IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 27, 2019, accepted December 14, 2019, date of publication December 19, 2019,
date of current version December 31, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2960925

Automatic Data Clustering Using Hybrid Firefly Particle Swarm Optimization Algorithm

MOYINOLUWA B. AGBAJE, ABSALOM E. EZUGWU, (Member, IEEE), AND ROSANNE ELS
School of Computer Science, University of KwaZulu-Natal, Pietermaritzburg Campus, Pietermaritzburg 3201, South Africa

Corresponding author: Absalom E. Ezugwu (ezugwua@ukzn.ac.za)

ABSTRACT The firefly algorithm is a nature-inspired metaheuristic optimization algorithm that has become an important tool for solving most of the toughest optimization problems in almost all areas of global optimization and engineering practices. However, as with other metaheuristic algorithms, the performance of the firefly algorithm depends on adequate parameter tuning. In addition, its diversification as a global metaheuristic can lead to reduced speed, as well as an associated decrease in the rate of convergence when applied to solve problems with large number of variables such as data clustering problems. Clustering is an unsupervised data analysis technique used for identifying homogeneous groups of objects based on the values of their attributes. To mitigate the aforementioned drawbacks, an improved firefly algorithm is hybridized with the well-known particle swarm optimization algorithm to solve automatic data clustering problems. To investigate the performance of the proposed hybrid algorithm, it is compared with four popular metaheuristic methods from literature using twelve standard datasets from the UCI Machine Learning Repository and the two moons dataset. The extensive computational experiments and results analysis carried out shows that the proposed algorithm not only achieves superior performance over the standard firefly and particle swarm optimization algorithms, but also exhibits high level of stability and can be efficiently utilized to solve other clustering problems with high dimensionality.

INDEX TERMS Automatic clustering, firefly algorithm, particle swarm optimization, hybrid metaheuristic, compact-separated validity index, Davies-Bouldin validity index.

I. INTRODUCTION

Data clustering is an unsupervised learning task that involves the classification or grouping of data objects in a distinctive manner such that the items in one group or cluster are well defined and different from objects in other clusters [39]. Simply put, based on their natural intrinsic characteristics, data objects within a cluster are more similar to each other than to those in other clusters, and so are placed together with a low intra-cluster distance. The objects in different clusters have a high inter-cluster distance [1]. The better the similarity and so the smaller the intra-cluster distance, the more compact and robust the cluster. Clustering analysis methods have been widely applied in fields such as mathematical programming, social network analysis, customer segmentation, market research, image segmentation, biological data analysis, data summarization [1]–[3], image retrieval [4], machine learning [5], [6], data mining [7]–[9], and data analysis [10], [11].

Clustering can be divided into two main categories; hierarchical and partitional [12]. On the one hand, hierarchical clustering algorithms are iterative-based clustering procedures that generate outputs that are similar to a hierarchical tree or dendrogram, which show a sequence of clustering, with each of the clusters belonging to a partition of the data objects [13], [54]. On the other hand, the partitional clustering algorithms start by decomposing the datasets into a set of disjoint clusters, based on certain optimization criteria. As discussed in Abraham *et al.* [54], such an optimization criterion involves minimizing some measure of dissimilarity among data points within each cluster, while maximizing the dissimilarity among different clusters. Since these techniques were first established, a number of different clustering methods have subsequently been proposed, namely, k-means, fuzzy c-means and simulated annealing, to solve clustering problems. These methods are highly dependent on initial solutions, thus making them easily entrapped within

The associate editor coordinating the review of this manuscript and approving it for publication was Victor S. Sheng.

local optima. Therefore, with most classical clustering algorithms having the defect of not being able to handle the task of clustering large and complex datasets, several proposals have been made with regards to the use of nature-inspired metaheuristic algorithms to solve data clustering problems. For example, in Abraham *et al.* [54], the role of several swarm intelligence algorithms in clustering different kinds of datasets was examined. The authors in [54] then proposed and described a newly nature-inspired technique for partitioning any dataset into an optimal number of groups through one run of optimization. Nanda and Panda [55] also gave an up-to-date review of all major nature inspired metaheuristic algorithms employed for partitional clustering. Furthermore, José-García and Gómez-Flores [56] presented a more comprehensive review of sixty-five clustering methods based on nature-inspired metaheuristics.

In the last few years, the firefly algorithm (FA) [14] has gained more popularity among global optimization researchers, due to its versatility, robustness and efficiency, coupled with its successful track record in dealing with diverse difficult optimization problems encountered in both science and engineering domains [59]–[61]. The FA is essentially an excellent optimization method, which has been in the forefront of other metaheuristic algorithms because of its multimodality, adaptability to problem landscapes by tuning its parameters, automatic subdivision of its populations into subgroups and ultimately its ability to solve many combinatorial, continuous and highly non-linear problems [16]–[22]. However, the FA, being a global metaheuristic algorithm, is greatly limited in performance because of its inability to strike a good balance between the intensification and diversification search processes; as a result the algorithm incurs high computational time even for simple task execution. Studies have indicated that this limitation can be resolved by hybridizing the FA with other state-of-the-art algorithms [57], [58]. The main objective of hybridization being to exploit the strengths and advantages of the individual algorithms that are been hybridized, which together reduce the deficiencies that individual algorithms might have, and greatly improving the resultant hybrid algorithm performance. Furthermore, it is noteworthy that hybridizing firefly with another algorithm should enhance the balance between exploitation and exploration which is a challenging task for most metaheuristic algorithms.

In this study, a hybrid of two algorithms, particle swarm optimization (PSO) and firefly, referred to in this paper as FAPSO, is proposed with the main aim of overcoming the aforementioned limitations of the firefly algorithm, such as premature convergence toward local minima, due to the random initialization of cluster centres. Furthermore, the study also investigates the applicability of the hybrid FAPSO to optimally determine the number of clusters for any given dataset, based on the two most commonly used clustering validity indices, namely the compact-separated (CS) validity index [23] and Davies-Bouldin (DB) validity index [24]. Most clustering methods would require a well-defined objective function and hence in this study, a Euclidean-based distance measure is selected, coupled with the two aforementioned validity indices used for computing the fitness function of each solution obtained. However, appropriate selection of the validity index is vital to the quality of the final clustering solution generated, because a bad choice may result in improper partitioning of clusters, even if the actual number of clusters seems traceable. Hence, with a good choice of validity measures, most metaheuristic algorithms are able to automatically partition datasets into an optimal number of clusters and also handle noise or outlier detection associated with the datasets. The proposed FAPSO algorithm is validated on twelve ground truths datasets from the UCI Machine Learning Repository [25], [26] and the two moons datasets [65]; the results of which are used to compare the effectiveness of the FAPSO algorithm with other clustering techniques including, automatic clustering DE (ACDE) [27], genetic clustering with an unknown number of clusters K (GCUK) [28], dynamic clustering particle swarm optimization (DCPSO) [18], classical differential evolution (DE) [49], classical FA [14] and classical PSO [15]. The experimental results show that the FAPSO outperforms the other contesting algorithms in most cases; not only in accuracy of results, but also in a statistically meaningful way.

The rest of the paper is structured as follows: Section II presents a review of literature on automatic data clustering. Section III describes the problem statement, methodology of the proposed hybrid firefly particle swarm optimization algorithm, and its implementation for automatic data clustering problems. Section IV discusses the study's experimental settings, datasets' characteristics, parameter configurations and results from simulation experiments for both the proposed FAPSO and other representative algorithms. Finally, Section V provides the concluding remarks and future research directions.

II. LITERATURE REVIEW

Many clustering algorithms, including the FA and PSO, have been proposed and implemented specifically for tackling the tasks of traditional clustering problems [18], [63]. However, only a few studies have focused on the application of these metaheuristic techniques to solve automatic data clustering problems. For example, Kuo and Zulvia [29] implemented an improved particle swarm optimisation (PSO) for automatic data clustering. Their proposed algorithm, called automatic data clustering using PSO (ACPSO), was used to address two main issues in automatic clustering. The first section was meant to address problems in determining the number of clusters, while the second section handled the representation of the cluster centroid. Further, they employed a sigmoid function to handle infeasible solutions and used the k-means algorithm to adjust the cluster centroids. Their results showed that their proposed ACPSO algorithm outperformed the other algorithms to which it was compared, in terms of accuracy and consistency. Nanda and Panda [30] proposed a multi-objective immunized PSO algorithm (MOIMPSO) to

classify actions of 3D human models. Their proposed algorithm provided a suitable Pareto optimal archive for unsupervised problems by automatically evolving cluster centres and simultaneously optimizing two different objective functions. Also, from the Pareto optimal archive, a single best solution that satisfies users' requirements was provided. Analysis of results showed that the proposed algorithm performed better than did the related algorithms in terms of accuracy of results and computational time. A kernel-based modified PSO by Abraham *et al.* [31] and a multi-elitist PSO by Das *et al.* [32] were developed and implemented for automatic clustering of complex data. Both studies are similar and the authors in [32] proposed an algorithm called Kernel_MEPSO. In this algorithm, instead of using the conventional square-measure distance approach, they adopted a kernel-induced similarity function, which enabled data that is non-separable in its original form to be clustered into homogenous groups in a high-dimensional feature space transformation. Comparison of the Kernel_MEPSO with other algorithms showed its superiority in a statistically significant way, although the Kernel_MEPSO did not outperform in all test cases.

Automatic clustering using a genetic algorithm, called AGCUK, was implemented by Liu *et al.* [33]. They designed a noising selection and division-absorption mutation to provide a balance between selection pressure and population diversity. Further, they used the DB index [24] to validate the effectiveness of their algorithm. Experimental results showed that AGCUK outperformed other algorithms in terms of providing the correct number of clusters and with lower rates of misclassification. A two-stage genetic algorithm (TGCA) for automatic clustering was implemented by He and Tan [34]. Here, the authors proposed an algorithm using the selection and mutation operators of the classical genetic algorithm (GA). In this way they could exploit the search capability of the algorithm by changing the probabilities of these operators according to the consistency of the number of clusters in the population. The proposed method, focuses, firstly, on searching for the best number of clusters and, then gradually moves to finding the globally optimal cluster centres. A comparison with other algorithms showed TGCA had a better performance in searching for the number of clusters and a higher accuracy.

Chowdhury *et al.* [35] proposed an automatic clustering approach based on invasive weed optimization (IWO). The algorithm made use of the fitness function from genetic algorithms as its validity measure, to automatically partition data appropriately without having former knowledge of its naturally occurring groups. The performance of the algorithm was compared to that of other algorithms and results showed that IWO partitioned data better than they did. However, the optimal data partitioning was derived with a minimally sized population, which also reduced computational time. According to Peng *et al.* [36], membrane computing was used to inspire an automatic clustering algorithm, in which a tissue-like membrane system with fully connected structures was designed as its computing framework, to determine automatically the most appropriate number of clusters as well as the optimal partitioning. Also, a modification of the velocity-position model was developed as evolution rules for communication. Comparison with other algorithms showed that the proposed membrane algorithm effectively determined the most appropriate number of clusters; being more robust and showing better clustering effects on high-dimensional datasets.

An improved artificial bee colony optimization with K-means algorithm (iABC) was proposed by Kuo and Zulvia [37] for automatic data clustering for customer segmentation. The proposed algorithm improved the onlooker bees in the classical ABC by directing their movements to a better location. The improved ABC provided a better initial centroid to the K-means algorithm, thus giving an improved and faster method for the onlooker bees to find solution than had could be found with the onlooker bees' original movement. Results indicated that the proposed iABC was better and steadier than other algorithms, although with a relatively high computational time. Further, when the method was applied to customer segmentation, results showed that it classified customers appropriately so organizations would be able to identify potential customers and design the most suitable marketing strategy. Likewise, Kuo *et al.* [38] used the bee colony optimization algorithm with kernel clustering (AKC-BCO) to solve the problem of automatic data clustering. Their proposed algorithm was implemented to determine the appropriate number of clusters as well as to correctly assign data points to clusters. They accomplished this by using a kernel function, which increased the clustering capability. Experimental results showed that AKC-BCO was superior to other algorithms in terms of speed of convergence, lack of local optimum trapping, and better and more stable clustering results. Further, AKC-BCO was applied to the real-life case of a prostate cancer prognosis system. Results here revealed that AKC-BCO clustered patients' test data appropriately and was able to predict survival chance for patients diagnosed with the disease.

