

Non-centroid-based discrete differential evolution for data clustering

Tanapon Poonthong, Jeerayut Wetweeraopong

Department of Mathematics, Faculty of Science, Khon Kaen University, Khon Kaen, Thailand

Article Info

Article history:

Received Jun 7, 2024

Revised Sep 3, 2024

Accepted Sep 28, 2024

Keywords:

Clustering accuracy

Data clustering

Differential evolution

Discrete optimization

Intra-cluster distance

Non-spherical shape clusters

ABSTRACT

Data clustering can find similarities and hidden patterns within data. Given a predefined number of groups, most partitional clustering algorithms use representative centers to determine their corresponding clusters. These algorithms, such as K-means and optimization-based algorithms, create and update centroids to give (hyper) spherical shape clusters. This research proposes a non-centroid-based discrete differential evolution (NCDDE) algorithm to solve clustering problems and provide non-spherical shape clusters. The algorithm directs the population of discrete vectors to search for data group labels. It uses a novel discrete mutation strategy analogous to the continuous mutation in classical differential evolution. It also combines a sorting mutation to enhance convergence speed. The algorithm adaptively selects crossover rates in high and low ranges. We use the UCI datasets to compare the NCDDE with other continuous centroid-based algorithms by intra-cluster distance and clustering accuracy. The results show that NCDDE outperforms the compared algorithms overall by intra-cluster distance and achieves the best accuracy for several datasets.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Jeerayut Wetweeraopong

Department of Mathematics, Faculty of Science, Khon Kaen University

Khon Kaen, 40002, Thailand

Email: wjeera@kku.ac.th

1. INTRODUCTION

The rapid growth of the internet, social media, and digital technologies has generated vast amounts of data that require data analysis tools to uncover hidden patterns and identify similar groups based on their attributes. Clustering algorithms are often employed to explore and categorize datasets, dividing data points into clusters or groups where the data within each group are more similar than those from other groups. These algorithms have been extensively applied in various fields, including image processing, data analysis, and pattern recognition [1], [2], as well as natural language processing, text mining [3], [4], and social network analysis and community detection [5], [6]. This research focuses on partitional clustering methods, which determine groups by optimizing similarity measures and transforming the task into an optimization problem. Most of these methods are centroid-based, using centroids or representative centers to define clusters. Examples include K-means [7] and population-based approaches such as genetic algorithms (GA) [8], particle swarm optimization (PSO) [9], differential evolution (DE) [10], artificial bee colony (ABC) [11], and ant colony optimization (ACO) [12]. These algorithms create and update centroids to minimize the distance between data points and their corresponding cluster centers and give spherical-shaped clusters.

We propose a non-centroid-based discrete differential evolution (NCDDE) algorithm for data clustering. The algorithm does not use centroids. It directly searches for the group labels for each data point, which allows the discovery of non-spherical clusters. The NCDDE employs a novel discrete mutation that interprets the continuous mutation of DE to the discrete space of integer labels. It adaptively combines the sorting mutation with the classic mutation to enhance convergence speed and select crossover rates from high and low ranges to suit the problems. This research's main contribution is a non-centroid-based discrete method that is competitive in giving the best intra-cluster distance and clustering accuracy for several datasets and providing non-spherical shape clustering results.

The remainder of the paper is organized as follows: section 2 provides the background knowledge, reviews the centroid-based optimization algorithms for data clustering, and presents the UCI datasets. Section 3 describes the proposed NCDDE algorithm. Section 4 performs the preliminary experiment to find a suitable mutation strategy for the NCDDE and the comparison experiments with other methods on the UCI datasets using intra-cluster distance and clustering accuracy. Section 5 provides insight discussion. Finally, we provide a conclusion in the last section.

2. LITERATURE REVIEW

This section provides backgrounds on the similarity measure, clustering accuracy, and the classical DE algorithms, as well as an overview of centroid-based optimization methods for data clustering on UCI datasets.

2.1. Similarity measure and intra-cluster distance

We use a similarity measure to calculate the similarity of a clustering result, considering the similarity of data points in each cluster through a distance function. This research employs the Euclidean distance to calculate the distance between two points, as (1):

$$Dist(x_i, x_j) = \sqrt{\sum_{m=1}^D (x_{im} - x_{jm})^2} \quad (1)$$

where x_i and x_j are data points and D is the dimension of each data point.

Let $Q = \{Q_1, Q_2, \dots, Q_K\}$ be a clustering result consisting of clusters Q_1 to Q_K and c_1, c_2, \dots, c_K be the representative centers where c_i is the center of the cluster Q_i . To measure the similarity of Q , we use the intra-cluster distance (f), which calculates the sum of distances between data points and their corresponding centers by (2):

$$f(Q) = \sum_{i=1}^K \sum_{x \in Q_i} Dist(c_i, x) \quad (2)$$

where x is a data point. The low value of $f(Q)$ indicates a suitable clustering result.

