International Journal of Mathematics and
Computer Science, **19**(2024), no. 3, 705–714

Adaptive differential evolution with archive strategy for solving partitional clustering problems

Tanapon Poonthong, Pikul Puphasuk, Jeerayut Wetweerapong

Department of Mathematics
Faculty of Science
Khon Kaen University
Khon Kaen 40002, Thailand

email: tanapon.p@kkumail.com, ppikul@kku.ac.th, wjeera@kku.ac.th

(Received December 1, 2023, Accepted January 9, 2024,
Published February 12, 2024)

Abstract

Clustering is an essential data exploration technique applied to many disciplines and applications such as data mining, image processing, bioinformatics, and machine learning. A clustering method identifies hidden patterns in a dataset and combines similar data points into clusters. The problems are challenging when they have many data points, attributes, and cluster partitions. In this paper, we propose an adaptive differential evolution with an archive strategy (ADEAS) to find candidate centroids and minimize their intra-cluster distance for solving partitional clustering problems. The archiving strategy stores inferior solutions during the selection operation to increase population diversity and create directions for guiding the search. We validate the proposed algorithm with several well-known methods using the UCI datasets. The results show that ADEAS outperforms the compared methods.

Key words and phrases: Adaptive differential evolution, Data clustering, Archive strategy.
AMS (MOS) Subject Classifications: 65K10, 68T10, 90C26.
ISSN 1814-0432, 2024, http://ijmcs.future-in-tech.net

1 Introduction

Data clustering is a data mining technique for grouping data objects so that objects in the same group are more similar to each other than objects in different groups. It has been used in many fields to identify patterns and relationships within the data. The partition-based clustering methods divide the data using similarity measures or validation indices to find optimal centroids. The intra-cluster distance is a widely used objective, and the K-means is a well-known method that iteratively improves the centroids. However, the clusters produced by K-means are sensitive to the initial centroids. Therefore, researchers have introduced evolutionary optimizations such as Genetic Algorithm (GA) [1], Differential Evolution (DE) [2], Ant Colony Optimization (ACO) [3], and Particle Swarm Optimization (PSO) [4] to find the optimal cluster solutions. Here, we propose an adaptive differential evolution algorithm with an archive strategy (ADEAS) for solving partitional clustering problems. The algorithm creates an archive for storing inferior solutions during DE's selection and reuses them in the mutation process to improve population diversification and searchability. Our contribution is the improved clustering method that can give optimal cluster partitions for various real-world problems.
The paper is organized as follows: In Section 2, we describe the clustering problems and review the evolutionary algorithms for data clustering. In Section 3, we introduce the adaptive differential evolution with an archive strategy (ADEAS). In Section 4, we present preliminary and comparison experiments using the UCI datasets. The performance comparison of ADEAS with other methods is shown in Section 5. Finally, we provide a conclusion in the last section.

2 Literature review

Let $S = \{x_1, x_2, \ldots, x_n\}$ be the set of n data points and k be the number of clusters. The partition-based clustering algorithm finds the optimal clusters with corresponding centroids by using some similarity measures. To measure the distance between two points, we use the Euclidean distance defined as follows:

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

where d is the dimension of each data point. Let $C = \{c_1, c_2, \ldots, c_k\}$ be a set of candidate centroids. The data points are assigned to the nearest cluster center c_i to obtain the set $Q = \{Q_1, Q_2, \ldots, Q_k\}$ of clusters corresponding to C. To evaluate Q, we use the intra-cluster distance calculated by

$$f(Q) = \sum_{i=1}^{k} \sum_{x \in Q_i} D(c_i, x)$$

The low value of $f(Q)$ indicates a suitable clustering result. Minimizing the intra-cluster distance over all possible clusters is challenging because this objective function is highly non-linear, multimodal, and non-separable. Thus various evolutionary optimization algorithms have been proposed. We focus on differential evolution algorithms.

