some data sets were balanced by using an up-sampling algorithm to overcome the class-imbalance problem. The experiment was carried out after removing the class label attribute from each data set. Subsequently, the performance was measured by comparing the predicted class labels with the original class labels.

Table 5
Data set description

Data set	Instances	No of Attributes	Attribute Characteristics	Classes
Appendicitis	106	07	Real	02
Breast Cancer	699	09	Real	02
Bupa Liver Disorder	345	06	Integer, Real	02
Ecoli	336	07	Real	08
Haberman's Survival	306	03	Real	02
Hepatitis	155	19	Categorical, Integer, Real	02
Indian Liver Patient	583	10	Integer, Real	02
Ionosphere	351	34	Integer, Real	02
Iris	150	04	Real	03
Lung Cancer	32	56	Integer	03
Mammographic Mass	961	06	Integer	02
Mushroom	8124	21	Integer	02
Seeds	210	07	Real	03
WDBC	569	30	Real	02
Zoo	101	16	Integer	07

5.2. Performance metrics

Accuracy is the most-used performance metric for evaluating the performance of classification and clustering algorithms. The accuracy performance metric is calculated from the confusion matrix (as shown in Eq. (44)). However, if the input dataset is highly imbalanced, the accuracy does not measure the performance perfectly [5].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{44}$$

Consider the scenario of an imbalance data set that has 90 negative instances and 10 positive instances; for example, if a classifier is able to classify all positive instances perfectly and zero instances from the negative class. If we only consider accuracy as a performance metric, then the overall performance will be calculated as 90%. Unfortunately, this classifier is useless, as the main intention is to classify the positive instances. In these types of data sets, the accuracy performance metric is not sufficient for measuring the performance of the classification model. Several methods have been proposed to address the above issue (which arises from the imbalanced nature of data sets). One such method is to assign a high cost for misclassifying minority class instances in order to minimize the error. Additionally, sampling is such a technique to overcome the imbalance issue – either by increasing the minority class instances

or by reducing the majority class instances by using up-sampling or down-sampling methods, respectively. Most importantly, we cannot apply a sampling method to a real problem for which classification is required; therefore, it is the job of researchers to select the appropriate performance metric for the domain of the classification.

Moreover, to maintain a trade-off between false negatives and false positives, several measures have been proposed [8]. Additionally, Matthew's correlation co-efficient (MCC – as shown in Eq. (45)) is such a measure that calculates the performance of classification models by considering the above two cases using a confusion matrix. The range of MCC lies between –1 to +1, where +1 denotes better performance, –1 denotes worse performance, and 0 means a coin-tossing classifier. Moreover, this performance metric is least influenced by the class-imbalance problem. MCC can be calculated as follows:

$$\frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{45}$$

On other hand, sensitivity accesses the instances of minority classes (true positive) in order to calculate a model's performance. Similarly, specificity (as shown in Eq. (46)) accesses negative class instances (false positives) in order to effectively measure performance. Additionally, precision is another such performance metric that measures a model's exactness; higher precision indicates better performance. From an information-retrieval point of view, precision calculates performance by checking for positive instances in a corpus. For instance, it checks a model's exactness by checking how well the classifier is able to classify the positive instances from the model's performance:

$$Specificity = \frac{TN}{TN + FP} \tag{46}$$

F-score is another commonly used performance metric that maintains a trade-off between precision and recall (sensitivity); an F-score will be zero if either precision or recall is evaluated as zero. The performance metric can be calculated as shown in Eq. (47):

$$F\text{-}score = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{47}$$

However, the F-score measure is sensitivity toward class-swapping problems. For instance, if positive class instances are swapped with negative class instances, then F-score will produce varying results. Despite the several flaws in F-score, it is still considered to be one of most widely used metrics among researchers.

5.3. Performance evaluation on individual data sets

The effectiveness of our proposed algorithm was accessed by using the accuracy, specificity, F-score, and MCC performance metrics. The results that were obtained after 100 independent simulations were used to make a comparative analysis with the GWO, JAYA, PSO, SCA, and traditional k-means algorithms. Additionally, we considered

15 machine-learning data sets from the UCI machine-learning repositories; also, we conducted a statistical test on the aforementioned algorithms using Duncan's multiple range test and the Friedman and Nemenyi hypothesis test at a 95% confidence interval to effectively measure the performance of our proposed algorithm. The above statistical analysis was comprised of two steps: (i) a result analysis using individual data sets; and (ii) considering all of the data sets together. In each table, * means that the same alphabets within a single column are statistically equivalent to each other with a 95% significance level (using Duncan's multiple range-based statistical test).

Table 6 demonstrates the mean and standard deviation statistical results of 100 independent simulations with 1000 iterations each for all of the algorithms on the Appendicitis data set. The results showed that the proposed algorithm provided the best mean result across all performance metrics as compared to the comparative algorithms. Moreover, the statistical test also confirmed the superior statistical performance of our proposed algorithm when compared to the JAYA, PSO, SCA, and k-means algorithms with respect to all of the performance metrics.

Table 7 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Breast Cancer data set. The results showed that the proposed algorithm achieved the highest mean result across all of the performance metrics than all of the other algorithms. Moreover, the statistical test also showed that the GWO algorithm provided the second-best mean result across all of the performance metrics.

Table 8 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Bupa data set. The result showed that our proposed algorithm achieved the best mean result among the comparative algorithms. However, the GWO, JAYA, and SCA algorithms also showed statistically superior results that did not differ significantly from our proposed algorithm with respect to the accuracy and MCC performance metrics. Similarly, our algorithm also achieved statistically equivalent performance with the superior mean results that were obtained by the SCA and k-means algorithms with respect to the specificity and F-score performance metrics.

Table 6

Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Appendicitis data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	$\mathbf{84.1198 \pm 0.3451}^{a}$	$\mathbf{0.7480 \pm 0.0263}^{a}$	$\mathbf{0.8545 \pm 0.0043}^{a}$	$\mathbf{0.6952 \pm 0.0068}^{a}$
GWO	83.0049 ± 0.2638^{b}	0.7229 ± 0.0238^{b}	0.8465 ± 0.0019^{b}	0.6764 ± 0.0035^{b}
JAYA	79.1217 ± 0.6563^{e}	0.7237 ± 0.1132^{b}	0.8027 ± 0.0183^{e}	0.6027 ± 0.0112^{e}
PSO	80.2758 ± 3.5350^{d}	0.6924 ± 0.0986^{c}	0.8227 ± 0.0270^{c}	0.6269 ± 0.0563^{d}
SCA	81.3237 ± 1.1710^{c}	0.7340 ± 0.0643^{ab}	0.8266 ± 0.0129^{c}	0.6391 ± 0.0197^{c}
K-means	77.6500 ± 0.2781^{f}	0.00399 ± 0.0004^{d}	0.8082 ± 0.0028^{d}	0.5861 ± 0.0071^{f}

Table 7
Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Breast Cancer data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	**97.8300 ± 0.2057**^{a}	**0.9932 ± 0.0045**^{a}	**0.9780 ± 0.0021**^{a}	**0.9571 ± 0.0041**^{a}
GWO	97.6846 ± 0.1363^{b}	0.9896 ± 0.0038^{b}	0.9766 ± 0.0014^{b}	0.9541 ± 0.0026^{b}
JAYA	96.9629 ± 0.8921^{d}	0.9733 ± 0.0235^{c}	0.9696 ± 0.0084^{d}	0.9397 ± 0.0176^{d}
PSO	97.1975 ± 0.6005^{c}	0.9892 ± 0.0126^{b}	0.9715 ± 0.0061^{c}	0.9448 ± 0.0118^{c}
SCA	97.6731 ± 0.2670^{b}	0.9895 ± 0.0060^{b}	0.9764 ± 0.0027^{b}	0.9539 ± 0.0054^{b}
K-means	94.3100 ± 0.0000^{e}	0.9103 ± 0.0000^{d}	0.9449 ± 0.0000^{e}	0.8881 ± 0.0000^{e}

