International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-2, July 2019

Integrating Bat Algorithm to Inverse Weighted K-means

Mohammed Alhanjouri, Ahmed Alghoul

Abstract: ***Inverse Weighted K-means less sensitive to poor initialization than the traditional K-means algorithm. Therefore, this paper introduce a new hybrid algorithm that integrates inverse weighted k-means algorithm with the optimization bat algorithm, which takes the advantages of both algorithms, from one side the quick convergence and the best global fitness values that obtained from using the bat algorithm and from other side the best clustering results that obtained from inverse weighted k-means algorithm. Moreover, to discuss in deeply the best choices of numerator and denominator powers to get best cluster integrity by getting the best value of cost function by comparing the results of the new algorithm with the inverse weighted k-means algorithm. Improved outcomes were achieved using the new hybrid algorithm.***

Index Terms: Bat algorithm, Inverse weighted k-means, k-means, optimization technique, Clustering.

I. INTRODUCTION

K-means clustering is one of the famous and the simplest clustering algorithms, in which is based on separating the data to groups or clusters, conditioned on there is a high similarity between the data in the same group and fewer similarities among other groups. This separating is continuously recalculating and relocating the nearest centroid for the data, and this leads to fine-tuning the shape of the clusters.

However, K-means clustering has two main problems. The first one is its performance depends on the starting position of the randomly generated initial centroids. The second one is the falling into local optima that do not give the best results; so, to get the best result we need to run the algorithm many times, and this is a computationally exhausting procedure. A heuristic approach was one approach to solve these problems using a variation of the K-means clusters or an optimization algorithm beside Density Based Spatial Clustering for applications of noise (DBSCAN) [1].

Inverse Weighted K-means (IWKM) is one of the variations of the K-means clustering algorithm in which has two modifications in calculating the performance function and locating the centroid for each cluster. This algorithm succeeded to solve the initial centroid position problem that was faced in the K-means algorithm [2]. However, it does not converge quickly to get the best fitness value, also there is not much researching especially on choosing the powers of numerator and denominator in objective function as will be shown below.

The meta-heuristic optimization algorithms or bio-inspired optimization algorithms are used with K-means to solve their problems, such as Bat algorithm, Cuckoo algorithm ...etc. These algorithms imitate the swarm behavior among groups in which they move toward a common goal to find the best solution. Due to these characteristics, the meta-heuristic optimization algorithms got a lot of attention from the researchers. Consequently, many researchers used hybrid combining algorithm such meta-heuristic optimization algorithm with data mining algorithms to get better results and solve the problems in the data mining algorithms.

The aim of this study is to integrate the bio-inspired optimization algorithms mainly the Bat algorithm (BA) with IWKM, called BIWKM, to overcome some of the limitations in the IWKM and to validate the effectiveness the performance of the BIWKM hybrid algorithm over IWKM. Many research papers combined Bat Algorithm with clustering algorithms, as [3], [4], [5], besides that Bat algorithm combined especially with K-means clustering algorithm [6], [7] which got better results than K-means clustering algorithm. Nonetheless, there no research paper combined bio-inspired optimization algorithms with IWKM algorithm [2],[8]. So, in this paper, we will combine the BA with IWKM algorithm to clustering the data and comparing the results with IWKM using artificial and real-life datasets. Additionally, the new hybrid BIWKM algorithm will be compared with the existing IWKM using the same datasets.

II. BAT ALGORITHM

Bats depend on a type of sonar called echolocation to determine the location of prey and avoid obstacles. They produce sound pulses while flying and listen to their reflected sound from surrounding objects to discover their locations and the type of them. The loudness of bats can reach to the highest value when they search for prey while decreasing to the lowest when they find prey and fly toward it.

The following assumptions are used to idealize applying algorithm [9]:

- All bats use echolocation to estimate distance, and they can distinguish between food/prey and the obstacles.

Revised Manuscript Received on 30 July 2019.
***** Correspondence Author
Mohammed A. Alhanjouri*, Computer Engineering Department, Islamic University of Gaza, Gaza, Palestine.
Ahmed Alghoul, Computer Engineering Department, Islamic University of Gaza, Gaza, Palestine.

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license http://creativecommons.org/licenses/by-nc-nd/4.0/

- Each bat flies randomly with velocity v_i at location x_i with a constant frequency of range f_{min} to f_{max} changing wavelength λ and the loudness A to find a prey. They can automatically change its frequency or wavelength of its emitted pulses and adjust the emission
- pulse rate r ∈ [0,1] according to the nearness to its target prey.
- The loudness can be ranged from a big positive value A_0 to a small defined valueA_{min}.

As the frequency range from [f_{min}, f_{max}], the corresponding wavelength range from [$\lambda_{min}, \lambda_{max}$]. So, if the used frequency is higher, the corresponding wavelength is shorter and the traveling distance is shorter. Because the frequency and wavelength are related then we fixed wavelength and changing frequency. Thus, for simplicity, we assume frequency ∈ [0 , f_{max}]. The pulse rate r is in the range of [0,1] where 0 means no pulses, and 1 means the maximum rate of pulse emission.

The BA algorithm has four main steps. These steps are discussed below:

The first step: Initializing the BA algorithm parameters

Each bat's position in the BA represents a possible solution to an optimization problem. A bat's position is represented by a solution (bat) x = ($x_1, x_2, x_i, \dots, x_D$) where x_i represented the decision variable on index i and D represents the dimensional space of x solution. The objective function to evaluate any solution x is represented by f(x).