2.2. Clustering accuracy

Clustering accuracy assesses the results of different clustering methods to a real-world benchmark dataset that includes the original group labels for each data point. A clustering method does not know these labels or the underlying data structure, and we categorize it as an unsupervised learning technique. Thus, different clustering methods may produce varying levels of accuracy. Clustering accuracy is the percentage of matches between the original group labels and those assigned by the clustering method. Its calculation uses (3):

$$\text{Clustering accuracy} = \frac{\text{Number of matched data points}}{\text{Number of all data points}} \times 100 \quad (3)$$

2.3. Classical differential evolution algorithm

The DE algorithm is an efficient population-based algorithm introduced by Storn and Price [10] for continuous optimization. The algorithm consists of initialization, mutation, crossover, and selection steps. The algorithm randomly generates vectors $x_i = [x_{ij}]$ where $i = 1, 2, 3, \dots, NP$ and $j = 1, 2, 3, \dots, D$, and finds the best vector x_{best} and its fitness function value f_{best} . The mutation step selects three random population vectors

x_{r1} , x_{r2} , x_{r3} and creates a mutant vector xm by adding the difference between two of them to a third vector by (4):

$$xm = x_{r1} + F(x_{r2} - x_{r3}) \quad (4)$$

where F is a scaling factor in a range of $[0, 1]$. The crossover operation constructs a trial vector xc by exchanging components of the target x_i and the mutant vectors as (5):

$$xc_j = \begin{cases} xm_j & \text{if } rand_j < CR \text{ or } j = I \\ x_{ij} & \text{otherwise} \end{cases} \quad (5)$$

where $rand_j$ is a uniform random number in a range of $[0, 1]$, $j = 1, 2, 3, \dots, D$, and I is a randomly fixed index from 1 to D . Then, the selection step compares the trial vector with the target vector. If the fitness of the trial vector is better, it replaces the target vector. Otherwise, the target vector remains. The algorithm terminates when a stopping criterion is met, such as reaching the maximum number of iterations. DE requires suitable control parameters (F and CR) to solve different problems. Many adaptive variants of DE have been proposed to manage the control parameters [13], [14].

2.4. Centroid-based optimization algorithms for clustering

Many researchers have proposed centroid-based algorithms to solve clustering problems using the DE algorithm and other population-based methods. Xiang *et al.* [15] presented a centroid-based differential algorithm with a shuffled strategy (DSDE) to solve clustering problems. It separates the population into two subgroups and merges them at the end of each generation. DSDE outperforms ACO, ABC, PSO, and PSOAG regarding intra-cluster distance and clustering accuracy on UCI datasets. Nayak *et al.* [16] proposed the cross-mutation-based differential evolution (CMDE), which hybridizes two mutation strategies for data clustering. By using weights to reduce the influence of the best vector and rearranging centroids, CMDE improves clustering results, offering better intra-cluster distances and accuracy on UCI datasets than the DE using a single mutation strategy. Mustafa *et al.* [17] combined a memetic algorithm and adaptive differential evolution mutation (AMADE) to identify candidate centroids for clustering. Utilizing the DEcurrent-to-best1 strategy for faster convergence, along with a restart phase to prevent premature convergence, AMADE outperforms GA, DE, HyGA, and HyDE algorithms on UCI datasets. The authors also provided the solutions of cluster centers for each dataset. Next, Tarkhaneh and Moser [18] introduced the archimedean spiral and neighborhood search-based mutation approach (ADENS) to solve clustering problems. The algorithm creates a spiral vector for mutation operation and uses neighborhood search to generate solutions for replacing poorly performing individuals. ADENS shows superior results on UCI datasets in minimizing intra-cluster distance and improving clustering accuracy compared to ICSK, DE-KM, and DSDE algorithms. Poonthong *et al.* [19] presented an adaptive differential evolution with archive strategy (ADEAS) for solving clustering problems. It finds candidate centroids and minimizes the intra-cluster distance. The algorithm employs an archive to store inferior solutions for enhancing population diversity. The result shows that ADEAS outperforms PSOPC, ACODE, ADEANS, VDEO, CMDE, and DSDE methods on UCI datasets.