Table 8
Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Bupa data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	**63.1775 ± 2.8151**^{a}	0.6294 ± 0.1252^{a}	0.6296 ± 0.0382^{ab}	**0.2701 ± 0.0575**^{a}
GWO	62.5075 ± 2.7603^{a}	0.5925 ± 0.1429^{a}	0.6329 ± 0.0421^{ab}	0.2586 ± 0 .0530^{a}
JAYA	62.3525 ± 3.4787^{a}	0.6164 ± 0.1605^{a}	0.6187 ± 0.0668^{b}	0.2587 ± 0.0705^{a}
PSO	57.9850 ± 3.5242^{b}	0.5479 ± 0.2379^{b}	0.5716 ± 0.1193^{c}	0.1764 ± 0.0639^{b}
SCA	62.9175 ± 2.6813^{a}	**0.6375 ± 0.1080**^{a}	0.6231 ± 0.0411^{ab}	0.2635 ± 0.0545^{a}
K-means	52.8050 ± 1.6390^{c}	0.2070 ± 0.1097^{c}	**0.6406 ± 0.0309**^{a}	0.0767 ± 0.0368^{c}

Table 9 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Ecoli data set. The results showed that the proposed algorithm achieved the second-best mean result among all of the algorithms in all of the performance metrics except for F-score. Additionally, the SCA algorithm achieved the best mean and statistically superior results as compared to the remaining algorithms with respect to all of the performance metrics. Moreover, the results that were obtained by the JAYA algorithm did not differ from our proposed algorithm; hence, both were considered to be statistical equivalent to each other.

Table 9
Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Ecoli data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	67.9217 ± 4.5816^{b}	0.9542 ± 0.0066^{b}	0.6656 ± 0.0519^{b}	0.6397 ± 0.0541^{b}
GWO	64.8940 ± 6.3598^{c}	0.9499 ± 0.0091^{c}	0.6324 ± 0.0664^{c}	0.6109 ± 0.0669^{c}
JAYA	67.6909 ± 5.6294^{b}	0.9538 ± 0.0080^{b}	0.6657 ± 0.0591^{b}	0.6407 ± 0.0630^{b}
PSO	50.4691 ± 8.2351^{d}	0.9292 ± 0.0118^{d}	0.5044 ± 0.0909^{d}	0.4700 ± 0.1023^{d}
SCA	**73.1541 ± 5.1768**^{a}	**0.9616 ± 0.0074**^{a}	**0.7176 ± 0.0586**^{a}	**0.6973 ± 0.0585**^{a}
K-means	22.5647 ± 6.7946^{e}	0.8894 ± 0.0097^{e}	0.2173 ± 0.0725^{e}	0.2490 ± 0.0690^{e}

Table 10 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Haberman's Survival data set. The results showed that our proposed algorithm achieved the best mean result for the F-score performance metric. With respect to the same performance metric, the next-best performance was achieved by GWO (67.2772); this shows a statistically equivalent result whose performance was not statistically different from the best result. With respect to the accuracy, specificity, and MCC performance metrics, GWO similarly showed the best results. However, the statistical results showed that the performance of GWO was not statistically different from our proposed algorithm.

Table 10

Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Haberman's Survival data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	67.0021 ± 2.3571^{a}	0.5725 ± 0.0933^{a}	$\mathbf{0.6975 \pm 0.0343^{a}}$	0.3518 ± 0.0422^{ab}
GWO	$\mathbf{67.2772 \pm 3.0389^{a}}$	$\mathbf{0.5873 \pm 0.1188^{a}}$	0.6960 ± 0.0398^{a}	$\mathbf{0.3585 \pm 0.0539^{a}}$
JAYA	64.0913 ± 2.8222^{c}	0.5497 ± 0.1124^{a}	0.6647 ± 0.0666^{b}	0.2956 ± 0.0547^{c}
PSO	58.3416 ± 4.6764^{d}	0.5506 ± 0.2693^{a}	0.5513 ± 0.1867^{c}	0.1999 ± 0.0919^{d}
SCA	66.0758 ± 2.4604^{b}	0.5641 ± 0.1006^{a}	0.6866 ± 0.0550^{ab}	0.3349 ± 0.0451^{b}
K-means	56.7232 ± 3.8905^{e}	0.2828 ± 0.0869^{b}	0.6608 ± 0.0456^{b}	0.1737 ± 0.0983^{e}

Table 11 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Hepatitis data set. The results showed that the JAYA algorithm had the best mean results across all of the performance metrics except F-score. Moreover, the GWO algorithm showed its best mean result in the F-score performance metric. However, the mean results that were obtained by our proposed algorithm did not statistically differ from the most-promising GWO and JAYA algorithms with respect to all of the performance metrics.

Table 11

Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Hepatitis data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	62.0991 ± 3.4906^{ab}	0.5891 ± 0.1570^{ab}	0.6223 ± 0.0805^{a}	0.2555 ± 0.0702^{a}
GWO	62.02883 ± 0.8488^{b}	0.5393 ± 0.1469^{d}	$\mathbf{0.6403 \pm 0.0762^{a}}$	0.2553 ± 0.0780^{a}
JAYA	$\mathbf{63.1504 \pm 3.7773^{a}}$	$\mathbf{0.6157 \pm 0.1338^{a}}$	0.6255 ± 0.0928^{a}	$\mathbf{0.2735 \pm 0.0716^{a}}$
PSO	57.8790 ± 3.9376^{d}	0.5864 ± 0.2424^{abc}	0.5342 ± 0.1721^{b}	0.1845 ± 0.0801^{c}
SCA	62.3564 ± 3.5145^{ab}	0.5829 ± 0.1451^{abc}	0.6283 ± 0.0818^{a}	0.2590 ± 0.0702^{a}
K-means	60.2482 ± 2.9155^{c}	$0.5511 \pm 0.1058b^{c}$	0.6128 ± 0.0792^{a}	0.2131 ± 0.0609^{b}

Table 12 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Indian Liver Patient data set. The results showed that the GWO algorithm had the best mean result and was statistically superior as compared to the others across all of the performance metrics except specificity. However, our proposed algorithm achieved statistical superior performance only in F-score performance metric.

Table 12

Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Indian Liver Patient data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	68.6980 ± 1.3679^{b}	$0.7637 \pm 0.0458b^{c}$	0.6600 ± 0.0256^{a}	0.3799 ± 0.0251^{b}
GWO	$\mathbf{69.7212 \pm 1.1622^{a}}$	0.7827 ± 0.0332^{ab}	$\mathbf{0.6679 \pm 0.0271^{a}}$	$\mathbf{0.4015 \pm 0.0188^{a}}$
JAYA	67.4790 ± 1.9574^{c}	$0.7638 \pm 0.0620b^{c}$	0.6383 ± 0.0621^{c}	0.3590 ± 0.0266^{c}
PSO	65.1085 ± 3.0997^{e}	0.7285 ± 0.1240^{d}	0.6112 ± 0.0883^{b}	0.3159 ± 0.0512^{e}
SCA	68.3283 ± 1.1186^{b}	0.7494 ± 0.0387^{c}	0.6598 ± 0.0260^{a}	0.3711 ± 0.0192^{b}
K-means	66.6675 ± 2.2640^{d}	$\mathbf{0.7853 \pm 0.0460^{a}}$	0.6187 ± 0.0522^{c}	0.3449 ± 0.0397^{d}

Table 13 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Ionosphere data set. The results showed that GWO achieved the highest means for the accuracy and MCC performance metrics, whereas JAYA achieved the highest mean in the specificity measure. Moreover, our proposed algorithm achieved statistically equivalent results with the most-promising results except the F-score performance metric. However, it achieved the second-highest significant result when compared to the remaining algorithms.