Das *et al.* [27] proposed an improved differential evolution (DE) algorithm for automatic clustering of unlabelled datasets, called ACDE. The proposed algorithm was able to automatically find optimal number of clusters, and was applicable for high-dimensional datasets. Further, the performance of the proposed method was validated with the CS-measure [23] and DB-measure [24]. From statistical analysis of experiments, ACDE outperformed the other state-of-the-art clustering algorithms used for comparison, although, it did not win for all the instances. Lee and Chen [40] implemented an improved differential evolution (ACDE-O) algorithm using crisp number oscillation for automatic clustering. The oscillation mechanism was used to improve the search possibility of finding more possible clusters in the case where poor initial choices resulted in a number of bad clusters. In comparison with another clustering algorithm, test results showed that ACDE-O was better at finding more suitable number of clusters. Saha *et al.* [41] implemented

an algorithm based on differential evolution (DE) and fuzzy clustering for automatic cluster evolution (ADEFC), which used the Xie-Beni index to assign points to different clusters. The Xie-Beni index was also used as the validity measure for the cluster partitioning and then the clusters' centres were encoded as vectors represented by 0's and 1's. Value 1 in the masker cell determines that the encoded centre of the vector is able to participate in the fuzzy cluster, and otherwise for value 0. In comparison to other algorithms, results showed that ADEFC consistently performed better than the other clustering techniques.

In another study, a parameter adaptive harmony search algorithm (ACPAHS) was developed by Kumar *et al.* [42] for automatic data clustering. The authors used a real-coded variable length harmony vector which was able to automatically detect the number of clusters. Further, assignment of data points to different cluster centres was done using a new approach of weighted Euclidean distance, which was able to detect any type of cluster regardless of their geometric shape. In addition, the authors applied their method to automatic image segmentation, and compared it with other existing clustering techniques. Experimental results showed ACPAHS outperformed other techniques in detecting the number of clusters automatically and gave a better clustering result. Murty *et al.* [43] implemented a teaching-learning based optimization (AUTO-TLBO) method for automatic data clustering. The effectiveness of their proposed method was evaluated with the CS-measure [23], and compared with other existing algorithms. Results showed that AUTO-TLBO was superior to other techniques in terms of optimally finding the number of clusters automatically with a fast convergence time. However, their method did not win in all the test instances. A bacterial evolutionary algorithm for automatic data clustering (ACBEA) was proposed by Das *et al.* [44]. The proposed method, according to the authors, was inspired by a biological microbial evolution model, which uses the operations of bacterial mutation, by mimicking the process occurring in bacteria at genetic level and improving the chromosome parts, and of gene transfer, with the exchange of information between chromosomes in the population. The operators were then modified to handle the variable length of chromosomes that encode different clustering classifications. When the performance of the proposed method was compared to that of other clustering algorithms, it showed superior accuracy of results.

All the methods mentioned above, despite their usefulness, however, suffer from one or more drawbacks. Some could not optimally obtain the precise number of clusters while others showed inefficient run times. Further, because some of the compiled results seemed inconclusive and somewhat controversial, we were doubtful that the experiments had been performed thoroughly. Also, reduction in the search ability of cluster centres occurred over high-dimensional dataset, i.e. there was non-scalability on high-dimensional data. The field of automatic data clustering is still a shallow area that has not gained much attention in the literature. Although, literature has shown the capabilities of such clustering algorithms as the genetic algorithm, particle swarm optimization, differential evolution, firefly algorithm etc., in other domains as mentioned earlier, we hope to achieve even more and better clustering optimality, robustness and result efficiency, in the scope of automatic data clustering. More so, we hope to achieve a near best automatic clustering algorithm for the chosen validity measures, which also allows parameter tuning for better optimization.

III. METHODOLOGY

This section provides a description of the clustering problem and detailed discussions on the methodology of the proposed hybrid FAPSO algorithmic design concept. The section also presents an overview and motivation for the design of the individual FA and PSO algorithms. Subsequently, the clustering design method for the FAPSO algorithm design is also discussed.

A. CLUSTERING PROBLEM DESCRIPTION

In this study, a hybrid firefly algorithm with particle swarm optimization is implemented to solve the automatic data clustering problem. The automatic clustering problem described in [48] is adopted for the implementation of the proposed FAPSO, which is designed to handle similar problems. Next, we present some of the essential clustering problem descriptions that is used by the FAPSO to solve automatic clustering problem.

Let the given dataset $D = \{d_1, d_2, \ldots, d_n\}$ be classified into non-overlapping groups of clusters $P = \{p_1, p_2, \ldots, p_n\}$, such that the dimension of $d_i (i = 1, 2, \ldots, n)$ is b. In each cluster, we have a centroid (cluster centre) c_i such that $c_i = (i = 1, 2, \ldots, K)$. We assume that $C = (c_1, c_2, \ldots, c_K)$ represents the centres of $P = (p_1, p_2, \ldots, p_K)$. For a b-dimensional data vector, the following conditions must hold:

$$P_i \cap P_j = \emptyset, \quad i, j = 1, 2, \ldots, K \text{ and } i \neq j \tag{1}$$

$$P_1 \cup P_2 \cup \ldots \cup P_K = D \tag{2}$$

$$P_i \subseteq D \quad \text{and } P_i \neq 0, \; i = 1, 2, \ldots, K. \tag{3}$$

During the initialization stage of FAPSO, the N swarm (population) size is defined as $X = (x_1, x_2, \ldots, x_n)$, which is akin to data points in relation to the clustering problem being addressed. As described above, let each member x_i in the swarm be a $K \times b$-dimensional vector, $D_{n \times b}$, defined as $X_i = x_1^*, x_2^*, \ldots, x_K^* = (x_{11}, x_{12}, \ldots, x_{1b}), (x_{21}, x_{22}, \ldots, x_{2b}), \ldots, (x_{K1}, x_{K2}, \ldots, x_{Kb})$. The goal of the optimization method carried out by the proposed FAPSO over the two validity indices, namely CS and DB for automatic data clustering is to minimize the sum of the distance between datasets $d_i (i = 1, 2, \ldots, n)$ and centre $c_i (i = 1, 2, \ldots, K)$. The lower and upper bounds of the number of groups in the population is defined respectively as Var_{min} represented as $l_j^* = min\{D_1, D_2, \ldots, D_b\}$ and Var_{max} denoted as $u_j^* = max\{D_1, D_2, \ldots, D_b\}$. Generally, the lower and upper bounds for the solution space are

$l = (l_1^*, l_2^*, \ldots, l_K^*)$ and $u = (u_1^*, u_2^*, \ldots, u_K^*)$, respectively. To solve the clustering problem, the ith particle X_i is evaluated as follows Eq. (4):

$$X_i = rand(1, K \times b) .^* (u - l) + l \tag{4}$$

where $rand(1, K \times b)$ is a vector of uniformly distributed random number which returns an integer between 0 and 1. The proposed hybrid firefly particle swarm optimization algorithm follows the concept of the squared error function of the k-means objective function [64], which in our method is given here as follows:

$$f(D, C) = \sum_{i=1}^{n} min \left\{ \|d_i - c_j\|^2 \,|\, j = 1, 2, \ldots, K \right\}. \tag{5}$$

Note that in our improved FA algorithm, the decision variables are the cluster centres, while the quality of each firefly with respect to its light intensity is evaluated using two cluster validity indices, namely, CS [23] and DB [24], given below in equations (6) and (7), respectively. Moreover, since the two validity function indices used in this paper are treated as an optimization problem with minimization objective functions, the best solution would be the solution with the minimum f_{CS} or f_{DB} value.

1) COMPACT-SEPARATED (CS) INDEX

The CS index estimates the ratio of the sum of within-cluster scatter to between-cluster separation. It has been shown that the CS index offers better accuracy in handling clusters with different dimensions, densities or sizes. Although, it is computationally more expensive than the DB index in terms of execution time, it does, however, produce more good quality solutions. Furthermore, a large value of a CS index indicates poor compactness or less separation of clusters, while a smaller value means good clustering. Hence, the CS validity index is computed as a fitness function, given in equation (6).

$$f_{CS} = \frac{\sum_{i=1}^{K} \left[\frac{1}{|D_n|} \sum_{X_i \in C_i} max_{Y_j \in C_i} \{V(X_i, X_j)\} \right]}{\sum_{i=1}^{K} [min_{j \in K, j \neq i} \{V(d_i, d_j)\}]} \tag{6}$$

where $|D_n|$ represents the number of data points in cluster P, the function $V(X_i, X_j)$ is the distance between within-cluster scatter X_i and between-cluster separation X_j, and $V(d_i, d_j)$ is the distance from the data points d to their centroid.

The CS index expressed in equation (6) represents the ratio of within-cluster scatters to between cluster separation, as mentioned earlier.

2) DAVIES-BOULDIN INDEX

The DB index estimates the quality of a clustering by evaluating the intra-cluster (average distances of all data points within a cluster from the centroid) to inter-cluster (the distance between two centroids) distances. As with the CS, for the DB index, the smaller the value, the better the compactness or separation, and otherwise for a large value. The DB measure is calculated as a fitness function, which is given as follows:

$$f_{DB} = \frac{1}{K} \sum\nolimits_{K}^{1} V_i \tag{7}$$

$$V_i = \max \left\{ \frac{W_i + W_j}{H_{ij}} | 1 \leq i, j \leq K, i \neq j \right\} \tag{8}$$

where V_i is the distance of a data point to its centroid, W_i and W_j are the average for all data points within clusters of their distance between data points and their centroids, and H_{ij} represents the inter-cluster distance between two centroids.

B. THE FIREFLY ALGORITHM

The FA is a swarm intelligence metaheuristic algorithm inspired by the behaviour of fireflies in a naturally occurring environment [14]. Fireflies are rhythmic in behaviour and exhibit light flashes which are unique to each firefly. These light flashes are used to either attract mating partners or to warn any potential predator of looming danger ahead. Accordingly, in the FA, two main characteristics guide its operation; light intensity and level of attractiveness x_{ik}. The light intensity varies and is dependent on the distance between two fireflies, in this case the emitting and observing fireflies, thus firefly i is attracted to firefly j, with a high intensity, and vice versa. Light intensity increases as the distance reduces, and gets weaker as the landscape increases, and this light intensity can be formulated as the objective function to be optimized. The distance also serves as a medium of communication for fireflies. According to the inverse square law, light intensity I decreases with the square of the distance r according to Eq. (9)

$$I \propto \frac{1}{r^2}, \quad I = \frac{1}{r^2} \tag{9}$$

where I is the light intensity and r denotes the distance between two fireflies. Attractiveness of a firefly in an environment is proportional to the light intensity I defined according to Eq. (10)

$$\beta(r) = \beta_0 e^{-\gamma r^2} \tag{10}$$

where β denotes attractiveness at $r = 0$ and γ is the light absorption coefficient. The distance between firefly i and firefly j is calculated by the Euclidean distance as Eq. (11)

$$r_{ij} = \|X_i - X_j\| = \sqrt{\sum_{k=1}^{d} (x_{ik} - x_{jk})^2} \tag{11}$$

where d is the dimension of the fireflies, r_{ij} is the distance between a pair of fireflies X_i and X_j with positions x_{ik} and x_{jk}. Hence, the movement of a less bright firefly i is governed by it being attracted to another brighter firefly j with more attractiveness as determined by Eq. (12)

$$X_i = X_i + \beta_0 e^{-\gamma r_{ij}^2} (X_j - X_i) + \alpha \varepsilon_i \tag{12}$$

where ε_i is a random number from Gaussian distribution and α is a randomization parameter [17]. The pseudocode for the basic firefly algorithm is presented in [14], while the modified FA for the problem at hand is shown in the Algorithm listing 1.

Algorithm 1 Pseudocode for FA

Begin
Initialize each firefly $X_{firefly}$ with C random cluster centres;
Calculate fitness function of the initial population using the two clustering validity indices function (f_{CS}, f_{DB}), given in Eqs. (6) and (7), Calculate fitness function of the initial population, which is directly proportional to the light intensity, using the two clustering validity indices function (f_{CS}, f_{DB}), given in Eqs. (6) and (7);
Light intensity I is determined according to Eq. (9);
Define the light absorption coefficient γ;
while termination conditions not met
  for $i = 1 : n$ for all n fireflies;
    for $j = 1 : n$ for all n fireflies;
      if $I(i) < I(j)$ **then**
        Move $X_{firefly}$ (i) towards $X_{firefly}(j)$ based on Eq. (12) to refine position of fireflies (clusters centre);
      end if
      Attractiveness varies with distance according to Eq. (10);
      Evaluate new solutions and update light intensity;
    end for
  end for
  Update the positions of fireflies by ranking and find the current best solution;
end while
End

C. PARTICLE SWARM OPTIMIZATION

The PSO algorithm is a swarm and population-based meta-heuristic method which simulates the social behaviour of a swarm of organisms, such as school of fish or a flock of birds [15]. The method has over the years gained much attention due to its simplicity of use and implementation, flexibility and versatility in solving optimization problems [45]. During the search stage, organisms move randomly with different velocities and use these velocities to update their individual positions, each candidate solution is a 'particle'. While moving within the search domain, a particle tries to attain its best velocity according to its own local best (p_{best}) value and its neighbour's global best (g_{best}) value. The change in position of a particle in the search domain depends on the current position of the particle, the current velocity of the particle, the distance between p_{best} and current position, and the distance between g_{best} and current position [12], [46]. The velocity guides the movement of particles within the search space. The control parameters of PSO are inertia weight, acceleration coefficients and velocity of particle. Given an n-dimensional search domain for particles $X = (x_1, x_2, \ldots, x_n)$ with positions $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ and velocities $V_i = (v_{i1}, v_{i2}, \ldots, v_{in})$, the particles can iteratively update their individual positions and velocities based on Eqs. (13) and (14), respectively.

