A STUDY OF
A HYBRID CLUSTERING USING
SWARM INTELLIGENCE TECHNIQUES
AND
K-MEANS ALGORITHM

A THESIS
SUBMITTED TO THE SCHOOL OF SCIENCE AND TECHNOLOGY,
MIDDLESEX UNIVERSITY, LONDON,
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTERS OF SCIENCE

DURDANE KOCACOBAN

OCTOBER, 2015

Supervisor: Xin-She Yang

Programme Leader: Gary Hearne

ABSTRACT

The history of clustering goes back to Plato (Wikipedia, 2015). He showed a categorizing approach in his Western Philosophy by using their similar characteristics (Wikipedia, 2015). Additionally this idea was expanded and improved by Aristotle in his Categories study. On this basis, the main idea in clustering can be summarized as the categorize objects as their similarities.

In data mining, the most popular clustering method is probably the K-means algorithm. The algorithm works by dividing the data points to (a predetermined number of) k clusters. While doing this partition, the purpose is to minimize intra-cluster distances and maximize inter-cluster distance. K-means is a powerful clustering method. Although there have been many clustering algorithms in the literature K-means clustering has always been one of popular methods.

On the other hand, K-means algorithm has some flaws such as the initialization problem and trapped at local minima. K-means is very sensitive to choose centroids. Undesirable selection of K-means can affect performance of K-means. Additionally sometimes data points tend to converge at local optima related with distance measures. As a consequence, researchers have proposed many modifications or hybrids. Recently their tendency is to use swarm intelligence algorithms. In this way, there exist hybrid studies combining nature-inspired algorithms and K-means. The well-known examples of them are Particle Swarm Optimization and Firefly algorithm. In this dissertation, we will focus on both Particle Swarm Optimization and Firefly algorithm. These two algorithms have some similar characteristics and both have strong convergence. They are used for improving cluster centroids by minimizing cluster distance.

In this dissertation, to improve K-means clustering algorithm, the hybrid approaches have been proposed by combining two swarm intelligence techniques (PSO and FA) with K-means. In hybrid algorithms, we evaluated to obtain the minimum error sum of squares with the help of PSO and FA, respectively. We used several benchmarking data set examples to validate the approaches and obtained three concluding remarks. Our first conclusion is that hybrid methods often obtain the much better results than classical K-means. The second conclusion is that the hybrid of PSO and K-means does not keep its performance for large data sets, while hybrid FA keeps. The third conclusion is that even though FA and K-means methods can obtain better results, but they are also the slowest method.

ACKNOWLEDGEMENT

The dissertation was submitted in partial fulfilment of the requirements for the degree Master of Science in The School of Science And Technology. The project work was carried out at the University of Middlesex, London. The task has been under the Supervision of Dr. Xin-She Yang at the University Of Middlesex.

First and foremost, I am pleased to thank my supervisor Dr. Xin-She Yang who acted as my supervisor for his valuable comments and enthusiastic support. Without his assistances, this work would not have been completed. Additionally, I thank him for his very careful reading and insightful suggestions during the writing of the dissertation, and his thoughtful guidance during my graduate study.

Furthermore I should like to gratefully appreciate my program leader and lecturer Professor Gary Hearne. His lecture guides me to understand data mining method therefore I am able to insert some data analysis techniques in my dissertation.

At last, my special thanks also go to my parents for all their emotional support and everything they gave me, their understanding, encouragement and love.

TABLE OF CONTENTS

LIST OF FIGURES AND TABLE

Chapter 1

INTRODUCTION

1.1. INTRODUCTION

Swarming of birds and fish is an awe-inspiring sight that to our eyes looks chaotic. Inside the swarm, there is very different picture. The flock moves in harmony and always aware of their nearest neighbour, chatting together. Nowadays, the importance of systems inspired by Nature has increased and attracted the attention of artificial intelligence researchers. When members of a swarm are thought separately, they are not seen very clever (Akyol and Alatas 2012). However when they move as a swarm they exhibit a collective intelligence behaviour (Akyol and Alatas 2012). While these individuals are moving in a multi-dimensional space, they keep in mind the best location where they discovered (Kucukdeniz 2009). Then they share good locations where they discovered by communicating with each other. Therefore they can adjust their position and speed according to these locations (Kucukdeniz 2009). By observing these movements, swarm intelligence based optimization algorithm has been developed.

The well-known swarm intelligence optimization algorithms are Firefly Algorithm, Artificial Fish-Swarm Algorithm, Wolf Colony Algorithm, Cat Swarm Optimization Algorithm and Artificial Bee Colony Algorithm, Ant Colony Optimization, Particle Swarm Optimization (Akyol and Alatas 2012). There are numerous effective studies about these algorithms. Further, nature-inspired algorithms have been started to be used in various areas such as pattern recognition, optimization, classification and clustering. In particular, clustering applications are noteworthy.

The purpose of clustering is to show the data set in the form smaller cluster (Abraham et al 2008). Therefore, it has an essential role in data mining. Clustering is one of the newest and most effective areas and recent trends tend to combine with swarm intelligence because these nature inspired algorithms have made a significant contribution by helping in local search and improving initial centroid points. As a consequence, the studies in this area have increased. Current studies have shown that swarm intelligence techniques give also good results for clustering problems even when taken alone. Nevertheless they are seen as not good enough to take over classical clustering algorithm such as K-means. Normally, optimization algorithms in both areas have some drawbacks. For example, while swarm optimization algorithms are generally good at global search, K-means is at local search. Therefore the idea that is the using together of both algorithms was developed out. The many studies have confirmed this idea. In this way, our study proposed to hybrid two swarm algorithms (particle swarm and firefly is known good at clustering) and K-means, individually. In this way, our aim is to investigate the performance of combined approaches.

1.2. THESIS OUTLINE

The rest part of the dissertation project is organized as follows: Chapter 2 represents the literature review that explores developments in the K-means, Particle Swarm Intelligence, Firefly Algorithm and their hybridization studies in more recent times and presents related works. Chapter 3 presents in detail investigation and mathematically description of worked swarm algorithms and K-means. Additionally, this part consists of basic algorithms structure of hybridization of PSO and FA to K-means. Chapter 4 presents simulation and experimental result part. Firstly, used data sets are introduced and then visualized both labelled and unlabelled graphs using hybrid algorithms with Matlab. Finally, we discuss and criticize results, which we obtained for known swarm and new mix algorithms. It also provides some suggestions. Chapter 5 concludes the study in the dissertation and points out possible directions for future research. Finally, we provided references we used and also Appendices part which includes Appendix A and B. Part A represents code listing PSO and K-means source code in Matlab, while B part represents the experimental results data.

Chapter 2
LITERATURE REVIEW

2.1. LITERATURE OF K-MEANS ALGORITHMS

Although the history of clustering is told dating back Aristotle it has always been in people life from the beginning. People split into categories to their products to use easily. K-means has always kept its important place in clustering algorithms. When I write K-means algorithm on the website https://library.mdx.ac.uk/summon/ the system represented over 10000 relevant resources about K-means. There was conference paper over 6000 journals article over 2000 and approximately 1000 books. This searching time period is ranging from 1930 up to date. As the purpose of this study is to focus on hybrid studies for K-means we will apply to limited studies, which emphasize origins and the important developments in K-means.

Let us start with emergence of K-means. The Steinhaus proposed K-means idea in 1956 and next year Lloyd prepared a study representing the standard algorithm of K-means (Karoussi, 2015). However this study published in 1982. As a consequence, J. MacQueen brought forward the first official publication on development of K-means (MacQueen, 1967). In 1975 J. A. Hartigan and M.A.Wong at Yale university proposed K-means algorithm efficiency via a publication. The main idea of algorithm was to divide of the variables to clusters in pre-specified number. At first, the centroids of clusters are determined randomly and then the distance to centres of members in each cluster is measured. Each member is assigned to nearest centroids (Wu, 2012). With some iteration, these clusters are updated. The process repeats until solutions stabilize. The clusters and its member numbers are determined as shortest distance to centroids. As the algorithm was suggested firstly by Lloyd (1982), it is also known as Lloyd's algorithm. In literature, there are many implementations for K-means. One of them was belonged to Kanungo et al in 2002. They suggested an easy and efficient implementation of K-means, which named as the filtering (Kanungo et al. 2002). The system calculates the kd-tree for data points instead of centres therefore it does not need an updating for centroids (Kanungo et al. 2002). In 2003, Likas et al proposed another hybrid study on K-means and global search called as global k-means. In order to minimize clustering error that is the main idea of K-means, the system uses a deterministic effective global clustering algorithm (Likas et al. 2003). The initial position in pure K-means has an important role in algorithm results. In contrast to K-means, the global k-means does not depend on initials positions and uses K-means for just local search part (Likas et al. 2003).

On the other hand, another study aspired to improve initial value sensitivity of K-means without ignoring to it. Sometimes the random initialization causes trouble such as the clusters have same members. Khan and Ahmad (2004) put forward cluster centre initialization algorithm (CCIA) with the aim of solving this problem. The CCIA employs K-means for determining starting centroids at first and improves these values (Khan and Ahmad, 2004). In 2007, Reymond and Henegan propounded a study claiming that kd-tress is better to initialize cluster centroids. Concurrently, Lu and his friends approached initialization problem via a hierarchical model treating to the problem as weighted clustering (Lu et al. 2008). The results demonstrated this new approach has a better performance and requires less iteration numbers (Lu et al. 2008).

The popularity of clustering came into also prominence in different areas such as biology and chemistry. The scientists Du and Wang who study on microarray technology proposed a method named as PK-means (P stands for particle-pair optimizer). It is used to cluster gene data sets and aims to reduce to stagnation problem in local search (Du and Wang, 2008). The technique has a simple structure and fast converge ability.

Another interesting idea was executed by Žalik in 2008. He suggested a new algorithm that is extension of classic K-means algorithm. The thing making the algorithm interesting is that it performs properly without pre-specified number of cluster (Žalik, 2008). The proper cluster number is provided by means of a special cost function (Žalik, 2008).

The researchers have never given up to study on the K-means due to its effectiveness but suffered from some uncontrollable parts such as initial centroids and number of group members. In 2003, Yi and Moon proposed a novel study that is extended K-means algorithm. As we mentioned the K-means has no control mechanism for adjusting the number of cluster member (Yi and Moon, 2013). However, this is important in areas such as logistic (Yi and Moon, 2013). The novel algorithm appeals to help of greedy algorithm to adjust it and it provides successful results (Yi and Moon, 2013). For initialization problem, recently many research emerge on K-means. The most notably of them are hybrid studies with nature inspired algorithms such as particle swarm and firefly algorithm. These algorithms have an amazing capacity for local search. Therefore they are used to improve centroid points that assigned from K-means.

2.2. LITERATURE OF PARTICLE SWARM OPTIMIZATION

Kennedy and Eberthart in USA commenced the particle swarm studies in 1995. These two successful researchers were impressed by workings of nature system and they brought to bear this system mathematically. They correlated to particle swarm with artificial life and more specifically bird flocking, animal herds and fish swarms (Kennedy and Eberthart 1995). The study put forward five properties of a swarm, suggested by Millonas, as follows:

- Proximity: the member should protect the distance with others
- Quality: the swarm has the capacity to interfere quality factors
- Diverse Response: the members should maintain their activities in every situation

- Stability: the different environments should not affect behaviour of swarm
- Adaptability: the swarm should be able to change behaviours for better conditions

The initial velocity and position formulation of particle swarm optimizer was suggested simply as:

$$v_i(t+1) = v_i(t) + c_1 * rand() * \big(p_i(t) - x_i(t)\big) + c_2 * rand() * (g(t) - x_i(t))$$
$$\mathrm{x_i(t+1) = x_i(t) + v_i(t+1)} \tag{2.1}$$

where $\mathrm{v_i(t)}$ and $\mathrm{x_i(t)}$ are velocity and position of particle i at iteration t, $\mathrm{v_i(t+1)}$ and $\mathrm{x_i(t+1)}$ are velocity and position of particle i at iteration $\mathrm{t+1}$, rand ( ) is random number between (0,1) and $\mathrm{c_1}$ and $\mathrm{c_2}$ are cognitive and social coefficients. Concurrently, Kennedy and Eberthart proposed a new optimizer using PSO (Eberthart and Kennedy, 1995). In this study, they discoursed on p_i = "gbest" local best and p_g = "lbest" global best concepts. They discovered that the changes on these values had an effect iteration number of algorithm. The used of local and global search together made possible to obtain better results (Eberthart and Kennedy, 1995). In 1998, a study committed in partnership with Eberthart evaluated partially the formulation of PSO' velocity formulation. It approached the dependency and importance of these parts. In study, the velocity update is separated three parts as follow:

$v_i(t)$ $\longrightarrow$ Previous velocity

$c_1 * rand() * \big(p_i(t) - x_i(t)\big)$ $\longrightarrow$ Social

$c_2 * rand() * (g(t) - x_i(t))$ $\longrightarrow$ Cognition

They discovered that without previous social and cognition parts, the particles move in a constant speed and direction (Shi and Eberthart 1998). Therefore, if the optimal solution is not on particles way they will unable to reach it. In an opposite case, velocity would depend on just particles location without velocity (Shi and Eberthart 1998). Then the particle at the best global location will remain up another takes over (Shi and Eberthart, 1998). Until that time, the particles will centre upon it at global best. As a consequence, the search space will narrow and it will turn local search (Shi and Eberthart, 1998). For preventing this problem, Shi and Eberthart put forward "w" inertia weight and they modified it to the formulation as follows:

$$v_i(t+1) = w * v_i(t) + c_1 * rand() * \big(p_i(t) - x_i(t)\big) + c_2 * rand() * (p_g(t) - x_i(t))$$
$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2.2}$$

The contribution of "w" inertia weight is to provide balance between local and global search (Shi and Eberthart 1998). Nevertheless still the system had some shortcomings in some instances. For example, when global best is discovered the all members gravitate towards this global best by ignoring their closest local minimizer. Therefore, global, local best and current position of particles will be same $p_i(t) = p_g(t) = x_i(t)$. in this circumstance, velocity update will just depend $w * v_i(k)$ (Bergh and Engelbrecht 2002). In 2002, Bergh and Engelbrecht propounded a study (Guaranteed Convergence PSO (GCPSO)) aiming to solve this problem. They modified the formulation by adding another update equation as follows:

$$p_i(t+1) = \begin{cases} x_i(t+1), & f(x_i(t+1)) < wf(p_i t) \\ p_i(t), & otherwise \end{cases} \tag{2.3}$$

Although Bergh and Engelbrecht proposed effective solution this problem it seems that the traditional PSO still has some problems and deficiencies. New modified method prevented to move all particles towards the global best, particles show a tendency the local minimizers could not been guaranteed (Bergh 2002). Bergh was aware deficiency of his own study and approached this problem in his dissertation to remove it. He put forward multi-start PSO that makes a restart with new numbers became different from those of previous searches when there occurs stagnation (Bergh 2002). Then, it compares previous global bests with new one and executes the most proper global best position (Bergh 2002). In following year, a study proposed by Merwe and Engelbrecht put forward an interesting idea. It showed PSO is also effective to cluster data and improve to K-means clustering. PSO was used to seed the initial clusters (Merwe and Engelbrecht, 2003).