Many researchers proposed adaptive differential evolution algorithms for data clustering. Xiang [5] presented the differential algorithm with a shuffled strategy (DSDE) to improve the clustering quality by randomly separating the population into two subpopulations. Both subgroups use DE/best/1 mutation strategy and merge at the end of each generation. The results of minimizing intra-cluster distances show that DSDE outperforms ACO, ABC, PSO, and PSOAG on the UCI datasets. Nayak et al. [6] proposed the cross-mutation-based differential evolution (CMDE) that combines two mutation strategies (DE/rand/1 and DE/best/1) for data clustering. The weight of using the best vector decreases from one at the start to zero at the maximum generation. Using intra-cluster distance, the CMDE outperforms DE/rand/1 and DE/best/1 and is competitive with the DSDE method on the UCI datasets. Alswaitti et al. [7] introduced the variance-based differential evolution algorithm (VDEO) for data clustering. The algorithm creates a mutant vector by randomizing five vectors for the switchable mutations between DE/rand/1 and DE/best/1. It calculates the data variances in each attribute dimension and uses them as different scaling factors (for each component) in the mutation process. The results on the UCI datasets show that the strategy provides solutions with less intra-cluster distance than DSDE and FSDE. Tarkhaneh and Moser [8] proposed the improved differential evolution algorithm using the Archimedean Spiral and neighborhoods search-based mutation approach for cluster analysis (ADENS). It uses the scaling factor in the range according to the number of function evaluations to generate solutions that replace poorly performing ones. The algorithm outperforms ICSK, DE-KM, and DSDE in minimizing intra-cluster distance. Wu et al. [9] presented the adaptive differential evolution called ACODE for data clustering. The algorithm combines the ant colony strategy to select three mutation strategies (DE/rand/1, DE/current-to-pbest/1, and DE/current-to-rand/1) and two crossover strategies (binomial and exponential). The results show that ACODE outperforms CSO, SL-PSO, DSDE, and EPSDE algorithms using intra-cluster distance. Sharma and Chhabra [10] proposed the PSO algorithm with polygamous crossover (PSOPC) that mates the best particle with other random particles using the arithmetic crossover to refine the exploration and exploitation strategy. The results on the UCI datasets show that the algorithm provides less intra-cluster distance than PSO, GA, FA, and GWO. However, these algorithms can only provide the best solutions for some problems in the UCI datasets. We propose using the adaptive differential evolution with archive strategy (ADEAS) that stores inferior population vectors to increase population diversity and enhance searchability.

3 The proposed ADEAS method

The ADEAS modifies the DEASC algorithm introduced in 2020 [11] to solve continuous optimization problems. It is an enhanced differential evolution algorithm by the switching crossover rate values CR in $[0, 0.1]$ or $[0.9, 1]$ and using scaling factor values F in $[0.5, 0.7]$. The algorithm can solve general well-known test functions with fast convergence speed and performs well for high dimensional test functions. The ADEAS adds the archive strategy that stores inferior solutions during selection and reuses them in the mutation process. The proposed algorithm can be described as follows:

1. Input and control parameters:
   Number of attributes of a data set: A, number of data clusters: K, dimension: $D = K \times A$, objective function to be minimized: intra-cluster distance function f, population size: $NP = 100$, maximum number of function evaluations: $maxnf$, scaling factor: F in the range of $[0.5, 0.7]$, crossover rate: CR in the range of $[0, 0.1]$ or $[0.9, 1]$, archive: Q of size NQ, the initial probabilities for using low or high crossover rates: $pc_1 = 0.1$ and $pc_2 = 0.9$, the initial counters corresponding to pc_1 and $pc_2 : nc_1 = nc_2 = 0$.

2. Initialization: Initial the population $P = [x_i]$ for $i = 1, 2, \ldots, NP$ where $x_i = [y_{11}, \ldots, y_{1A}, \ldots, y_{K1}, \ldots, y_{KA}]$ and $[y_{j1}, \ldots, y_{jA}]$ is a random data point for $j = 1, 2, \ldots, K$. Calculate the fitness values $f(x_i)$ and record the best vector and best fitness value, denoted by x_{best} and f_{best}, respectively.

3. Archive initialization: Initial the archive Q of size NQ with the random vectors the same way as P. Use Q in the mutation to diversify and guide the search and use it in the selection to store the inferior solutions. Set the index $iq = 0$ to update the archive Q.

4. Mutation: For each target vector x_i, choose distinct random vectors x_{r1} and $x_{r2} \in P$ different from x_i, and $x_q \in Q$. Create the mutant vector v_i by $v_i = x_{r1} + F \cdot (x_{r2} - x_q)$, where F is a random real number in the range $[0.5, 0.7]$. If a component of v_i is out of bound, then randomly adjust it to the bound.

5. Crossover: Generate a random number rd in $[0, 1]$. If $rd < pc_1$, then randomize CR in the range of $[0, 0.1]$; otherwise, randomize CR in the range of $[0, 9, 1]$. Create the trial vector u_i as follows.

$$u_{i,j} = \{ \begin{array}{l} v_{i,j};\ \text{if } R_j \leq CR \text{ or } j = I_i \\ x_{i,j};\ \text{otherwise,} \end{array} .$$

   where R_j is a random real value in $[0, 1]$ and I_i is a randomly fixed integer from $j = 1, 2, 3, \ldots, D$ used to guarantee a change of at least one component.