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{13}$$

$$V_i(t+1) = wV_i(t) + c_1 r_1 \left(p_{best_i}(t) - X_i(t)\right) + c_2 r_2 \left(g_{best}(t) - X_i(t)\right) \tag{14}$$

where t and $t+1$ denotes the successive iterations values of the algorithm. The parameter V_i is the velocity vector of the ith particle. The parameters c_1 *and* c_2 are acceleration coefficients, while r_1 *and* r_2 are random numbers in range [0,1]. The parameter w is the inertia weight, while p_{best} *and* g_{best} are the personal best and global best positions, respectively. The iterative process of Eqs. (13) and (14) continues until the termination criteria is satisfied. The standard algorithmic steps of the PSO are presented in [15], while the modified PSO algorithm for the clustering problem at hand is illustrated in Algorithm listing 2.

D. HYBRID FIREFLY PARTICLE SWARM OPTIMIZATION

The hybridization techniques described in this paper focus on exploiting the merits of both FA and PSO algorithms. There are two key issues when designing a hybrid clustering framework [47]. The first issue has to do with combining two or more different methods into a single framework, and the second problem is how to evaluate the optimal solution from the search of individual solutions. In this study, the implementation strategy is to integrate FA with PSO. This means that we start our search process by using the FA as our basic search algorithm, because of its strong exploitation ability, and then subsequently employ the PSO to get our optimal solution based on its great diversification ability. In FA, as described earlier, the movement of a firefly or fireflies in an environment depends principally on the attractiveness exhibited by the light intensity of the firefly. Hence, the initial behaviour of the firefly doesn't impede its current behaviour in the domain. Although, if the brightest firefly is found within the local search space, the population therefore becomes easily trapped, thus resulting in slow or premature convergence. Therefore, at the beginning of the search process, a fast convergence capability is required. Yang in [14], [48] highlighted the superiority and numerous global

Algorithm 2 Pseudocode for PSO

Begin
Initialize each particle X_{PSO} with C random cluster centres;
while termination criteria not met
for $i = 1 : n$ particle;
Calculate the fitness value of $X_{PSO}(i)$ using the two validity indices function: $f_{CS}, f_{DB.}$;
Assign $X_{PSO}(i)$ to the cluster that have nearest centroid to $X_{PSO}(i)$;
end for
Find the personal best and global best position of each particle;
Update the cluster centroids according to velocity updating and coordinate updating using Eqs. (13) and (14);
end for
end while
End

Algorithm 3 Pseudocode for FAPSO

Begin
Randomly initialize each $X_{firefly}$ with C random cluster centroid;
Calculate fitness value;
for $i = 1$ *to* n;
Evaluate fitness function (f_{CS}, f_{DB}) defined in Eqs. (6) and (7) and get the current best solution;
if current value of $X_{firefly}(i).Cost <= BestSol.Cost$ Update the current $X_{firefly}(i)$ as the best solution;
$BestSol = X_{firefly}(i)$;
end if
end for
while maximum iteration ($MaxIt$) limit is not reached
for $i = 1$ *to* n;
for $j = 1$ *to* n;
if $X_{firefly}(j) < X_{firefly}(i).Cost$
Move $X_{firefly}(i)$ towards $X_{firefly}(j)$ using Eqs. (10) and (11);
if $newsol.Cost <= newX_{firefly}(i).Cost$;
$newX_{firefly}(i)$ becomes the new solution;
if $newX_{firefly}(i).Cost <= bestSol.Cost$;
Update $newX_{firefly}(i)$ as the new solution;
end if
end if
end if
Randomly initialize $X_{PSO}(i) \leftarrow newX_{firefly}(i)$;
determine the personal best and global best position of each particle $X_{PSO}(i)$;
Calculate $X_{PSO}(i)$ fitness value using the function $f_{CS}, f_{DB.}$;
if $X_{PSO}(i).Cost <= newX_{firefly}(i).Cost$
$X_{p_{best}} \leftarrow X_{PSO}(i)$;
else if new $X_{PSO}(i)$ fitness value is smaller than overall best fitness value, update the new value as the global best
$X_{g_{best}} \leftarrow newX_{PSO}(i)$; Update the cluster centroids according to velocity and coordinates updating using Eqs. (13) and (14);
end if
end if
end for
end for
end while
End

advantages that FA has over PSO. For instance, FA needs no historical knowledge of local best position before making a further global search for an optimal solution. Furthermore, FA does not have the drawbacks associated with velocity initialization and instability for high velocity of particles [14], as is the case with PSO. In contrast to FA, PSO has the edge in

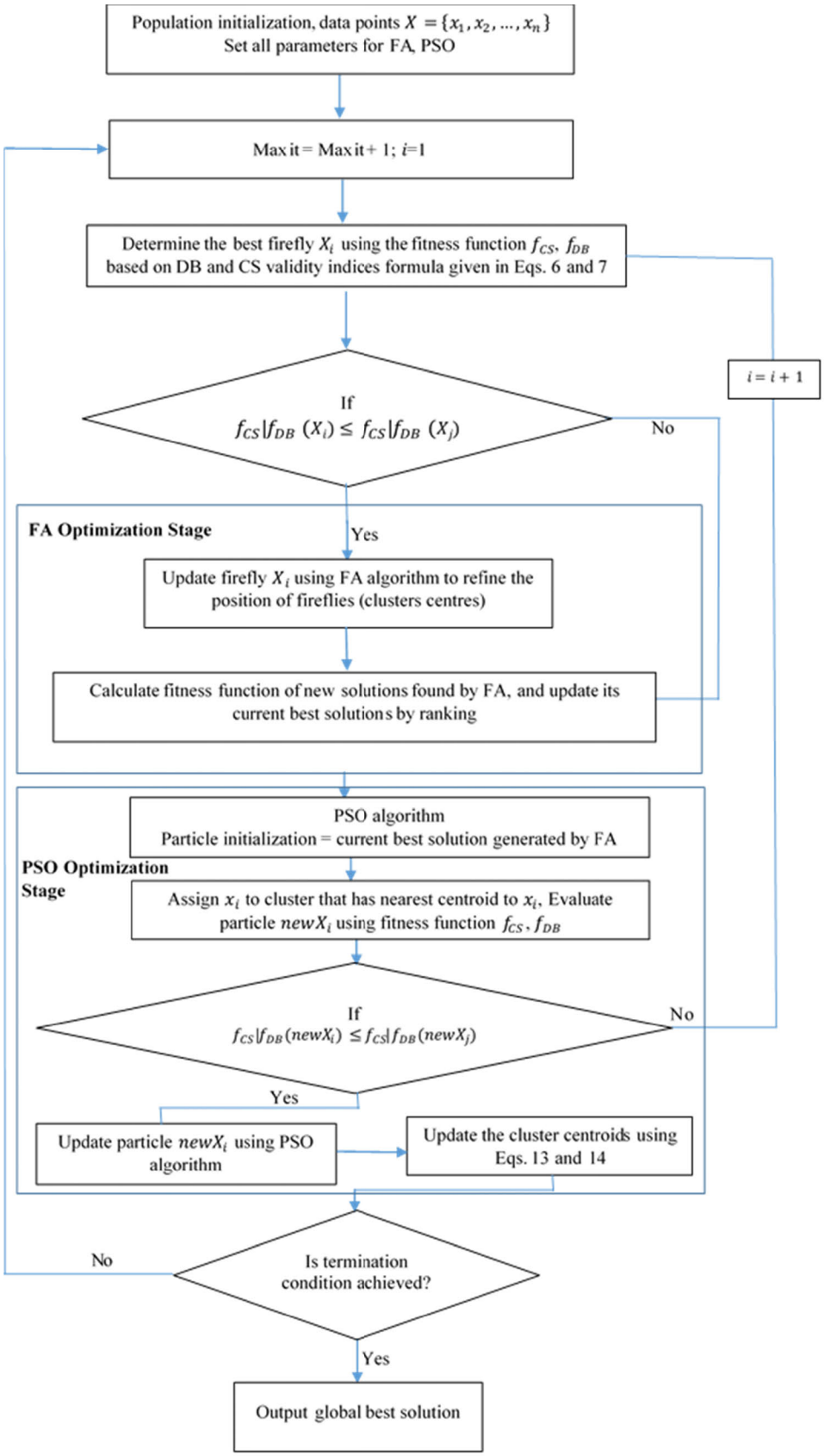


FIGURE 1. Flowchart of hybrid FAPSO algorithm for automatic clustering.

TABLE 1. Parameter setting for classical FA, classical PSO, and proposed hybrid FAPSO.

FA		PSO		FAPSO	
Parameter	Value	Parameter	Value	Value	Parameter
$nPop$	25	$nPop$	25	$nPop$	25
$MaxIt$	200	$MaxIt$	200	$MaxIt$	200
$gamma$	1	w	1	$gamma$	1
$beta0$	2	c_1	2	$beta0$	2.0
$alpha$	0.2	c_2	0.2	$alpha$	0.2
$alpha\_damp$	0.98	$wdamp$	0.99	$alpha\_damp$	0.98
				w	1
				$wdamp$	0.99
				c_1	1.5
				c_2	2.0

relation to its great explorative ability. PSO relies on the initial position of particles (p_{best}) from the population to guide it in further searches in the neighbouring regions that look promising, to find its optimal position (g_{best}). This, however, gives PSO a more explorative advantage over FA. As earlier stated in Section IIIB above, PSO is controlled mainly with the inertia weight (w), velocity (V) and acceleration coefficients (c_1 *and* c_2), thus giving the proposed algorithm a good balance between exploration and exploitation. The PSO velocity is used to generate the positions of new particles in the domain, and if the velocities of particles are not correctly obtained, it can cause them to swing back and forth around the optimal solution, thus, causing the iteration process to slow down. In contrast, fireflies move regardless of their initial positions.

In this study, we propose the integration of FA and PSO by exploiting the strengths of each individual algorithm's characteristics into a single efficient framework to solve automatic data clustering problems. Since the FA has no velocity and historical knowledge attributes for its operation, we first use FA as the local search mechanism to fine tune our local solution, and then implement the PSO for the global search task due to its ability to converge rapidly and strong explorative ability. The effectiveness and efficiency of the proposed FAPSO method is evaluated using the CS validity index [23] and DB validity index [24]. These indices also help to determine the appropriate optimal number of clusters and find the best partitioning for the detected clusters. The FAPSO starts with the random initialization process by creating the initial population of fireflies. Thereafter, the fitness values of each solution of the FA is computed using the CS and DB indices, after which the population is updated by

TABLE 2. Parameters setting for the competing algorithms.

GCUK		DCPSO		ACDE		classical DE	
Parameter	Value	Parameter	Value	Parameter	Value	Parameter	Value
Pop_size	50	Pop_size	100	Pop_size	10*dim	Pop_size	10*dim
$Cross-over$	0.8	$Inertia\ weight$	0.72	CR_{max}	1	CR	0.9
$Mutation\ probability$	0.001	c_1, c_2	1.494	CR_{min}	0.5	F	0.8
K_{max}	20	K_{max}	20	K_{max}	20	K_{max}	20
K_{min}	2	K_{min}	2	K_{min}	2	K_{min}	2

TABLE 3. Characteristics of the thirteen datasets.

Datasets	Type of dataset	Number of data points (N)	Dimension of dataset (D)	Number of clusters (k)	References
Breast	UCI dataset	699	9	2	[25,26]
Compound	Shape set	399	2	6	[25,51]
Flame	Shape set	240	2	2	[25,52]
Glass	UCI dataset	214	9	7	[25,26]
Iris	UCI dataset	150	4	3	[25,26]
Jain	Shape set	373	2	2	[25,53]
Pathbased	Shape set	300	2	3	[25,54]
Spiral	Shape set	312	2	3	[25,54]
Statlog (Heart)	UCI dataset	270	13	2	[25,26]
Thyroid	UCI dataset	215	5	2	[25,26]
Two moons	-	10,000	2	2	[66]
Wine	UCI dataset	178	13	3	[25,26]
Yeast	UCI dataset	1,484	8	10	[25,26]

TABLE 4. Numerical results for hybrid FAPSO based on the CS and DB validity indices over 40 independent runs.