The researchers revealed also new studies to improve and modify to classic PSO. In 2005, Janson and Middendorf proposed the hierarchical particle swarm optimizer. In this method the particles are calibrated in a dynamic hierarchy, which implies the neighborhood system (Janson and Middendorf, 2005). The system moves the particles upward or downward hierarchy as the quality of their best solutions found so far (Janson and Middendorf, 2005). They used regular tree as a method. The particles are put hierarchically by using tree at each iteration before update velocity and position (Janson and Middendorf, 2005). They showed H-PSO is faster than classic PSO.

Every researcher approached various problems in PSO and propounded new modified methods. Hui Wang et al executed opposition based PSO in 2007. The method prevents that the particles oscillate between previous and global best particle (Wang et al, 2007). Also it provides that the particles expand their search space by adding searching neighbours of global best (Wang et al, 2007). In previous years, although many new PSO algorithms were suggested the stagnation problem of PSO did not prevent totally. Evers and Ghalia prepared a study called as regrouping PSO. The system works by regrouping automatically the swarm when stagnation situations such as focusing on local optima (Evers and Ghalia, 2009).

From 2010, the points of interest were slightly changed. The scientists started to focus on PSO clustering subject. For example, in 2011, in contradistinction to our study, Cura claimed that pure PSO does better clustering than hybrid studies (Cura, 2011). After that, new studies on this subject have begun to emerge.

2.3. LITERATURE OF FIREFLY ALGORITHMS

Firefly algorithm is one of recent popular nature inspired algorithm. In 2009, Xin-She Yang proposed the firefly algorithm at Cambridge University. As the name suggests, the algorithm was inspired by behaviour of fireflies (Yang, 2008). In a nutshell, firefly is based on three rules (Yang and He, 2013):

- There interaction between two fireflies is independent of genders.
- The attractiveness is proportional to the brightness. Therefore the system would allow to move toward closer adjacent. This rule is especially important for clustering problems.
- The brightness of a firefly is designated by the landscape of the objective function.

Although the firefly algorithm (FA) has some similarities with particle swarm optimization, many studies have confirmed that FA is more effective than some swarm intelligence algorithm. To illustrate, Xin-She Yang who is creator of algorithm exhibited it with a study in 2009. In this study, he presented a performance benchmarking among firefly, genetic algorithm and particle swarm by using well-known optimization test functions (Yang, 2009). Interestingly enough, FA performed 100% for all test functions except first two, while GA and PSO were showing 100% performance in only one function (Yang, 2009). This situation has been inspiring for researchers engaged in various studies on GA and PSO to compare performances.

In 2010, Yang and Deb put forward a study comparing eagle strategy (hybridization of Lévy walk and Firefly) and PSO. As is evident from its name, Eagle strategy is a metaheuristic algorithm inspired by food searching ability in nature of eagles (Yang and Deb 2010). The difference between results of two methods was significant. It is possible that this study might be one of the studies displaying performance difference of FA from other metaheuristic algorithms. Another comparative performance study was propounded by Rajini and David in following year. The study claimed that FA is the best metaheuristic algorithm when it is compared with other algorithms. In this study, a Flexible Neural Tree model was generated by using Ant Colony and FA (Rajini and David 2011).

After FA proved itself, some modification studies appeared on it. In 2011, Farani et al proposed the Gaussian FA. It was asserted for improving some deficiencies of FA such as entrapping into local optimums (Farani et al. 2011). Using Gaussian distribution created modification. For same problem, Tilahun and Ong propounded another modification study in next year. In 2012, Farahani et al. proposed a hybrid to solve this stagnation problem for local optima. The hybridization of genetic algorithm and firefly was suggested to treat global search and obtain better results (Farahani et al. 2012). At same year, Abdullah et al. put forward also a hybrid combining FA with the evolutionary operations of Differential Evolution. It was named as Hybrid Evolutionary Firefly Algorithm and purposed to improve searching and knowledge sharing ability of FA (Abdullah et al. 2012). Furthermore, another study was suggested by three academics at Kalyani University in the same year. It is a new hybridization study that includes back propagation algorithm (BPNN) into the FA (Nandy et al. 2012). The results showed that the hybridization of FA and BPNN converged faster than those of genetic algorithm and BPNN and the hybrid provided better results (Nandy et al. 2012).

Recent research studies related with FA are on classification and clustering areas in data mining. As the nature inspired algorithms have amazing ability for local search, it has taken scientists' attention. They began to use in their studies to get more quality results. Alweshah and Abdullah proposed one of these works in this year. The study hybrids FA and probabilistic neural network (PNN) by aiming to solve classification problem (Alweshah and Abdullah, 2015). Method works by hybridizing FA and Simulated Annealing (SA) because SA both checks the randomness steps in FA and optimizes the weights of PNN (Alweshah and Abdullah, 2015). These effective results of FA influenced us to insert it our study.

Chapter 3

METHODOLOGY

3.1. WHAT IS THE CLUSTERING?

Unsupervised learning is to be given only variables without any information and ensured several results by using these variables. More clearly, Unsupervised approaches can also be a classification method used when we do not know how to put into classes. Clustering is an unsupervised classification technique used in pattern recognition problems. It divides to unlabelled data sets to a particular number of clusters as their similarities (Xu and Wunsch 2009). Today, we can see many real life examples of it. To illustrate, when you added familiar people on a social media, the website creates a specific group as your added peoples and gives you suggestion to add also. When the lack of information about data, some problems steps in the clustering (Breaban and Luchian 2011). For examples,

- Specify the distance function
- Specify the relevant characteristics
- Designate the clustering criteria
- Validate the solution

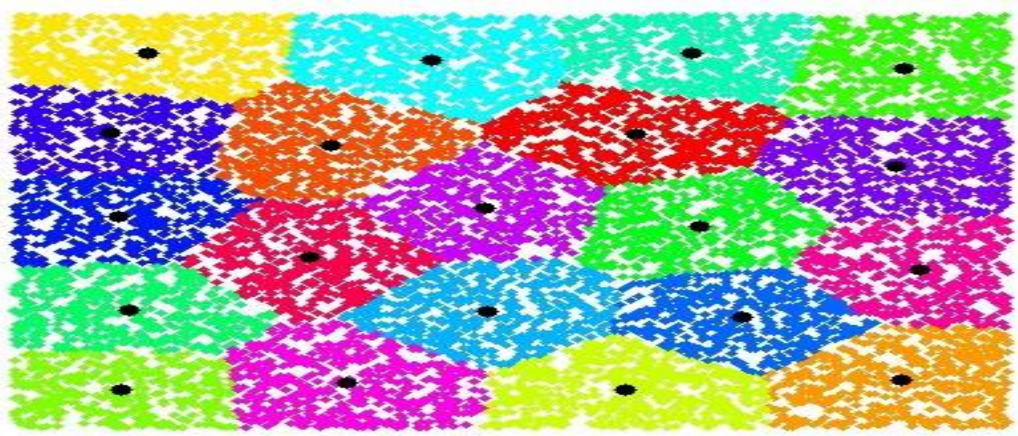

Figure 3. 1. K-means clustering of 5000 randomly sampled data points (VLFeat, n.d.)

These problems could release as different examples. Therefore for each example could be needed different criteria optimization (Breaban and Luchian 2011).

The well-known algorithms K-means takes account global structure of data set (Breaban and Luchian 2011). Reasonably, the account of local and global structure of data will be more helpful to cluster therefore our study will be on improvement of both structure. To develop it, some hybridization methods will be appealed.

3.2. K MEANS

K-means algorithm is one of the earliest clustering algorithms. As we mentioned in literature review part, Hugo Steinhaus firstly suggested the idea in 1956 and simultaneously Stuart Lloyd proposed the algorithm that went by the name of the standard algorithm. Nevertheless, in 1967, the term "K-means' was firstly emerged by James MacQueen. As the name suggests, the idea of algorithm is to partition k clusters according to the means of these clusters. K-means is based on an objective function. This objective function is based upon searching the nearest points to cluster centroids (Wu 2012). The purpose is to separate by doing maximum the similarities between elements of each cluster and minimum the similarities among clusters. These centroids are determined randomly at first. After, they are updated permanently up to reach discrete and best clusters. Let's explain in detail and represent mathematically.

Suppose we have a $\mathrm{X} = \{\mathrm{x}_1, \mathrm{x}_2, \dots, \mathrm{x}_\mathrm{n}\}$ data set and each points has d dimensions. In the algorithm, as a first, k random points are selected and data is divided k clusters in the way of$\{C_1 \dots C_k\}$. Properly, the algorithm of K-means can be summarized as follow;

Fundamental Working System of K-means Clustering Algorithm

---

1. Initialize cluster centroids randomly
2. ***Repeat***
   (a) Assign each member of clusters according to the closest cluster centroids

   Where the distance of members to the centroids is determined by;

   $$d(c_k, x_i) = \sqrt{\sum_{j=1}^{d}(c_{kj} - x_{ij})^2} \quad (3.1)$$

   where k and d subscript cluster number and dimension, respectively.

   i represents ith data point ($i = 1 \dots n$). c_k is kth centroid.

   (b) Recalculate cluster centroids of each cluster
3. ***Until*** converge criterion is met.

The figuration of K-means algorithm is demonstrated as follows:

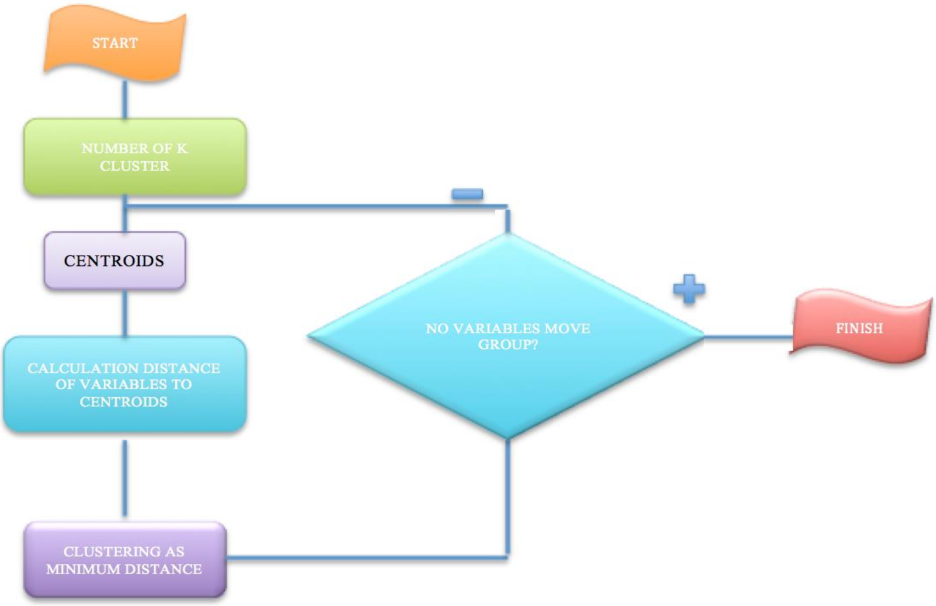

Figure 3. 2. The K-means Algorithm (Ramsey, 2014)

K-means clustering objective function is generalized by cluster sum of squares as follows:

$$\mathrm{SSE} = \sum_{l=1}^{k} \sum_{\forall x_i \in C_k} d(c_k, x_i)^2 \qquad (3.2)$$

SSE is the easiest and most used clustering measure criterion. It measures and provides the sum of distance data points and centroids. Therefore the smallest SSE gives us the best clustering result obtained. The distance between two objects is measured by using Euclidean distance given in equation (3.1). In final stage of iterations, the centroids are recalculated as following equation;

$$\mathrm{c_k} = \frac{1}{\mathrm{n_j}} \sum_{\forall \mathrm{x_i} \in \mathrm{C_j}} \mathrm{x_i} \qquad (3.3)$$

Where $\mathrm{n_j}$ performs the number of data points in the jth cluster, while $\mathrm{x_i}$ specifies data points in the jth cluster.