6. Selection and updating of the archive: Replace x_i with u_i if the fitness value of u_i is better than that of x_i and store x_i in the archive Q as the element at the index iq. Increment iq by 1 (modulo NQ). If the fitness value of u_i is also better than f_{best}, update x_{best} and f_{best}.

7. Updating control parameters: Increase nc_1 by 1 if a better solution is randomized with CR in the range $[0, 0.1]$, otherwise, increase nc_2 by 1 when CR is in the range $[0.9, 1]$. If $nc_1 + nc_2 \geq 100$, add 10 to both nc_1 and nc_2 to prevent them from being 0. Then, update pc_1 and pc_2 using the formulas $pc_1 = 0.9pc_1 + 0.1nc_1/(nc_1 + nc_2.)$ and $pc_2 = 1 - pc_1$, respectively. Reset nc_1 and nc_2 to 0 after updating pc_1 and pc_2.

8. Stopping condition: Repeat steps $4 - 7$ until the f_{best} value has stayed the same for the specified period of generations.

4 Experimental Designs

We conduct two experiments: a preliminary experiment to find the suitable setting for the ADEAS method and another one to compare ADEAS using the obtained setting with six other selected methods. Both methods are performed on nine UCI datasets. The datasets indicate the number of instances, attributes, and classes in Table 1. The optimal values present the best-known lowest intra-cluster distance values.

Table 1: The description of UCI datasets.

Name	Instances	Attributes	Classes	Optimal values
Iris	150	4	3	96.65548
Wine	178	13	3	16292.18464
Glass	214	9	6	210.00105
Thyroid	215	5	3	1866.46617
Haberman	306	3	2	2566.98890
Balance scale	625	4	3	1423.82040
Cancer	683	9	2	2964.38697
Vowel	871	3	6	148967.24081
Cmc	1473	9	3	5532.18472

4.1 Finding the Suitable Setting for ADEAS

The first experiment uses population size $NP = 100$ and archive size $NQ = 4 \cdot NP$. The algorithm stops when the number of function evaluations reaches $maxnf = 200000 \cdot NP$, or f_{best} is the same for 1000 consecutive generations. We explore the search capabilities of DEASC [11] with and without the archive Q and combine two initialization methods: I1 and I2. Method I1 randomly creates the centroids with values in the ranges of data attributes, while Method I2 creates the centroids from the data points. Thus the experiment consists of four combination methods: DEASCI1, DEASCI2, ADEASI1, and ADEASI2. The performance comparison between these four variants and the classical DE algorithm ($F = 0.5, CR = 0.9$) on the Iris, Wine, Glass, and Vowel datasets uses 30 independent runs for each method. We record the number of successful runs that match the optimal intra-cluster distance values in Table 1.

4.2 Comparing ADEAS with Other Optimization Algorithms

The second experiment compares the performance of ADEAS with other optimization algorithms on the Iris, Wine, Glass, Thyroid, Haberman Balance scale, Cancer, Vowel, and CMC datasets. The compared methods are

PSOPC, ACODE, ADENS, VDEO, CMDE, and DSDE methods [5–10]. We take their results reported in the respective original papers. The ADEAS uses the suitable setting from Experiment 1 and performs 30 independent runs. We record the number of successful runs (NS), the mean of best function values (Mean_fb), the mean of function evaluations (Mean_nf), the standard deviation (SD), and the percentage standard deviation (%SD).

5 Experimental results

5.1 Suitable Setting for ADEAS

The results of the preliminary experiment are reported in Table 2. The table shows the mean and the percentage of the standard deviation of the lowest intra-cluster distance values, function evaluations, and the number of successful runs for each dataset.

Table 2: Performance comparison of DE, DEASCI1, DEASCI2, ADEASI1, and ADEASI2 over 30 independent runs.