	CS-index				DB-index			
Dataset	Best	Worst	Average	StDev.	Best	Worst	Average	StDev.
Breast	**0.6025**	1.0795	0.8513	0.1402	0.6519	1.0881	0.6808	0.1032
Compound	0.5032	0.7732	0.5827	0.0558	**0.4932**	0.6770	0.5024	0.0365
Flame	**0.3683**	0.5366	0.4488	0.0350	0.5969	0.7073	0.6427	0.0405
Glass	0.0608	0.0608	0.0608	0.0000	**0.3336**	0.6136	0.5140	0.1206
Iris	**0.4260**	0.5999	0.4879	0.0478	0.5689	0.6640	0.5725	0.0149
Jain	**0.4673**	0.5867	0.5232	0.0288	0.6326	0.6514	0.6442	0.0062
Pathbased	**0.4988**	0.6360	0.5524	0.0350	0.6238	0.6695	0.6416	0.0167
Spiral	**0.5567**	0.7226	0.6340	0.0367	0.7270	0.7988	0.7373	0.0182
Statlog	0.2937	0.2937	0.2937	0.0000	**0.2038**	0.2052	0.2040	0.0003
Thyroid	**0.4271**	0.4271	0.4271	0.0000	0.4813	0.5027	0.4896	0.0097
Two moons	0.6613	0.7852	0.6812	0.0244	0.6575	0.6613	0.6592	0.0010
Wine	**0.5085**	0.7940	0.6429	0.0724	0.5639	0.8850	0.7981	0.0408
Yeast	**0.3120**	0.4169	0.3512	0.0290	0.4379	0.7770	0.5888	0.0978
Average	0.4374	0.5932	0.5028	0.0389	0.5363	0.6847	0.5904	0.0389

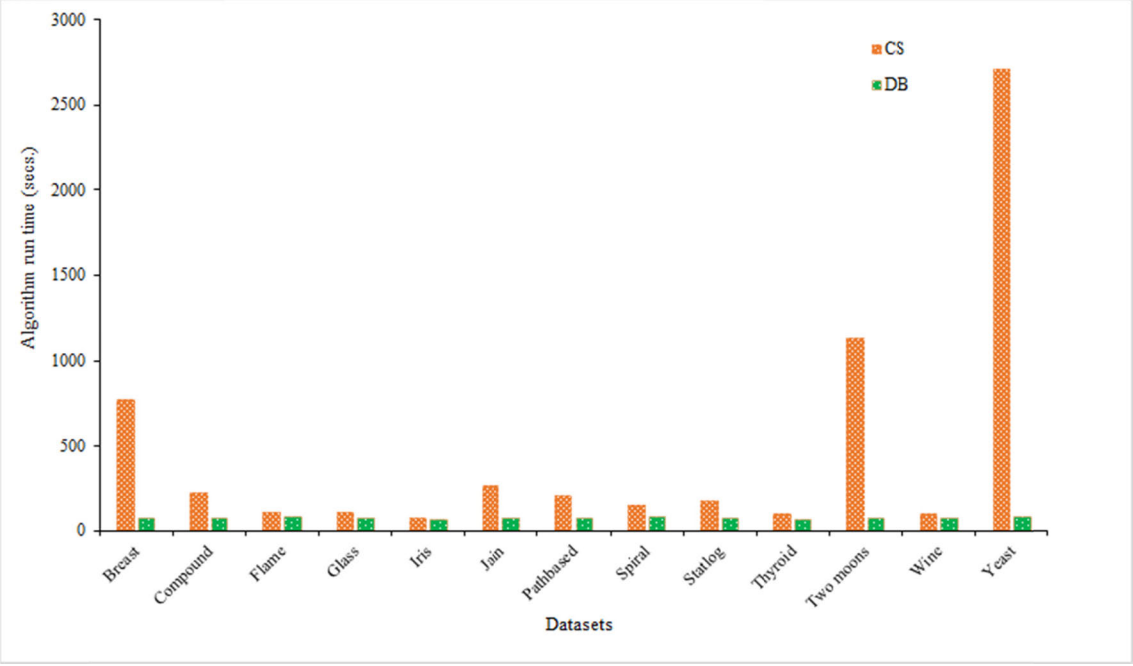


FIGURE 2. Average execution run time consumed by hybrid FAPSO on CS and DB measures for all the datasets for 40 replications.

using the operators of FA. Subsequently the same process is repeated iteratively for the PSO operators still within the first cycle of the evaluation phase of the FAPSO implementation. Note that the PSO uses the best solution generated by the FA search results as its initial search population. Iteratively, the positions and velocities of the new solutions generated by PSO are updated according to Eqs. (5) and (6) respectively. In evaluation, we compare the previous local best value and global best value with the new population, and update the particle with the best fitness values as our global best or optimal solution. Similarly, the same CS and DB indices are used by the PSO to compute the final fitness function of each solution, which the FAPSO then uses to determine the best candidate solution and make the necessary updates. Finally, the best solution is determined based on which solution has the smallest CS index value or DB index value. The two

phases of the FAPSO algorithm are repeated until the termination condition is reached. Algorithm listing 3 shows these steps of the hybrid FAPSO, while Figure 1 illustrates the design of the proposed method.

IV. EXPERIMENTATION

This section describes the experimental configuration and perimeter settings used for the performance evaluation of the proposed FAPSO algorithm for the task of automatic data clustering. The description of the benchmark datasets used for the validation of FAPSO is also given here. Finally, simulation results, discussions and comparison of FAPSO with results from literature are presented.

A. SYSTEM CONFIGURATION AND PARAMETER SETTING

The experiments were carried out using a 3.60 GHz Intel(R) Core(TM) i7-7700 processor with 16 GB memory on Windows 10 operating system. The entire algorithm was programmed in MATLAB R2018b and the statistical analysis test for the validation of the statistical significant difference of the obtained FAPSO results was carried out using IBM SPSS Version 25. Results from existing algorithms in the literature are compared with those of the FAPSO to ascertain the superiority or otherwise of the new clustering algorithm. These algorithms include ACDE [27], GCUK [28], DCPSO [18], DE [49], FA [14] and PSO [15]. The parameter settings for both FAPSO and the competing techniques are as described in Tables 1 and 2. The following descriptive statistics were used to report the computed numerical solutions obtained by the FAPSO: average cost, best cost, worst cost and standard deviation. Each of the obtained results reports the objective function of the clustering result determined by the CS and DB validity indices. The average computational time required to find the clustering solutions by the proposed algorithm is also reported.

In the experiments, FA, PSO and FAPSO had a maximum of 200 generations and a population size of 25. Twelve datasets (seven real-life datasets [25], [26] and five artificial shape sets [25], [26], [50]–[53]) and two moons dataset were used to test the effectiveness of the proposed algorithm against the compared algorithms. Although, ACDE, GCUK and DCPSO used only five real datasets from the UCI Machine Learning Repository for their experiments. The dataset design is discussed in detail and summarized in Table 3 below.

B. DATASETS DESIGN

The twelve benchmark datasets that were used for the study experimentation are taken from the UCI Machine Learning Repository of the University of California. Table 3 summarizes the datasets with respect to their type, their number of data points, dimension of the dataset, and number of clusters.

The experiment begins by executing each of the three individual algorithms (FA, PSO and hybrid FAPSO) separately with an initial randomly generated population size of 25, over 200 iterations. Furthermore, the algorithms are replicated 40 times for each of the thirteen datasets. The numerical analysis of FAPSO on CS and DB validity measures on the benchmark datasets are presented in Table 4. In Table 4, Best, Worst, Average and StDev. denote the optimal clustering solution, worst clustering solution, mean clustering solution and standard deviation. The corresponding

TABLE 5. Comparison of hybrid FAPSO with FA and PSO over 40 independent runs.

Dataset	Algorithm	CS Index		DB Index	
		Mean Sol.	StDev.	Mean Sol.	StDev.
Breast	FA	0.6423	0.0754	0.6519	0.0002
	PSO	0.9561	0.1815	0.6646	0.0801
	FAPSO	0.8513	0.1402	0.6808	0.1032
Compound	FA	0.6976	0.1159	0.4932	0.0000
	PSO	0.7558	0.0596	0.4959	0.0173
	FAPSO	0.5827	0.0558	0.5024	0.0365
Flame	FA	0.3880	0.0070	0.7229	0.0405
	PSO	0.4278	0.0947	0.7752	0.0024
	FAPSO	0.4488	0.0350	0.6427	0.0405
Glass	FA	0.0608	0.0000	0.6091	0.0000
	PSO	0.0617	0.0022	0.6023	0.0990
	FAPSO	0.0608	0.0000	0.5140	0.1206
Iris	FA	0.5582	0.0556	0.5700	0.0000
	PSO	0.6685	0.1427	0.5700	0.0000
	FAPSO	0.4879	0.0478	0.5725	0.0149
Jain	FA	0.6537	0.0059	0.6490	0.0015
	PSO	0.6581	0.0363	0.6510	0.0010
	FAPSO	0.5232	0.0288	0.6442	0.0062
Pathbased	FA	0.5801	0.1040	0.6683	0.0067
	PSO	0.7720	0.1160	0.6674	0.0087
	FAPSO	0.5524	0.0350	0.6416	0.0167
Spiral	FA	0.6816	0.0665	0.7480	0.0308
	PSO	0.6951	0.1273	0.7801	0.0295
	FAPSO	0.6340	0.0367	0.7373	0.0182
Statlog	FA	0.2945	0.0054	0.2039	0.0003
	PSO	0.2997	0.0132	0.2698	0.1333
	FAPSO	0.2937	0.0000	0.2040	0.0003
Thyroid	FA	0.5271	0.0186	0.4937	0.0097
	PSO	0.5668	0.0553	0.4981	0.0297
	FAPSO	0.4271	0.0000	0.4896	0.0097
Two moons	FA	0.7211	0.0471	0.6016	0.0027
	PSO	0.7795	0.0192	0.6592	0.0010
	FAPSO	0.6812	0.0244	0.5997	0.0000
Wine	FA	0.8426	0.0720	0.8050	0.0087
	PSO	0.8867	0.0307	0.8196	0.0369
	FAPSO	0.6429	0.0724	0.7981	0.0408
Yeast	FA	0.4770	0.0528	0.6026	0.0864
	PSO	0.5313	0.0896	0.6808	0.1257
	FAPSO	0.3512	0.0290	0.5888	0.0978

TABLE 6. Comparison of hybrid FAPSO with results from literature over 40 independent runs.

Dataset	Algorithm	CS-index		DB-index	
		Mean Sol.	StDev.	Mean Sol.	StDev.
Breast	FAPSO	0.8513	0.1402	0.6808	0.1032
	Classical FA	0.6423	0.0754	0.6519	**0.0002**
	Classical PSO	0.9561	0.1815	0.6646	0.0801
	ACDE	**0.4532**	**0.0034**	0.5203	0.0060
	DCPSO	0.4854	0.0090	0.5754	0.0730
	GCUK	0.6089	0.0160	0.6328	0.0020
	Classical DE	0.8984	0.3810	**0.5199**	0.0070
Glass	FAPSO	**0.0608**	**0.0000**	0.5140	0.1206
	Classical FA	**0.0608**	**0.0000**	0.6091	**0.0000**
	Classical PSO	0.0617	0.0022	0.6023	0.0990
	ACDE	0.3324	0.4870	1.0092	0.0083
	DCPSO	0.7642	0.0730	1.5152	0.0730
	GCUK	1.4743	0.2360	1.8371	0.0340
	Classical DE	0.7782	0.6430	1.6673	0.0040
Iris	FAPSO	0.4879	0.0478	0.5725	0.0149
	Classical FA	0.5582	0.0556	0.5700	**0.0000**
	Classical PSO	0.6685	0.1427	0.5700	**0.0000**
	ACDE	0.6643	0.0970	**0.4645**	0.0220
	DCPSO	0.7361	0.6710	0.6899	0.0080
	GCUK	0.7282	2.0030	0.7377	0.0650
	Classical DE	0.7633	**0.0390**	0.5822	0.0670
Wine	FAPSO	0.6429	0.0724	0.7981	0.0408
	Classical FA	0.8426	0.0720	0.8050	**0.0087**
	Classical PSO	0.8867	0.0307	0.8196	0.0369
	ACDE	0.9249	0.0320	3.0432	0.0210
	DCPSO	1.8721	**0.0370**	4.3432	0.2320
	GCUK	1.5842	0.3280	5.3424	0.3430
	Classical DE	1.7964	0.8020	3.3923	0.0920

average computational time needed to determine the respective optimal solution for the proposed algorithm is presented in Figure 2. Furthermore, comparisons of the results for the hybrid FABSO are compared with those from FA and PSO and with results from literature are presented in Tables 5 and 6 respectively. The Friedman test statistic was also carried out to further validate the statistical significant difference among the various clustering results obtained by the respective algorithms. The statistical results for Friedman's mean are presented in Table 7, while those for the Wilcoxon post-hoc tests are presented in Table 8. When comparing the performance of the proposed hybrid FAPSO with that for other metaheuristics algorithms, we focused on the quality of solution determined by the CS and DB validity indices as our clustering objective function, and also the computational time consumed by each of these two validity measures to find the solution on each of clustering dataset. The results that are presented in four decimal places, show the datasets on which FAPSO outperformed the other algorithms, while the results marked in boldface, indicate where the other algorithms did better than or the same as FAPSO.