The K-means is a heuristic method therefore it cannot guarantee optimum results. As the centroids are selected randomly at start, results would change depending on these points. K-means generally works very fast. However, sometimes, it might work very slowly due to initialization problem. As a result of this initialization problem, it gives very unsatisfying results. To solve this problem, there have been many studies. Recent studies mostly focus on metaheuristic methods such as particle swarm optimization and firefly algorithm, etc. As they have unique for local search, they are used to improve initial centroids that assigned randomly by K-means. The best two algorithms on this topic are particle swarm and firefly algorithm. Therefore we will cover just these two topics.

3.3. PARTICLE SWARM INTELLIGENCE

The Particle Swarm Optimization (PSO) is a stochastic optimization algorithm, which was inspired from swarm behaviours such as bird flocks, fish schools and animal herds, and James Kennedy (Social Psychologist, Purdue University) and Russell Eberthart (Electrical Engineer, Bureau of Labour Statistics) found it out in 1995 (Parsopoulos and Vrahatis 2010).

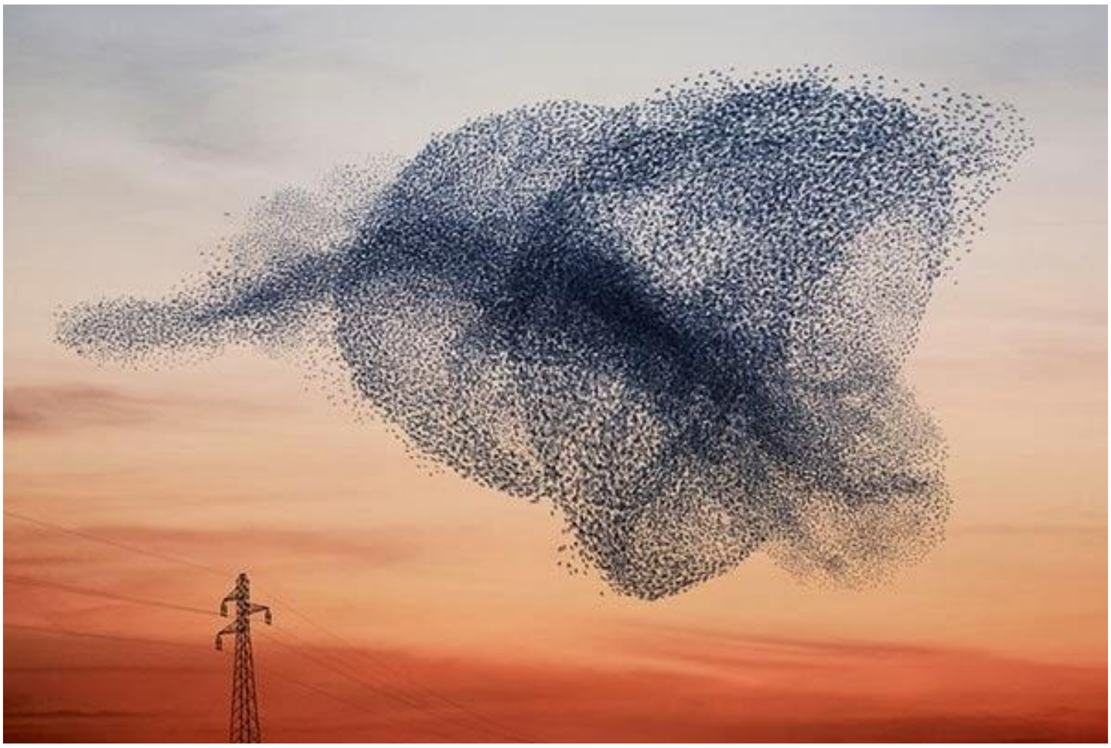

Figure 3. 3. The Particle Swarm Optimization (DL Cade, 2014)

It is a heuristic optimization technique and tries to reach optimal solution just like other optimization algorithms make. It seems the most popular among of swarm intelligence methods. PSO uses particles as the name suggests. In PSO, each particle remembers its best positions and the best positions of other particles taking place in its neighbourhood (Kennedy and Eberthart 1995). It classifies these positions as pbest and gbest sets. That is, pbest shows a set of the best positions of particle, which are the best converging points to optimal solution. gbest represents a set of the best positions of neighbour particles, which are the best converging points to optimal solution (Kennedy and Eberthart 1995). The main aim of the PSO is to speed up to reach of the particles to their pbest and gbest positions (Kennedy and Eberthart 1995). This is a main idea of the algorithm. Let's look at the algorithm mathematically.

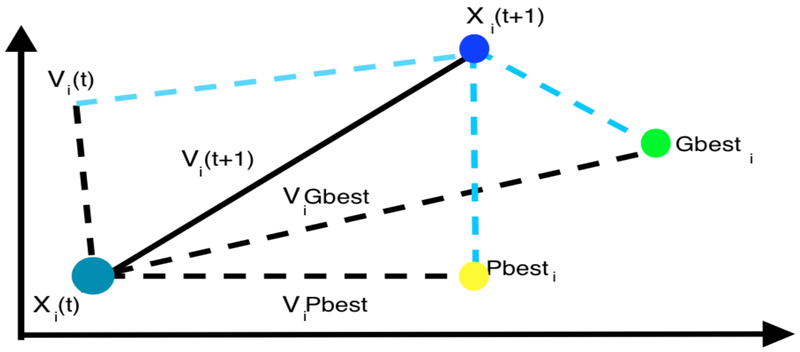

Figure 3. 4. The Particle Swarm Optimization (inspired) (Hosseini, 2014)

Let A is a search space in $\mathbb{R}^n$ and $f: A \rightarrow Y \subseteq \mathbb{R}$ is our objective function. As we mentioned earlier PSO employs the particles. The swarm of these particles is defined as a set:

$$\boldsymbol{S} = \{x_1, x_2, \dots, x_N\} \tag{3.4}$$

where N particles (N number specified by users) and x_i is defined as,

$$x_i = (x_{i1}, x_{i2}, \dots, x_{in})^{\mathrm{T}} \in A \qquad i = 1, \dots, N \tag{3.5}$$

The $f\ (x)$ function is valid for all particles. Each variable has a function in the form of $f_i = f\ (x_i)\ \in Y$ (Parsopoulos and Vrahatis 2010). As these particles refer to swarm members it is supposed that particles move in the space$A$. Therefore, each particle normally has a proper velocity emitted as:

$$v_i = (v_{i1}, v_{i2}, \dots, v_{in})^{\mathrm{T}} \qquad i = 1, \dots, N \tag{3.6}$$

where t refers iteration number. For i th particle, $x_i(t)$ denotes particle position, while $v_i(t)$ is velocity (Parsopoulos and Vrahatis 2010). In this swarm each particle has a memory and they have capability to store their $best$ positions discovered during its search process. This $memory$ set is figured as:

$$\boldsymbol{P} = \{p_1, p_2, \dots, p_N\} \tag{3.7}$$

p_i is defined as,

$$p_i = (p_{i1}, p_{i2}, \dots, p_{in})^{\mathrm{T}} \in A \qquad i = 1, \dots, N \tag{3.8}$$

such that t is iteration number, these positions are figured as:

$$p_i(t) = \underset{t}{\arg min}\ f_i(t) \tag{3.9}$$

Seeing that the swarm members are social creatures they have the capacity to share their experience about best positions. Therefore it also allows storing the global best position. Let g is the index of the best position in$\boldsymbol{P}$.

$$p_g(t) = \arg\min_{t} f(p_i(t)) \tag{3.10}$$

At last, PSO' velocity and position are mathematically generated as follows:

$$\mathcal{V}_{\mathrm{ij}}(\mathrm{t}+1) = \mathcal{V}_{\mathrm{ij}}(\mathrm{t}) + c_1\mathrm{r}_1(\mathrm{p}_{\mathrm{ij}}(\mathrm{t}) - \mathrm{x}_{\mathrm{ij}}(\mathrm{t})) + c_1\mathrm{r}_1(\mathrm{p}_{\mathrm{gj}}(\mathrm{t}) - \mathrm{x}_{\mathrm{ij}}(\mathrm{t})) \tag{3.11}$$
$$x_{ij}(t+1) = x_{ij}(t) + \mathcal{V}_{\mathrm{ij}}(t) \tag{3.12}$$
$$i = 1, \dots, N, \quad j = 1, \dots, N$$

such that t is the iteration number, r_1, r_2 are random variables normally distributed in $[0,1]$. c_1 is $cognitive$ parameter and c_2 is $social$ parameter. The PSO also has a control mechanism for possible next best particles. The particle is updated its position via following equation depending on the next position is better or not. The new better position of x_i at iteration $t+1$ iteration is:

$$p_i(t+1) = \begin{cases} x_i(t+1), & f(x_i(t+1)) \leq f(p_i(t)) \\ p_i(t), & otherwise \end{cases} \tag{3.13}$$

When the index g (general best position) is determined, PSO iterations are terminated (Parsopoulos and Vrahatis 2010).

In the end, we want to emphasize some highlights of PSO. While PSO algorithm is running, following points should be considered. The first highlight is the reason for our particle number selection in application part of study.

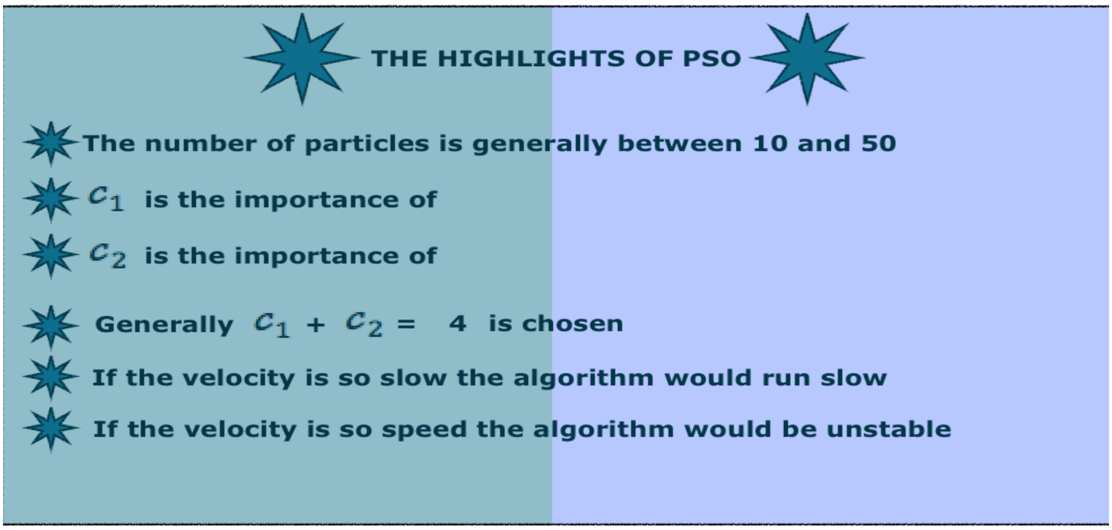

Figure 3. 5. The Highlights in The Particle Swarm Optimization

3.4. FIREFLY ALGORITHM

3.4.1. Firefly behaviour and habitat

Fireflies are an interesting insect species, which belong to the beetle group in arthropods. They have approximately 2000 species and the most of them live in the tropical and temperate areas (Lukasik and Zak, 2009). These insects are a natural light source. They are capable of producing natural light by means of photogenic cells in the back of the transparent cuticle layer and in the abdominal region located close to the body surface. They produce chemically cold light, which become yellow, green and pale-red lights. While they are flying they produce flickering light they blink their lights in the case of short periods.

Figure 3. 6. The Firefly Lights (Anon, 2015)

This rhythmic system separates them from other similar species and construct them as a special communication method (Yang, 2010). The velocity and frequency of flashing and the lag of response flashing, each of them has a special meaning such as the taking of its prey attention or allure potential partners. This communication method was considered as an interesting subject by science world and the firefly algorithm proposed.

3.4.2. Firefly Algorithm

In 2008, the intelligent swarm communication signalling of fireflies inspired researcher to put forward an algorithm based on fireflies' flashing behaviour (Yang, 2008). The underlying luminescence characteristics in the algorithm are summarized with the following three rules (Yang, 2009):

Fireflies are accepted as epicene, therefore fireflies light flickering to each other regardless of their sex.

Expressiveness, it depends on the degree of brightness. A firefly has less brightness would move towards a brighter firefly. If there is no a brighter firefly than itself it moves randomly.

To obtain optimum results, the brightness of a firefly is determined by the objective function or type of the problem.

The firefly algorithm base on two important matters: the variation of light intensity and formulation of the attractiveness (Yang, 2010).

For the created m fireflies, the optimization objective function that gives the fitness value of x_i solution for each i firefly is determined

$$f(x^*) = \min_{x \epsilon S} f(x) \tag{3.14}$$

The purpose is to find x^* solution which maximises (or minimises) to the f(x) objective function (Yang, 2010). The cost value obtained with the application result of each firefly solution to the objective function refers to the brightness of the firefly (Yang, 2008).