Other population-based methods and their hybrids can potentially solve clustering problems. Abualigah *et al.* [20] combined two well-known optimization techniques, Harris Hawks optimization and differential evolution (H-HHO), to balance global and local searches. Experimental results on the UCI dataset show that the algorithm outperforms K-means, TLBO, GSA, ITGO, and classical DE algorithms regarding intra-cluster distance and cluster accuracies. The authors also provided optimal centroids. Sight *et al.* [21] proposed a moth-flame optimization algorithm (MFO) for data clustering. It uses a logarithmic spiral function to control the search. Experiments on UCI datasets demonstrate that MFO can achieve better intra-cluster distances than BHA, MVO, HHO, GWO, and K-means and identify the centroids generated by the proposed algorithm. Sharma and Chhabra [22] developed a clustering solution by integrating PSO with a polygamous approach and crossover (PSOPC). The algorithm sets up each particle with cluster centroids chosen from the dataset. It generates offspring particles by merging the best particles with random ones to enhance exploration and exploitation. The PSOPC achieves better intra-cluster distance and data clustering accuracy than PSO, GA, DE, FA, and GWO. More recently, Singh *et al.* [23] proposed an opposition learning-based Harris Hawks optimizer (OHHO) for solving the data clustering problem, which incorporates the opposition learning technique into the exploration phase of the Harris Hawks optimizer algorithm. Experiments on UCI datasets show that

the proposed OHHO method outperforms MVO, BOA, SCA, HHO, SSA, and GWO algorithms by intra-cluster distance. The authors also provided optimal centroids for each dataset.

2.5. UCI datasets

The UCI machine learning repository [24] is a data resource that provides a wide range of datasets for data clustering. This research uses the Iris, Wine, Glass, Thyroid, Haberman, Liver, Cancer, Vowel, and CMC datasets. Table 1 lists their names, numbers of instances, attributes, and classes.

Table 1. The description of UCI datasets

Name	Instances	Attributes	Classes
Iris	150	4	3
Wine	178	13	3
Glass	214	9	6
Thyroid	215	5	3
Haberman	306	3	2
Liver	345	6	2
Cancer	683	9	2
Vowel	871	3	6
CMC	1473	9	3

3. THE PROPOSED NON-CENTROID-BASED DISCRETE DIFFERENTIAL EVOLUTION ALGORITHM

We present a NCDDE algorithm to solve clustering problems. It operates on a population of discrete vectors where the number of components equals the number of data (D). Components are integer labels of data groups ranging from 1 to a total number of clusters (K). Data corresponding to components with this group label can be used to calculate a group centroid or other representative center. The algorithm implements a new discrete mutation that follows the concept of continuous mutation of classical DE but does not use a scaling factor. Like DEASC [25], the NCDDE uses a crossover rate CR in the low range $[0,0.1]$ and high range $[0.9,1]$. It also combines the sorting mutation to improve convergence speed. The details of NCDDE are as follows:

3.1. Discrete mutation

The discrete mutation selects three population vectors x_{r1} , x_{r2} and x_{r3} where $r1$, $r2$, and $r3$ are random distinct indices and also different from i . The algorithm considers x_{r1} as the base vector and creates the mutant vector xm where xm_j is the same as $x_{r1,j}$ when the components $x_{r2,j}$ and $x_{r3,j}$ are the same. Otherwise, xm_j is randomized from 1 to K when $x_{r2,j}$ and $x_{r3,j}$ are different. The discrete mutation is (6):

$$xm_{i,j} = \begin{cases} x_{r1,j} & \text{if } x_{r2,j} = x_{r3,j} \\ I_{rand} & \text{if } x_{r2,j} \neq x_{r3,j} \end{cases} \quad (6)$$

where $j = 1, 2, 3, \dots, D$ and I_{rand} is a random integer from 1 to K .

Figure 1 illustrates the discrete mutation for the following three x_{r1} , x_{r2} and x_{r3} vectors of integer labels where the number of cluster $K = 3$ and the dataset comprises 9 data points. The components $*$ of xm are random integers from 1 to K . At the beginning period of generations, the mutant vectors are diversified. When the search progresses, the population vectors will have better fitness values through the selection and become more similar. The mutation process will create mutant vectors that are more intensified and converge toward an optimal solution. They will gradually and indirectly keep the components that are good among them. It happens when the corresponding components of x_{r1} , x_{r2} and x_{r3} are the same. The crossover operation plays a role in mixing the contents of a target vector and a mutant one to obtain a trial vector (to compare and compete with the target).