Name	Statistics	DE	DEASCI1	DEASCI2	ADEASI1	ADEASI2
Iris	NS	**30**	**30**	**30**	**30**	**30**
	Mean_fb	96.65548	96.65548	96.65548	96.65548	96.65548
	SD	0.00	0.00	0.00	0.00	0.00
	Mean_nf	27884.80	77472.50	75929.40	67495.67	86738.77
	%SD	10.98	19.24	17.14	19.24	14.44
Glass	NS	0	14	19	1	**27**
	Mean_fb	213.27063	210.188016	210.0393	210.41496	210.03490
	SD	0.93	0.10	0.05	0.04	0.11
	Mean_nf	154142.27	3323269.37	1374300.50	731463.87	903432.87
	% SD	8.82	17.03	52.59	15.97	13.85
Thyroid	NS	3	23	**30**	26	**30**
	Mean_fb	1887.833	1868.23515	1866.46617	1868.842	1866.46617
	SD	0.38	0.32	0.00	0.00	0.00
	Mean_nf	48395.87	109700.87	88165.83	109565.17	100613.67
	% SD	5.78	16.90	15.70	17.66	17.00
Vowel	NS	20	**30**	**30**	**30**	**30**
	Mean_fb	148973.3652	148967.2408	148967.2408	148967.2408	148967.2408
	SD	0.02	0.00	0.00	0.00	0.00
	Mean_nf	99544.23	1152481.40	874733.77	330985.90	364567.97
	% SD	17.01	55.74	21.13	11.65	12.78

We focus on the maximum number of successful runs, highlighted in bold. Each algorithm achieves all 30 successful runs for Iris. ADEASI2 obtains 27 successful runs for Glass and outperforms the others. DEASCI2 and ADEASI2 are comparable and give the best performance for Thyroid. In the case of Vowel, all algorithms accomplish a maximum of 30 successful runs except for DE. From these results, we select ADEASI2, using the archive Q and creating the initial centroids from the data point, as the suitable setting for our proposed algorithm and use it in the second comparison experiment.

5.2 Performance Comparison of ADEAS with Other Optimization Algorithms

Table 3 reports the comparison results of our proposed method and six well-known evolutionary methods on UCI datasets. The table presents the lowest value of Best, Mean_fb, and SD in bold text. The unavailable results of some methods on some datasets are indicated by “ - ”. All algorithms give the same results for Iris and Haberman. ADEAS, PSOPC, ACODE, and DSDE achieve the lowest of best values for Wine, whereas ADEAS gives the lowest Mean_fb and SD values. ADEAS demonstrates lower Best, Mean_fb, and SD values than other algorithms for Glass. ADEAS, ACODE, VDEO, and DSDE achieve the lowest of best values for Thyroids, whereas ADEAS gives the lowest Mean_fb and SD values. ADEAS, ACODE, and VDEO achieve the lowest of best values for the Balance scale dataset, whereas ADEAS gives the lowest Mean_fb and SD values. All algorithms give the best results for Cancer except for VDEO. ADEAS and DSDE achieve the lowest of best values for the Vowel dataset, whereas ADEAS gives the lowest Mean_fb and SD values. ADEAS, PSOPC, and ADENS achieve the lowest of best values for CMC, whereas CMDE, DSDE, and ADEAS give the lowest Mean_fb and SD values. Consequently, ADEAS outperforms the six compared methods by achieving the lowest values for Best, Mean_fb, and SD on all nine datasets.

6 Discussion

The ADEAS algorithm randomly selects initial centroids from data points so they are near some data points from the beginning, while the initial centroids with random values may be far from all data points. This initialization method also prevents the cluster with no data points. ADEAS employs the archive strategy that stores inferior solutions to enhance population diversity and create directions for guiding the search where different vectors from the archive to current population solutions tend to approach optimal solutions. The algorithm uses the directions to generate the mutation vectors. As a result, ADEAS can find the best intra-cluster distances for all nine datasets by the stopping condition that allows the population to generate new candidate solutions.

Table 3: Performance comparison of PSOPC, ACODE, ADENS, DSDE, VDEO, DSDE, and ADEAS algorithms.