Also, the BA parameters have to be initialized in this step as follow:

- N: The population size of bats
- f_{min} and f_{max}: The minimum and the maximum frequency range which is used to choose a specific value of frequency f_j that help to determine the velocity and location of a specific bat in a specific time.
- v_j: Velocity vector for each bat to alter its current position by adding a value to generate a new location that is nearer to the best bat position.
- A_j: Loudness vector values for all bats the help to updating the current position
- r_j: Pulse rate vector for all bats

The second step: Initializing the bat population matrix:

The BA starts with random generation bat population matrix of size N × D, which N is the bat population and D is the number of decision variables. The sets of bat location vectors are the contents of this matrix. Also, in this step, we calculate the objective function value for every solution (bat) and the best global bat location that corresponding to the best objective function value is stored in x_{Gbest} variable.

The Bat population matrix of size N and dimension D is

$$\begin{bmatrix} x_{1,1} & \cdots & x_{1,D} \\ \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,D} \end{bmatrix}.$$

The Third step: Adjusting the current bat population:

Each bat x_j flies with velocity v_j that affected by its randomly generated frequency f_j. Then the new bat position is calculated as:

$$f_j = f_{min} + (f_{max} - f_{min}) \times U(0,1) \quad (1)$$

$$v'_{j,i} = v_{j,i} + (x_{j,i} - x_{Gbest,i}) \times f_j \quad (2)$$

$$x'_{j,i} = x_{j,i} + v'_{j,i} \quad (3)$$

For the local search, the bat generates new position through local search strategy with random walk:

$$x'_{j,i} = x_{best,i} + \in A^{(t)} \quad (4)$$

Where $x_{best,i}$ is the best local position for the bat j, $A^{(t)}$ is the mean of all loudness for all bats at current iteration t, ∈ is a random number in the range [-1, 1].

To summarize the new bat position:

if $U(0.1) > r_j$

$$x'_{j,i} = x_{best,i} + \in A^{(t)}$$

else

$$x'_{j,i} = x_{j,i} + v'_{j,i}$$

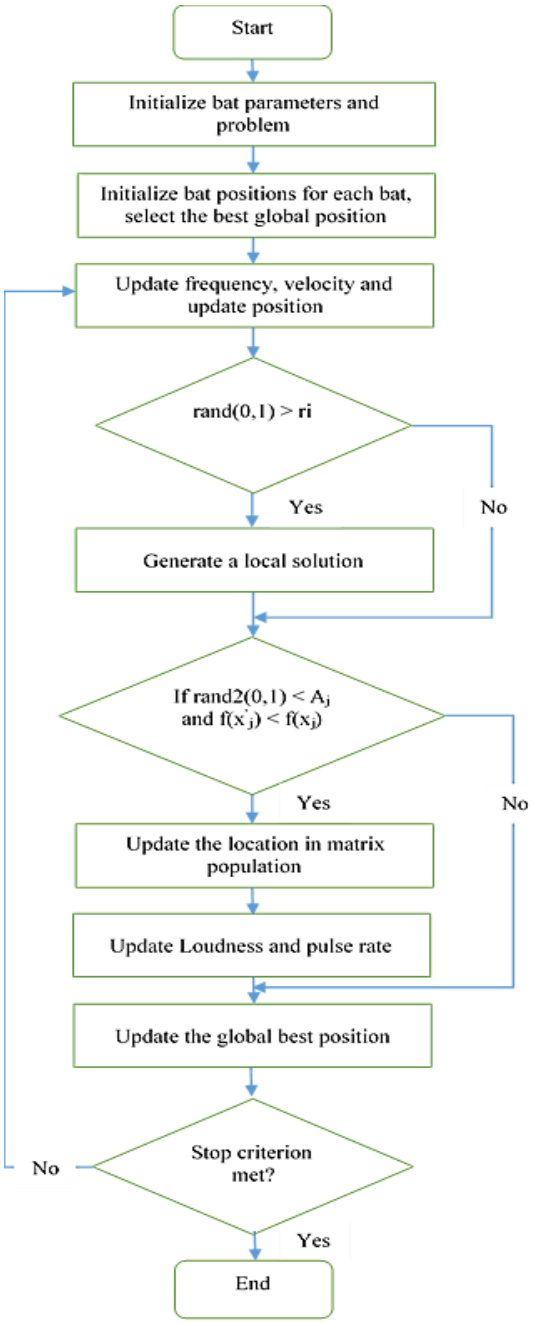


Figure 1. Flowchart of bat algorithm

Retrieval Number: B3564078219/19©BEIESP
DOI: 10.35940/ijrte.B3564.078219
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

The $x_{j,i}$ is replaced by the new bat position $x'_{j,i}$ if the $x'_{j,i}$ is better or not more loudness than the current A_j. The global best position $\boldsymbol{x_{Gbest}}$ is altered if the $f(\boldsymbol{x'_j}) < f(\boldsymbol{x_{Gbest}})$.
The fourth step: The termination in which the algorithm is repeated until the termination criterion is met. Figure 1 shows the flowchart of the Bat algorithm [9].