C. RESULTS AND DISCUSSION

The numerical result presented in Tables 5 and 6 show clearly that the hybrid FAPSO is an efficient algorithm for solving automatic data clustering problem. The FAPSO also offers fast convergence and a high level of stability, as shown in Figures 3 and 4. This means that the other non-hybrid metaheuristic algorithms, namely, FA and PSO, demonstrated lesser performances than did the FAPSO. In Table 4, we show

TABLE 7. Ranks achieved by the Friedman mean test on FA, PSO and hybrid FAPSO.

Dataset	CS-Index			DB-Index		
	FA	PSO	FAPSO	FA	PSO	FAPSO
Breast	**1.15**	2.60	2.25	**1.49**	2.88	1.64
Compound	2.10	2.61	1.29	**1.94**	1.98	2.09
Flame	**1.23**	2.05	2.73	1.99	2.89	1.13
Glass	**1.93**	2.15	1.93	1.8	2.73	1.48
Iris	2.15	2.73	1.13	**1.99**	**1.99**	2.03
Jain	2.49	2.51	1.00	1.9	2.73	1.38
Pathbased	**1.59**	2.71	1.70	2.23	2.54	1.24
Spiral	2.31	2.14	1.55	1.73	2.68	1.60
Statlog	1.94	2.16	1.90	**2.01**	1.86	2.13
Thyroid	2.29	2.71	1.00	1.96	2.3	1.74
Two moons	1.83	2.90	1.28	1.90	3.00	1.10
Wine	2.23	2.73	1.05	1.89	2.33	1.79
Yeast	2.30	2.70	1.00	**1.71**	2.55	1.74

a comparison of the results obtained for the CS and DB validity measures over 40 independent runs. To make the comparison of the validity measures fair, the populations for FA, PSO and FAPSO were initialized using the same random swarm sizes. In Table 4, we present the results that show how FAPSO performed on both CS and DB measures for all the datasets. The values marked in boldface, show where the CS-index obtained better results than the DB-index in some of the dataset instances; namely, Breast, Flame, Iris, Jain, Pathbased, Spiral, Thyroid, Wine and Yeast datasets. Conversely, the DB index performed better than the CS-index in some instances on the following datasets, specifically the Compound, Glass and Statlog. Although, the CS-index is able to achieve its results at the expense of high computational times, as was shown in Figure 2, while the DB-index with lesser computational times, has poorer results. Furthermore, on average, we deduced that, the CS measure generally performed better than DB measure, thus indicating that the CS measure achieves a superior inter-cluster distance than does the DB measure. This clearly shows that the CS-index has a better partitioning ability and better solution than does the DB measure. For the Yeast dataset the FAPSO had its best optimal result on the CS measure, the DB produced its optimal solution on the Statlog dataset. Also, both validity measures yielded the worst results on the Breast dataset. It is to be noted that the objective function of the proposed method is a minimization case, so the least values obtained by the CS and DB indices are considered to indicate the best results. Hence, for FAPSO, the CS measure produced better results than the DB index.

As discussed earlier, although the CS measure outperformed the DB measure in terms of offering good partitioning and better quality of solution, the computational time for the CS index is relatively poor compared with that for the DB. In Figure 2, where the CS index is represented by orange-dotted bars and, the DB index by green-dotted bars, it is clearly shown that the DB has a better run time compared to the CS measure. The Yeast dataset had the highest average run time for CS index.

In Table 5, the numerical results for the standard FA, PSO and the proposed hybrid FAPSO algorithm are shown. The experiments, which was carried out for 40 replications, gives the comparison of the three aforementioned algorithms, and

TABLE 8. *p*-values produced by the Wilcoxon rank-sum test for equal medians.

Dataset	CS-Index			DB-Index		
	FA vs. PSO	FAPSO vs. FA	FAPSO vs. PSO	FA vs. PSO	FAPSO vs. FA	FAPSO vs. PSO
Breast	0	0	0.011	0	0.211	0
Compound	0.008	0	0	0.317	0.068	0.225
Flame	0	0	0.002	0	0	0
Glass	0.024	1	0.024	0.002	0	0
Iris	0	0	0	1	0.285	0.285
Jain	0.441	0	0	0	0	0
Pathbased	0	0.375	0	0.12	0	0
Spiral	0.914	0.001	0.014	0	0.313	0
Statlog	0.034	0.317	0.008	0.221	0.122	0.535
Thyroid	0	0	0	0.648	0.082	0.107
Two moons	0	0	0	0	0	0
Wine	0	0	0	0.038	0.361	0.029
Yeast	0.002	0	0	0.002	0.675	0.002

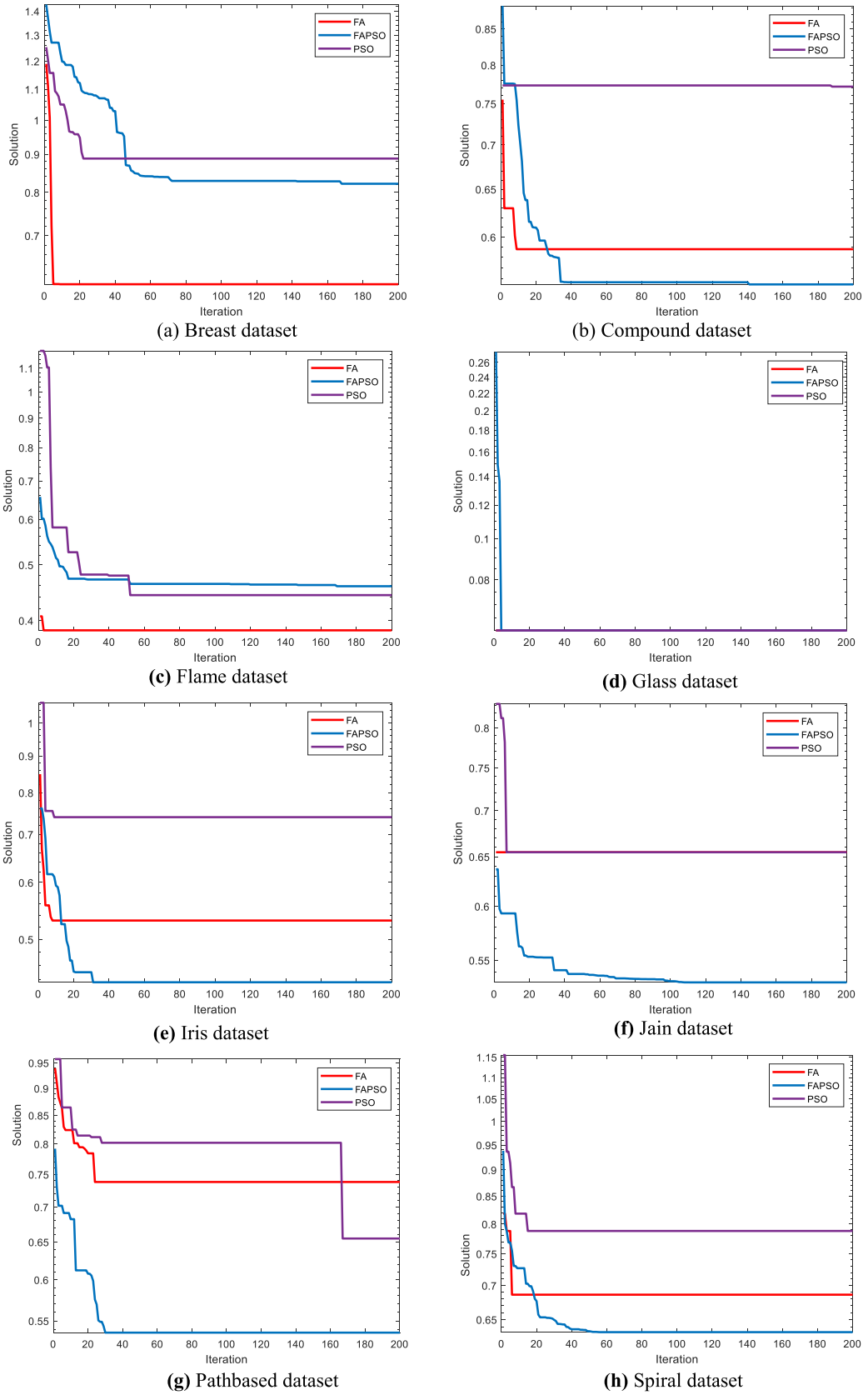
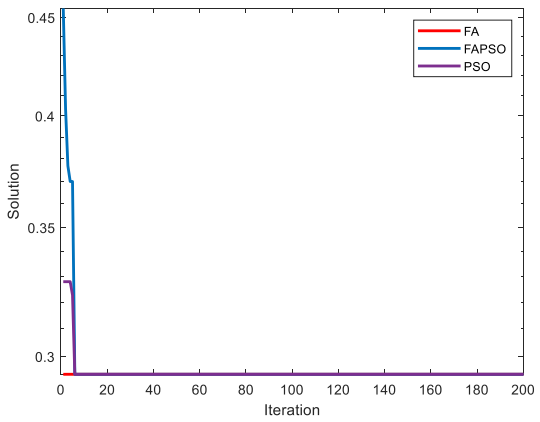
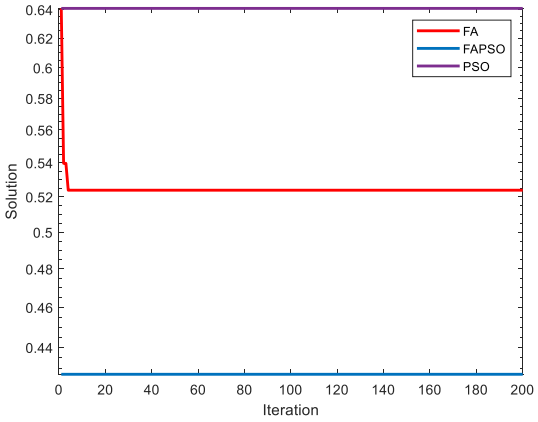


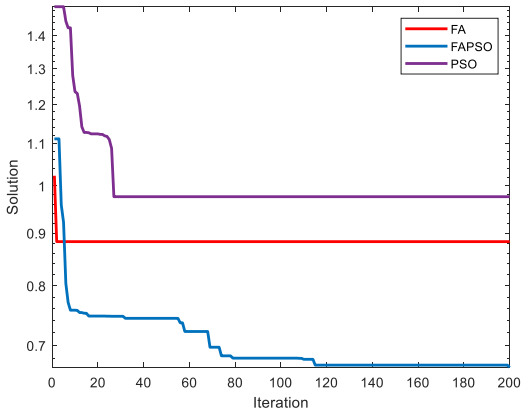
FIGURE 3. Convergence curves on CS-index of each algorithm on all the test datasets.