$x_{r1} =$	1	1	2	2	1	3	2	3	3
$x_{r2} =$	1	1	1	2	1	2	3	3	3
$x_{r3} =$	1	2	1	1	2	3	3	2	3
$xm =$	1	*	2	*	*	*	2	*	3

Figure 1. Discrete mutation to generate x_m from vectors x_{r1} , x_{r2} , and x_{r3}

3.2. Sorting mutation

The sorting mutation selects x_{r1}^* with the lowest fitness values from the randomly selected vectors x_{r1} , x_{r2} , and x_{r3} . Let x_{r2}^* and x_{r3}^* be the remaining vectors. The algorithm considers x_{r1}^* as the base vector and creates the mutant vector xm from x_{r1}^* , x_{r2}^* , and x_{r3}^* using the discrete mutation.

3.3. The combination of discrete and sorting mutations

NCDDE adaptively combines discrete and sorting mutations according to the corresponding probabilities $pm1$ and $pm2$ calculated from their success in selection. The probabilities are initialized to 0.5 and updated by the counters $nm1$ and $nm2$. If a randomly generated number is less than $pm1$, the algorithm selects the discrete mutation to generate xm ; otherwise, it selects the sorting mutation. The probabilities are updated by these counters.

3.4. Adaptive control parameter CR

The algorithm employs the crossover rate CR values within the ranges of $[0, 0.1]$ and $[0.9, 1]$ to generate a trial vector based on the corresponding probabilities $pc1$ and $pc2$, which are calculated from their success in selection. The probabilities are initialized to 0.5 and updated by the counters $nc1$ and $nc2$. A trial vector xc is generated by (7):

$$xc_j = \begin{cases} xm_j & \text{if } rand_j < CR \text{ or } j = I \\ x_{ij} & \text{otherwise} \end{cases} \quad (7)$$

where $rand_j$ is a uniform random number in a range of $[0, 1]$, $j = 1, 2, 3, \dots, D$, and I is a randomly fixed index from 1 to D . The pseudo-code of NCDDE is presented in Algorithm 1.

4. EXPERIMENTAL DESIGN

We design three experiments. The first preliminary experiment finds a suitable mutation strategy for NCDDE. The second experiment compares the NCDDE method with five compared methods using intra-cluster distance. In the third experiment, we compare the performance of the NCDDE method with five compared methods using clustering accuracy.

4.1. Finding a suitable mutation strategy for the NCDDE algorithm

This preliminary experiment compares the performance of NCDDE algorithms using three different mutation strategies: discrete mutation (DM), sorting mutation (SM), and adaptive mutation (AM). DM is a simple discrete mutation. SM positions the best vector by fitness function from three random vectors as the base vector. AM adaptively uses discrete and sorting mutations based on their success in creating a better solution in the selection. Each algorithm runs on the Iris, Wine, Glass, Thyroid, Haberman, and Cancer datasets. We set the population size $NP=50$ and the maximum number of generations $maxGen = 200000$. The algorithm terminates when it reaches $maxGen$, or f_{best} is the same for $nimp = 1000$ consecutive generations.

Table 2 shows the mean and the percentage of the standard deviation of the obtained intra-cluster distances and the number of function evaluations on 30 runs and highlights the best value in boldface. The NCDDE algorithm with AM gives the lowest mean values for all datasets except for Thyroid, where the DM gives a slightly lower mean than SM and AM. Thus, we choose NCDDE with the adaptive mutation as our proposed algorithm. Figure 2 shows the convergence graphs of the NCDDE with discrete, sorting, and adaptive mutations.

Algorithm 1 NCDDE algorithm

```

1: Set the number of clusters  $K$  and dimension  $D =$  the number of data
2: Set the control parameters  $CR, pm1, nm1, nm2, pc1, nc1, nc2, nimp$ 
3: Set the maximum number of generations  $maxGen$ 
4: Initialize the population  $P$  of  $NP$   $D$ -dimensional integer vectors  $x_i$  where each  $x_{i,j}$  is from 1 to  $K$ .
5: Find the best vector  $x_{best}$  and its best fitness value  $f_{best}$ 
6: for  $g=1:maxGen$  do
7:   for  $i=1:NP$  do
8:     if  $rand(0, 1) < pm_1$  then
9:       Create a mutant vector  $xm$  using discrete mutation by eq. (6)
10:    else
11:      Create a mutant vector  $xm$  using sorting mutation
12:    end if
13:    Apply the crossover operation eq. (7) to get a trial vector  $xc$ 
14:    Calculate  $f(xc)$  and  $nf = nf + 1$ 
15:    if  $f(xc) < f(x_i)$  then
16:       $x_i = xc$  and  $f(x_i) = f(xc)$ 
17:      if  $xm$  is created by eq. (6) then
18:        Increase  $nm1 = nm1 + 1$ 
19:      else
20:        Increase  $nm2 = nm2 + 1$ ;
21:      end if
22:      if  $nm1 + nm2 \geq 100$  then
23:        Adjust  $nm1 = nm1 + 10$  and  $nm2 = nm2 + 10$ 
24:        Update  $pm1 = 0.9pm1 + 0.1nm1/(nm1 + nm2)$ 
25:        Set  $nm1 = 0$  and  $nm2 = 0$ 
26:      end if
27:      if  $CR$  in range  $[0, 0.1]$  then
28:        Increase  $nc1 = nc1 + 1$ 
29:      else
30:        Increase  $nc2 = nc2 + 1$ 
31:      end if
32:      if  $nc1 + nc2 \geq 100$  then
33:        Adjust  $nc1 = nc1 + 10$  and  $nc2 = nc2 + 10$ 
34:        Update  $pc1 = 0.9pc1 + 0.1nc1/(nc1 + nc2)$ 
35:        Set  $nc1 = 0$  and  $nc2 = 0$ 
36:      end if
37:      if  $f(xc) < f_{best}$  then
38:        Update  $x_{best}$  and  $f_{best}$ 
39:      end if
40:    end for
41:  end for
42:  if  $f_{best}$  remains unchanged for  $nimp$  generations then
43:    Stop
44:  end if
45: end for
46: Report  $x_{best}, f_{best},$  and  $nf$ 