III. INVERSE WEIGHTED K-MEANS ALGORITHM (IWKM)

It is introduced to solve the initial starting centroids position in the K-means clustering algorithm by using two modifications to the original K-means clustering algorithm in calculating the cost function [10]. The first one is using the sum of the inverse of distances from a specific data point to every centroid, and the second one is multiplication the inverse sum by weight which is the minimum distance of the same data point to the centroids as shown in (5) for the cost function. Where m_j is the centroid in the jth cluster, K is the maximum number of clusters, m_l is the centroid that has the minimum distance to the s_i data point, and N is the number of data points in all clusters. IWKM calculate the centroid in a way that differ from how K-means calculate it as shown in (6), where V_l is the set of indices of data points that have the minimum distance to m_l centroid and V_j is the complement of V_l.

$$J_1 = \sum_{i=1}^{N}\left[\sum_{j=1}^{K}\frac{1}{\|s_i - m_j\|^p}\right]\min_{j=1}^{K}\|s_i - m_j\|^n \quad (5)$$

$$m_l = \frac{\sum_{i\in V_l} s_i a_{il} + \sum_{i\in V_{j,j\neq l}} s_i b_{il}}{\sum_{i\in V_l} a_{il} + \sum_{i\in V_{j,j\neq l}} b_{il}} \quad (6)$$

$$a_{il} = -(n-p) * \|s_i - m_l\|^{n-p-2} - n * \|s_i - m_l\|^{n-2} * \left(\sum_{j=1}^{K}\|s_i - m_j\|^{-p}\right) \quad (7)$$

$$b_{ij} = p * \frac{\|s_i - m_l\|^n}{\|s_i - m_j\|^{p+2}} \quad (8)$$

As shown in (6) the centroid calculated by a way that has a relationship between all data points that have the minimum distance to this centroid and with other clusters that do not have the minimum distance to their data points, accordingly IWKM behaves in a better way than standard K-means. Figure 2 shows a flowchart of how IWKM algorithm used to calculate the centroids, also this algorithm depends on two basic steps for computing the centroids. The first one is computing the value of ail in equation (7) and the second one is computing the value if b_{ij} in equation (8). Moreover, the flowchart shows the convergence done and the algorithm stops running when the difference between the current position of the centroid and the previous position is small. IWKM algorithm changes the position of the centroids gradually until they become approximately at the center of the cluster for each centroid, besides that the algorithm not falling into local optima. Nevertheless, we will see in the results and experiment section the value of the objective function cost, and the number of iterations to converge can be reduced using the BIWKM hybrid algorithm.

IV. INTEGRATING BIO-INSPIRED BAT ALGORITHM INTO IWKM (BIWKM HYBRID) ALGORITHM

The BIWKM hybrid algorithm is a new algorithm that integrates BA with IWKM algorithm to get the best clustering result in calculating objective function and to reduce the number of iterations to converge.

Likewise, both of IWKM and BA use the same formulas as in (5) and (6) to calculate the cost function and determine the position of the centroid respectively.