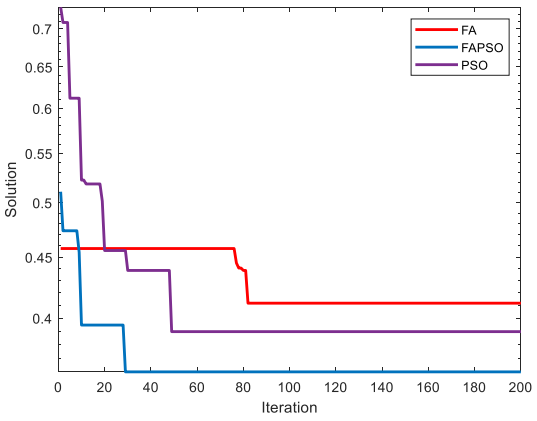
(i) Statlog (Heart) dataset



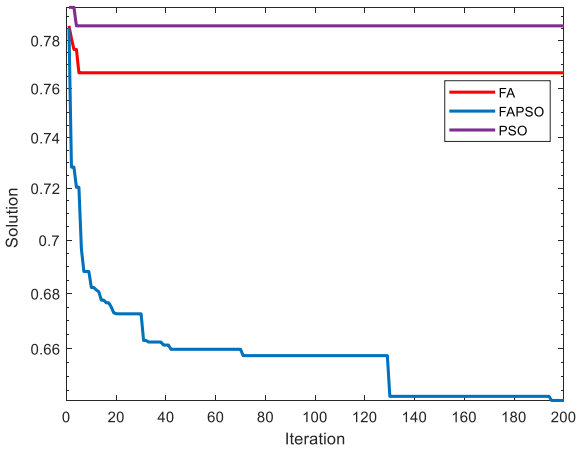
(j) Thyroid dataset



(k) Wine dataset



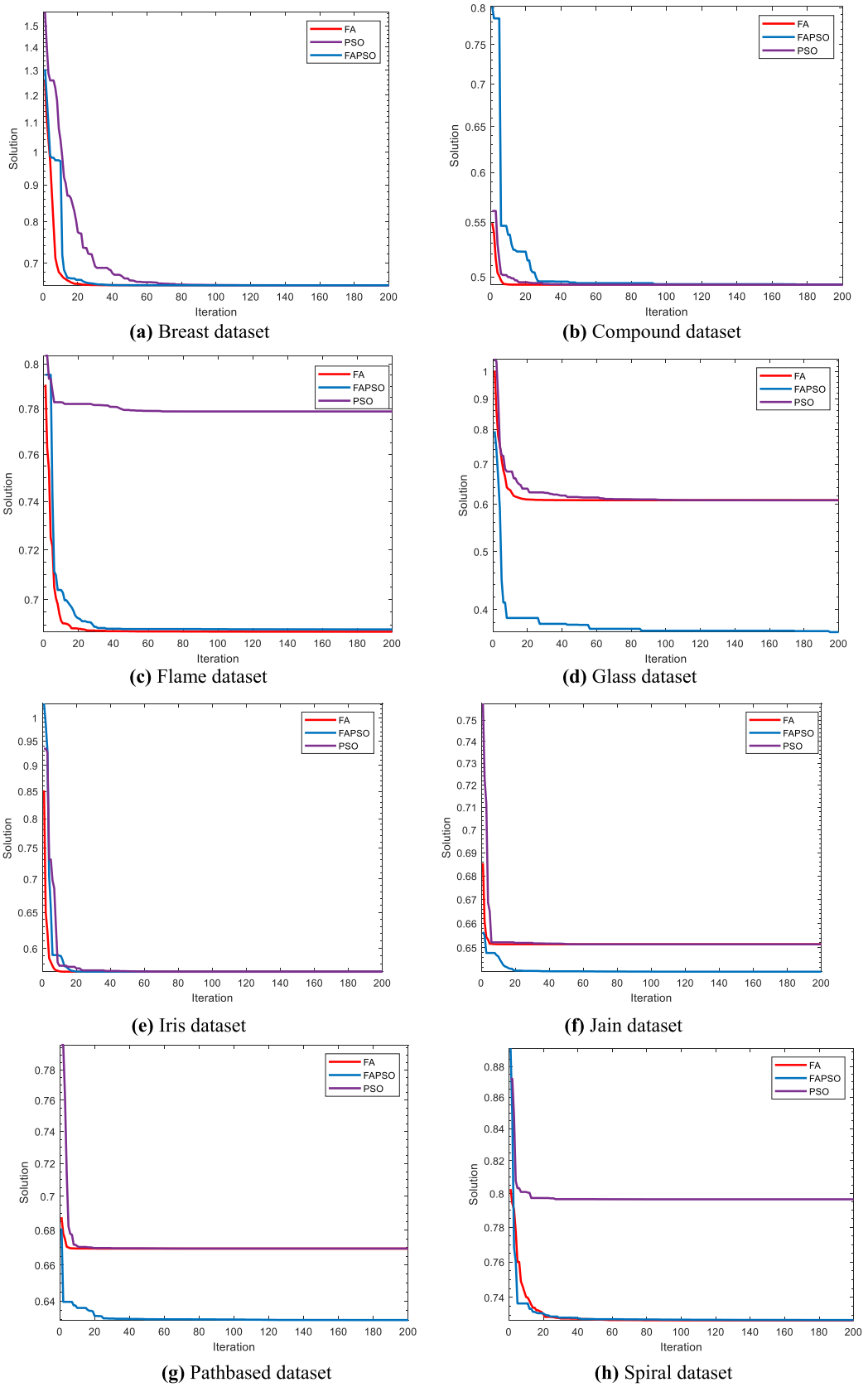
(l) Yeast dataset



(m) Two moons dataset

FIGURE 3. ***(Continued.)*** **Convergence curves on CS-index of each algorithm on all the test datasets.**

the results presented are the Average solution and Standard deviation. Although in some instances of the test datasets the proposed hybrid FAPSO was outperformed by the single FA and PSO algorithms, highlighted in bold in Table 5, the hybrid method however did show superior performance for most of the datasets. For the Breast dataset, we observe that the FA obtained very close values for both CS index and DB index; here it also outperforms our proposed FAPSO on both clustering indices. Similarly, for Compound and Flame datasets, the DB and CS had better clustering solutions than



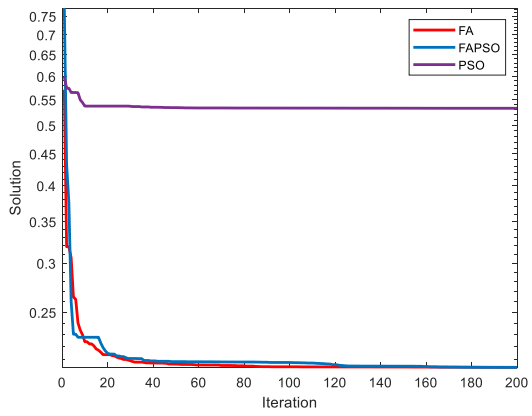
(a) Breast dataset **(b)** Compound dataset

(c) Flame dataset **(d)** Glass dataset

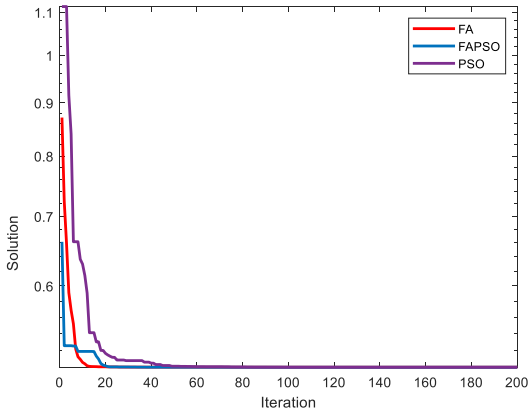
(e) Iris dataset **(f)** Jain dataset

(g) Pathbased dataset **(h)** Spiral dataset

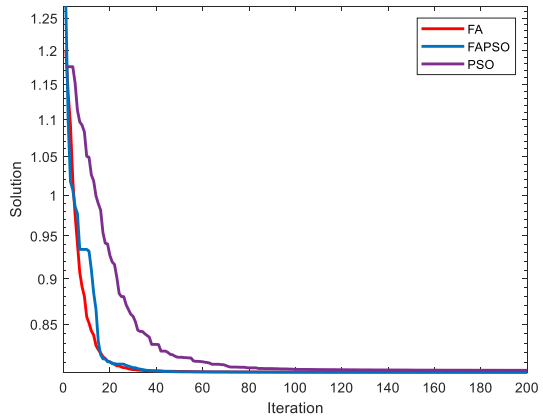
FIGURE 4. Convergence curves on DB-index of each algorithm on all the test datasets.



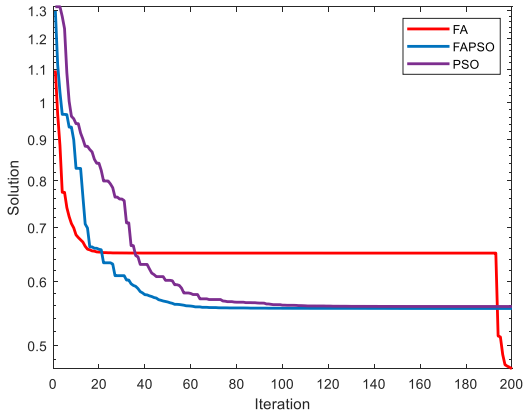
(i) Statlog (Heart) dataset



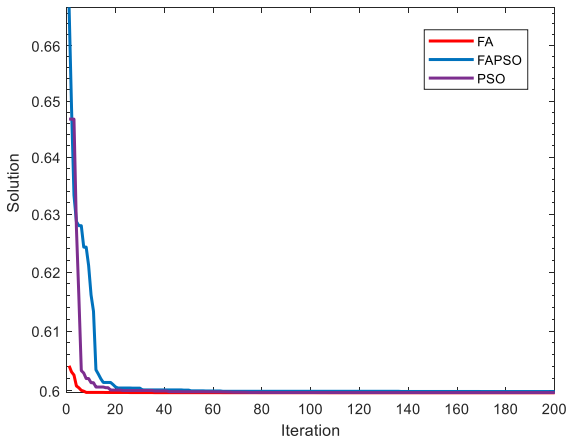
(j) Thyroid dataset



(k) Wine dataset



(l) Yeast dataset

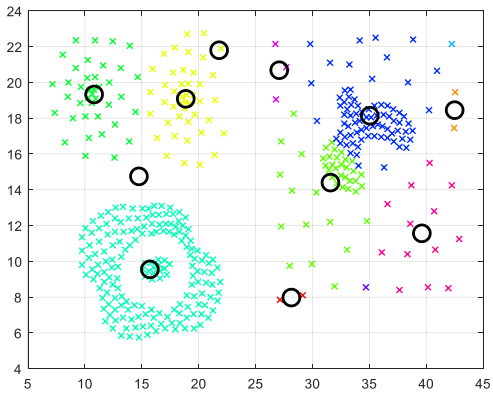


(m) Two moons dataset

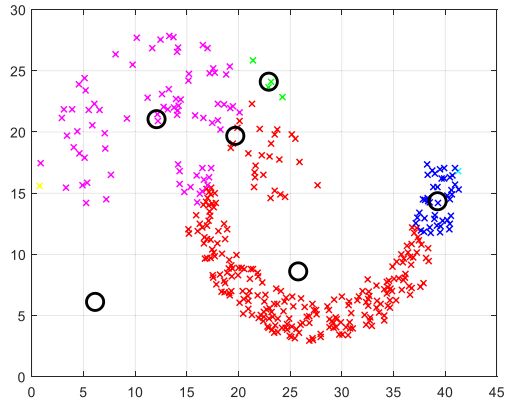
FIGURE 4. ***(Continued.)*Convergence curves on DB-index of each algorithm on all the test datasets.**

did both the PSO and FAPSO. However, the overall results clearly show that FAPSO is superior than both FA and PSO in terms of the quality of clustering solutions obtained by these three algorithms.

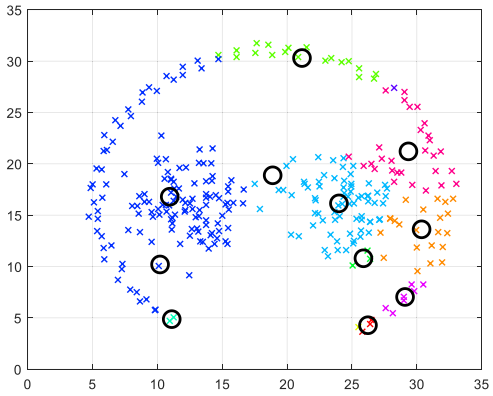
Table 6 presents a comparison between the clustering performance of the proposed FAPSO algorithm and results from literature. The detailed analysis of the results is based on the individual algorithm competitiveness when executed on four



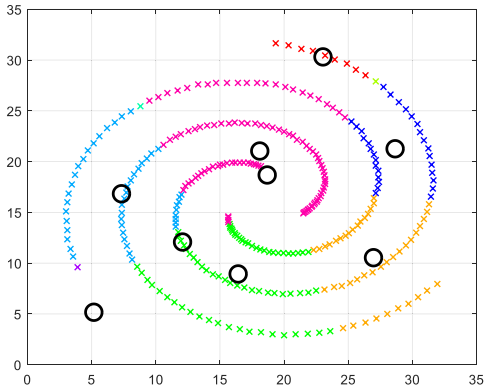
(a) Compound dataset



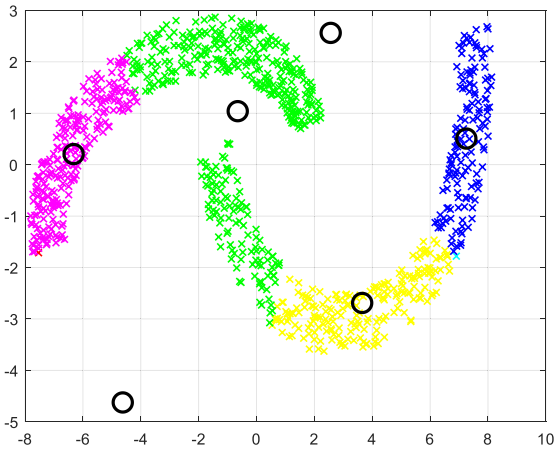
(b) Jain dataset



(c) Pathbased dataset



(d) Spiral dataset



(e) Two moons dataset

FIGURE 5. Clustering illustrations of hybrid FAPSO using CS-index for some selected datasets.

ground truth datasets, namely Breast, Glass, Iris, and Wine. The following observations are evident.