Table 13

Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Ionosphere data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	70.2854 ± 4.1247^{a}	0.7435 ± 0.1464^{ab}	0.6853 ± 0.0619^{b}	0.4217 ± 0.0815^{a}
GWO	$\mathbf{70.7856 \pm 4.6489^{a}}$	0.7544 ± 0.1167^{ab}	0.6921 ± 0.0527^{b}	$\mathbf{0.4258 \pm 0.0972^{a}}$
JAYA	69.5900 ± 4.7908^{a}	$\mathbf{0.7677 \pm 0.1457^{a}}$	0.6633 ± 0.0929^{c}	0.4113 ± 0.0884^{a}
PSO	67.5961 ± 5.1558^{b}	0.7187 ± 0.1395^{b}	0.6511 ± 0.1046^{c}	0.3650 ± 0.0982^{b}
SCA	70.5022 ± 4.7309^{a}	0.7145 ± 0.1387^{b}	0.6994 ± 0.0576^{ab}	0.4213 ± 0.0928^{a}
K-means	67.2112 ± 6.9085^{b}	0.5378 ± 0.2271^{c}	$\mathbf{0.7133 \pm 0.0214^{a}}$	0.3679 ± 0.1079^{b}

Table 14 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Iris data set. The results showed that the highest mean result was achieved by our proposed algorithm across all of the performance metrics. Addition-

ally, the GWO, JAYA, and SCA algorithms showed statistically equivalent results with our proposed algorithm.

Table 15 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Lung Cancer data set. The results showed that the JAYA algorithm achieved the highest mean result across all of the performance metrics. However, our proposed algorithm showed a statistically equivalent result with the most-promising JAYA algorithm across all of the performance metrics except MCC. Similarly, GWO also achieved a statistically equivalent result with the most-promising results across all of the performance metrics.

Table 14

Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Iris data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	**96.8800 ± 1.4108^{a}**	**0.9844 ± 0.0071^{a}**	**0.9687 ± 0.0142^{a}**	**0.9538 ± 0.0208^{a}**
GWO	95.0800 ± 1.6383^{a}	0.9754 ± 0.0082^{a}	0.9507 ± 0.0165^{a}	0.9274 ± 0.0241^{a}
JAYA	96.5196 ± 1.8118^{a}	0.9826 ± 0.0091^{a}	0.9651 ± 0.0182^{a}	0.9486 ± 0.0265^{a}
PSO	77.7929 ± 10.7622^{b}	0.8890 ± 0.0538^{b}	0.7407 ± 0.1367^{b}	0.6985 ± 0.1445^{b}
SCA	95.9595 ± 2.5065^{a}	0.9798 ± 0.0125^{a}	0.9595 ± 0.0252^{a}	0.9404 ± 0.0369^{a}
K-means	67.6793 ± 21.6234^{c}	0.8384 ± 0.1081^{c}	0.6513 ± 0.2364^{c}	0.6342 ± 0.2158^{c}

Table 15

Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Lung Cancer data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	66.0248 ± 6.3476^{ab}	0.8301 ± 0.0317^{ab}	0.6307 ± 0.0719^{a}	0.5097 ± 0.0903b^{c}
GWO	67.4868 ± 7.7042^{a}	0.8374 ± 0.0385^{a}	0.6399 ± 0.0871^{a}	0.5349 ± 0.1110^{ab}
JAYA	**67.8965 ± 7.3476^{a}**	**0.8395 ± 0.0367^{a}**	**0.6471 ± 0.0858^{a}**	**0.5396 ± 0.1032^{a}**
PSO	55.3584 ± 7.5928^{c}	0.7768 ± 0.0380^{c}	0.5005 ± 0.0831^{c}	0.3775 ± 0.1001^{d}
SCA	65.0763 ± 5.9857^{b}	0.8254 ± 0.0299^{b}	0.6066 ± 0.0746^{b}	0.5028 ± 0.0841^{c}
K-means	39.9762 ± 2.7303^{d}	0.6999 ± 0.0136^{d}	0.3407 ± 0.0547^{d}	0.2337 ± 0.0447^{e}

Table 16 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Mammographic Mass data set. The results showed that, for all of the performance metrics except specificity, our proposed algorithm achieved the highest mean result and was statistically superior to the other algorithms. However, our proposed algorithm achieved the second-highest mean rank in the specificity performance metric. Additionally, the GWO, JAYA, and SCA algorithms were statistically

equivalent to our proposed algorithm in the specificity performance metric. Furthermore, the SCA algorithm showed statistically equivalent results with our proposed algorithm in the accuracy, F-score, and MCC performance metrics.

Table 16
Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Mammographic Mass data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	**75.6877 ± 4.0665**^{a}	0.7959 ± 0.1034^{b}	**0.7445 ± 0.0515**^{a}	**0.5247 ± 0.0797**^{a}
GWO	72.4648 ± 4.1258^{b}	0.7900 ± 0.1026^{b}	0.7019 ± 0.0552^{b}	0.4625 ± 0.0829^{c}
JAYA	74.5636 ± 4.1091^{a}	0.7703 ± 0.0972^{b}	0.7359 ± 0.0549^{a}	0.5010 ± 0.0812^{b}
PSO	64.4080 ± 4.8593^{d}	**0.8444 ± 0.1335**^{a}	0.5227 ± 0.1687^{c}	0.3342 ± 0.0625^{e}
SCA	75.0362 ± 4.4172^{a}	0.7820 ± 0.0856^{b}	0.7396 ± 0.0555^{a}	0.5085 ± 0.0875^{ab}
K-means	68.1900 ± 1.3207^{c}	0.6395 ± 0.0455^{c}	0.6942 ± 0.0185^{b}	0.3667 ± 0.0279^{d}

Table 17 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Mushroom data set. The results showed that our proposed algorithm had the highest mean results and was statistically significant in the accuracy and specificity performance metrics. Moreover, GWO showed higher mean results than the proposed algorithm in both the F-score and MCC performance metrics. However, our proposed algorithm showed a statistically equivalent result with the GWO algorithm (except in the case of F-score).

Table 17
Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Mushroom data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	**73.8688 ± 2.6607**^{a}	**0.6552 ± 0.1314**^{a}	0.7562 ± 0.0359^{b}	0.5014 ± 0.0482^{a}
GWO	73.7408 ± 2.8921^{a}	0.5889 ± 0.1357^{c}	**0.7703 ± 0.0260**^{a}	**0.5139 ± 0.0435**^{a}
JAYA	73.5915 ± 3.2323^{a}	0.6308 ± 0.1373^{ab}	0.7589 ± 0.0366^{b}	0.4996 ± 0.0568^{a}
PSO	69.7969 ± 2.4989^{c}	0.4706 ± 0.0875^{d}	0.7519 ± 0.0373^{b}	0.4550 ± 0.0589^{c}
SCA	72.1397 ± 2.3134^{b}	0.5977 ± 0.1464b^{c}	0.7484 ± 0.0383^{b}	0.4794 ± 0.0459^{b}
K-means	65.8667 ± 5.8492^{d}	0.3812 ± 0.1035^{e}	0.7332 ± 0.0406^{c}	0.3791 ± 0.1318^{d}

Table 18 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Seeds data set. The results showed that our proposed algorithm achieved the best mean results; it can also be observed that the performance results of the GWO, JAYA, and SCA algorithms were not statistically different than our

most-promising proposed algorithm. Additionally, the JAYA algorithm achieved the second-highest mean results with respect to all of the performance metrics.