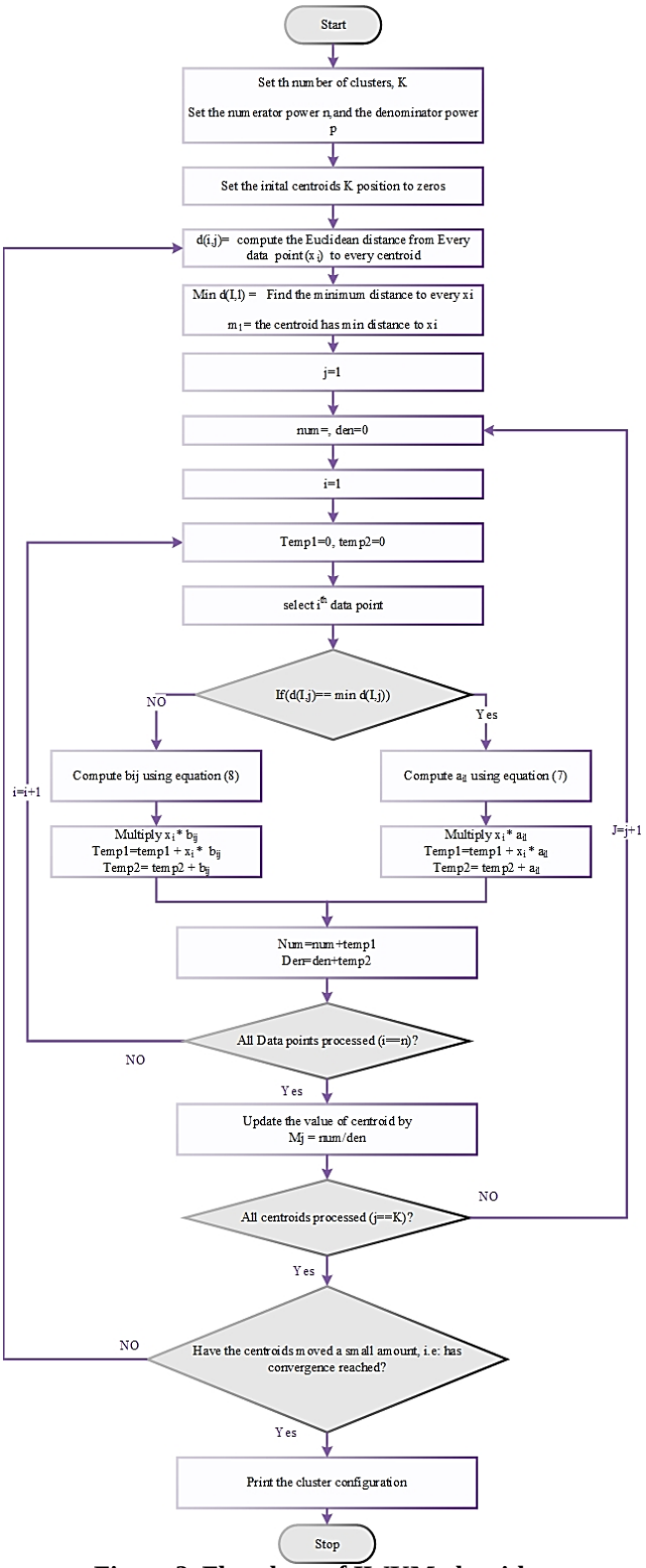


Figure 2. Flowchart of IWKM algorithm

Otherwise, the BIWKM hybrid algorithm differs from IWKM algorithm in exploration/ searching step, in which we search to optimize the solution that every bat population will get it, and compare it with the previous solution. If it is

better, the bat position is updated, save the current solution, and continue to search for a better solution until a convergence condition is met. This search is done heuristically for all bats in the population to get the best global solution by comparing the local best solution for every bat with others, and finally, the best global solution is gotten. we concentrate below on the new modifications to this algorithm, the remains the same as mentioned above in BA and IWKM algorithm sections.

We can split the BIWKM algorithm to three steps as follows:

The first step: The Initialization step, We suppose S is the total number of data points to be clustered, D is the number of features in each data point, K is the number of clusters to be formed, and is a predefined user parameter. N is the bat population size, and every bat (x) is represented by a solution (bat) x = $(x_{11}, x_{12}, x_{13}, \dots, x_{1\,K*D})$ where x_i represented the decision variable on index i and K×D represents the dimensional space of x solution. So, every bat represents a solution x, this solution holds the centroid positions. Additionally, we defined the minimum frequency, the maximum frequency, the loudness and the pulse rate. Then we assigned data points to clusters for each bat randomly and string them in a classification matrix (classMat) with size N × S, which every row represents a bat. This classMat determines the final classification for every data point for each bat. For example: if N=100, S=500, D=2 and K=3, then the population matrix size is (100,3×2) = (100,6)

$$x = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N1} & x_{N2} & x_{N3} & x_{N4} & x_{N5} & x_{N6} \end{bmatrix}$$

Also, every row in the population matrix represents a solution that consists of three centroids because of K=3 in this example. i.e.: for the first row that represents the first bat x_i, the first and second columns represents the first centroid position $[x_{11}, x_{12}]$, the third and fourth columns represents the second centroid position $[x_{13}, x_{14}]$, and the fifth and sixth columns represents the third centroid position $[x_{15}, x_{16}]$. Besides that, we notice every centroid consists of two elements, and this because the dimension D=2, so if D=3, this means that every centroid consists of three elements, and so on.

The classMat size is N × S =100*500, and the value of every element in the matrix is 1,2, or 3 because of K=3 in this example.

$$\text{classMat} = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1S} \\ k_{21} & k_{22} & \cdots & k_{2S} \\ \vdots & \vdots & \vdots & \vdots \\ k_{N1} & k_{N2} & \cdots & k_{NS} \end{bmatrix}$$

where the element of index i, j represent a sample number with index j, and the k_{ij} element consists of a value of {1, 2,…, K}, which represents a cluster number that the corresponding sample with index j of bat x_i belongs to it.

Calculating the best position of centroids are the main goal of this hybrid algorithm by updating the position of centroids iteratively using a meta-heuristic optimization BA. to achieve this goal and find the optimal centroid position by using the equations (6) to (8) with taking in consideration these equations calculate the centroid positions for one bat iteratively. Besides that, m_l is a row vector of dimension equal to D, represents a one centroid in every row in the population matrix.The Second step: The Exploration/search step: this step is the main step whereas every bat search for the best centroid position depending on the local search and the global search for the optimized BA. the bat in the local search stage finds the best local position from its previous positions.