```

Table 2. Performance comparison of NCDDE with discrete, sorting, and adaptive mutations

Datasets	Statistics	DM	SM	AM
Iris	Mean_fb	97.27	97.30	97.27
	SD	0.08	0.12	0.09
	Mean_nf	250402.17	75532.77	132506.13
	%SD	32.09	38.76	41.88
Wine	Mean_fb	16578.44	16542.41	16539.73
	SD	1.12	0.10	0.06
	Mean_nf	271041.53	68561.17	131980.70
Glass	%SD	30.33	40.61	42.59
	Mean_fb	248.38	248.91	247.65
	SD	3.46	4.19	2.90
Thyroid	Mean_nf	814838.27	749808.34	791164.30
	%SD	38.63	35.85	33.65
	Mean_fb	1997.07	2004.50	2001.06
Haberman	SD	0.78	0.29	0.43
	Mean_nf	336957.13	116094.21	173266.17
	%SD	44.60	41.35	66.14
Cancer	Mean_fb	2625.11	2625.16	2625.11
	SD	0.00	0.01	0.00
	Mean_nf	34931.55	20382.72	23668.03
Cancer	%SD	40.78	7.13	13.12
	Mean_fb	2984.13	2985.28	2984.12
	SD	0.01	0.15	0.01
Cancer	Mean_nf	62734.14	41432.64	43969.60
	%SD	42.70	3.59	3.72