Table 18
Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Seeds data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	**91.0237 ± 0.6098**^{a}	**0.9551 ± 0.0030**^{a}	**0.9103 ± 0.0059**^{a}	**0.8670 ± 0.0091**^{a}
GWO	90.7475 ± 0.7325^{a}	0.9537 ± 0.0037^{a}	0.9075 ± 0.0072^{a}	0.8627 ± 0.0108^{a}
JAYA	90.9189 ± 0.8722^{a}	0.9546 ± 0.00436^{a}	0.9091 ± 0.0088^{a}	0.8657 ± 0.0127^{a}
PSO	69.7148 ± 15.2418^{b}	0.8486 ± 0.0762^{b}	0.6655 ± 0.1830^{b}	0.5798 ± 0.1965^{b}
SCA	89.8999 ± 1.5595^{a}	0.9495 ± 0.0078^{a}	0.8985 ± 0.0162^{a}	0.8514 ± 0.0226^{a}
K-means	62.4334 ± 21.2349^{c}	0.8122 ± 0.1062^{c}	0.6122 ± 0.2192^{c}	0.5650 ± 0.2147^{b}

Table 19 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the WDBC data set. The results showed that, across all of the performance metrics, the JAYA algorithm showed the best mean results and was statistically superior to its comparative algorithms. Additionally, the mean performance of the GWO algorithm was not statistically different from the JAYA algorithm across all of the performance metrics. Similarly, the mean performance of the SCA algorithm was statistically equivalent to the most-promising JAYA algorithm across all of the performance metrics except specificity.

Table 19
Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on WDBC data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	89.7380 ± 1.1390^{b}	0.8948 ± 0.0376^{b}	0.8975 ± 0.0121^{b}	0.7968 ± 0.0227^{b}
GWO	90.6228 ± 0.6068^{a}	0.9193 ± 0.0279^{a}	0.9048 ± 0.0077^{a}	0.8141 ± 0.0123^{a}
JAYA	**90.7250 ± 0.8613**^{a}	**0.9165 ± 0.0344**^{a}	**0.9061 ± 0.0104**^{a}	**0.8167 ± 0.0166**^{a}
PSO	87.7152 ± 3.9506^{c}	0.8872 ± 0.0912^{bc}	0.8730 ± 0.0534^{c}	0.7668 ± 0.0621^{c}
SCA	90.1944 ± 1.2834^{ab}	0.8749 ± 0.0507^{c}	0.9044 ± 0.0124^{a}	0.8083 ± 0.0239^{a}
K-means	77.8938 ± 2.6842^{d}	0.5602 ± 0.0563^{d}	0.8191 ± 0.0180^{d}	0.6209 ± 0.0403^{d}

Table 20 demonstrates the mean and standard deviation results that were obtained for 100 independent simulations with 1000 iterations each for all of the algorithms on the Zoo data set. The results showed that, across all of the performance metrics, the SCA algorithm achieved the highest mean results. Additionally, the statistical test confirms that our proposed algorithm is showing second best performance among the remaining algorithms in all performance measures.

Table 20
Performance of different algorithms considering accuracy, specificity, F-score, and MCC metrics on Zoo data set

	Accuracy Mean ± Std. Dev.	Specificity Mean ± Std. Dev.	F-score Mean ± Std. Dev.	MCC Mean ± Std. Dev.
Proposed	68.8963 ± 8.2976^{b}	0.9482 ± 0.0138^{b}	0.6719 ± 0.0824^{b}	0.6443 ± 0.0970^{b}
GWO	65.2430 ± 9.4536^{c}	0.9421 ± 0.0158^{c}	0.6436 ± 0.0895^{c}	0.6152 ± 0.1001^{c}
JAYA	70.0181 ± 8.1827^{b}	0.9500 ± 0.0137^{b}	0.6824 ± 0.0801^{b}	0.6620 ± 0.0931^{b}
PSO	50.7065 ± 12.9577^{d}	0.9178 ± 0.0216^{d}	0.4928 ± 0.1301^{d}	0.4711 ± 0.1283^{d}
SCA	$\mathbf{73.0321 \pm 8.7358^{a}}$	$\mathbf{0.9551 \pm 0.0146^{a}}$	$\mathbf{0.7141 \pm 0.0858^{a}}$	$\mathbf{0.7050 \pm 0.0904^{a}}$
K-means	27.7289 ± 9.4555^{e}	0.8795 ± 0.0158^{e}	0.2656 ± 0.0969^{e}	0.3035 ± 0.0866^{e}

5.4. Considering all data sets together

The effectiveness of the proposed algorithm was evaluated considering the results of the 15 data sets with respect to the accuracy, specificity, F-score, and MCC performance metrics. Additionally, we applied the Friedman and Nemenyi hypothesis test on the obtained results. The mean rank results that were obtained from the statistical test are presented in Figures 6, 7, 8, and 9 for the accuracy, specificity, F-score, and MCC performance metrics, respectively.

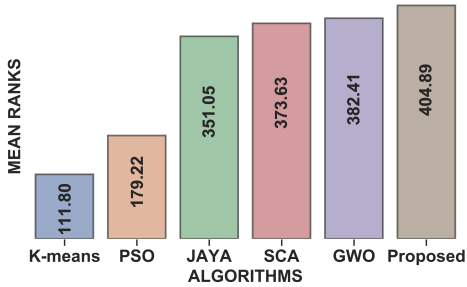

Figure 6. Mean rank of meta-heuristic algorithms for clustering using 15 benchmark data sets on accuracy as performance metric with p-value = .000 and critical distance = 1.9

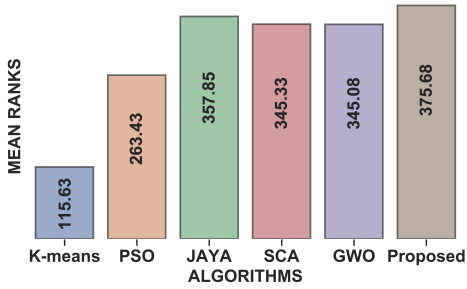

Figure 7. Mean rank of meta-heuristic algorithms for clustering using 15 benchmark data sets on specificity as performance metric with p-value = .000 and critical distance = 1.9

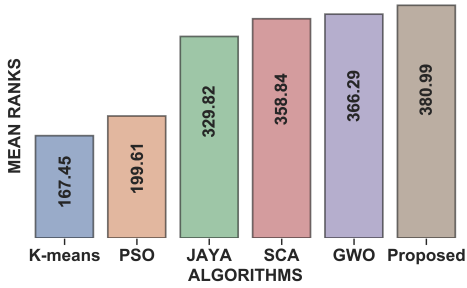

Figure 8. Mean rank of meta-heuristic algorithms for clustering using 15 benchmark data sets on F-score as performance metric with p-value = .000 and critical distance = 1.9

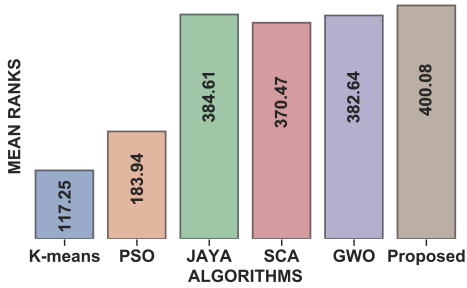

Figure 9. Mean rank of meta-heuristic algorithms for clustering using 15 benchmark data sets on MCC as performance metric with p-value = .000 and critical distance = 1.9

Figure 10 demonstrates a boxplot visualization of the clustering results considering the accuracy performance metric on 15 different data sets for the different clustering algorithms (GWO, PSO, k-means, JAYA, SCA, and our proposed GWOJAYA). It can be seen that the majority of the good clustering results were achieved by our proposed algorithm from among 15 different experimental works. The good clustering results were achieved by the Appendicitis, Breast Cancer, Bupa, Haberman's Survival, Ionosphere, Iris, Mammographic Mass, Mushroom, and Seeds data sets considering the accuracy performance metric only.

Figure 11 demonstrates a boxplot visualization of the clustering results considering the specificity performance metric on 15 different data sets for the different clustering algorithms (GWO, PSO, k-means, JAYA, SCA, and and our proposed algorithm). It can be seen that the majority of the good clustering results were achieved by our proposed algorithm from among 15 different experimental works. The good clustering results were achieved by the Appendicitis, Breast Cancer, Bupa, Haberman's Survival, Indian Liver Patient, Iris, Mushroom, and Seeds data sets considering the specificity performance metric only.