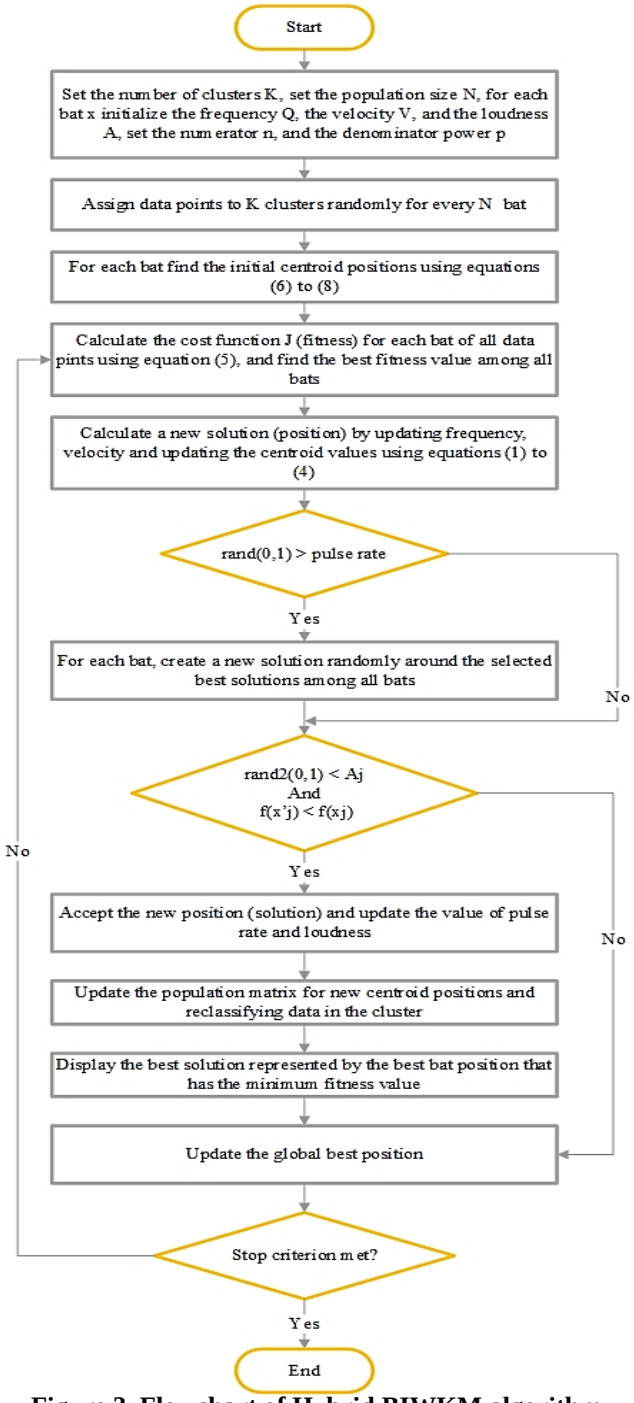


Figure 3. Flowchart of Hybrid BIWKM algorithm

Retrieval Number: B3564078219/19©BEIESP
DOI: 10.35940/ijrte.B3564.078219
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Additionally, the bat in the global search finds the best position of the centroid of all current bats positions as described in BA section, then it updates its frequency, positions, loudness, and velocity using equations (1) to (4). The third step: Centroid positions update and reclassifying data in the clustering step: The decision for the best global bat position is taken from either the previous positions for the same bat or from the current positions of all bats is taking depending on calculating the cost function in equation (5) for every bat. Accordingly, the bat positions will be updated if the value of the cost function is the best (the minimum) as shown in Figure 3. After that, the data is classified again for the new values of the updated centroid positions for every bat. This will be repeated until a convergence criterion is met or reached a maximum number of iterations.

V. EXPERIMENTS AND RESULTS

To verify our new bio-inspired hybrid algorithm BIWKM to clustering data we executed experiments to test its efficiency. The new hybrid algorithm tested using six datasets download from UCI machine learning repository https://archive.ics.uci.edu/ml/ datasets.php. These datasets are Glass, Iris, Libras, Wine, Haberman's Survival and Synthetic Control Chart Time Series. Likewise, we used an artificial dataset called Mouse dataset downloaded from https://elki-project.github.io/datasets/, which can be easily visualized and compared the results visually with IWKM and BIWKM clustering algorithms. The size of datasets, the number of attributes and the number of clusters are shown in Table 1.

Table 1. Datasets Information

Dataset Name	No. of Instances	No. of Attributes	No. of Clusters
Glass	214	9	6
Iris	150	4	3
Libras	360	90	15
Wine	178	13	3
Haberman's Survival	306	3	2
Synthetic Control Chart Time Series	600	60	6
Mouse	500	2	3

The environment of the experiment is carried out using MATLAB programs and conducted on HP ProBook laptop with core i7-6500u CPU and 8 GB RAM. The results of the hybrid algorithm are compared with IWKM clustering algorithm as a testing reference. The comparison depends on processing each dataset five times to measure the average CPU time taken to finish running the program, the number of iterations to converge, and the average best cost function (fitness) values using different values of numerator power (n) and denominator power (p) which are clarified in the cost function in equation (6). The values of p are {1, 2, 3, 4, 5} and $n = p$, $p+1$, $p+2$, and $p+3$, Moreover, n must be equal or larger than p to converge to algorithm in an efficient way as shown in Table 2.

Figure 4 shows some samples of the clustering results of Mouse datasets using the BIWKM and IWKM clustering algorithms with changing the value of p and n. with each cluster has a different color than the others.

It can be shown that both of the algorithms are able to accomplish approximately the same clustering results, However, the BIWKM algorithm accomplishes a fairly good result than IWKM algorithm. The best clustering result was when $p = n = 2$ for BIWKM algorithm.