- **Breast Cancer Wisconsin (Original) dataset.** Table 6 demonstrates that ACDE outperformed other algorithms, including the proposed hybrid FAPSO. ACDE achieved the optimal Mean and StDev. values on the CS measure of all the algorithms, which is better than the Mean and StDev. values obtained by DE on the DB index. Although, DE had the best results on the DB index of all the algorithms, the StDev. value of FA is much less than that for the other algorithms. Nevertheless, it cannot be said conclusively that the results achieved from ACDE and DE on CS index and DB index are superior to our result, because, the data points used in the

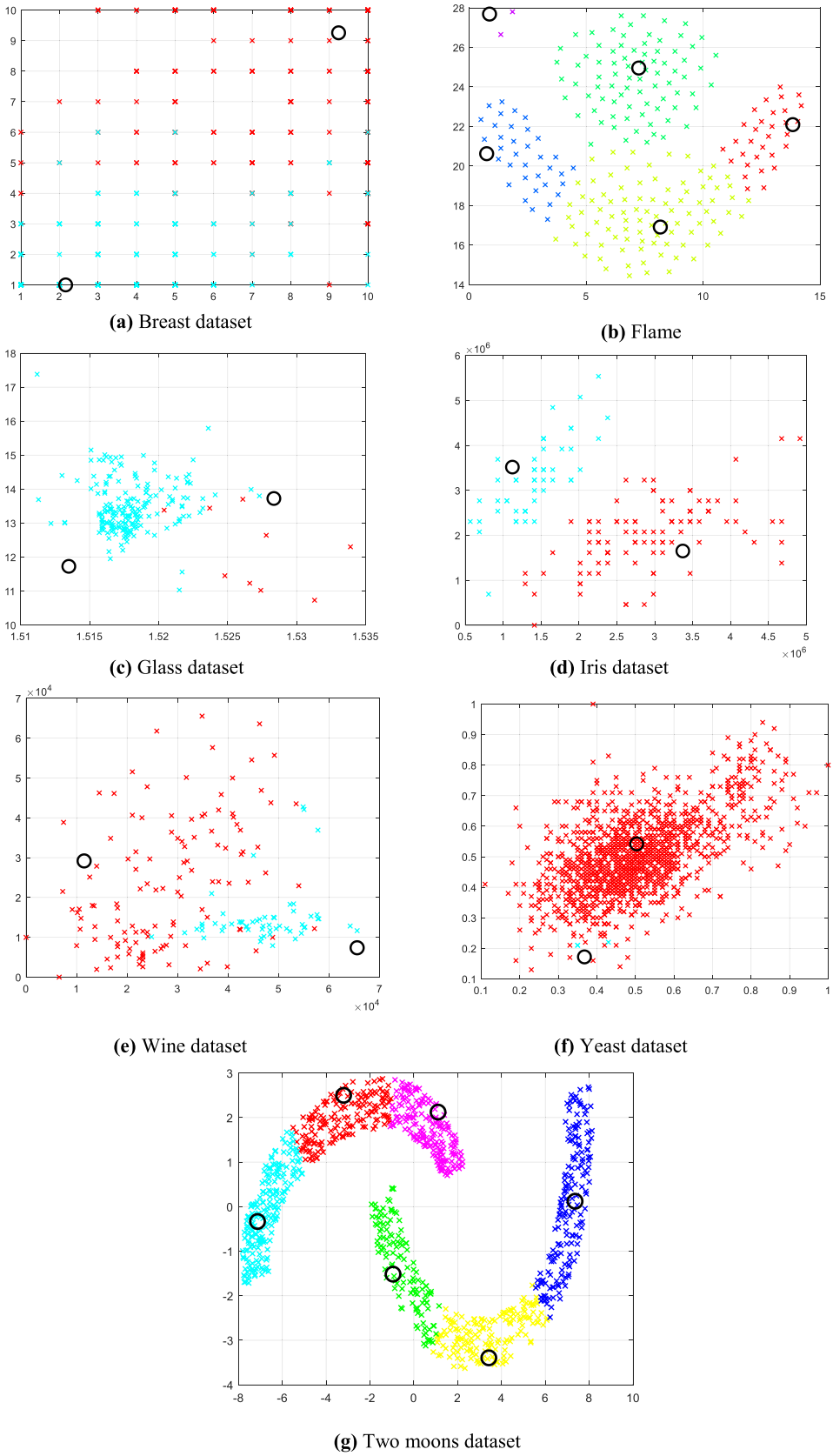


FIGURE 6. Clustering illustrations of the hybrid FAPSO using DB-index for some selected datasets.

ACDE algorithm was 683, while we used the standard original data points of 699.

- **Glass dataset.** FASPO and FA achieved identical results with the smallest values for the Mean and StDev. on the CS-index. Likewise, for DB-index, FAPSO outperformed the other algorithms in terms of its optimal Mean value. However, the variation in the StDev. for the FA is better than that of the FAPSO. The values of DB index achieved for other algorithms are less than that of FAPSO.
- **Iris dataset.** On the Iris dataset, the Mean value achieved by the FAPSO on the CS measure is better than that of the other algorithms, although with DE having a better variation on the computed StDev. results. The StDev. values are so close that differences between results for this dataset are not obvious. ACDE algorithm had a better Mean result on DB index than did the other algorithms, but also, the StDev. of both FA and PSO is considerably less than for the other methods.
- **Wine dataset.** The FAPSO algorithm again achieves the smallest Mean value on the CS index, but with a high variation of StDev. as compared to those of PSO, ACDE, GCUK and DCPSO. Likewise, on the DB index, FAPSO had the least Mean value of all algorithms, but with FA, PSO, and ACDE having the least variation of StDev.

A Friedman's statistical test was carried out to further validate the above numerical results and claims of superior performance made for the proposed clustering algorithm. The results presented in Table **7** highlight that the FAPSO is the best performing algorithm on both CS and DB clustering validity measures. Although, FA and PSO ranked better than FAPSO in some instances, this does not, however, negate the claim that that FAPSO is a strong, efficient and effective algorithm for automatic data clustering analysis. For the thirteen datasets used, on the one hand, for the CS measure the FA algorithm ranked better than PSO and FAPSO on four datasets, while FAPSO had optimal ranking on eight datasets. On the other hand, for the DB measure, FA had the best ranking on four datasets, while PSO ranked best on the Iris dataset; but FAPSO was superior in seven instances. With the Iris dataset, FA and FAPSO on DB index had the same mean rank. To further establish a proper comparison between FA, PSO and FAPSO, a post hoc test was performed to compare the control method and the other set of algorithms, and to also show statistical significance between the sets of algorithms, as presented in Table 8. Generally, we observed that the statistical significance is more evident for the CS index than the DB index, in the related samples of FAPSO versus. FA and FAPSO versus PSO. Hence, this further strengthens our claims regarding the capability of the CS validity index, with its extreme statistical significance on almost all the datasets.

The Wilcoxon rank-sum test for equal medians was applied to the earlier Friedman's statistics test results, so as to draw a more meaningful statistical conclusion. Table 8 reports the p-values obtained by the Wilcoxon statistics test for the pairwise comparison on the CS and DB measures, of three groups formed by FAPSO versus FA, FAPSO versus PSO and FA versus PSO. We could not carry out any statistical test on the existing FA and PSO algorithms because the required raw data had not been published by the original authors. The Wilcoxon test establishes a statistical significance between our pairwise groups, as we cannot base our statistical judgement on the Friedman test alone. Also, the Wilcoxon rank-sum test compares the null hypothesis that two values are samples from a continuous distribution with equal medians, against the alternative that they are not. On both the CS and DB measures, almost all the values presented in Table 8 are less than 0.05, that is the 5% significance level, which gives strong evidence against the null hypothesis, and an indication that the hybrid FAPSO results are statistically significant. Further, a nonparametric test, which is similar to ANOVA, called the Friedman mean-rank test was run. This test can be used to establish significant differences between the behaviour of two or more algorithms, and was performed and computed among FA, PSO and hybrid FAPSO. In Table 7 for the Friedman mean ranking, the values highlighted in bold show where the competing algorithms ranked better than FAPSO. Generally, on average, the statistical results reveal FAPSO has a better mean rank than both FA and PSO.

Figures 3 and 4 present graphical comparisons between FAPSO and the other two algorithms in terms of convergence rate for each of the thirteen datasets. Overall on both CS and DB measures, FAPSO clearly converges faster than FA and PSO. Whereas the curves are not smoothly represented on CS index, DB had better, that is smoother, curves. Similarly, PSO, converged fairly better than FA on the DB index, while its convergence on CS index is seemingly poor. The convergence rates for FA on both measures, emerged slightly better on DB index than for the CS index.

The individual clustering samples for some selected datasets for hybrid FAPSO on CS and DB indices are shown in Figures 5 and 6, respectively. In Figure 5a (Compound dataset), the dataset was clearly divided into six classes, with a small part of the magenta and yellow classes being mixed with the blue class. Also, there were outliers in the green class that were not properly grouped. For the Jain dataset (b), we had three distinct clusters with a few green outliers attached to the magenta class. Also in Pathbased (c) and Spiral (d) datasets, we had five and six clearly separated classes, respectively. The moon dataset (e) also generated five well compacted clusters with no outliers. Likewise, for DB index as shown in Figure 6, four of the selected datasets had perfect clustering which are well separated as presented on each graph. The exception was the Yeast dataset (f), which had a few outliers around it.

V. CONCLUSION

In this paper, a new hybrid approach has been proposed to solve automatic data clustering problems. Because cluster

analysis has been applied to many fields, many clustering algorithms have consequently been proposed. However, most of those algorithms require a predefined number of clusters; but in real world problems this information is seldom available. In this study, we have proposed and implemented an automatic clustering algorithm based on the FA and PSO algorithms. One important feature of the proposed FAPSO clustering algorithm is that it is able to determine automatically, in real time, the optimal number of clusters even for those datasets with high dimensions, where the possibility of tracking the number of clusters may be close to impossible. The simulation results have demonstrated that the hybrid FAPSO outperformed the two state-of-the-art clustering algorithms, namely FA and PSO, and other existing metaheuristic algorithms, namely, ACDE, GCUK, DCPSO and DE, in a statistically meaningful way over most of the benchmark datasets used for the experiments. Furthermore, the FAPSO has been shown to be superior in terms of its capability of obtaining better clustering solution quality, optimal number of clusters, and improved convergence speed. Overall, between the two clustering validity indices used to evaluate the performance of the proposed hybrid algorithm, the CS index proved to be a better and more efficient clustering metric for FAPSO than did the DB index, even though its run time was inferior to that of the DB index. Moreover, the CS index provided to be better than the DB index in clustering solutions in terms of cluster compactness and cohesion. For future research and possible extension of the proposed FAPSO, it will be interesting to consider the option of introducing Levy flights to hasten the movement of fireflies, thus reducing the timing of foraging. Other local search approaches can also be integrated into the FAPSO to solve complex real-world clustering problems. Finally, FAPSO having proved to be an efficient algorithm, it could be further explored in other engineering optimization fields.

CONFLICT OF INTERESTS

The authors declare that there is no conflict of interests regarding the publication of the paper.

REFERENCES

[1] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Math. Program.*, vol. 79, nos. 1–3, pp. 191–215, 1997.

[2] C. C. Aggarwal, Ed., *Data Classification: Algorithms and Applications*. Boca Raton, FL, USA: CRC Press, 2014.

[3] G. Gan, C. Ma, and J. Wu, *Data Clustering: Theory, Algorithms, and Applications*, vol. 20. Philadelphia, PA, USA: SIAM, 2007.

[4] M. Karthikeyan and P. Aruna, "Probability based document clustering and image clustering using content-based image retrieval," *Appl. Soft Comput.*, vol. 13, no. 2, pp. 959–966, 2013.

[5] Q. He, X. Jin, C. Du, F. Zhang, and Z. Shi, "Clustering in extreme learning machine feature space," *Neurocomputing*, vol. 128, no. 5, pp. 88–95, 2014.

[6] L. M. Abualigah, A. T. Khader, E. S. Hanandeh, and A. H. Gandomi, "A novel hybridization strategy for krill herd algorithm applied to clustering techniques," *Appl. Soft Comput.*, vol. 60, pp. 423–435, Nov. 2017.

[7] L. M. Abualigah and A. T. Khader, "Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering," *J. Supercomputing*, vol. 73, no. 11, pp. 4773–4795, 2017.

[8] L. M. Abualigah, A. T. Khader, M. A. Al-Betar, and O. A. Alomari, "Text feature selection with a robust weight scheme and dynamic dimension reduction to text document clustering," *Expert Syst. Appl.*, vol. 84, pp. 24–36, Oct. 2017.

[9] L. M. Abualigah, A. T. Khader, M. A. Al-Betar, and M. A. Awadallah, "A krill herd algorithm for efficient text documents clustering," in *Proc. IEEE Symp. Comput. Appl. Ind. Electron. (ISCAIE)*, May 2016, pp. 67–72.