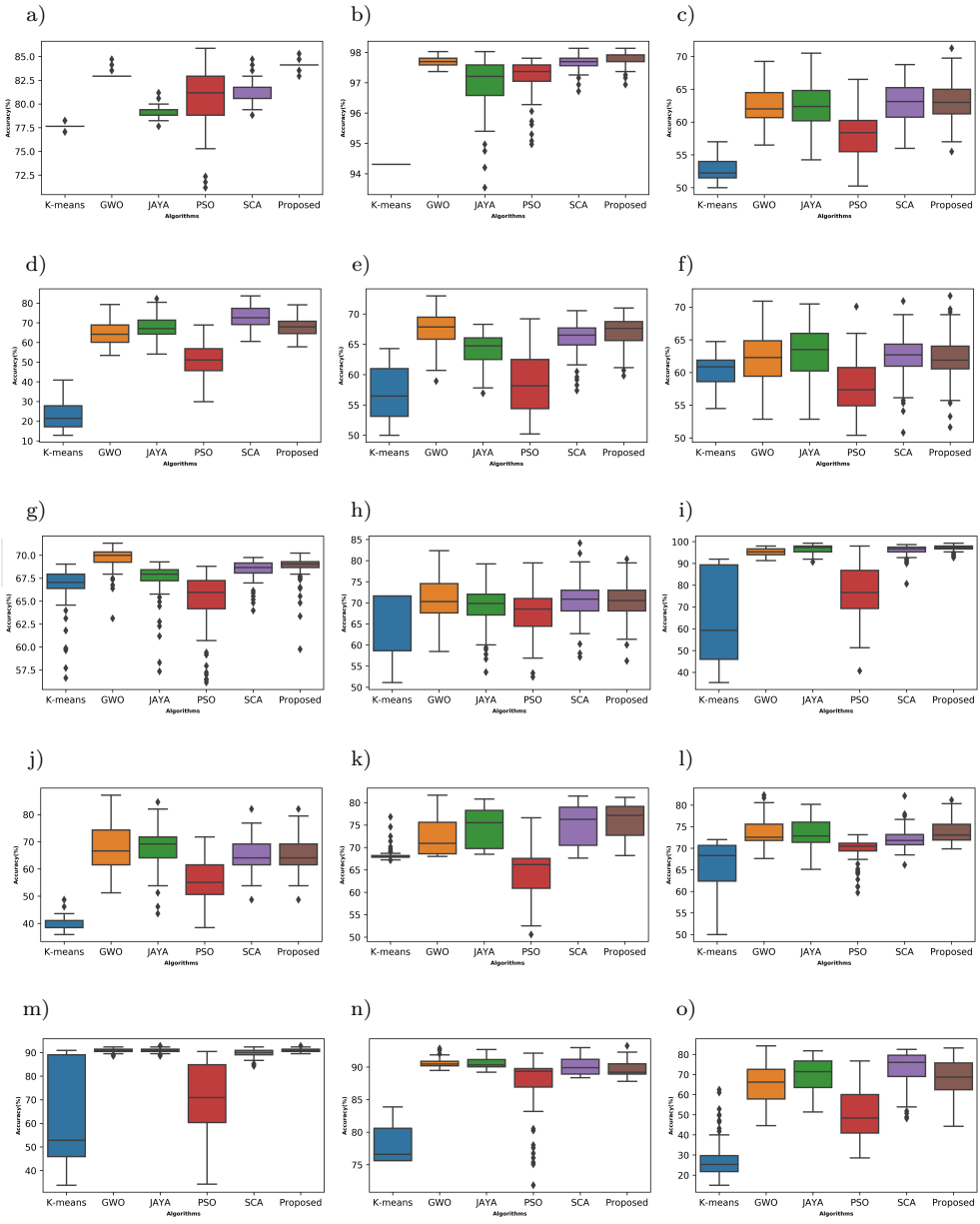

Figure 10. Accuracy metrics among GWO, PSO, k-means, JAYA, SCA, and our proposed algorithm for 15 data sets: a) Appendicitis; b) Breast Cancer; c) Bupa; d) Ecoli; e) Haberman's Survival; f) Hepatitis; g) Indian Liver Patient; h) Ionosphere; i) Iris; j) Lung Cancer; k) Mammographic Mass; l) Mushroom; m) Seeds; n) WDBC; o) Zoo

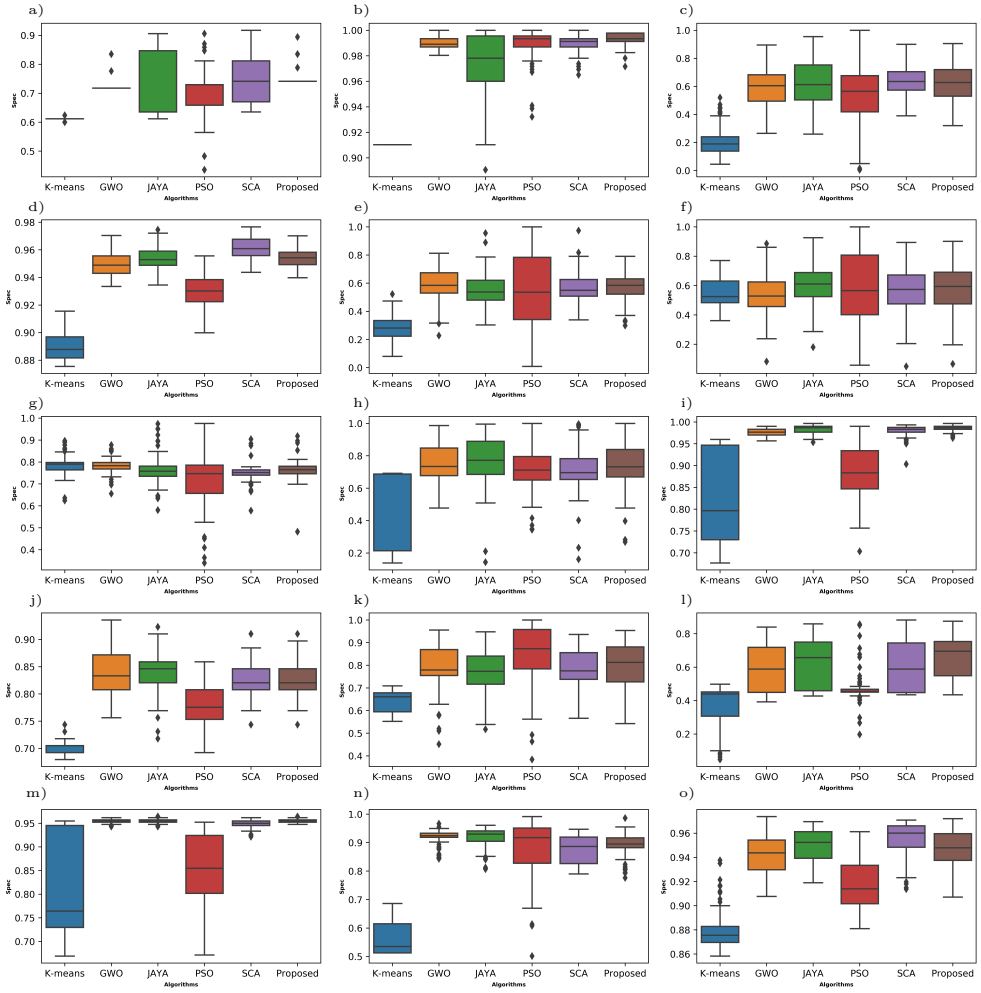

Figure 11. Specificity metrics among GWO, PSO, k-means, JAYA, SCA, and our proposed algorithm for 15 data sets: a) Appendicitis; b) Breast Cancer; c) Bupa; d) Ecoli; e) Haberman's Survival; f) Hepatitis; g) Indian Liver Patient; h) Ionosphere; i) Iris; j) Lung Cancer; k) Mammographic Mass; l) Mushroom; m) Seeds; n) WDBC; o) Zoo

Figure 12 demonstrates a boxplot visualization of the clustering results considering the F-score performance metric on 15 different data sets for the different clustering algorithms (GWO, PSO, k-means, JAYA, SCA, and our proposed algorithm). It can be seen that the majority of the good clustering results were achieved by our proposed algorithm from among 15 different experimental works. The good clustering results were achieved by the Appendicitis, Breast Cancer, Bupa, Haberman's Survival, Hepatitis, Iris, Mammographic Mass, Mushroom, and Seeds data sets considering the F-score performance metric only.