Table 2. Different Combinations Of *P* And *N* Values

p	$n = p$	$n = p+1$	$n = p+2$	$n = p+3$
1	1	2	3	4
2	2	3	4	5
3	3	4	5	6
4	4	5	6	7
5	5	6	7	8

The performance of both algorithms is measured using the cost/fitness function value using different values of n and p, whereas this value takes into consideration the relation between every point with every centroid. In addition to that, we measured the CPU time of running both algorithms for one iteration, also we measured the fast convergence of both algorithms by taking the differences between the first and last

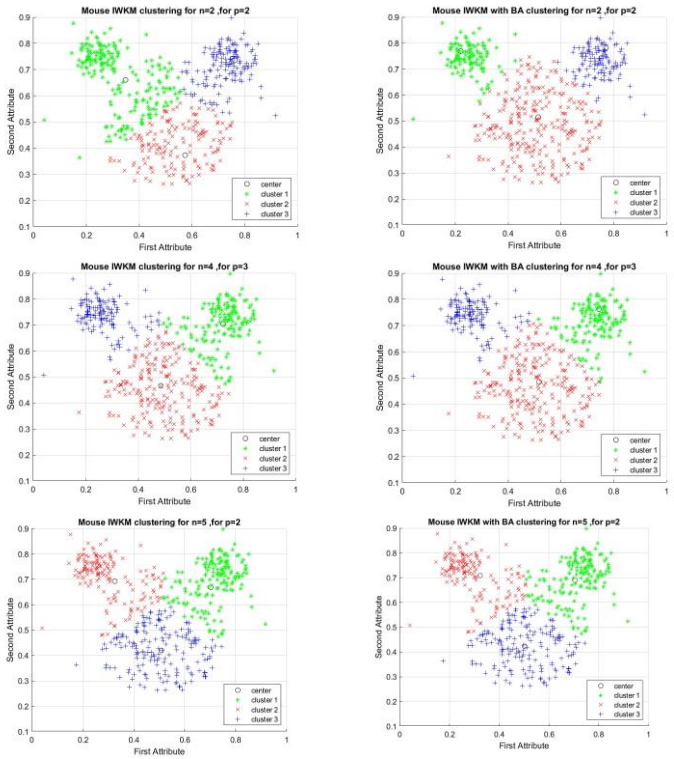
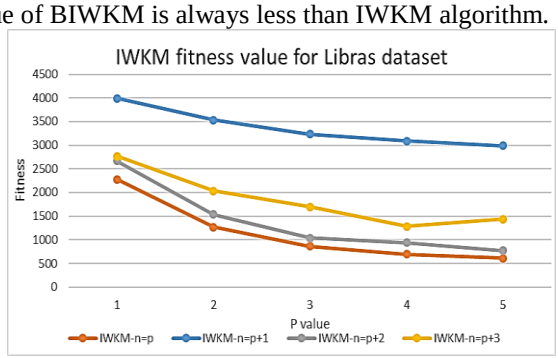


Figure 2. Sample Of Clustering Results Using IWKM And BIWKM For Mouse Dataset

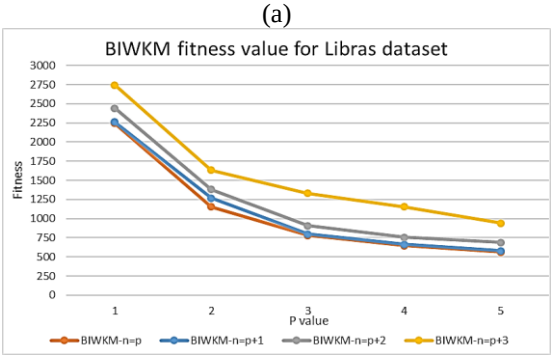
fitness values and if the change is small then the algorithm is converged quickly.

Figure 5 shows a sample of the best global fitness values of Libras dataset for IWKM in part (a) and the BIWKM in part (b) by different values of p and n. For example in BIWKM chart for $n = p+3$ we notice that there is a reduction in the fitness value as p increase, and the same thing

apply to all other lines in each chart. In addition, we notice for the same values of *n* and *p* in both algorithms, the fitness value of BIWKM is always less than IWKM algorithm.



(a)



(b)

Figure 3. Sample Of The Best Global Fitness Results Of Libras Dataset For IWKM In (A), And The BIWKM In (B) Using Different Values Of *P* And *N*.

The Tables 3-9 compares a sample of results of running two algorithms using seven datasets, we take one raw from each value of *p* and *n* for every dataset and displays the best, the worst and the average values of the objective/fitness function.

Table 3. Comparison of IWKM and BIWKM algorithms with respect to fitness values for Haberman dataset

n=	*p*	*n*	IWKM			IWKM		
			Best	Worst	Ave.	Best	Worst	Ave.
P	2	2	306	611	394	306	306	306
P+1	2	3	3531	7116	3637	3531	3531	3531
P+2	2	4	44483	109068	46492	44483	44483	44483
P+3	2	5	79353810	2123698	846734	776569	776583	776572

Table 4. Comparison of IWKM and BIWKM algorithms with respect to fitness values for Libras dataset

n=	*p*	*n*	IWKM			IWKM		
			Best	Worst	Ave.	Best	Worst	Ave.
P	2	2	1265	5398	1859	1154	1220	1176
P+1	2	3	1296	8841	2174	1269	1269	1269
P+2	2	4	1534	15208	2844	1380	1385	1384
P+3	2	5	2033	27538	4195	1632	1747	1648

Table 5. Comparison of IWKM and BIWKM algorithms with respect to fitness values for Synthetic dataset

n=	*p*	*n*	IWKM			IWKM		
			Best	Worst	Ave.	Best	Worst	Ave.
P	2	2	1102.92	3592.65	1486	870	1043.56	880
P+1	2	3	46377	276152	55689	46377	46377	46377
P+2	2	4	1918348	23507675	2543347	1918348	1918348	1918348
P+3	2	5	38904599	239632167	154236636	88537135	88904599	88561456

The fitness function makes relation with every point, with every centroid, and with the best centroid for every data point. As a result, the fitness function value measures the cluster integrity of the algorithm. The BIWKM algorithm gets better fitness values than IWKM algorithm in all values of *p* and *n*. As *p* and *n* increases, the values of fitness functions increase, because the term that raised to power n increases in the numerator, and the term that in the denominator raised to power p decreases.