Data	Stat	Methods						
		PSOPC	ACODE	ADENS	VDEO	CMDE	DSDE	ADEAS
Iris	Best	**96.66**	**96.66**	**96.66**	**96.66**	**96.66**	**96.66**	**96.66**
	Mean_fb	**96.66**	**96.66**	**96.66**	**96.66**	**96.66**	**96.66**	**96.66**
	SD	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Wine	Best	**16292.18**	**16292.18**	16292.19	16292.43	16292.2	**16292.18**	**16292.18**
	Mean_fb	16292.54	16292.23	16292.67	16293.56	16292.2	16292.39	**16292.18**
	SD	0.61	0.15	0.60	0.87	0.00	0.25	**0.00**
Glass	Best	210.43	210.43	-	210.40	-	210.05	**210.00**
	Mean_fb	219.28	213.25	-	213.62	-	212.73	**210.03**
	SD	11.32	5.71	-	1.99	-	1.68	**0.11**
Thyroid	Best	-	**1866.47**	-	**1866.47**	1908.96	**1866.47**	**1866.47**
	Mean_fb	-	1879.14	-	1867.51	1908.96	1874.00	**1866.47**
	SD	-	12.04	-	0.91	0.00	11.77	**0.00**
Balance scale	Best	-	**1423.82**	-	1423.83	-	-	**1423.82**
	Mean_fb	-	1423.89	-	1425.13	-	-	**1423.82**
	SD	-	0.12	-	0.88	-	-	**0.00**
Cancer	Best	**2964.39**	-	**2964.39**	2964.41	**2964.39**	**2964.39**	**2964.39**
	Mean_fb	**2964.39**	-	**2964.39**	2964.43	**2964.39**	**2964.39**	**2964.39**
	SD	**0.00**	-	**0.00**	0.02	**0.00**	**0.00**	**0.00**
Haber-man	Best	**2566.99**	-	-	**2566.99**	-	-	**2566.99**
	Mean_fb	**2566.99**	-	-	**2566.99**	-	-	**2566.99**
	SD	**0.00**	-	-	**0.00**	-	-	**0.00**
Vowel	Best	-	-	-	149073.62	149946.00	**148967.24**	**148967.24**
	Mean_fb	-	-	-	149682.56	149946	149193.97	**148967.24**
	SD	-	-	-	615.17	0.00	373.45	**0.00**
CMC	Best	**5532.18**	-	**5532.18**	-	**5532.18**	**5532.18**	**5532.18**
	Mean_fb	5532.20	-	5532.22	-	**5532.18**	**5532.18**	**5532.18**
	SD	0.01	-	0.05	-	**0.00**	**0.00**	**0.00**

7 Conclusion

We have presented an adaptive differential evolution with an archive strategy for solving partitional clustering problems. The algorithm employs the archive strategy to enhance population diversity and improve searchability. Experimental results show that the ADEAS outperforms the classic DE and several well-known population-based algorithms. ADEAS can find the best intra-cluster distances for UCI datasets and becomes a tool for data analysis, pattern recognition, and knowledge discovery.

Acknowledgment

Tanapon Poonthong thanks the Science Achievement Scholarship of Thailand for financial support. This research was partially supported by the Fundamental Fund of Khon Kaen University and the National Science, Research and Innovation Fund (NSRF).

References

[1] J. H. Holland, Genetic algorithms, Scientific American, **267**, no. 1, (1992), 66–73.

[2] R. Storn, K. Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, Journal of global optimization, **11**, (1997), 341–359.

[3] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, IEEE Computational Intelligence Magazine, **1**, no. 4, (2006), 28–39.

[4] J. Kennedy, R. Eberhart, Particle swarm optimization, Proceedings of the International Conference on Neural Networks, **4**, (1995), 1942–1948.

[5] W. L. Xiang, N. Zhu, S. F. Ma, X. L. Meng, M. Q. An, A dynamic shuffled differential evolution algorithm for data clustering, Neurocomputing, **158**, (2015), 144–154.

[6] S. K. Nayak, P. K. Rout, A. K. Jagadev, A cross mutation-based differential evolution for data clustering, International Journal of Data Mining, Modelling and Management, **9**, no. 1, (2017), 17–38.

[7] M. Alswaitti, M. Albughdadi, N. A. M. Isa, Variance-based differential evolution algorithm with an optional crossover for data clustering, Applied Soft Computing, **80**, (2019), 1–17.

[8] O. Tarkhaneh, I. Moser, An improved differential evolution algorithm using Archimedean spiral and neighborhood search based mutation approach for cluster analysis, Future Generation Computer Systems, **101**, (2019), 921–939.

[9] G. Wu, W. Peng, X. Hu, R. Wang, H. Chen, Configuring differential evolution adaptively via path search in a directed acyclic graph for data clustering, Swarm and Evolutionary Computation, **55**, (2020), 100690.

[10] M. Sharma, J. K. Chhabra, An efficient hybrid PSO polygamous crossover based clustering algorithm, Evolutionary Intelligence, **14**, no. 3, (2021), 1213–1231.

[11] P. Puphasuk, J. Wetweerapong, An enhanced differential evolution algorithm with adaptation of switching crossover strategy for continuous optimization, Foundations of Computing and Decision Sciences, **45**, no. 2, (2020), 97–124.