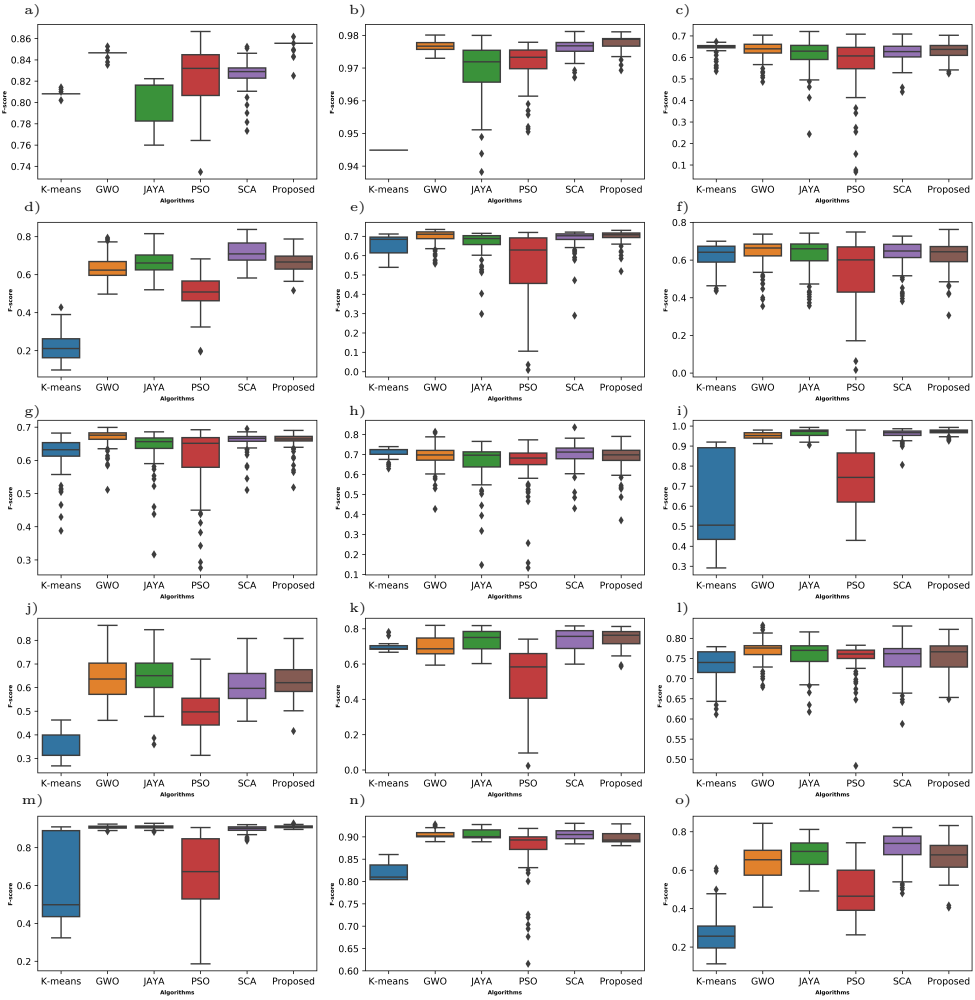

Figure 12. F-score metrics among GWO, PSO, k-means, JAYA, SCA, and our proposed algorithm for 15 data sets: a) Appendicitis; b) Breast Cancer; c) Bupa; d) Ecoli; e) Haberman's Survival; f) Hepatitis; g) Indian Liver Patient; h) Ionosphere; i) Iris; j) Lung Cancer; k) Mammographic Mass; l) Mushroom; m) Seeds; n) WDBC; o) Zoo

Figure 13 demonstrates a boxplot visualization of the clustering results considering the MCC performance metrics on 15 different data sets for the different clustering algorithms (GWO, PSO, k-means, JAYA, SCA, and our proposed algorithm). It can be seen that the majority of the good clustering results were achieved by our proposed algorithm from among 15 different experimental works. The good clustering results were achieved by the Appendicitis, Breast Cancer, Bupa, Haberman's Survival, Ionosphere, Iris, Mammographic Mass, and Seeds data sets considering the MCC performance metric only.

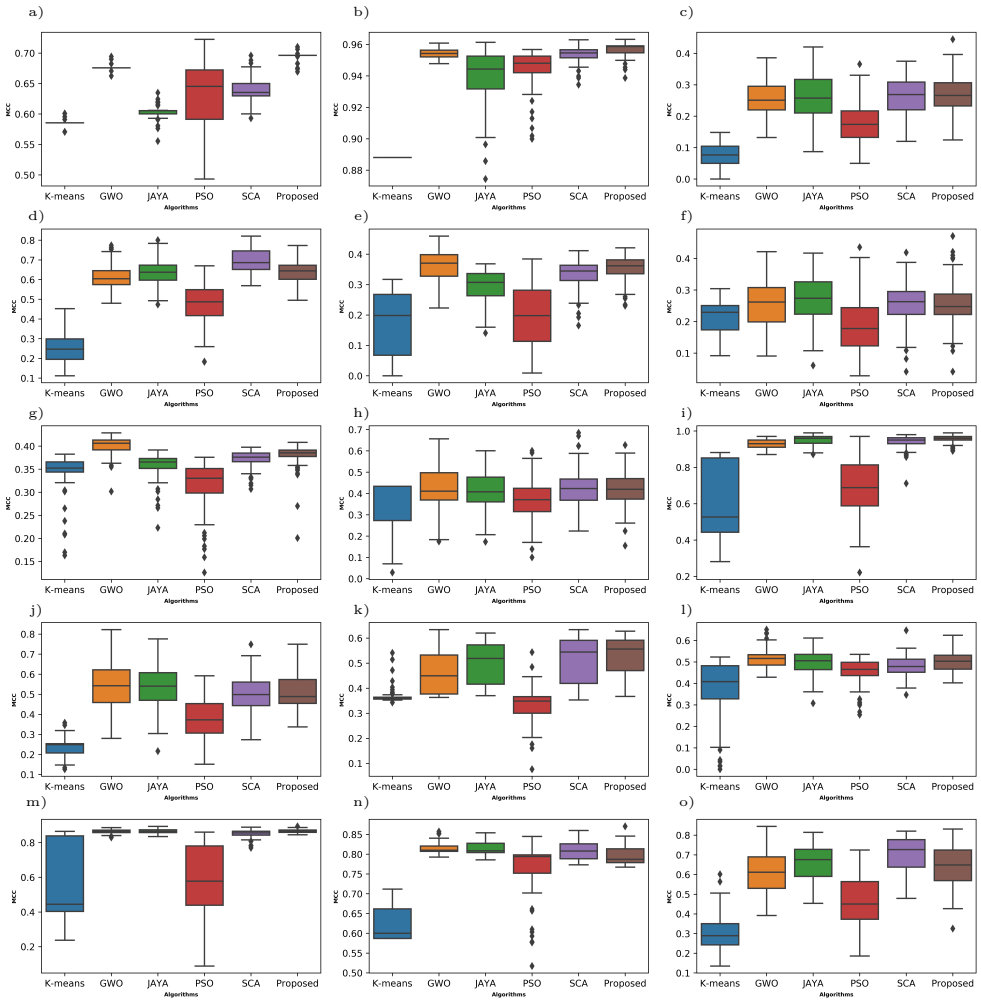

Figure 13. Matthew's correlation coefficient (MCC) metrics among GWO, PSO, k-means, JAYA, SCA, and our proposed algorithm for 15 data sets: a) Appendicitis; b) Breast Cancer; c) Bupa; d) Ecoli; e) Haberman's Survival; f) Hepatitis; g) Indian Liver Patient; h) Ionosphere; i) Iris; j) Lung Cancer; k) Mammographic Mass; l) Mushroom; m) Seeds; n) WDBC; o) Zoo