Table 6. Comparison of IWKM and BIWKM algorithms with respect to fitness values for Iris dataset

n=	*p*	*n*	IWKM			IWKM		
			Best	Worst	Ave.	Best	Worst	Ave.
P	2	2	160.8	449.2	185.6	160.8	160.8	160.8
P+1	2	3	126.4	905.8	146.8	126.4	126.4	126.4
P+2	2	4	105.9	2036.5	148.1	105.9	105.9	105.9
P+3	2	5	105.1	5787.8	235.3	104.2	104.2	104.2

Table 7. Comparison of IWKM and BIWKM algorithms with respect to fitness values for Wine dataset

n=	*p*	*n*	IWKM			IWKM		
			Best	Worst	Ave.	Best	Worst	Ave.
P	2	2	206	529	218	206	206	206
P+1	2	3	20902	134135	23895	20902	20902	20902
P+2	2	4	3192125	52433984	4195606	3192000	3192007	3192000
P+3	2	5	7.99E+8	269E+8	26.89E+8	6.13E+8	7.2E+8	6.2E+8

Table 8. Comparison of IWKM and BIWKM algorithms with respect to fitness values for Mouse dataset

n=	*p*	*n*	IWKM			IWKM		
			Best	Worst	Ave.	Best	Worst	Ave.
P	2	2	675.5	1497.9	719.5	635.8	635.8	635.8
P+1	2	3	81.1	345.1	90.3	81.0	81.1	81.01
P+2	2	4	12.83	91.743	14.91	12.75	12.75	12.75
P+3	2	5	2.254	27.289	2.961	2.253	2.254	2.253

Table 9. Comparison of IWKM and BIWKM algorithms with respect to fitness values for Glass dataset

n=	*p*	*n*	IWKM			IWKM		
			Best	Worst	Ave.	Best	Worst	Ave.
P	2	2	247.5	642.0	366.2	217.0	241.3	219.0
P+1	2	3	456.5	1808.3	501.5	456.5	456.5	456.5
P+2	2	4	1166.4	6856.0	1352.4	1164.7	1164.9	1164.7
P+3	2	5	3768.9	32569.3	4546.0	3766.5	3767.3	3766.6

The BIWKM algorithm find approximately the best global fitness value from the first iteration, and in the next iterations the reduction in the fitness values is minor, i.e,. The difference between first fitness value in the first iteration and the last one is very small, though the IWKM algorithm starts from a large fitness value then the value reduces in the next iterations, so there is a large reduction in the fitness value. However, both of algorithms reduced to nearly the same fitness value but the BIWKM is always the minimum value.

The difference between the first and last fitness values in BIWKM is approximately less than one hundred in almost data sets except for the Synthetic dataset especially when the ***n=p***+3, However, the difference is much less than the IWKM algorithm.

For example, the difference in last and first value for Glass dataset when ***p***=2 and ***n***=4 are 5687.944 in IWKM algorithm, even though the difference in last and first value for is 0.1894 in BIWKM algorithm for the same dataset. The reason for that in BIWKM algorithm each bat in the population find a fitness value and the algorithm select the best globally one, on another hand, this does not exist in IWKM algorithm.

Table 10 shows the minimum fitness values for all datasets for different combinations of ***p*** and ***n***. The minimum fitness values measured at ***p*** is equal to 5, and the most minimum value is when ***p*** =5, and ***n*** = ***p***+3 =8 for BIWKM algorithm, this means if the value of p increases, and the difference between p and n increases, then the fitness value is the most minimum. In contrast, the maximum fitness value of fitness function is measured when ***p***=1 as shown in Table 11, in addition to that, as the difference between p and n increases, the fitness value increases. As a result, the most maximum fitness value is when ***p***=1 and ***n***=***p***+3 = 4.

Table 10. Comparison Of Minimum Fitness Values For Different Values Of *N* And *P* For All Datasets

n =	***p***	***n***	**IWKM**	**BIWKM**
p	5	5	151.0148	150
p+1	5	8	64.80109	64.79843
p+2	5	8	10.00895	9.893861
p+3	5	8	1.754061	1.736755

There is no difference in clustering results using any values of n in condition n must be greater than or equal to *p* for BIWKM algorithm. Otherwise, the values of n must be greater than p for IWKM algorithm to converge smoothly without oscillating; meanwhile, there are two problems if the value of n is equal to *p* in IWKM algorithm. The first one is a high oscillating in the convergence of the algorithm as shown in Figure 8, the second one is the clustering result is bad as shown in Figure 4 in the top left image. The first reason is as mentioned in the previous two paragraphs and the other reason regard with the first term in the cost function for the data point,

$$s_i = \frac{\|S_i - m_1\|^n}{\|S_i - m_1\|^p} + \cdots + \frac{\|S_i - m_1\|^n}{\|S_i - m_j\|^p}$$

is equal to one and the other terms have values smaller than one but has a high effect on the cost function, thus leads to a high oscillating as shown in Figure 8. On the contrary, when n is larger than p then the first term has a value larger than one because the numerator is greater than the denominator, also the other terms is much less than one so can be neglected.

