Dynamic Partitional Clustering Using Evolution Strategies

C.-Y. Lee

Department of Mechanical Engineering
California Institute of Technology
MC 104-44
1200 E. California Blvd.
Pasadena, CA 91125, U.S.A.
cinyoung@caltech.edu

E.K. Antonsson

Department of Mechanical Engineering
California Institute of Technology
MC 104-44
1200 E. California Blvd.
Pasadena, CA 91125, U.S.A.
erik@design.caltech.edu

Abstract

A novel evolution strategy implementing variable length genomes is developed to address the problem of dynamic partitional clustering. As opposed to static, dynamic partitional clustering does not require the a priori *specification of the number of clusters. Results of the algorithm are presented and discussed for 2-D touching and non-touching cluster test cases.*

1 Introduction

An important aspect of data analysis is the determination of data structures or, equivalently, pattern discovery. For small data sets, human beings are quite adept at identifying these patterns. However, as the amount and dimensionality of data grows beyond the grasp of human minds, automation of pattern discovery becomes crucial.

Here, only automation of partitional clustering (shortened to clustering in the sequel) is addressed. Clustering divides, or partitions, a data set into regions of high similarity, as defined by some distance metric, called clusters. In most instances, a cluster is identified by a prototypical vector called the cluster center. Hence, the problem of cluster optimization is twofold: optimization of (1) cluster centers and (2) number of clusters. The latter aspect has often been neglected in previous approaches as these approaches typically fix the number of clusters *a priori*. For example, the popular k-means clustering technique requires prior specification of the number of clusters. K-means also has the disadvantage of being a gradient descent-like search such that it is easily trapped in local optima. More robust algorithms, based on evolutionary algorithms, have been developed as a result. Still, these approaches have required prior specification of cluster number.

To combat this shortcoming of fixing cluster number *a priori*, a novel evolution strategy (ES) is developed. The proposed ES implements variable length genomes that allow the algorithm to effectively search for both optimal cluster center positions and cluster number. Note that, since cluster number is optimized during run time, such clusterings are referred to as 'dynamic'. The remainder of the paper is organized to: (i) briefly discuss previous evolutionary clustering approaches, (ii) develop the variable length ES for dynamic clustering, (iii) present preliminary results of the proposed ES on two test cases for which the clusterings are known, and (iv) conclude with a summary and a discussion of future research directions.

2 Previous Evolutionary Algorithm Approaches

The genetic algorithm presented in [1] and most of the evolutionary approaches reviewed therein implement fixed length coding schemes that require the cluster number to be chosen *a priori*. Thus, for these methods to find optimal clusterings, they must be repeated for cluster numbers ranging from 1 to the number of data points, which is often an extremely time consuming process.

As opposed to the static clusterings described in [1], [2] introduces a dynamic clustering algorithm based on evolutionary programming. In this approach, cluster number is modified by simple insertion and deletion

operators that add or remove a random number of cluster centers from a clustering. Results of the algorithm are promising; however, the number of clusters is relatively small (≤ 5). Moreover, the presence of crossover as a search operator is absent.

3 Dynamic Clustering Using Evolution Strategies

The proposed evolution strategy is an application of the extended genome (exG) representation introduced in [3]. exG was developed specifically to tackle problems of variable complexity. In this case, cluster number gives rise to a variable complexity search. Details of the extension of exG to the clustering problem follow.

3.1 Problem Description

The proposed evolution strategy is developed to solve 2-D spatial data clustering problems, primarily, because the results of 2-D clusterings are easily interpreted graphically. This does not represent a limitation of the proposed ES, since the ES is easily generalizable to any dimension as will be seen later.

3.2 Coding scheme

In accordance with the problem's 2-D nature, each gene or cluster center is an ordered pair denoted as (x, y). Individual genes comprise the genome to make a complete clustering. The genes are ordered within a genome by ascending x value. A three cluster genome can then be written as $[(x_1, y_1), (x_2, y_2), (x_3, y_3)]$ with $x_1 \leq x_2 \leq x_3$. Also encoded by each genome are self-adaptive mutation parameters (one for each data dimension).

3.3 Initialization

The step wise initialization procedure is given as

FOR each population member,
  Randomly choose genome length, n,
  uniformly from 10 to 35.
  FOR each cluster, n,
    Randomly select cluster center inside
    data bounds.
  Order the genome.

While the range of cluster numbers is specified *a priori*, there are no restrictions on the evolved number of clusters since the search operators, to be discussed subsequently, are able to adapt cluster number. A conservative estimate of the cluster number range has been chosen in this case.

3.4 Search operators

Mutation of genes follows the guidelines of standard ES's in which a Gaussian random variable with zero mean and standard deviation σ_x or σ_y perturbs both the x and y values of every cluster center/gene. σ_x and σ_y are the self-adaptive mutation parameters and are adjusted as in [4].