As it was mentioned before, it is assumed that the attractiveness of a firefly is directly proportional with its light intensity (Yang, 2010). We can choose a particular position x and express the light intensity with $I(x)$ function, then $I(x) \propto F(x)$. On the other hand attractiveness β would depend on the viewer visibility distance, that means it would show a difference as the distance r_{ij} between ith and jth fireflies (Yang, 2010). As much as the distance decreases then the brightness declines. In addition to this, the light is psychically absorbed by environment; therefore the degree of absorption should be taken into account.

3.4.2.1. Light Intensity

Mathematically, the brightness $I(r)$ show variety with distance r with regard to inverse square law, $I(r) = I_s/r^2$ where I_s is brightness at source (Yang, 2010). However, in the case of $r = 0$, a singularity situation occurs. To eliminate this problem, I_s/r^2 expression is approximated to $I_0 e^{-\gamma r^2}$ by using Gaussian (Yang, 2010).

$$I(r) = I_0 e^{-\gamma r^2} \tag{3.15}$$

(If this formulation declines slow, $I(r) = \frac{I_0}{1+\gamma r^2}$ can be substituted)
Therefore $I(x)$ function can be generated as:

$$I = I_0 e^{-\gamma r} \tag{3.16}$$

where I_0 is the light intensity at $r = 0$ and γ is the light absorption coefficient.

3.4.2.2. Attractiveness

The brightness and attractiveness is directly proportional. By using this information the attractiveness β is generated as follows:

$$\beta = \beta_0 e^{-\gamma r^2} \tag{3.17}$$

where β_0 is the light intensity at $r = 0$ and γ is the light absorption coefficient (Yang, 2010).

3.4.2.3. Distance and Movement

Similar with PSO, fireflies update their positions and move better place. Suppose we get two fireflies at x_i and x_j locations. The Cartesian distance between them as shown:

$$r_{ij} = \|x_i - x_j\| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{3.18}$$

where $x_{i,k}$ expresses the kth component in the x_i coordinate of ith firefly (Yang, 2010). If we have two-dimensional search space it can be expressed as follows:

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{3.19}$$

For two adjacent fireflies i and j, the movement of less brightness one (i) to other firefly (j) is calculated following formulation:

$$x_i = x_i + \beta_0 e^{-\gamma r^2{}_{ij}} (x_j - x_i) + \alpha \left(rand - \frac{1}{2}\right) \tag{3.20}$$

such that α is randomization parameter and disturbed uniformly in [0,1].

3.5. HOW TO HYBRIDIZE PSO and K-MEANS

The K-means is easy applicable and efficient clustering method. Nevertheless, it has also some drawbacks like other heuristic methods. The most important problems are sensitive initialization and converging at local optima. For solving these problems, the various hybridization methods were asserted in K-means clustering (Breaban and Luchian 2011). One of these algorithms is PSO. It is a known fact that K-means algorithm inclines to converge quicker than PSO (Merwe and Engelbrecht, 2003). Nevertheless, it generally does less accurate clustering (Merwe and Engelbrecht, 2003). On the other hand, PSO algorithm is so much better than K-means algorithm for searching local optima because PSO has ability to maintain, recombine and compare several candidate solutions at the same time (Ding et al. 2011).

For the first time, Mohamed Omran et al. proposed PSO based hybrid clustering algorithms in 2002 (Ding et al. 2011). Their study demonstrates that PSO algorithm does better the K-means algorithm (Ding et al. 2011). After that, the study on hybridization has increased dramatically. Some of them suggested starting with K-means for initialization, while some is K-means. As different from Omran' treatise, our algorithm will start with K-means for initializing centroids, and then PSO will improve and update them, in final part again K-means will step in to perform updated centroids. Previously Merwe et al. proposed an algorithm combining PSO and K-means similar with we planned. In addition to this algorithm, we will use again K-means at last part. Therefore we will use this algorithm just as a basis.

The Working System of PSO based K-means Clustering Algorithm
1. Initialize k random cluster centres and execute K-means algorithm.
2. ***For*** iteration=1 to maximum iterations do
3. ***For*** particle= 1 to all particles do
4. ***For*** all x_i dataset variables
5. Calculate Euclidean distance of all x_i and all cluster centroids obtained by K-means.
6. ***End*** for
7. Assign each variable to its closest cluster centroid.
8. ***End*** for
9. Calculate global fitness and local fitness
10. Update cluster centroids according to velocity updating
11. ***End*** for
12. Again, start up K-means with updated cluster centroids.

In the algorithm, just first particle is matched with clusters centroids obtained by K-means and rest of them are initiated randomly (Ding et al. 2011). This method demonstrates that the seeding to the initial swarm particles with K-means clusters outperforms the PSO clustering (Merwe and Engelbrecht, 2003). This PSO based clustering algorithm is termed as PSO-KMEANS.

3.6. HOW TO HYBRIDIZE K-MEANS and FIREFLY

In data analysis, the clustering is most used technique. As we mentioned previous parts, when clustering expression has always run through the minds K-means clustering. It has been a clustering method that is the most studied and done scientific research on. Firefly Algorithm is another metaheuristic approach to improve deficiencies of K-means. However, we will introduce and use modified FA instead of standard algorithm to improve our results.

3.6.1. Modified Firefly Algorithm

In pure FA, a brighter firefly can attract fireflies over the period of iterations (Hassanzadeh and Meybodi, 2012). Therefore fireflies stagnate around local optima rather than global optima, thus algorithm ability decreases (Hassanzadeh and Meybodi, 2012). To improve these shortcomings, Hassanzadeh and Meybodi proposed modified firefly algorithm (MFA). MFA is so similar with standard algorithm; just global optima were added to position equation. Therefore the algorithm would update global optima in each iteration and this situation would prevent the fireflies stagnate on local optima. As the modified algorithm is so similar with classic FA, we will give just modified part of it. The position update part was modified as follow:

$$x_i = x_i + \beta_0 e^{-\gamma r^2{}_{ij}}\left(x_j - x_i\right) + \beta_0 e^{-\gamma r^2{}_{i,gbest}}\left(x_{gbest} - x_i\right) + \alpha\left(rand - \frac{1}{2}\right) \qquad (3.21)$$

where, $gbest$ is global optima and x_{gbest} is the coordinate of global optima (Hassanzadeh and Meybodi, 2012).

Just as in PSO-clustering algorithm, our aim is to improve K-means algorithm results by initializing with clusters updated by MFA. In the proposed method, the clustering has three stages that are set clusters with K-means, update and improve clusters with MFA and initialize K-means with MFA updated clusters. Our objective function refers Euclidean distance. The propounded hybrid-clustering algorithm can be summarized as follows:

The Working System of FA based K-means Clustering Algorithm

1. Initialize k random cluster centres and execute K-means algorithm.
    2. ***For*** iteration=1 to maximum iterations do
        3. ***For*** firefly= 1 to all fireflies do
            4. ***For*** all x_i dataset variables
                5. Calculate Euclidean distance of all x_i and all cluster centroids obtained by K-means.
            6. ***End*** for
            7. Assign each variable to its closest cluster centroid.
        8. ***End*** for
        9. Calculate global fitness and local fitness
        10. Calculate Euclidean distances of all x_i to swarm global best position
        10. Update cluster centroids
    11. ***End*** for
12. Again, start up K-means with updated cluster centroids.

This MFA based clustering algorithm is named as K-MFA.

Chapter 4

SIMULATION AND RESULTS

4.1. DATA SETS AND INPUT PARAMETERS

The purpose of chapter 4 is to evaluate how bio-inspired optimization algorithms may outperform K-means algorithm. As we mentioned in detail before, nature-inspired algorithms are good to take over and improve local search of K-means. Additionally, the applying them K-means structure is easy. In this chapter we will represent and compare PSO, FA and K-means performances on data sets. These comparisons will be done by taken as a basis Error Sum of Square (SSE) and Mean Square Error (MSE) of them because SSE provides the sum of squared differences between data variables and cluster centroids and MSE gives a measure of how close a fitted line is to data points. It means that the more SSE is small the more data set is well clustered.

For each data set, in general, we appealed to 10 particles and 20 iterations for analysing because results did not change anything for bigger values. Therefore we decided to immobilize them at these parameters. Furthermore, we used different parameter values for both algorithm PSO-K-means and FA-K-means. For PSO-K-means, w inertia weight was attributed to iteration numbers inspired by Mathworks PSO package and used following equation:

w(iteration)=((iterations - iteration)*($w_{max} - w_{min}$))/(iterations-1) + w_{min} (4.1)

where w_{max} is maximum inertia and it is taken 0.9, while w_{min} is minimum and taken 0.4.
The acceleration constants c_1 *and* c_2 at equation (3.10) are taken both 1.49.

On the other hand, for FA-K-means, α randomization parameter and β attractiveness are linked to iteration numbers as following equations:

$$\alpha(\text{iteration})=\alpha_0 * (0.8^{(iteration)}) \quad (4.2)$$
$$\beta(\text{iteration})=\beta_0 * e^{-\gamma*(r_{ij}(\text{iteration}))^2} \quad (4.3)$$
$$\beta_2(\text{iteration})=\beta_0 * e^{-\gamma*(r_{i,gbest}(\text{iteration}))^2} \quad (4.4)$$

α_0 and β_0 are calculated as 0.5 and 0.3, respectively.

We run our new clustering algorithms on five well-known data sets. These data set obtained by UCI Machine Learning Repository also providing total 333 most popular data sets. Our utilised data sets summarized as follows:

Iris dataset: R.A. Fisher donated this multivariate data set July 1, 1988. The data was taken from “https://archive.ics.uci.edu/ml/datasets/Iris”. It consists of 150 variables and 4 attributes, which are sepal-length (cm), sepal-width (cm), petal-length (cm) and petal-width (cm). The original data set includes also species attribute including three iris type Setosa, Versicolour and Virginica. They show classes of datasets. Data set originally gives three iris species information and each one has 50 records. Therefore data set should be separated 3 classes.

Wine dataset: Stefan Aeberhard donated this multivariate data set July 1, 1991. The data was taken from https://archive.ics.uci.edu/ml/datasets/Wine. Dataset consists of 178 variables and 13 attributes, which are Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines and Proline. These data are the results of a chemical analysis of wines grown in the same region in Italy and gives three types of wines. Therefore data set should be separated 3 classes.

Glass dataset: Vina Spiehler donated this multivariate data set September 1, 1987. The data was taken from https://archive.ics.uci.edu/ml/datasets/Glass+Identification. Dataset consists of 214 variables and 9 attributes, which are RI: refractive index, Na: Sodium, Mg: Magnesium, Al: Aluminum, Si: Silicon, K: Potassium, Ca: Calcium, Ba: Barium and Fe: Iron. The original data set includes another attribute that is type of glass "building_windows_float_processed, building_windows_non_float_processed, vehicle_windows_float_processed, vehicle_windows_non_float_processed, containers, tableware and headlamps". They show classes of datasets. Therefore data set should be separated 7 classes.

WDBC dataset: Nick Street donated this multivariate data set October 1, 1995. The data was taken https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic). Dataset consists of 569 variables and 30 attributes, which the first ten attributes are radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, and symmetry and fractal dimension. The original data set has also diagnosis attribute "M = malignant, B = benign". They show classes of datasets. Therefore data set should be separated 2 classes.

Yeast dataset: Paul Horton donated this multivariate data set August 1, 1996. The data was taken from https://archive.ics.uci.edu/ml/datasets/Yeast. Dataset consists of 1484 variables and 8 attributes, which are mcg (McGeoch's method), gvh (Heijne's method), alm (the ALOM), mit (mitochondrial), erl (endoplasmic reticulum lumen), pox (peroxisomal), vac (vacuolar) and nuc (nuclear). Dataset should be separated 10 classes.

4.2. SIMULATION

In simulation part of dissertation, we will introduce how to work of hybrid algorithm with help of Matlab software. As to visualize all data sets for three algorithms will cover a lot of place and unnecessary in study, it will be given just PSO-KMEANS works for Iris and Wine data sets. Additionally, Particles also are visualized separately to see how to move.