6. Conclusion

In this paper, we have proposed a meta-heuristic-based hybrid clustering algorithm using the GWO and JAYA algorithms. The proposed algorithm enjoys the explorative and exploitative skills of both algorithms in order to maintain a trade-off between them. To check the superiority of our proposed algorithm, we conducted a non-parametric test with 100 independent simulations and 1000 iterations each for all of

the tested algorithms and for each data set separately. Additionally, the Friedman and Nemenyi hypothesis mean rank test showed that our proposed algorithm achieved the highest mean rank results across 23 mathematical benchmark functions. Additionally, the null hypothesis with no significance difference in mean was rejected with a critical distance of 1.3. With respect to all of the performance metrics, it could also be observed that the GWO algorithm achieved the second-highest mean rank as compared to the comparative algorithms while evaluating performance. However, this algorithm had a statistically equivalent mean rank to SCA in only the specificity performance metric. To measure the exact significant differences among the algorithms and to evaluate performance, we conducted Duncan's multiple range test for 100 independent simulations with 1000 iterations each and on each data set separately. The obtained statistical results signify that the proposed algorithm achieved statistically superior performance in the accuracy performance metric for the Appendicitis, Breast Cancer, Bupa, Haberman's Survival, Hepatitis, Ionosphere, Iris, Lung Cancer, Mammographic Mass, Mushroom, and Seeds data sets among the remaining data sets except for the three data sets i.e. the Indian Liver Patient, WDBC, and Zoo. With respect to all of the performance metrics, our proposed algorithm achieved statistically significant results for the Appendicitis, Breast Cancer, Bupa, Haberman's Survival, Hepatitis, Iris, and Seeds data sets. By considering a minimum of any three performance metrics at a time among the four performance metrics, our proposed algorithm showed statistically superior performance in 11 data sets: Appendicitis, Breast Cancer, Bupa, Haberman's Survival, Hepatitis, Ionosphere, Iris, Lung Cancer, Mammographic Mass, Mushroom, and Seeds. For the remaining four data sets (i.e., the Ecoli, Indian Liver Patients, WDBC, and Zoo data sets), our proposed algorithm achieved the second-highest mean rank across all of the performance metrics. The statistical results signified the robustness and reliability of our proposed algorithm for clustering problems; hence, a better trade-off between exploration and exploitation was achieved. Therefore, a significant improvement in performance was achieved with the hybridization of the GWO and JAYA algorithms. Considering the above conclusive remark, we suggest some future research direction for our proposed algorithm that can be summarized as follows:

- can be adapted to solve large-scale global optimization problems;
- introducing this algorithm for image segmentation;
- introducing this algorithm for artificial neural network weight balancing and hyperparameter selection of deep neural network models;
- use of this algorithm for stabilizing training of generative adversarial network.

Acknowledgements

We would like to thank Prof. Dr. C.R. Tripathy, Vice-Chancellor, Biju Patnaik University of Technology for his valuable guidance and suggestions for successfully completing this research paper.

References

[1] Abdel-Basset M., Manogaran G., El-Shahat D., Mirjalili S.: A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem, *Future Generation Computer Systems*, vol. 85, pp. 129–145, 2018.

[2] Aljarah I., Mafarja M., Heidari A.A., Faris H., Mirjalili S.: Clustering analysis using a novel locality-informed grey wolf-inspired clustering approach, *Knowledge and Information Systems*, 2020. doi: 10.1007/s10115-019-01358-x.

[3] Aljarah I., Mafarja M., Heidari A.A., Faris H., Mirjalili S.: Multi-verse Optimizer: Theory, Literature Review, and Application in Data Clustering. In: S. Mirjalili, J. Song Dong, A. Lewis (eds.), *Nature-Inspired Optimizers*, pp. 123–141, Springer Cham, 2020.

[4] Aydilek I.B.: A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems, *Applied Soft Computing*, vol. 66, pp. 232–249, 2018.

[5] Bekkar M., Djemaa H.K., Alitouche T.A.: Evaluation Measures for Models Assessment over Imbalanced Data Sets, *Journal of Information Engineering and Applications*, vol. 3(10), pp. 27–38, 2013.

[6] Bolaños-Martinez D., Bermudez-Edo M., Garrido J.L.: Clustering Study of Vehicle Behaviors Using License Plate Recognition. In: *International Conference on Ubiquitous Computing and Ambient Intelligence*, pp. 784–795, Springer, 2023.

[7] Cheng M.Y., Prayogo D.: Symbiotic organisms search: a new metaheuristic optimization algorithm, *Computers & Structures*, vol. 139, pp. 98–112, 2014.

[8] Chicco D., Jurman G.: The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation, *BMC Genomics*, vol. 21(1), pp. 1–13, 2020.

[9] Dhiman G., Kumar V.: Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications, *Advances in Engineering Software*, vol. 114, pp. 48–70, 2017.

[10] Dorigo M., Di Caro G.: Ant colony optimization: a new meta-heuristic. In: *Proceedings of the 1999 Congress on Evolutionary Computation – CEC99 (Cat. No. 99TH8406)*, vol. 2, pp. 1470–1477, IEEE, 1999.

[11] El-Ashmawi W.H., Ali A.F., Slowik A.: An improved jaya algorithm with a modified swap operator for solving team formation problem, *Soft Computing*, vol. 24, pp. 16627–16641, 2020.

[12] Emary E., Zawbaa H.M., Grosan C., Hassenian A.E.: Feature subset selection approach by gray-wolf optimization. In: *Afro-European Conference for Industrial Advancement*, pp. 1–13, Springer, 2015.

[13] Eskandar H., Sadollah A., Bahreininejad A., Hamdi M.: Water cycle algorithm – A novel metaheuristic optimization method for solving constrained engineering optimization problems, *Computers & Structures*, vol. 110, pp. 151–166, 2012.

[14] Gao Z.M., Zhao J.: An improved grey wolf optimization algorithm with variable weights, *Computational Intelligence and Neuroscience*, vol. 2019, 2019.

[15] Gholami K., Olfat H., Gholami J.: An intelligent hybrid JAYA and crow search algorithms for optimizing constrained and unconstrained problems, *Soft Computing*, vol. 25(22), pp. 14393–14411, 2021.

[16] Gunduz M., Aslan M.: DJAYA: A discrete Jaya algorithm for solving traveling salesman problem, *Applied Soft Computing*, vol. 105, p. 107275, 2021.

[17] Guo C., Tang H., Niu B., Lee C.B.P.: A survey of bacterial foraging optimization, *Neurocomputing*, vol. 452, pp. 728–746, 2021.

[18] Hatamlou A., Abdullah S., Hatamlou M.: Data clustering using big bang–big crunch algorithm. In: *International Conference on Innovative Computing Technology*, pp. 383–388, Springer, 2011.

[19] Hatamlou A., Abdullah S., Nezamabadi-pour H.: Application of Gravitational Search Algorithm on Data Clustering. In: *International Conference on Rough Sets and Knowledge Technology*, pp. 337–346, Springer, 2011.

[20] Hatamlou A., Abdullah S., Nezamabadi-pour H.: A combined approach for clustering based on K-means and gravitational search algorithms, *Swarm and Evolutionary Computation*, vol. 6, pp. 47–52, 2012.

[21] Holland J.H.: *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, MIT Press, 1992.