[10] L. M. Abualigah, A. T. Khader, M. A. Al-Betar, and E. S. Hanandeh, "A new hybridization strategy for krill herd algorithm and harmony search algorithm applied to improve the data clustering," *management*, vol. 9, no. 11, pp. 1–10, 2017.

[11] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A hybrid strategy for krill herd algorithm with harmony search algorithm to improve the data clustering," *Intell. Decis. Technol.*, vol. 12, no. 1, pp. 3–14, 2018.

[12] M. Ramadas and A. Abraham, *Metaheuristics for Data Clustering and Image Segmentation*. Cham, Switzerland: Springer, 2019.

[13] D.-X. Chang, X.-D. Zhang, C.-W. Zheng, and D.-M. Zhang, "A robust dynamic niching genetic algorithm with niche migration for automatic clustering problem," *Pattern Recognit.*, vol. 43, no. 4, pp. 1346–1360, 2010.

[14] X. S. Yang, *Firefly Algorithm, Nature Inspired Metaheuristic Algorithms*. Frome, U.K.: Luniver Press, 2010

[15] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. IEEE 6th Int. Symp. Micro Mach. Hum. Sci. (MHS)*, Oct. 1995, pp. 39–43.

[16] T. Rui, S. Fong, X. S. Yang, and S. Deb, "Nature-inspired clustering algorithms for Web intelligence data," in *Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Intell. Agent Technol.*, vol. 3, Dec. 2012, pp. 147–153.

[17] I. Fister, I. Fister, Jr., X.-S. Yang, and J. Brest, "A comprehensive review of firefly algorithms," *Swarm Evol. Comput.*, vol. 13, pp. 34–46, Dec. 2013.

[18] M. Omran, A. Salman, and A. Engelbrecht, "Dynamic clustering using particle swarm optimization with application in unsupervised image classification," in *Proc. 5th World Enformatika Conf. (ICCI)*, Prague, Czech Republic, 2005, pp. 199–204.

[19] M. Omran, A. P. Engelbrecht, and A. Salman, "Particle swarm optimization method for image clustering," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 19, no. 3, pp. 297–321, 2005.

[20] L. M. Abualigah, A. T. Khader, M. A. AlBetar, and E. S. Hanandeh, "Unsupervised text feature selection technique based on particle swarm optimization algorithm for improving the text clustering," in *Proc. Int. Conf. Comput. Sci. Eng. (EAI)*, 2017, p. 169.

[21] M. G. Omran, A. Salman, and A. P. Engelbrecht, "Dynamic clustering using particle swarm optimization with application in image segmentation," *Pattern Anal. Appl.*, vol. 8, no. 4, p. 332, 2006.

[22] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A new feature selection method to improve the document clustering using particle swarm optimization algorithm," *J. Comput. Sci.*, vol. 25, pp. 456–466, Mar. 2018.

[23] C. H. Chou, M. C. Su, and E. Lai, "A new cluster validity measure and its application to image compression," *Pattern Anal. Appl.*, vol. 7, no. 2, pp. 205–220, 2004.

[24] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979.

[25] *Clustering Basic Benchmark*. Accessed: Nov. 25, 2019. [Online]. Available: http://cs.joensuu.fi/sipu/datasets/

[26] K. Bache and M. Lichman. (2013). UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA, USA. [Online]. Available: http://archive.ics.uci.edu/ml/

[27] S. Das, A. Abraham, and A. Konar, "Automatic clustering using an improved differential evolution algorithm," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 1, pp. 218–237, Jan. 2008.

[28] S. Bandyopadhyay and U. Maulik, "Genetic clustering for automatic evolution of clusters and application to image classification," *Pattern Recognit.*, vol. 35, no. 6, pp. 1197–1208, 2002.

[29] R. Kuo and F. E. Zulvia, "Automatic clustering using an improved particle swarm optimization," *J. Ind. Intell. Inf.*, vol. 1, no. 1, pp. 46–51, 2013.

[30] S. J. Nanda and G. Panda, "Automatic clustering algorithm based on multiobjective immunized PSO to classify actions of 3D human models," *Eng. Appl. Artif. Intell.*, vol. 26, nos. 5–6, pp. 1429–1441, 2013.

[31] A. Abraham, S. Das, and A. Konar, "Kernel based automatic clustering using modified particle swarm optimization algorithm," in *Proc. ACM 9th Annu. Conf. Genetic Evol. Comput.*, Jul. 2007, pp. 2–9.

[32] S. Das, A. Abraham, and A. Konar, "Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm," *Pattern Recognit. Lett.*, vol. 29, no. 5, pp. 688–699, Apr. 2008.
[33] Y. Liu, X. Wu, and Y. Shen, "Automatic clustering using genetic algorithms," *Appl. Math. Comput.*, vol. 218, no. 4, pp. 1267–1279, Oct. 2011.
[34] H. He and Y. Tan, "A two-stage genetic algorithm for automatic clustering," *Neurocomputing*, vol. 81, pp. 49–59, Apr. 2012.
[35] A. Chowdhury, S. Bose, and S. Das, "Automatic clustering based on invasive weed optimization algorithm," in *Proc. Int. Conf. Swarm, Evol., Memetic Comput.* Berlin, Germany: Springer, 2011, pp. 105–112.
[36] H. Peng, J. Wang, P. Shi, A. Riscos-Núñez, and M. J. Pérez-Jiménez, "An automatic clustering algorithm inspired by membrane computing," *Pattern Recognit. Lett.*, vol. 68, pp. 34–40, Dec. 2015.
[37] R. J. Kuo and F. E. Zulvia, "Automatic clustering using an improved artificial bee colony optimization for customer segmentation," *Knowl. Inf. Syst.*, vol. 57, pp. 331–357, Nov. 2018.
[38] R. J. Kuo, Y. D. Huang, C.-C. Lin, Y.-H. Wu, and F. E. Zulvia, "Automatic kernel clustering with bee colony optimization algorithm," *Inf. Sci.*, vol. 283, pp. 107–122, Nov. 2014.
[39] A. K. Jain, "Data clustering: 50 years beyond k-means," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Berlin, Germany: Springer, Sep. 2008, pp. 3–4.
[40] W. P. Lee and S. W. Chen, "Automatic clustering with differential evolution using cluster number oscillation method," in *Proc. IEEE 2nd Int. Workshop Intell. Syst. Appl.*, May 2010, pp. 1–4.
[41] I. M. U. Saha, and S. Bandyopadhyay, "A new differential evolution based fuzzy clustering for automatic cluster evolution," in *Proc. IEEE Int. Adv. Comput. Conf.*, Mar. 2009, pp. 706–711.
[42] V. Kumar, J. K. Chhabra, and D. Kumar, "Automatic data clustering using parameter adaptive harmony search algorithm and its application to image segmentation," *J. Intell. Syst.*, vol. 25, no. 4, pp. 595–610, 2016.
[43] M. R. Murty, A. Naik, J. V. R. Murthy, P. P. Reddy, S. C. Satapathy, and K. Parvathi, "Automatic clustering using teaching learning based optimization," *Appl. Math.*, vol. 5, no. 8, p. 1202, 2014.
[44] S. Das, A. Chowdhury, and A. Abraham, "A bacterial evolutionary algorithm for automatic data clustering," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 2403–2410.
[45] F. Marini and B. Walczak, "Particle swarm optimization (PSO)," *Chemometrics Intell. Lab. Syst.*, vol. 149, pp. 153–165, Dec. 2015.
[46] D. W. van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 1, Dec. 2003, pp. 215–220.
[47] X. W. Xia, L. Gui, G. He, C. Xie, B. Wei, Y. Xing, R. Wu, and Y. Tang, "A hybrid optimizer based on firefly algorithm and particle swarm optimization algorithm," *J. Comput. Sci.*, vol. 26, pp. 488–500, May 2018.
[48] X. S. Yang, "Firefly algorithms for multimodal optimization," in *Proc. Int. Symp. Stochastic Algorithms*. Berlin, Germany: Springer, Oct. 2009, pp. 169–178.
[49] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.
[50] C. T. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters," *IEEE Trans. Comput.*, vol. C-100, no. 1, pp. 68–86, Jan. 1971.
[51] L. Fu and E. Medico, "FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data," *BMC Bioinf.*, vol. 8, no. 1, p. 3, 2007.
[52] A. K. Jain and M. H. C. Law, "Data clustering: A user's dilemma," in *Proc. Int. Conf. Pattern Recognit. Mach. Intell.* Berlin, Germany: Springer, 2005, pp. 1–10.
[53] H. Chang and D.-Y. Yeung, "Robust path-based spectral clustering," *Pattern Recognit.*, vol. 41, no. 1, pp. 191–203, 2008.
[54] A. Abraham, S. Das, and S. Roy, "Swarm intelligence algorithms for data clustering," in *Soft Computing for Knowledge Discovery and Data Mining*. Boston, MA, USA: Springer, 2008, pp. 279–313.
[55] S. J. Nanda and G. Panda, "A survey on nature inspired metaheuristic algorithms for partitional clustering," *Swarm Evol. Comput.*, vol. 16, pp. 1–18, Jun. 2014.
[56] A. José-García and W. Gómez-Flores, "Automatic clustering using nature-inspired metaheuristics: A survey," *Appl. Soft Comput.*, vol. 41, pp. 192–213, Apr. 2016.
[57] T. Hassanzadeh and M. R. Meybodi, "A new hybrid approach for data clustering using firefly algorithm and K-means," in *Proc. IEEE 16th CSI Int. Symp. Artif. Intell. Signal Process. (AISP)*, May 2012, pp. 7–11.
[58] J. Senthilnath, S. N. Omkar, and V. Mani, "Clustering using firefly algorithm: Performance study," *Swarm Evol. Comput.*, vol. 1, no. 3, pp. 164–171, 2011.
[59] M. Louzazni, A. Khouya, K. Amechnoue, A. Gandelli, M. Mussetta, and A. Crăciunescu, "Metaheuristic algorithm for photovoltaic parameters: Comparative study and prediction with a firefly algorithm," *Appl. Sci.*, vol. 8, no. 3, p. 339, 2018.
[60] M. Louzazni, A. Khouya, K. Amechnoue, M. Mussetta, and A. Crăciunescu, "Comparison and evaluation of statistical criteria in the solar cell and photovoltaic module parameters' extraction," *Int. J. Ambient Energy*, pp. 1–13, Aug. 2018, doi: 10.1080/01430750.2018.1517678.
[61] A. E. Ezugwu, O. J. Adeleke, A. A. Akinyelu, and S. Viriri, "A conceptual comparison of several metaheuristic algorithms on continuous optimisation problems," *Neural Comput. Appl.*, pp. 1-45, 2019, doi: 10.1007/s00521-019-04132-w.
[62] A. E. Ezugwu and F. Akutsah, "An improved firefly algorithm for the unrelated parallel machines scheduling problem With sequence-dependent setup times," *IEEE Access*, vol. 6, pp. 54459–54478, 2018.
[63] H. Banati and M. Bajaj, "Performance analysis of firefly algorithm for data clustering," *Int. J. Swarm Intell.*, vol. 1, no. 1, pp. 19–35, 2013.
[64] Y. Zhou, H. Wu, Q. Luo, and M. Abdel-Baset, "Automatic data clustering using nature-inspired symbiotic organism search algorithm," *Knowl.-Based Syst.*, vol. 163, pp. 546–557, Jan. 2019.
[65] X. Zhang, J. Li, and H. Yu, "Local density adaptive similarity measurement for spectral clustering," *Pattern Recognit. Lett.*, vol. 32, no. 2, pp. 352–358, 2011.

MOYINOLUWA B. AGBAJE received the B.Sc. degree (Hons.) in computer science from the Federal University of Technology, Akure, Nigeria, in 2015. She is currently pursuing the master's degree with the University of KwaZulu-Natal, Durban, South Africa. Her research interests include deep learning, machine learning, big data analytics, artificial intelligence, computer vision, and nature-inspired optimization.

ABSALOM E. EZUGWU received the B.Sc. degree in mathematics from the Department of Computer Science and the M.Sc. and Ph.D. degrees in computer science from Ahmadu Bello University, Zaria, Nigeria. He is currently a Senior Lecturer with the School of Mathematics, Statistics, and Computer Science, University of KwaZulu-Natal, Durban, South Africa. He has published articles relevant to his research interest in internationally referred journals and edited books, conference proceedings, and local journals. His main research interests include parallel algorithms design in cloud and grid computing environments, artificial intelligence with specific interest in computational intelligence, and metaheuristic solutions to real-world global optimization problems. He is a member of IAENG and ORSSA.

ROSANNE ELS received the M.Sc. degree in computer science from the University of Natal, Pietermaritzburg, South Africa, in 1997, and the PGCE degree from the University of KwaZulu-Natal, Pietermaritzburg, in 2008. She is currently a Lecturer of Computer Science with the School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal. Her research interests include parallel computing, deep learning, machine learning, data mining, artificial intelligence, computer science education, and nature inspired optimization.