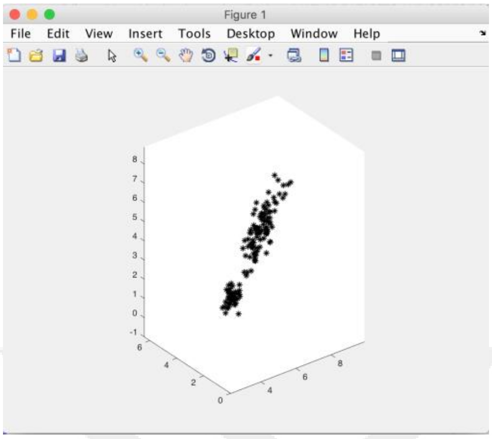

Figure 4. 1. PSO-KMEANS Iris data set at first iteration

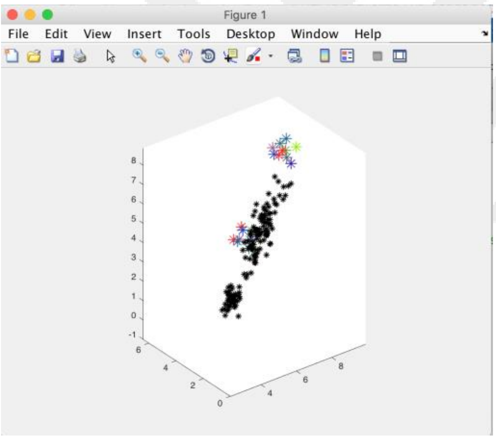

Figure 4. 2. PSO-KMEANS Iris data set at 15th iteration with particles

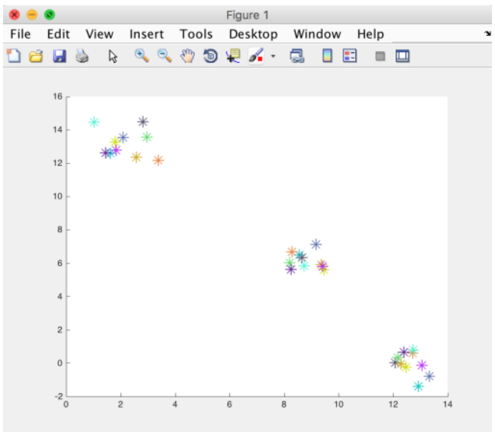

Figure 4. 3. PSO-KMEANS Iris data set particles movement

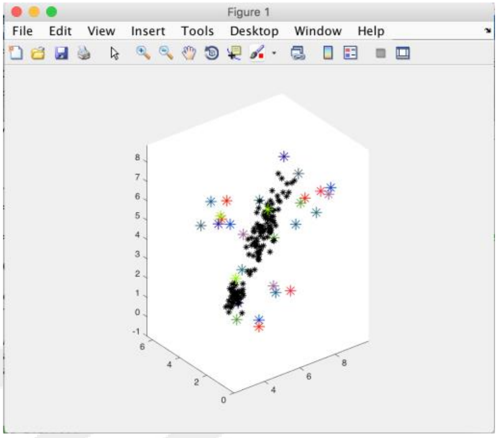

Figure 4. 4. PSO-KMEANS Iris data set at first iteration with particles

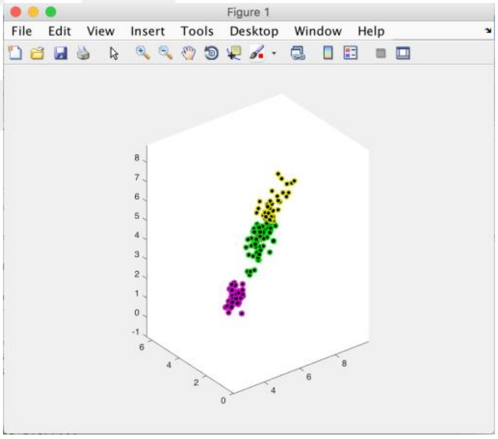

Figure 4. 5. PSO-KMEANS Iris data set clusters

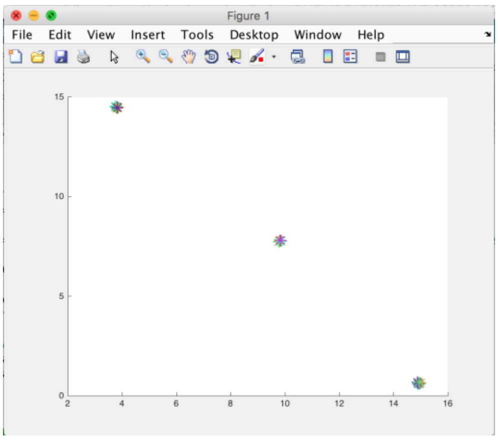

Figure 4. 6. PSO-KMEANS Iris data set particles movement

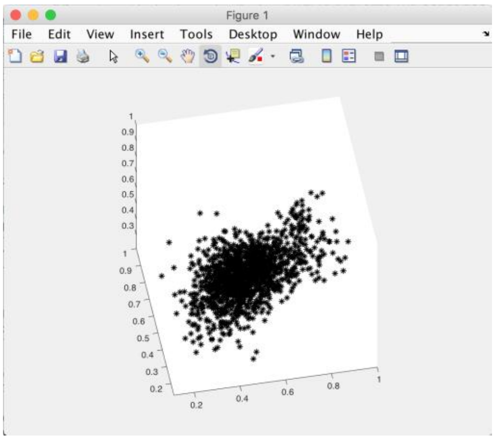

Figure 4. 7. PSO-KMEANS Wine data set at first iteration

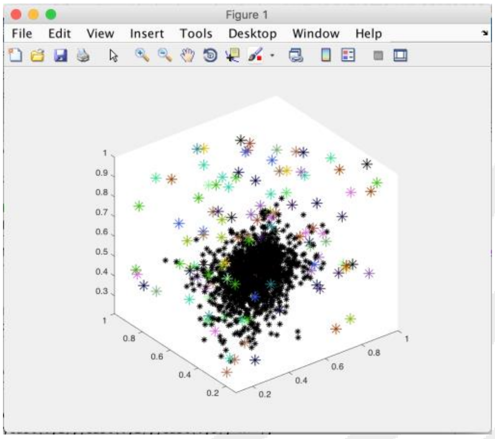

Figure 4. 8. PSO-KMEANS Wine data set at 15th iteration with particles

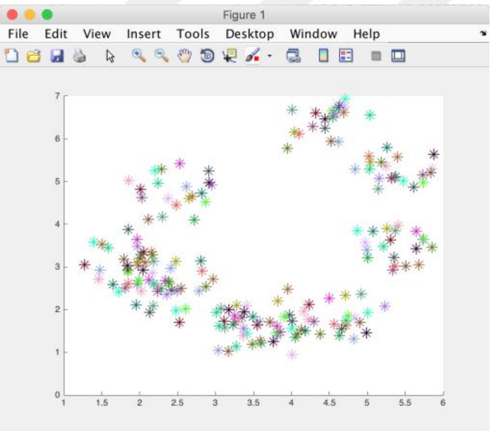

Figure 4. 9. PSO-KMEANS Iris data set particles movement

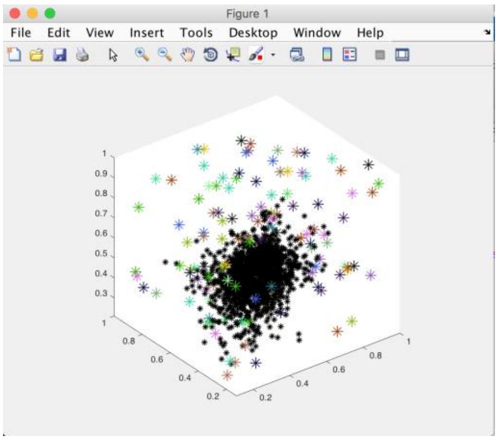

Figure 4. 10. PSO-KMEANS Wine data set at first iteration with particles

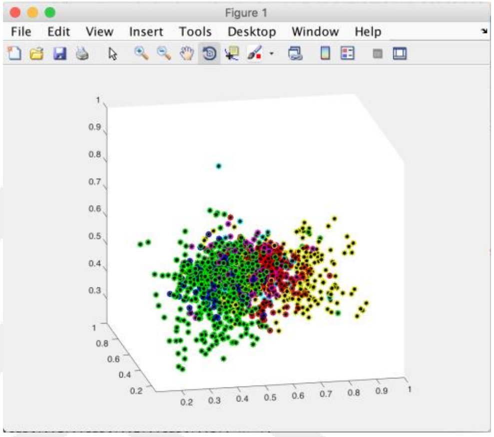

Figure 4. 11. PSO-KMEANS Wine data set clusters

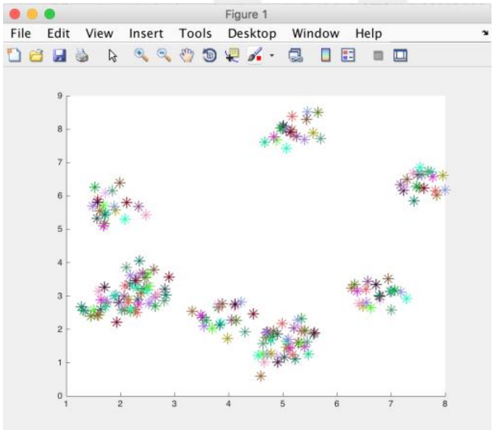

Figure 4. 12. PSO-KMEANS Iris data set particles movement

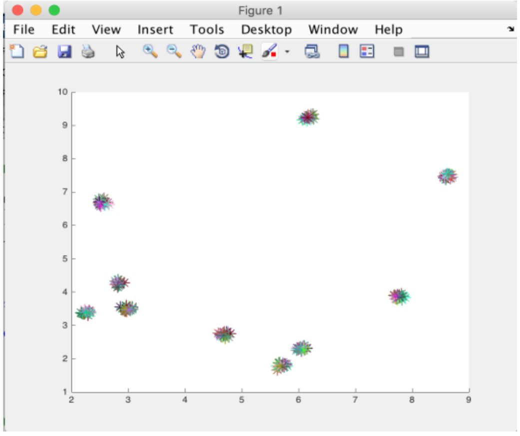

Figure 4. 13. PSO-KMEANS Iris data set particles movement

4.3. RESULTS AND CRITICAL ANALYSIS

While we were analyzing our data sets we used Matlab software, which are fourth generation programming language and stands for **Mat**rix **Lab**oratory. It was proposed by Math Works. We used R2015b (8.6.0.267246) Matlab version prepared for 64-bit (maci64).

After 20 iterations, obtained results in terms of SSE and MSE were tabled as noted below:

IRIS	*SSE*	*MSE*	*CALCULATION TIME (SEC)*
K-MEANS	142.754	0.9517	0.012532
PSO-KMEANS	78.8514	0.5257	2.361729
FA-KMEANS	78.8514	0.5257	2.232143
WINE	*SSE*	*MSE*	*CALCULATION TIME (SEC)*
K-MEANS	2.62932e+06	14771.4337	0.009874
PSO-KMEANS	2370689.6868	13318.4814	2.585321
FA-KMEANS	2370689.6868	13318.4814	3.554762
GLASS	*SSE*	*MSE*	*CALCULATION TIME (SEC)*
K-MEANS	444.739	2.0782	0.010270
PSO-KMEANS	377.3577	1.7634	4.222622
FA-KMEANS	377.3577	1.7634	6.553267
WDBC	*SSE*	*MSE*	*CALCULATION TIME (SEC)*
K-MEANS	5.05168e+07	88781.7784	0.020167
PSO-KMEANS	47264841.9168	83066.5060	5.242084

FA-KMEANS	**47264841.9168**	**83066.5060**	**8.831755**
YEAST	***SSE***	***MSE***	***CALCULATION TIME (SEC)***
K-MEANS	**62.6133**	**0.0422**	**0.036647**
PSO-KMEANS	**71.9003**	**0.0485**	**32.690386**
FA-KMEANS	**45.9827**	**0.0310**	**64.344378**

Table 4. 1. Performance Comparison of K-means, PSO-KMEANS and FA-KMEANS

As you see on the table, three clustering algorithms work well on worked data sets. In this project worked data sets are ranged as their sizes. Therefore we will evaluate them in order of size. Initially, Iris data set is the smallest among five data sets and it was grouped into 3 clusters because data set consists of three different iris species that are Versicolor, Setosa and Virginica. Each of them has 50 records. When we look at the first part of table, the SSE for K-means is 142.754. However, it is 78.8514 for PSO-K-MEANS and FA-K-MEANS. As the both algorithm has similar characteristic, it is a normal that they get nearly same results. On the other hand, MSE represents SSE value for each data point and it is proportional to SSE. Therefore we do not need to evaluate it separately. However they have a difference in the calculation times for all data set. For example, Iris, K-means is very fast, while PSO-KMEANS and FA-KMEANS is much slower.

Our second data set is Wine. It includes chemical analysis of Wines raised in Italy and obtained from three different plants. As a consequence, we tried to split data as these plant species. As we see on the table, a similar situation exists for Wine data set. PSO-KMEANS and FA-KMEANS still gives better results than classic K-means. For this part, results are 2.62932e+06 and 2370689.6868, respectively. Then again, hybrids are still also much slower. It is a gripping situation, FA-KMEANS works also slower than PSO-KMEANS in every time.

Another worked data set is Glass. Glass data set was prepared to investigate criminal cases by using their oxide content (i.e. Fe, Na etc). For this data set, the SSE result of K-means obtained is 444. 739. PSO-KMEANS and FA-KMEANS results are same and much preferable like previous data sets with 377.3577. The time difference between K-means and hybrids are highly clarified in this example

Another popular and large data set worked on is WDBC, which stands for Wisconsin Diagnostic Breast Cancer. As the name suggests, the data set consists of malignant or benign tumor types with 10 features values. Our purpose is to cluster two groups data set as malignant or benign tumor type. When we divided WDBC dataset to two clusters we obtained approximately 50516800 SSE value for K-means, while PSO-KMEANS and FA-KMEANS were 47264841.9168. As long as the data set size expanded, the calculation time of PSO-KMEANS and FA-KMEANS increased. However the calculation time of K-means grew up slightly. This situation can be clearly seen on final data set.

The final data set worked on is Yeast that includes prediction of protein regions in yeast bacteria through 8 bio-statistical tests. It was the largest data set we analyzed and grouped into 10 clusters. These data sets results were surprisingly different from others and emphasized performance difference between PSO-KMEANS and FA-KMEANS. PSO-KMEANS provided the worst clustering SSE value, while FA-KMEANS provides the best result. The SSE values of K-means, PSO-K-MEANS and FA-K-MEANS were 62.6133, 71.9003 and 45.9827, respectively. However the calculation time sorting was totally opposite. K-means was still incredibly fast. On the other hand, PSO-KMEANS is slow but double time faster than FA-KMEANS. Nevertheless, the performance of FA-K-MEANS is non-contestable among others. As a consequence, the size of data sets affects the performance of PSO-K-MEANS. However FA-K-MEANS continues to provide better results than others even if it works progressively much slower.