As shown Table 12 the time in seconds taken to finish running the algorithm in IWKM is much less than in BIWKM algorithm, this is because we used bat population of 400 with a number of iterations is 50. However, in IWKM algorithm there is no bat population, we only depend on the number of iterations which is equal to 50, as a result, the time needed for IWKM to finish running is much smaller than BIWKM algorithm. But if measure the time needed for one bat to finish 50 iterations in BIWKM algorithm the result is approximately less than in IWKM algorithm, consequently, the time in BIWKM depends on population size.

Table 11. Comparison Between Two Algorithms With Respect To The Time Taken To Running The Algorithms For Different Values Of *N* And *P* For All Datasets

Dataset	***p***	***n***	**IWKM time**	**BIWKM time**	**BIWKM time/400**
Haberman	5	8	0.154	46.827	0.117
Libras	4	5	1.021	308.792	0.771
Synthetic	3	6	0.832	219.695	0.549
Iris	5	6	0.104	32.758	0.081
Wine	3	6	0.100	34.809	0.087
Glass	2	5	0.137	40.780	0.101
Mouse	1	4	0.191	78.523	0.196

VI. CONCLUSION

The benefits of inverse weighted k-means clustering algorithm and bio-inspired bat algorithm is combined in a new hybrid algorithm to get better global fitness values, clustering results, quicker convergence than the inverse weighted k-means clustering algorithm using different values of the numerator (*n*) and denominator (*p*) powers of the cost function. Besides that, overcoming the oscillating problem in the inverse weighted k-means clustering algorithm. For future work, we will use this new algorithm to make image segmentation to get improved results. The result of BIWKN gave more effective global fitness values than result of IWKM.

REFERENCES

1. M. Alhanjouri and R. Ahmed, "New Density Based Clustering Technique: GMDBSCAN-UR", International Journal of Advanced Research in Computer Science, Vol. 3, No. 1, Feb. 2012. Published by ISSN: 0976-5697.
2. W. Barbakh, C. Fyfe, Inverse weighted clustering algorithm. Computing and Information Systems 2007; 11(2).
3. T. Johnson, G. children, P. Fadte, A. Falari, Enhanced k strange points clustering using bat inspired algorithm. International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR) 2018; 8(2):77-82.
4. G. Dashora, P. Awwal, An overview of particle swarm optimization and bat algorithm for data clustering. International Journal of Trend in Research and Development; 3(1):2394-9333.
5. N. Mohan, R. Sivaraj, R. Devi Priya, A comprehensive review of bat algorithm and its applications to various optimization problems. Asian Journal of Research in Social Sciences and Humanities; 6(11):676-690.
6. R. Tang, S. Fong, X. Yang, S. Deb, Integrating Nature-inspired Optimization Algorithms to K-means Clustering. Seventh International Conference on Digital Information Management (ICDIM) 2012:116-123.
7. S. Fong, S. Deb, X. Yang, Y. Zhaung, Towards enhancement of performance of k-means clustering. The Scientific World Journal 2014.
8. W. Barbakh, M. Crowe, C. Fyfe, A family of novel clustering algorithms. International Conference on Intelligent Data Engineering and Automated Learning 2006:283-290.
9. X. Yang, A New Metaheuristic Bat-Inspired Algorithm-for code -Xin-She Yang-2010. Nature Inspired Cooperative Strategies for Optimization (NISCO 2010); 284:65-74.
10. M. Alhanjouri and H. Hejazi, "DIC Structural HMM based IWAK-means to Enclosed Face Data", International Journal of Computer Applications (IJCA), Vol. 18, No. 4,pp. 43-50, March 2011. Published by Foundation of Computer Science. NY., USA.
11. M. Alhanjouri and R. Ahmed, "GMDBSCAN-UR as a new density-based clustering technique", International Conference for computing and information (ICCI-2012), University of Helwan, Cairo, Egypt. Dec. 2012.

AUTHORS PROFILE

Mohammed Ahmed Alhanjouri, received Bachelor of Electrical and Communications Engineering (honor), Master of Electronics Engineering (excellent), and Ph.D. in electrical communications, his dissertation about chromosome classification and DNA fingerprint. He has worked at Arab Academy for Science & Technology & Maritime Transport (Alex., Egypt) from 2002 to 2006. Then he worked as Assistant Professor, with responsibility: director of projects and research center, and head of computer Engineering department at Islamic University of Gaza, Palestine. Also he worked at University of Palestine as dean of library, then as head of Software Eng. Dept. Currently, he is serving as Associate Professor with responsibilities: Dean of Quality and Development, Chairman of IT Affairs, and Director of Excellence and eLearning Center at Islamic University of Gaza. His research areas are: artificial Intelligence, optimization techniques, image/speech recognition, machine learning, bioinformatics, neural network, genetic algorithm. He published more than 50 scientific article in different journals and conferences.

Ahmed Mohammed Alghoul graduated from Computer Engineering Faculty (University of Jordan- Jordan) in 2006. Currently, he is a postgraduate student at Faculty of Engineering, Computer Engineering Department in Islamic University in Palestine. He worked as a teaching assistant in University College of applied science in 2016. He is working now as a Head of Operating Systems Department in the Governmental Datacenter. His primary research interest is in the area of optimization techniques, Artificial Intelligence, Pattern Recognition, Networking and Security.