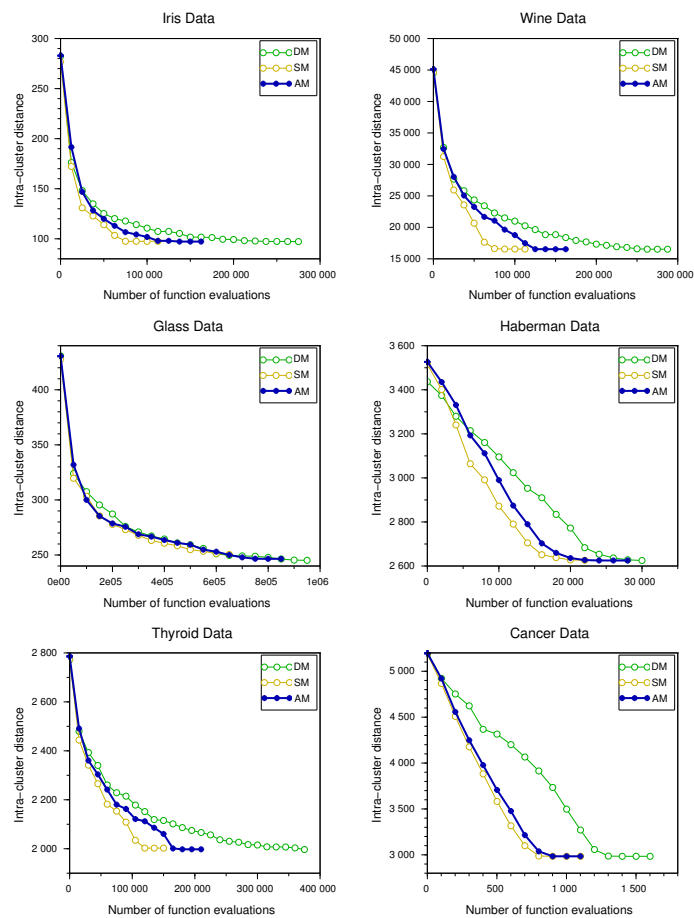


Figure 2. Convergence graphs of NCDDE with DM, SM, and AM on six UCI datasets

4.2. Performance comparison of the NCDDE with other continuous centroid-based algorithms by intra-cluster distance

The second experiment compares the intra-cluster distance achieved by NCDDE with ADEAS, HHO, AMADE, ADENS, and DSDE algorithms on the Iris, Wine, Glass, Vowel, Cancer, CMC, Thyroid, and Liver datasets. The NCDDE generates candidate cluster solutions without using centroid. It obtains centroids from data corresponding to components with each group label and calculates the intra-cluster distances. We use the centroid results from the compared algorithm's original papers to calculate the intra-cluster distances. The NCDDE performs 30 independent runs. Table 3 highlights the obtained lowest best intra-cluster distances for each dataset in bold text. The unavailable results of some algorithms on some datasets are indicated by “-”.

NCDDE, ADEAS, AMADE, and DSDE give the same lowest intra-cluster distance for Iris. ADEAS, AMADE, and DSDE achieve the lowest values for Vowel. ADEAS and DSDE give the lowest values for Liver. ADEAS achieves the lowest value for Glass. NCDDE is the only algorithm that gives the best values for the remaining datasets: Wine, Cancer, CMC, and Thyroid. Consequently, NCDDE outperforms the compared algorithms overall.

Table 3. Intra-cluster distances obtained by NNCDE and the compared algorithms

Datasets	ADEAS	H-HHO	AMADE	ADENS	DSDE	NCDDE
Iris	97.22	97.72	97.22	97.33	97.22	97.22
Wine	16555.07	16544.17	16555.07	17871.92	16555.07	16530.54
Glass	215.47	310.94	215.50	-	215.49	236.94
Thyroid	1986.72	-	-	-	1986.72	1981.91
Liver	9982.95	-	-	10332.15	9982.95	10037.78
Cancer	2984.90	3001.62	2984.90	3010.76	2984.90	2984.07
Vowel	149331.28	-	149331.28	-	149331.28	245965.46
CMC	5541.65	5546.51	5541.65	5561.39	5541.65	5541.64

4.3. Performance comparison of the NCDDE with other algorithms by clustering accuracy

Using the results from the previous experiment, we compare the clustering accuracy of NCDDE with ADEAS, H-HHO, AMADE, ADENS, and DSDE. Each UCI dataset defines the original group labels for all data points. Clustering accuracy is the percentage of the number of data points that are assigned by an algorithm to their correct groups. Table 4 highlights the highest clustering accuracy obtained for each dataset in bold text.

Table 4. Clustering accuracies obtained by NNCDE and the compared algorithms

Datasets	ADEAS	HHO	AMADE	ADENS	DSDE	NCDDE
Iris	90.00	94.00	90.00	89.33	90.00	90.00
Wine	71.91	71.35	71.91	62.92	71.91	70.79
Glass	52.80	43.93	52.34	-	52.34	43.46
Thyroid	58.14	-	-	-	58.14	71.63
Liver	50.43	-	-	55.36	50.43	50.43
Cancer	96.49	95.61	96.49	95.02	96.49	96.78
Vowel	48.91	-	48.91	-	48.91	42.94
CMC	39.44	39.31	39.44	39.10	39.44	39.51

ADEAS, AMADE, and DSDE give the highest clustering accuracy for Wine and Vowel datasets. HHO achieves the highest clustering accuracy for Iris. ADEAS gives the highest clustering accuracy for Glass. ADENS achieves the highest clustering accuracy for Liver, and NCDDE gives the highest accuracy for Cancer, CMC, and Thyroid. We can observe that ADEAS and NCDDE can provide the best accuracy for three different datasets, whereas HHO and ADENS provide the best accuracy for one dataset. The result indicates that different algorithms can achieve the best accuracy for different datasets.

5. DISCUSSION

In subsection 4.1, we compare the performance of NCDDE using three different mutation strategies: DM, SM, and AM that adaptively uses both discrete and sorting mutations. The discrete mutation requires more function evaluations, while the sorting mutation provides the smallest number of function evaluations but cannot provide the best intra-cluster distance. The adaptive mutation can achieve the best intra-cluster distance with the number of function evaluations between those used by the discrete and the sorting mutations.

Subsection 4.2 presents a performance comparison of the NCDDE algorithm with ADEAS, HHO, AMADE, ADEANS, and DSDE algorithms by intra-cluster distance on eight datasets. The NCDDE algorithm is competitive and can provide the lowest intra-cluster distance for four datasets. In subsection 4.3, we compare the NCDDE algorithm's performance with the other algorithms in clustering accuracy. The NCDDE has the highest clustering accuracy for three datasets, and each of the compared algorithms has the highest clustering accuracy on some datasets.