Chapter 5
CONCLUSION

5.1. CONCLUSIONS

In this dissertation, our main purpose is to demonstrate how nature-inspired algorithms outperform k-clustering methods by introducing two-hybrid methods (PSO-K-MEANS and K-MEANS-FA). The deficiencies existing in the structure of K-means causes some problems. As we emphasized throughout the whole dissertation, the first and most important problem is sensitive initialization of K-means. Some times it might end up with obtaining poor results and partition the data sets. Another important problem is that K-means falls into a trap at local optima.

Two approaches have been proposed to overcome with the restrictions of K-means by originating current hot research topics. They were PSO-KMEANS and FA-KMEANS. These nature inspired algorithms are employed to work with discrete data variables and converge them towards global best. Therefore PSO and FA fit to remove K-means shortcomings. This situation rationalizes integrating K-means with these swarm algorithms.

This dissertation proposes and evaluates performances of the new two integrated clustering algorithms, which are developed by hybridizing to classic K-means clustering with PSO and FA. In proposed methods, it was started with K-means algorithm to seed PSO initial points by using centroids. Then algorithms stepped in to improve centroids and obtain better results. Therefore, K-means was again used by initializing improved centroids by bio-inspired algorithm. These methods applied to five most popular data sets that are IRIS (150 variables, 4 dimension), WINE (178 variables, 13 dimension), GLASS (214 variables, 9 dimension), WDBC (569 variables, 30 dimension) and YEAST (1484 variables, 8 dimension). For each data set, Pure and Hybrid K-means was run for maximum of 20 iterations. The two swarm intelligence optimization approaches were contrasted with classic K-means algorithms. The comparison was done by predicating on the minimum Error Sum of Squares. As time factor is important, the computation time of them was also taken into account.

5.1. LIMITATIONS AND DISCUSSIONS

Even though we got satisfactory outcomes during our study. They do have some limitations.
The first limitation is that PSO-KMEANS did not work proper on large data sets. Second is that FA-KMEANS maintained its performance even on large data sets; however, it is very slow. If we try it on larger data set with over 10.000 variables, it may take over one-day. These properties may reduce the attractiveness of PSO and FA because today data sets are very large sizes.

The experimental results demonstrated that classic K-means, PSO-KMEANS and

FA-KMEANS are effective methods for clustering data in variant size. PSO-KMEANS and FA-KMEANS converged better than K-means and gave lower sum square of error values for all data sets except YEAST. For YEAST data set, only FA-K-means algorithm provided much preferable results. Although the iteration numbers were increased, PSO-K-means still fell behind of K-means in respect to performance. As a consequence, it can be evaluated FA provided a better fit with K-means.

5.3. FUTURE RESEARCH

A main contribution of this dissertation is to provide a basis to study methods hybridizing nature-inspired optimization algorithms with clustering methods. It is naturally predicted that new swarm intelligence techniques will be proposed in the future. Therefore, this research area will extend correspondingly. It can be expected that new bio-inspired optimization algorithms will be more effective than currents. On the other hand researchers might focus to improve PSO and FA further by introducing other capabilities.

BIBLIOGRAPHY

BIBLIOGRAPHY

Abdullah, A., Deris, S., Mohamad, M.S. & Hashim, S.Z.M. 2012, "A New Hybrid Firefly Algorithm for Complex and Nonlinear Problem" in , 2012th edn, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 673-680.

Abraham, A., Das, S. & Roy, S. 2008, "Swarm Intelligence Algorithms for Data Clustering" in Springer US, Boston, MA, pp. 279-313.

Ahmadyfard, A. & Modares, H. 2008, "Combining PSO and k-means to enhance data clustering", IEEE, , pp. 688.

Akyol, S. and Alatas, B. 2012, Guncel Suru Zekasi Optimizasyon Algoritmalari. Nevşehir Üniversitesi Fen Bilimleri Enstitü Dergisi, 1, pp.36-50.

Alweshah, M. & Abdullah, S. 2015, "Hybridizing firefly algorithms with a probabilistic neural network for solving classification problems", Applied Soft Computing, vol. 35, pp. 513-524.

Anon, (2015). [image] Available at: https://idreamofbiochemistry.wordpress.com [Accessed 22 Sep. 2015].

Breaban, M. & Luchian, H. 2011, "PSO aided k-means clustering: introducing connectivity in k-means", ACM, , pp. 1227.

Cura, T. 2011; 2012, "A particle swarm optimization approach to clustering", Expert Systems with Applications, vol. 39, no. 1, pp. 1582.

Ding, J., Shao, J., Huang, Y., Sheng, L., Fu, W. & Li, Y. 2011, "Swarm intelligence based algorithms for data clustering", , pp. 577.

DL Cade, (2014). Murmurations. [image] Available at: http://petapixel.com/2013/11/08/stunning-photographs-starling-murmurations-arent/ [Accessed 22 Sep. 2015].

Du, Z., Wang, Y. & Ji, Z. 2008, "PK-means: A new algorithm for gene clustering", Computational Biology and Chemistry, vol. 32, no. 4, pp. 243-247.

Eberthart , R. & Kennedy, J. 1995, "A new optimizer using particle swarm theory", , pp. 39.

Evers, G.I. & Ben Ghalia, M. 2009, "Regrouping particle swarm optimization: A new global optimization algorithm with improved performance consistency across benchmarks", IEEE , pp. 3901.

Farahani, S.M., Abshouri, A.A., Nasiri, B. & Meybodi, M.R. 2011, "A Gaussian Firefly Algorithm", International Journal of Machine Learning and Computing, vol. 1, no. 5, pp. 448.

Farahani, S. M.; Abshouri, A. A.; Nasiri, B.; Meybodi, M. R. (2012). "Some hybrid models to improve firefly algorithm performance". Int. J. Artificial Intelligence 8 (S12): 97–117.

Hamed Hosseini, Mehdi Shahbazian, and Mohammad Ali Takassi, “The Design of Robust Soft Sensor Using ANFIS Network.” Journal of Instrumentation Technology, vol. 2, no. 1 (2014): 9-16. doi: 10.12691/jit-2-1-3.

Hassanzadeh, T. & Meybodi, M.R. 2012, "A new hybrid approach for data clustering using firefly algorithm and K-means", IEEE , pp. 007.

Janson, S. & Middendorf, M. 2005, "A hierarchical particle swarm optimizer and its adaptive variant", IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 35, no. 6, pp. 1272-1282.

Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R. & Wu, A.Y. 2002, "An efficient k-means clustering algorithm: analysis and implementation", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 881-892.

Karoussi, E. (2015). Data Mining K-Clustering Problem. Postgraduate. University of Agder

Kennedy, J. & Eberthart , R. 1995, "Particle swarm optimization", pp. 1942.

Khan, S.S. & Ahmad, A. 2004, "Cluster center initialization algorithm for K-means clustering", Pattern Recognition Letters, vol. 25, no. 11, pp. 1293-1302.

Koziel, S. & Yang, X. 2011, Computational optimization, methods and algorithms, Springer.

Kucukdeniz, T. 2009, Suru Zekasi Optimizasyon Teknigi Ve Tedarik Zinciri Yonetiminde Bir Uygulama. Ph.D. Istanbul University.

Likas, A., Vlassis, N. & J. Verbeek, J. 2003, "The global k-means clustering algorithm", Pattern Recognition, vol. 36, no. 2, pp. 451-461.

Lu, J.F., Tang, J.B., Tang, Z.M. & Yang, J.Y. 2008, "Hierarchical initialization approach for K-Means clustering", Pattern Recognition Letters, vol. 29, no. 6, pp. 787-795.

Lukasik, S. and Zak, S. (2009) ‘Firefly algorithm for continuous constrained optimization tasks', ICCCI 2009, Lecture Notes in Artificial Intelligence (Eds. N. T. Ngugen, R. Kowalczyk, S. M. Chen), 5796, pp. 97-100.

Yi, F. & Moon, I. 2013, "Extended K-Means Algorithm", IEEE, pp. 263.

MacQueen, J. "Some methods for classification and analysis of multivariate observations", Proc.Fifth Berkeley Symp.on Math.Statist.and Prob., Vol.1 (Univ.of Calif.Press, 1967), 281-297.

Nandy S, Sarkar PP, Das A. 2012, 'Analysis of nature-inspired firefly algorithm-based backpropagation neural network training", Int J Comput Appl; 43(22): 8–16

Parsopoulos, K.E. & Vrahatis, M.N. 2010, Particle swarm optimization and intelligence: advances and applications, Information Science Reference (Isr), US.

Rajini, A. & David, V.K. 2011, A Comparative Performance Study on Hybrid Swarm Model for Micro array Data, Foundation of Computer Science, New York.

Ramsey, L. (2014). [image] Available at: http://lukeramsey.io/clustering-with-K-means [Accessed 22 Sep. 2015].

Redmond, S. and Heneghan, C. (2007). A method for initialising the K-means clustering algorithm using kd-trees. Pattern Recognition Letters, 28(8), pp.965-973.

Shi, Y. & Eberthart, R. 1998, "A modified particle swarm optimizer", pp. 69.

Wang, H., Li, H., Liu, Y., Li, C. & Zeng, S. 2007, "Opposition-based particle swarm algorithm with cauchy mutation", IEEE, pp. 4750.

Wikipedia, (2015). Categorization. [online] Available at: https://en.wikipedia.org/wiki/Categorization [Accessed 3 Oct. 2015].

Wu, J. 2012, Advances in K-means Clustering: A Data Mining Thinking, 1. Aufl.; 1 edn, Springer Berlin Heidelberg, Berlin, Heidelberg.

Xu, R., Wunsch, D.C., IEEE Computational Intelligence Society & Books24x7, I. 2009; 2008, Clustering, 1. Aufl; 1 edn, Wiley, Oxford

Van den Bergh, F. & Engelbrecht, A.P. 2002, "A new locally convergent particle swarm optimiser", pp. 6 pp. vol.3.

Van den Bergh, F. 2002, "An analysis of particle swarm optimizers", ProQuest Dissertations Publishing.

Van der Merwe, D.W. & Engelbrecht, A.P. 2003, "Data clustering using particle swarm optimization", , pp. 215.

VLFeat, (n.d.). K-means clustering of 5000 randomly sampled data points. [image] Available at: http://www.vlfeat.org/overview/K-means.html [Accessed 22 Sep. 2015].

Yang, X. S., 2008, Introduction to computational mathematics. Singapore: World Scientific Pub.

Yang, X. S., 2009, "Firefly algorithms for multimodal optimization", pp. 169.

Yang, X. S., 2014, Nature-inspired optimization algorithms, Elsevier.

Yang, X. S., 2010, Firefly algorithm, Lévy flights and global optimization, in: Research and Development in Intelligent Systems XXVI (Eds M. Bramer, R. Ellis, M. Petridis), Springer London, pp. 209-218.

Yang, X. S., & He, X. 2013, "Firefly Algorithm: Recent Advances and Applications".

Yang, X. S., & Deb, S. 2010, "Eagle Strategy Using Lévy Walk and Firefly Algorithms for Stochastic Optimization" in Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 101-111.

Yusiong, J. 2012, Optimizing Artificial Neural Networks using Cat Swarm Optimization Algorithm. IJISA, 5(1), pp.69-80.

Žalik, K.R. 2008, "An efficient k′-means clustering algorithm", Pattern Recognition Letters, vol. 29, no. 9, pp. 1385-1391.