[22] Hou Y., Gao H., Wang Z., Du C.: Improved Grey Wolf Optimization Algorithm and Application, *Sensors*, vol. 22(10), 3810, 2022.

[23] Hruschka E.R., Campello R.J.G.B., Freitas A.A., de Carvalho A.C.P.L.F.: A Survey of Evolutionary Algorithms for Clustering, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 39(2), pp. 133–155, 2009.

[24] Ikotun A.M., Ezugwu A.E., Abualigah L., Abuhaija B., Heming J.: K-means Clustering Algorithms: A Comprehensive Review, Variants Analysis, and Advances in the Era of Big Data, *Information Sciences*, vol. 622, pp. 178–210, 2022.

[25] Jafar O.A.M., Sivakumar R.: Ant-based clustering algorithms: A brief survey, *International Journal of Computer Theory and Engineering*, vol. 2(5), pp. 787–796, 2010.

[26] Kamboj V.K., Bath S.K., Dhillon J.S.: Solution of non-convex economic load dispatch problem using Grey Wolf Optimizer, *Neural Computing and Applications*, vol. 27(5), pp. 1301–1316, 2016.

[27] Kar M.K., Kumar S., Singh A.K., Panigrahi S.: A modified sine cosine algorithm with ensemble search agent updating schemes for small signal stability analysis, *International Transactions on Electrical Energy Systems*, vol. 31(11), e13058, 2021.

[28] Kennedy J., Eberhart R.: Particle swarm optimization. In: *Proceedings of ICNN'95 – International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, 1995.

[29] Kirkpatrick S., Gelatt C.D., Vecchi M.P.: Optimization by simulated annealing, *Science*, vol. 220(4598), pp. 671–680, 1983.

[30] Lam A.Y.S., Li V.O.K.: Chemical-reaction-inspired metaheuristic for optimization, *IEEE Transactions on Evolutionary Computation*, vol. 14(3), pp. 381–399, 2009.

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

[32] Mirjalili S.: SCA: a sine cosine algorithm for solving optimization problems, *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.

[33] Mirjalili S., Lewis A.: The whale optimization algorithm, *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

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

[35] Mittal N., Singh U., Sohi B.S.: Modified Grey Wolf Optimizer for Global Engineering Optimization, *Applied Computational Intelligence and Soft Computing*, vol. 2016, 2016.

[36] Mostafa A., Fouad A., Houseni M., Allam N., Hassanien A.E., Hefny H., Aslanishvili I.: A Hybrid Grey Wolf Based Segmentation with Statistical Image for CT Liver Images. In: *International Conference on Advanced Intelligent Systems and Informatics*, pp. 846–855, Springer, 2016.

[37] Nadimi-Shahraki M.H., Taghian S., Mirjalili S.: An improved grey wolf optimizer for solving engineering problems, *Expert Systems with Applications*, vol. 166, 113917, 2021.

[38] Nanda S.J., Panda G.: A survey on nature inspired metaheuristic algorithms for partitional clustering, *Swarm and Evolutionary Computation*, vol. 16, pp. 1–18, 2014.

[39] Niknam T., Amiri B.: An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis, *Applied Soft Computing*, vol. 10(1), pp. 183–197, 2010.

[40] Niknam T., Firouzi B.B., Nayeripour M.: An efficient hybrid evolutionary algorithm for cluster analysis. In: *World Applied Sciences Journal*, vol. 4, pp. 300–307, Citeseer, 2008.

[41] Panigrahi S., Behera H.S.: Nonlinear time series forecasting using a novel self-adaptive TLBO-MFLANN model, *International Journal of Computational Intelligence Studies*, vol. 8(1-2), pp. 4–26, 2019.

[42] Rahman M.A., Islam M.Z.: A hybrid clustering technique combining a novel genetic algorithm with K-Means, *Knowledge-Based Systems*, vol. 71, pp. 345–365, 2014.

[43] Rao R.V.: Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems, *International Journal of Industrial Engineering Computations*, vol. 7(1), pp. 19–34, 2016.

[44] Rao R.V., Savsani V.J., Vakharia D.P.: Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, *Computer-Aided Design*, vol. 43(3), pp. 303–315, 2011.

[45] Shelokar P.S., Jayaraman V.K., Kulkarni B.D.: An ant colony approach for clustering, *Analytica Chimica Acta*, vol. 509(2), pp. 187–195, 2004.

[46] Shial G., Sahoo S., Panigrahi S.: Community Detection and Disease identification using Meta-heuristic based Clustering methods. In: *2022 IEEE India Council International Subsections Conference (INDISCON)*, pp. 1–6, IEEE, 2022.

[47] Shial G., Sahoo S., Panigrahi S.: Identification and Analysis of Breast Cancer Disease using Swarm and Evolutionary Algorithm. In: *2022 IEEE Region 10 Symposium (TENSYMP)*, pp. 1–6, IEEE, 2022.

[48] Shial G., Sahoo S., Panigrahi S.: An Enhanced GWO Algorithm with Improved Explorative Search Capability for Global Optimization and Data Clustering, *Applied Artificial Intelligence*, 2023. doi: 10.1080/08839514.2023.2166232.

[49] Shial G., Tripathy C., Panigrahi S., Sahoo S.: An Improved GWO Algorithm for Data Clustering. In: S.K. Panda, R.R. Rout, R.C. Sadam, B.V.S. Rayanoothala, K.C. Li, R. Buyya (eds.), *Computing, Communication and Learning*, pp. 79–90, Springer Cham, 2022.

[50] Storn R., Price K.: Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, vol. 11(4), pp. 341–359, 1997.

[51] Thilagavathy R., Sabitha R.: Using cloud effectively in concept based text mining using grey wolf self organizing feature map, *Cluster Computing*, vol. 22(5), pp. 10697–10707, 2019.

[52] Thirumoorthy K., Muneeswaran K.: A hybrid approach for text document clustering using Jaya optimization algorithm, *Expert Systems with Applications*, vol. 178, 115040, 2021.

[53] Wang H., Geng Q., Qiao Z.: Parameter tuning of particle swarm optimization by using Taguchi method and its application to motor design. In: *2014 4th IEEE International Conference on Information Science and Technology*, pp. 722–726, IEEE, 2014.

[54] Wu C., Wang Z.: A modified fuzzy dual-local information c-mean clustering algorithm using quadratic surface as prototype for image segmentation, *Expert Systems with Applications*, vol. 201, 117019, 2022.

[55] Ying L., Yu L., Wei-Neng C., Jun Z.: A hybrid differential evolution algorithm for mixed-variable optimization problems, *Information Sciences*, vol. 466, pp. 170–188, 2018.

[56] Yu K., Qu B., Yue C., Ge S., Chen X., Liang J.: A performance-guided JAYA algorithm for parameters identification of photovoltaic cell and module, *Applied Energy*, vol. 237, pp. 241–257, 2019.

[57] Yu X., Xu W., Li C.: Opposition-based learning grey wolf optimizer for global optimization, *Knowledge-Based Systems*, vol. 226, 107139, 2021.

[58] Zhang Y., Chi A., Mirjalili S.: Enhanced Jaya algorithm: A simple but efficient optimization method for constrained engineering design problems, *Knowledge-Based Systems*, vol. 233, 107555, 2021.

Affiliations

Gyanaranjan Shial
Sambalpur University, Department of Computer Science & Application, Sambalpur University Institute of Information Technology, Burla, Odisha, India, 768019
gyanaranjan.vssut@gmail.com

Sabita Sahoo
Sambalpur University, Department of Mathematics, Burla, Odisha, India, 768019,
sabitamath@suniv.ac.in

Sibarama Panigrahi
Sambalpur University, Department of Computer Science & Application, Sambalpur University Institute of Information Technology, Burla, Odisha, India, 768019,
panigrahi.sibarama@gmail.com

Received: 08.08.2022
Revised: 23.01.2023
Accepted: 23.01.2023