The NCDDE searches for cluster labels without using centroids and can provide non-spherical shape clusters. Figure 3 shows the clustering results obtained by the non-centroid-based NCDDE algorithm and the centroid-based ADEAS algorithm on the dataset of 36 data points. They divide the data into 4 clusters represented by different colors and the centroid markers "x". NCDDE gives the elliptic-liked shape clusters and intra-cluster distance equal to 43.68, lower than the 43.76 of ADEAS.

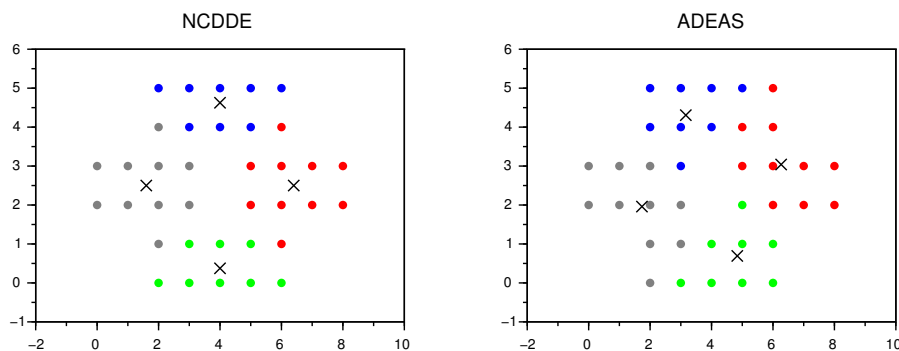


Figure 3. Clustering results of non-centroid-based and centroid-based algorithms

6. CONCLUSION

We have presented a NCDDE for data clustering. The algorithm directly searches for the group labels for each data and uses no representative centers. It combines discrete and sorting mutations and adaptively chooses high and low ranges crossover rates. Comparison experiments of NCDDE with ADEAS, HHO, AMADE, ADEANS, and DSDE show that NCDDE can provide lower intra-cluster distances on several UCI datasets and achieves the best clustering accuracy for Cancer, CMC, and Thyroid. In addition, the proposed algorithm is flexible and can give non-spherical clusters. For future work, we will apply NCDDE to real-world, large-scale datasets and explore its effectiveness in computer vision applications such as image segmentation and medical image analysis.

ACKNOWLEDGEMENT

This research received partial funding from the Fundamental Fund of Khon Kaen University and the National Science, Research, and Innovation Fund (NSRF). The author thanks the Science Achievement Scholarship of Thailand for the financial support.




REFERENCES

- [1] M. V. Nichita, M. A. Paun, V. A. Paun, and V. P. Paun, "Image clustering algorithms to identify complicated cerebral diseases. Description and Comparison," *IEEE Access*, vol. 8, pp. 88434-88442, 2020, doi: 10.1109/ACCESS.2020.2992937.
- [2] M. Wang and W. Deng, "Deep face recognition with clustering based domain adaptation," *Neurocomputing*, vol. 393, pp. 1-14, 2020, doi: 10.1016/j.neucom.2020.02.005.
- [3] T. Bezdán *et al.*, "Hybrid fruit-fly optimization algorithm with K-means for text document clustering," *Mathematics*, vol. 9, no. 16, p. 1929, 2021, doi: 10.3390/math9161929.
- [4] R. L. Rose, T. G. Puranik, and D. N. Mavris, "Natural language processing based method for clustering and analysis of aviation safety narratives," *Aerospace*, vol. 7, no. 10, p. 143, 2020, doi: 10.3390/aerospace7100143.
- [5] Q. Cai, M. Gong, L. Ma, S. Ruan, F. Yuan, and L. Jiao, "Greedy discrete particle swarm optimization for large-scale social network clustering," *Information Sciences*, vol. 316, pp. 503-516, 2015, doi: 10.1016/j.ins.2014.09.041.
- [6] L. N. Ferreira and L. Zhao, "Time series clustering via community detection in networks," *Information Sciences*, vol. 326, pp. 227-242, 2016, doi: 10.1016/j.ins.2015.07.046.