APPENDICES

Appendix A

1. K-means source code in Matlab

Kmeans IRIS

```
clear;
close all;
tic

centroids = 3;            % centroids
hybrid_pso = true;        % enable/disable hybrid_pso

load fisheriris.mat
dataset_size = size (meas);

% EXECUTE K-MEANS
if hybrid_pso
    fprintf('Running Matlab K-Means Version\n');
    [idx,KMEANS_CENTROIDS,sumd,D] =
kmeans(meas,centroids,'start','uniform','display','iter');
    fprintf('\n');
end

mse_KMEANS=sum(sumd)/150;
fprintf('\nEnd, Mean Square Error of Kmeans is %5.4f\n',
mse_KMEANS);
toc
```

2. PSO-Kmeans source code in Matlab

PSO-Kmeans IRIS

```
clear;
close all;
tic
% INIT PARTICLE SWARM
centroids = 3;            % centroids
dimensions = 4;           % how many dimensions in each centroid
particles = 10;           % how many particles in the swarm, aka how
many solutions
iterations = 20;          % iterations of the optimization alg.
simtime=0.01;             % simulation delay between each iteration
hybrid_pso = true;        % enable/disable hybrid_pso

% GLOBAL PARAMETERS
Min_Inertia_weight=0.4;
Max_Inertia_weight=0.9;
c1 = 1.49; %COGNITIVE
c2 = 1.49; %SOCIAL


% LOAD DEFAULT CLUSTER
load fisheriris.mat
dataset_size = size (meas);
```

```
% EXECUTE K-MEANS
if hybrid_pso
    fprintf('Running Matlab K-Means Version\n');
    [idx,KMEANS_CENTROIDS,sumd,D] =
kmeans(meas,centroids,'start','uniform','display','iter');
    fprintf('\n');
end


% SETTING UP PSO DATA STRUCTURES
swarm_vel = rand(centroids,dimensions,particles)*2;
swarm_best = zeros(centroids,dimensions);
c = zeros(dataset_size(1),particles);
ranges = max(meas)-min(meas); %%scale
swarm_pos = rand(centroids,dimensions,particles);



for iteration=1:iterations

    %CALCULATE EUCLIDEAN DISTANCES TO ALL CENTROIDS
    distances=zeros(dataset_size(1),centroids,particles);
    for particle=1:particles
        for centroid=1:centroids
            distance=zeros(dataset_size(1),1);
            for data_vector=1:dataset_size(1)
                %meas(data_vector,:)
                dis-
tance(data_vector,1)=norm(KMEANS_CENTROIDS(centroid,:)-
meas(data_vector,:));
            end
            distances(:,centroid,particle)=distance;
        end
    end




    %ASSIGN MEASURES with CLUSTERS
    for particle=1:particles
        [value, index] = min(distances(:,:,particle),[],2);
        c(:,particle) = index;
    end





    %CALCULATE GLOBAL FITNESS and LOCAL FITNESS:=swarm_fitness
    average_fitness = zeros(particles,1);
    for particle=1:particles
        for centroid = 1 : centroids
            if any(c(:,particle) == centroid)
                lo-
cal_fitness=mean(distances(c(:,particle)==centroid,centroid,particle
));
                average_fitness(particle,1) = aver-
age_fitness(particle,1) + local_fitness;
            end
        end
        average_fitness(particle,1) = average_fitness(particle,1) /
centroids;
        swarm_fitness(particle) = average_fitness(particle,1);
```

```
        swarm_best(:,:,particle) = swarm_pos(:,:,particle);     %LO-
CAL BEST FITNESS

    end
    [global_fitness, index] = min(swarm_fitness);       %GLOBAL BEST
FITNESS
    swarm_overall_pose = swarm_pos(:,:,index);          %GLOBAL BEST
POSITION



    % SOME INFO ON THE COMMAND WINDOW
    fprintf('%3d. global fitness is
%5.4f\n',iteration,global_fitness);

    pause(simtime);



    % SAMPLE r1 AND r2 FROM UNIFORM DISTRIBUTION [0..1]
    r1 = rand;
    r2 = rand;



    % UPDATE CLUSTER CENTROIDS
    for particle=1:particles
        w(iteration)=((iterations - iteration)*(Max_Inertia_weight -
Min_Inertia_weight))/(iterations-1) + Min_Inertia_weight;
        inertia = w(iteration) * swarm_vel(:,:,particle);
        cognitive = c1 * r1 * (swarm_best(:,:,particle)-
swarm_pos(:,:,particle));
        social = c2 * r2 * (swarm_overall_pose-
swarm_pos(:,:,particle));
        vel = inertia+cognitive+social;

        swarm_pos(:,:,particle) = swarm_pos(:,:,particle) + vel ;
% UPDATE PARTICLE POSE
        swarm_vel(:,:,particle) = vel;
% UPDATE PARTICLE VEL
    end
end

% START UP KMEANS WITH UPDATED CLUSTER CENTROIDS

for particle=1:particles
[idx2,C,sumd2(:,particle)]=kmeans(meas,centroids,'start',swarm_pos(:
,:,particle));


if any(sum(sumd2(:,particle))<sum(sumd))
    sse=sum(sumd2(:,particle));
fprintf('\nEnd, Sum Square Error is %5.4f\n', sse);


end
end


mse_KMEANS=sum(sumd)/150;
fprintf('\nEnd, Mean Square Error of Kmeans is %5.4f\n',
mse_KMEANS);
```

```
mse_HYBRID=sum(sumd2)/150;
fprintf('\nEnd, Mean Square Error of Kmeans and PSO is %5.4f\n',
mse_HYBRID);

toc
toc
```

3. FA-Kmeans source code in Matlab
FA-Kmeans IRIS

```
clear;
close all;
tic
% INIT PARTICLE SWARM
centroids = 3;          % centroids
dimensions = 4;         % how many dimensions in each centroid
particles = 10;         % how many particles in the swarm, aka how
many solutions
iterations = 20;        % iterations of the optimization alg.
simtime=0.01;           % simulation delay between each iteration
hybrid_pso = true;      % enable/disable hybrid_pso
alpha0=0.5;
beta0=0.3;


% LOAD DEFAULT CLUSTER
load fisheriris.mat
dataset_size = size (meas);

% EXECUTE K-MEANS
if hybrid_pso
    fprintf('Running Matlab K-Means Version\n');
    [idx,KMEANS_CENTROIDS,sumd,D] =
kmeans(meas,centroids,'start','uniform','display','iter');
    fprintf('\n');
end




% SETTING UP PSO DATA STRUCTURES
swarm_vel = rand(centroids,dimensions,particles)*2;
swarm_best = zeros(centroids,dimensions,particles);
c = zeros(dataset_size(1),particles);
ranges = max(meas)-min(meas); %%scale


% KMEANS_INIT

swarm_pos = rand(centroids,dimensions,particles);



for iteration=1:iterations

    %CALCULATE EUCLIDEAN DISTANCES TO ALL CENTROIDS
    rdis=zeros(dataset_size(1),centroids,particles);
    for particle=1:particles
```

```
        for centroid=1:centroids
            distance=zeros(dataset_size(1),1);
            for data_vector=1:dataset_size(1)
                %meas(data_vector,:)
               dis-
tance(data_vector,1)=norm(KMEANS_CENTROIDS(centroid,:)-
meas(data_vector,:));
            end
            rdis(:,centroid,particle)=distance;
        end
    end




    %ASSIGN MEASURES with CLUSTERS
    for particle=1:particles
        [value, index] = min(rdis(:,:,particle),[],2);
        c(:,particle) = index;
    end


    %CALCULATE GLOBAL FITNESS and LOCAL FITNESS:=swarm_fitness
    average_fitness = zeros(particles,1);
    for particle=1:particles

        for centroid = 1 : centroids
            if any(c(:,particle) == centroid)
                lo-
cal_fitness=mean(rdis(c(:,particle)==centroid,centroid,particle));
                average_fitness(particle,1) = aver-
age_fitness(particle,1) + local_fitness;
            end
        end

        average_fitness(particle,1) = average_fitness(particle,1) /
centroids;
        swarm_fitness(particle) = average_fitness(particle,1);
        swarm_best(:,:,particle) = swarm_pos(:,:,particle);     %LO-
CAL BEST FITNESS




    end
    [global_fitness, index] = min(swarm_fitness);       %GLOBAL BEST
FITNESS
    swarm_overall_pose = swarm_pos(:,:,index);          %GLOBAL BEST
POSITION


     rdis2=zeros(dataset_size(1),centroids,particles);
    for particle=1:particles
        for centroid=1:centroids
            distance2=zeros(dataset_size(1),1);
            for data_vector=1:dataset_size(1)
                %meas(data_vector,:)
               dis-
tance2(data_vector,1)=norm(swarm_overall_pose(centroid,:)-
meas(data_vector,:));
            end
            rdis2(:,centroid,particle)=distance2;
```

```
        end
    end





 % SOME INFO ON THE COMMAND WINDOW
    fprintf('%3d. global fitness is
%5.4f\n',iteration,global_fitness);

    pause(simtime);




    alpha(iteration)=alpha0.*(0.8.^(iteration));
    beta(iteration)=beta0.*exp(-distance(iteration).^2);
    beta2(iteration)=beta0.*exp(-distance2(iteration).^2);


    % UPDATE CLUSTER CENTROIDS
 for particle=1:particles



swarm_pos(:,:,particle) = swarm_pos(:,:,particle)+...
    (swarm_overall_pose-
swarm_pos(:,:,particle)).*beta2(iteration)+...
    (swarm_best(:,:,particle)-
swarm_pos(:,:,particle)).*beta(iteration) +alpha(iteration).*(rand-
0.5);




 end







end

for particle=1:particles
[idx2,C,sumd2(:,particle)]=kmeans(meas,centroids,'dist','sqEuclidean
','start',swarm_pos(:,:,particle));


if any(sum(sumd2(:,particle))<sum(sumd))
    sse=sum(sumd2(:,particle));
fprintf('\nEnd, Sum Square Error is %5.4f\n', sse);


end
end
```

```
mse_KMEANS=sum(sumd)/150;
fprintf('\nEnd, Mean Square Error of Kmeans is %5.4f\n',
mse_KMEANS);
mse_HYBRID=sum(sumd2)/150;
fprintf('\nEnd, Mean Square Error of Kmeans and PSO is %5.4f\n',
mse_HYBRID);
toc
```

Appendix B

1. Experimental Results Data

1.1. KMEANS

KMEANS_iris

Running Matlab K-Means Version

iter	phase	num	sum
1	1	150	151.01
2	1	7	146.368
3	1	5	144.721
4	1	3	144.333
5	1	4	143.924
6	1	3	143.61
7	1	1	143.542
8	1	2	143.414
9	1	2	143.023
10	1	2	142.823
11	1	1	142.786
12	1	1	142.754
13	2	0	142.754

Best total sum of distances = 142.754

End, Mean Square Error of Kmeans is 0.9517

Elapsed time is 0.012532 seconds.

KMEANS_wine

Running Matlab K-Means Version

iter	phase	num	sum
1	1	178	7.10528e+06
2	1	25	3.6457e+06
3	1	15	2.9532e+06
4	1	5	2.7907e+06
5	1	5	2.71311e+06
6	1	1	2.70573e+06
7	1	1	2.70378e+06
8	1	5	2.66128e+06
9	1	3	2.64323e+06
10	1	1	2.64054e+06
11	1	1	2.63794e+06
12	1	1	2.6367e+06

13	1	1	2.63356e+06
14	2	0	2.62932e+06

Best total sum of distances = 2.62932e+06

End, Mean Square Error of Kmeans is 14771.4337

Elapsed time is 0.009874 seconds.

KMEANS_glass

Running Matlab K-Means Version

iter	phase	num	sum
1	1	214	771.081
2	1	40	466.405
3	1	4	456.641
4	1	3	452.647
5	1	2	448.686
6	2	1	444.744
7	2	0	444.739

Best total sum of distances = 444.739

End, Mean Square Error of Kmeans is 2.0782

Elapsed time is 0.010270 seconds.

KMEANS_wdbc

Running Matlab K-Means Version

iter	phase	num	sum
1	1	569	8.24381e+07
2	1	102	5.86589e+07
3	1	23	5.41095e+07
4	1	17	5.26316e+07
5	1	15	5.21595e+07
6	1	16	5.17022e+07
7	1	16	5.10842e+07
8	1	11	5.07527e+07
9	1	4	5.06889e+07
10	1	4	5.06379e+07
11	1	5	5.0592e+07
12	1	3	5.05534e+07
13	1	4	5.05246e+07
14	1	1	5.05178e+07
15	2	0	5.05168e+07

Best total sum of distances = 5.05168e+07

End, Mean Square Error of Kmeans is 88781.7784

Elapsed time is 0.020167 seconds.

KMEANS_yeast

Running Matlab K-Means Version

iter	phase	num	sum
1	1	1484	96.7376
2	1	391	78.1299
3	1	216	73.9324

4	1	154	71.1086
5	1	93	70.1455
6	1	55	69.8341
7	1	26	69.7563
8	1	22	69.7025
9	1	22	69.6453
10	1	31	69.4725
11	1	60	68.8838
12	1	71	67.6788
13	1	100	65.3099
14	1	88	63.4932
15	1	46	62.911
16	1	25	62.7443
17	1	20	62.6622
18	1	13	62.6351
19	1	3	62.6299
20	1	1	62.6294
21	1	1	62.6288
22	1	2	62.6273
23	2	1	62.6136
24	2	0	62.6133

Best total sum of distances = 62.6133

End, Mean Square Error of Kmeans is 0.0422

Elapsed time is 0.036647 seconds.

1.2. PSO-KMEANS

PSO_KMEANS_iris
Running Matlab K-Means Version

iter	phase	num	sum
1	1	150	308.751
2	1	21	149.373
3	1	11	144.752
4	1	3	143.349
5	1	2	142.786
6	1	1	142.754
7	2	0	142.754

Best total sum of distances = 142.754

1. global fitness is 0.7158
2. global fitness is 0.7158
3. global fitness is 0.7158
4. global fitness is 0.7158
5. global fitness is 0.7158
6. global fitness is 0.7158
7. global fitness is 0.7158
8. global fitness is 0.7158
9. global fitness is 0.7158
10. global fitness is 0.7158
11. global fitness is 0.7158

12. global fitness is 0.7158
13. global fitness is 0.7158
14. global fitness is 0.7158
15. global fitness is 0.7158
16. global fitness is 0.7158
17. global fitness is 0.7158
18. global fitness is 0.7158
19. global fitness is 0.7158
20. global fitness is 0.7158

End, Sum Square Error is 78.8514

End, Mean Square Error of Kmeans is 0.9517

End, Mean Square Error of Kmeans and PSO is 0.5257

Elapsed time is 2.361729 seconds.

PSO_KMEANS_wine
Running Matlab K-Means Version

iter	phase	num	sum
1	1	178	4.69646e+06
2	1	10	3.73059e+06
3	1	7	3.4782e+06
4	1	9	3.24154e+06
5	1	8	2.86256e+06
6	1	6	2.7455e+06
7	1	3	2.70573e+06
8	1	1	2.70378e+06
9	1	5	2.66128e+06
10	1	3	2.64323e+06
11	1	1	2.64054e+06
12	1	1	2.63794e+06
13	1	1	2.6367e+06
14	1	1	2.63356e+06
15	2	0	2.62932e+06

Best total sum of distances = 2.62932e+06

1. global fitness is 105.7115
2. global fitness is 105.7115
3. global fitness is 105.7115
4. global fitness is 105.7115
5. global fitness is 105.7115
6. global fitness is 105.7115
7. global fitness is 105.7115
8. global fitness is 105.7115
9. global fitness is 105.7115
10. global fitness is 105.7115
11. global fitness is 105.7115
12. global fitness is 105.7115
13. global fitness is 105.7115
14. global fitness is 105.7115
15. global fitness is 105.7115
16. global fitness is 105.7115

17. global fitness is 105.7115
18. global fitness is 105.7115
19. global fitness is 105.7115
20. global fitness is 105.7115

End, Sum Square Error is 2370689.6868

End, Mean Square Error of Kmeans is 14771.4337

End, Mean Square Error of Kmeans and PSO is 13318.4814

Elapsed time is 2.585321 seconds.

PSO_KMEANS_glass
Running Matlab K-Means Version

iter	phase	num	sum
1	1	214	897.531
2	1	31	665.836
3	1	26	481.9
4	1	7	461.858
5	1	4	456.704
6	1	3	453.361
7	1	2	451.983
8	1	1	450.249
9	1	1	448.91
10	1	1	447.133
11	2	0	444.739

Best total sum of distances = 444.739

1. global fitness is 1.1993
2. global fitness is 1.1993
3. global fitness is 1.1993
4. global fitness is 1.1993
5. global fitness is 1.1993
6. global fitness is 1.1993
7. global fitness is 1.1993
8. global fitness is 1.1993
9. global fitness is 1.1993
10. global fitness is 1.1993
11. global fitness is 1.1993
12. global fitness is 1.1993
13. global fitness is 1.1993
14. global fitness is 1.1993
15. global fitness is 1.1993
16. global fitness is 1.1993
17. global fitness is 1.1993
18. global fitness is 1.1993
19. global fitness is 1.1993
20. global fitness is 1.1993

End, Sum Square Error is 377.3577

End, Mean Square Error of Kmeans is 2.0782

End, Mean Square Error of Kmeans and PSO is 1.7634

Elapsed time is 4.222622 seconds.

PSO_KMEANS_wdbc
Running Matlab K-Means Version

iter	phase	num	sum
1	1	569	1.34192e+08
2	1	159	7.08446e+07
3	1	84	5.38038e+07
4	1	44	5.00987e+07
5	1	15	4.91516e+07
6	1	3	4.90971e+07
7	1	5	4.89576e+07
8	1	6	4.86669e+07
9	1	5	4.85474e+07
10	1	4	4.84707e+07
11	1	5	4.8371e+07
12	1	3	4.82814e+07
13	1	5	4.80761e+07
14	1	2	4.79836e+07
15	1	3	4.79302e+07
16	1	3	4.78613e+07
17	1	2	4.78263e+07
18	1	4	4.76644e+07
19	1	4	4.74999e+07
20	2	0	4.74999e+07

Best total sum of distances = 4.74999e+07

1. global fitness is 341.1832
2. global fitness is 341.1832
3. global fitness is 341.1832
4. global fitness is 341.1832
5. global fitness is 341.1832
6. global fitness is 341.1832
7. global fitness is 341.1832
8. global fitness is 341.1832
9. global fitness is 341.1832
10. global fitness is 341.1832
11. global fitness is 341.1832
12. global fitness is 341.1832
13. global fitness is 341.1832
14. global fitness is 341.1832
15. global fitness is 341.1832
16. global fitness is 341.1832
17. global fitness is 341.1832
18. global fitness is 341.1832
19. global fitness is 341.1832
20. global fitness is 341.1832

End, Sum Square Error is 47264841.9168

End, Mean Square Error of Kmeans is 83479.5294

End, Mean Square Error of Kmeans and PSO is 83066.5060

Elapsed time is 5.242084 seconds.

PSO_KMEANS_yeast
Running Matlab K-Means Version

iter	phase	num	sum
1	1	1484	96.7376
2	1	391	78.1299
3	1	216	73.9324
4	1	154	71.1086
5	1	93	70.1455
6	1	55	69.8341
7	1	26	69.7563
8	1	22	69.7025
9	1	22	69.6453
10	1	31	69.4725
11	1	60	68.8838
12	1	71	67.6788
13	1	100	65.3099
14	1	88	63.4932
15	1	46	62.911
16	1	25	62.7443
17	1	20	62.6622
18	1	13	62.6351
19	1	3	62.6299
20	1	1	62.6294
21	1	1	62.6288
22	1	2	62.6273
23	2	1	62.6136
24	2	0	62.6133

Best total sum of distances = 62.6133

1. global fitness is 0.1440
2. global fitness is 0.1440
3. global fitness is 0.1440
4. global fitness is 0.1440
5. global fitness is 0.1440
6. global fitness is 0.1440
7. global fitness is 0.1440
8. global fitness is 0.1440
9. global fitness is 0.1440
10. global fitness is 0.1440
11. global fitness is 0.1440
12. global fitness is 0.1440
13. global fitness is 0.1440
14. global fitness is 0.1440
15. global fitness is 0.1440
16. global fitness is 0.1440
17. global fitness is 0.1440
18. global fitness is 0.1440
19. global fitness is 0.1440
20. global fitness is 0.1440

End, Sum Square Error is 71.9003

End, Mean Square Error of Kmeans is 0.0422

End, Mean Square Error of Kmeans and PSO is 0.0485

Elapsed time is 32.690386 seconds.

1.3. FA-KMEANS

FIREFLY_KMEANS_iris
Running Matlab K-Means Version

iter	phase	num	sum
1	1	150	172.232
2	1	14	146.355
3	1	6	144.721
4	1	3	144.333
5	1	4	143.924
6	1	3	143.61
7	1	1	143.542
8	1	2	143.414
9	1	2	143.023
10	1	2	142.823
11	1	1	142.786
12	1	1	142.754
13	2	0	142.754

Best total sum of distances = 142.754

1. global fitness is 0.7158
2. global fitness is 0.7158
3. global fitness is 0.7158
4. global fitness is 0.7158
5. global fitness is 0.7158
6. global fitness is 0.7158
7. global fitness is 0.7158
8. global fitness is 0.7158
9. global fitness is 0.7158
10. global fitness is 0.7158
11. global fitness is 0.7158
12. global fitness is 0.7158
13. global fitness is 0.7158
14. global fitness is 0.7158
15. global fitness is 0.7158
16. global fitness is 0.7158
17. global fitness is 0.7158
18. global fitness is 0.7158
19. global fitness is 0.7158
20. global fitness is 0.7158

End, Sum Square Error is 78.8514

End, Mean Square Error of Kmeans is 0.9517

End, Mean Square Error of Kmeans and PSO is 0.5257

Elapsed time is 2.232143 seconds.

FIREFLY_KMEANS_wine
Running Matlab K-Means Version

iter	phase	num	sum
1	1	178	2.87638e+06
2	1	6	2.67951e+06
3	1	2	2.64923e+06
4	1	1	2.64323e+06
5	1	1	2.64054e+06
6	1	1	2.63794e+06
7	1	1	2.6367e+06
8	1	1	2.63356e+06
9	2	0	2.62932e+06

Best total sum of distances = 2.62932e+06

1. global fitness is 105.7115
2. global fitness is 105.7115
3. global fitness is 105.7115
4. global fitness is 105.7115
5. global fitness is 105.7115
6. global fitness is 105.7115
7. global fitness is 105.7115
8. global fitness is 105.7115
9. global fitness is 105.7115
10. global fitness is 105.7115
11. global fitness is 105.7115
12. global fitness is 105.7115
13. global fitness is 105.7115
14. global fitness is 105.7115
15. global fitness is 105.7115
16. global fitness is 105.7115
17. global fitness is 105.7115
18. global fitness is 105.7115
19. global fitness is 105.7115
20. global fitness is 105.7115

End, Sum Square Error is 2370689.6868

End, Mean Square Error of Kmeans is 14771.4337

End, Mean Square Error of Kmeans and FA is 13318.4814

Elapsed time is 3.554762 seconds.

FIREFLY_KMEANS_glass
Running Matlab K-Means Version

iter	phase	num	sum

1	1	214	834.383
2	1	27	578.162
3	1	13	448.023
4	1	2	444.759
5	2	0	444.739

Best total sum of distances = 444.739

1. global fitness is 1.1993
2. global fitness is 1.1993
3. global fitness is 1.1993
4. global fitness is 1.1993
5. global fitness is 1.1993
6. global fitness is 1.1993
7. global fitness is 1.1993
8. global fitness is 1.1993
9. global fitness is 1.1993
10. global fitness is 1.1993
11. global fitness is 1.1993
12. global fitness is 1.1993
13. global fitness is 1.1993
14. global fitness is 1.1993
15. global fitness is 1.1993
16. global fitness is 1.1993
17. global fitness is 1.1993
18. global fitness is 1.1993
19. global fitness is 1.1993
20. global fitness is 1.1993

End, Sum Square Error is 377.3577

End, Mean Square Error of Kmeans is 2.0782

End, Mean Square Error of Kmeans and FA is 1.7634

Elapsed time is 6.553267 seconds.

FIREFLY_KMEANS_wdbc
Running Matlab K-Means Version

iter	phase	num	sum
1	1	569	8.87784e+07
2	1	66	5.53916e+07
3	1	36	4.87892e+07
4	1	10	4.8319e+07
5	1	5	4.80761e+07
6	1	2	4.79836e+07
7	1	3	4.79302e+07
8	1	3	4.78613e+07
9	1	2	4.78263e+07
10	1	4	4.76644e+07
11	1	4	4.74999e+07
12	2	0	4.74999e+07

Best total sum of distances = 4.74999e+07

1. global fitness is 341.1832

2. global fitness is 341.1832
3. global fitness is 341.1832
4. global fitness is 341.1832
5. global fitness is 341.1832
6. global fitness is 341.1832
7. global fitness is 341.1832
8. global fitness is 341.1832
9. global fitness is 341.1832
10. global fitness is 341.1832
11. global fitness is 341.1832
12. global fitness is 341.1832
13. global fitness is 341.1832
14. global fitness is 341.1832
15. global fitness is 341.1832
16. global fitness is 341.1832
17. global fitness is 341.1832
18. global fitness is 341.1832
19. global fitness is 341.1832
20. global fitness is 341.1832

End, Sum Square Error is 47264841.9168

End, Mean Square Error of Kmeans is 83479.5294

End, Mean Square Error of Kmeans and FA is 83066.5060

Elapsed time is 8.831755 seconds.

FIREFLY_KMEANS_yeast
Running Matlab K-Means Version

iter	phase	num	sum
1	1	1484	97.1437
2	1	345	70.5087
3	1	217	60.7833
4	1	169	57.5073
5	1	142	55.415
6	1	108	54.2081
7	1	82	53.4016
8	1	48	52.989
9	1	38	52.6955
10	1	30	52.5812
11	1	21	52.515
12	1	19	52.4681
13	1	26	52.3723
14	1	21	52.3146
15	1	18	52.2605
16	1	13	52.2214
17	1	9	52.2013
18	1	7	52.1847
19	1	6	52.1659
20	1	5	52.1547
21	1	4	52.1524
22	1	3	52.1493

23	1	5	52.1026
24	1	7	52.0862
25	1	4	52.0828
26	1	3	52.0819
27	1	3	52.0809
28	1	3	52.0793
29	1	3	52.0755
30	1	3	52.0733
31	1	3	52.0715
32	1	3	52.07
33	1	1	52.0699
34	1	4	52.0665
35	1	5	52.0615
36	1	8	52.0527
37	1	6	52.0437
38	1	5	52.0346
39	2	1	51.7664
40	2	1	51.7609
41	2	1	51.7367
42	2	1	51.6458
43	2	1	51.5202
44	2	1	51.3831
45	2	1	51.2012
46	2	1	51.0744
47	2	1	50.9753
48	2	1	50.9136
49	2	1	50.8846
50	2	1	50.8319
51	2	1	50.7986
52	2	1	50.7854
53	2	1	50.7817
54	2	1	50.7739
55	2	1	50.7026
56	2	1	50.5465
57	2	1	50.252
58	2	1	50.1257
59	2	1	50.083
60	2	1	50.0521
61	2	1	50.031
62	2	0	50.0293

Best total sum of distances = 50.0293

1. global fitness is 0.1686
2. global fitness is 0.1686
3. global fitness is 0.1686
4. global fitness is 0.1686
5. global fitness is 0.1686
6. global fitness is 0.1686
7. global fitness is 0.1686
8. global fitness is 0.1686
9. global fitness is 0.1686
10. global fitness is 0.1686
11. global fitness is 0.1686
12. global fitness is 0.1686

13. global fitness is 0.1686
14. global fitness is 0.1686
15. global fitness is 0.1686
16. global fitness is 0.1686
17. global fitness is 0.1686
18. global fitness is 0.1686
19. global fitness is 0.1686
20. global fitness is 0.1686

End, Sum Square Error is 45.9827

End, Mean Square Error of Kmeans is 0.0337

End, Mean Square Error of Kmeans and FA is 0.0310

Elapsed time is 64.344378 seconds.