- [7] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*, vol. 1, no. 14, Jun. 1967, pp. 281-297.
- [8] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66-73, 1992, doi: 10.1038/scientificamerican0792-66.
- [9] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4, Nov. 1995, pp. 1942-1948, doi: 10.1109/ICNN.1995.488968.
- [10] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341-359, 1997, doi: 10.1023/A:1008202821328.
- [11] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing*, vol. 8, no. 1, pp. 687-697, 2008, doi: 10.1016/j.asoc.2007.05.007.
- [12] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28-39, 2006, doi: 10.1109/MCI.2006.329691.
- [13] M. F. Ahmad, N. A. M. Isa, W. H. Lim, and K. M. Ang, "Differential evolution: A recent review based on state-of-the-art works," *Alexandria Engineering Journal*, vol. 61, no. 5, pp. 3831-3872, 2022, doi: 10.1016/j.aej.2021.09.013.
- [14] M. Pant, H. Zaheer, L. Garcia-Hernandez, and A. Abraham, "Differential Evolution: A review of more than two decades of research," *Engineering Applications of Artificial Intelligence*, vol. 90, p. 103479, 2020, doi: 10.1016/j.engappai.2020.103479.
- [15] W. L. Xiang, N. Zhu, S. F. Ma, X. L. Meng, and M. Q. An, "A dynamic shuffled differential evolution algorithm for data clustering," *Neurocomputing*, vol. 158, pp. 144-154, 2015, doi: 10.1016/j.neucom.2015.01.058.
- [16] S. K. Nayak, P. K. Rout, and A. K. Jagadev, "A cross mutation-based differential evolution for data clustering," *International Journal of Data Mining, Modelling and Management*, vol. 9, no. 1, pp. 17-38, 2017, doi: 10.1504/IJDM.2017.082571.
- [17] H. M. Mustafa, M. Ayob, M. Z. A. Nazri, and G. Kendall, "An improved adaptive memetic differential evolution optimization algorithms for data clustering problems," *PLoS One*, vol. 14, no. 5, p. e0216906, 2019, doi: 10.1371/journal.pone.0216906.
- [18] O. Tarkhaneh and I. Moser, "An improved differential evolution algorithm using Archimedean spiral and neighborhood search based mutation approach for cluster analysis," *Future Generation Computer Systems*, vol. 101, pp. 921-939, 2019, doi: 10.1016/j.future.2019.07.026.
- [19] T. Poonthong, P. Puphasuk, and J. Wetweeraopong, "Adaptive differential evolution with archive strategy for solving partitioned clustering problems," *Computer Science*, vol. 19, no. 3, pp. 705-714, 2024.
- [20] L. Abualigah et al., "Hybrid Harris hawks optimization with differential evolution for data clustering," in *Metaheuristics in Machine Learning: Theory and Applications*, Cham: Springer International Publishing, pp. 267-299, 2021, doi: 10.1007/978-3-030-70542-8_12.
- [21] T. Singh, N. Saxena, M. Khurana, D. Singh, M. Abdalla, and H. Alshazly, "Data clustering using moth-flame optimization algorithm," *Sensors*, vol. 21, no. 12, p. 4086, 2021, doi: 10.3390/s21124086.
- [22] M. Sharma and J. K. Chhabra, "An efficient hybrid PSO polygamous crossover based clustering algorithm," *Evolutionary Intelligence*, vol. 14, no. 3, pp. 1213-1231, 2021, doi: 10.1007/s12065-019-00235-4.
- [23] T. Singh, S. S. Panda, S. R. Mohanty, and A. Dwibedy, "Opposition learning based Harris hawks optimizer for data clustering," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 7, pp. 8347-8362, 2023, doi: 10.1007/s12652-021-03600-3.
- [24] K. Bache and M. Lichman, "UCI machine learning repository," University of California, Irvine, School of Information and Computer Science, 2013. [Online]. Available: <http://archive.ics.uci.edu/ml>. (Accessed: Oct. 13, 2024).
- [25] P. Puphasuk and J. Wetweeraopong, "An enhanced differential evolution algorithm with adaptation of switching crossover strategy for continuous optimization," *Foundations of Computing and Decision Sciences*, vol. 45, no. 2, pp. 97-124, 2020, doi: 10.2478/fcds-2020-0007.

BIOGRAPHIES OF AUTHORS



Tanapon Poonthong    completed M.Sc. degree in Applied Mathematics from Khon Kaen University, Thailand in 2018. He is a Ph.D. student in Applied Mathematics, Khon Kaen University. His research area is optimization. He can be contacted at email: Tanapon.p@kku.ac.th.



Jeerayut Wetweeraopong    completed M.Sc. degree in Mathematics from West Virginia University, US in 1995 and Ph.D. degree in Mathematics from Khon Kaen University, Thailand in 2012. He is an assistant professor at Department of Mathematics, Khon Kaen University. He has been doing research in field of scientific computing and optimization. He can be contacted at email: wjeera@kku.ac.th.