In contrast to [2], crossover is used as the length changing operator. The reason for this is that the developed crossover operator can duplicate any deletion or insertion-like length change. In canonical two point crossover, a range of genes is selected over which to swap. This remains the same here except that the range is given by a range of x values over which to swap. Thus, if two parents have a different number of genes within the range, both genomes will exhibit length changes after crossover. When only a single gene is swapped between genomes, a simultaneous insertion and deletion operation occurs. This shows the porposed crossover operator's ability to mimic the action of insertion and deletion.

3.5 Selection strategy

A (10+60) ES selection strategy is used, where 10 indicates the number of parents, or individuals in the population, and 60 is the number of offspring produced each generation. The 10 best individuals from the combined pool of parents and offspring are selected for survival each generation (denoted by the '+'). Offspring are created by mutating and recombining each parent twice. Hence, a total of six offspring is generated per parent genome per generation.

3.6 Fitness evaluation

A modification of the Mean Square Error (MSE) clustering measure is chosen as the fitness function. The MSE is expressed as

$$\text{MSE } fitness = \sum_{i=1}^{n} \sum_{j=1}^{m_i} d(c_i, x_j^i)$$

where n is the number of clusters, c_i is the ith cluster center, m_i is the number of data points belonging to the ith cluster, x_j^i is the jth data point belonging to the ith cluster, and $d(a, b)$ is the euclidean distance between points a and b. Data points are assigned membership to the cluster with the nearest cluster center.

Unfortunately, the above fitness function is poorly suited for comparing clusterings that have a different number of clusters. This can be seen by imagining the plot of optimal MSE as a function of cluster number. Since the optimal MSE can always be decreased by adding a data point as a cluster center, fitness is a monotonically decreasing function of cluster number. Obviously, the best clustering is not *simply* the data set (since no new information is gained if this is the case). The problem is that there needs to be a penalty for model complexity (*i.e.*, number of clusters). This tradeoff between model complexity and goodness of fit is well known in function approximation and learning system theories [5].

A variety of methods have been developed to address the problem of model complexity. Two of the more prevalent methods, for clustering at least, are the Davies and Bouldin index [6] and the principle of Minimum Description Length, or MDL [2]. The Davies and Bouldin index requires the calculation of the clustering scatter or variation in addition to the distances between cluster centers and their data members. MDL also necessitates extra calculations, as seen by its formulation

$$\text{MDL}(\omega) = -\log_2(f(\mathbf{x}|\omega)) + \frac{1}{2}n\log_2|\mathbf{x}|$$

where $\mathbf{x}$ is the observed data, $f(\mathbf{x}|\omega)$ is the conditional likelihood function given parameter vector ω, n is the number of parameters, and $|\mathbf{x}|$ is the number of data points.

Instead of using either of the above fitness measures, a heuristic measure is used, since both the Davies and Bouldin index and the MDL principle require too much overhead when compared to MSE. So, a heuristic MSE is chosen, given by

$$\text{MSE heuristic } fitness = \sqrt{n+1}\sum_{i=1}^{n}\sum_{j=1}^{m_i} d(c_i, x_j^i)$$

The heuristic penalizes model complexity by multiplying the MSE fitness by a constant proportional to the square root of the number of clusters. The penalization factor was chosen primarily because it provided good clustering results for a variety of data sets. However, the effectiveness of the penalization term is not without basis. To see this, recall the plot of MSE fitness versus cluster number. The curve exhibits a point where the slope changes dramatically since the addition of new cluster centers after this point has little effect on the MSE fitness measure. Intuitively, the optimal clustering occurs at this 'knee', as it provides the best performance per unit change in cluster number. By multiplying the curve by some function of cluster number, the 'knee' can be made into a minimum. The chosen penalization factor seems to be one such function. However, these functions are dependent on 'knee' location and curvature making the choice of such a function difficult.

3.7 Termination criteria

The evolution strategy is terminated either when the number of iterations exceeds 200 or when the average population fitness varies less than 0.001 from one generation to the next. As with the fitness penalization factor, the maximum number of iterations is chosen empirically. In most cases, after 200 generations, the self-adaptive mutation parameters are so small that significant improvements in the heuristic MSE no longer occur.

4 Results

The proposed evolution strategy is tested on two data sets for which the clusterings are known. In both cases, twenty cluster centers are used to generate the test data. For each cluster center, fifty data points are generated according to a Gaussian with the cluster center as the mean and a random number as the variance. The first test case distributes the cluster centers such that the resultant data set has no overlapping, or non-touching, clusters. Conversely, the second test case has overlapping, or touching, clusters. While overlap can take on many definitions, the distinction between no overlap and some overlap can clearly be seen in the test data as shown in Figures 1 and 4. Since these two test cases are representative of all clustered data sets, the performance of the proposed ES will be indicative of its effectiveness in dynamic clustering applications. For both test cases, the ES was run a total of eleven times. Results follow.

4.1 Non-touching clusters

The data and clustering for the non-touching cluster test case are shown in Figure 1. A typical evolved clustering is is shown in Figure 2. For comparison, Figure 3 shows plots of a typical evolved clustering and the actual clustering used to generate the data.

The fitness measure of the test data using the actual cluster centers is 84.6963. This compares to the average evolved fitness of 135.054 with standard deviation 95.2274. Furthermore, the average number of

clusters of the evolved clusterings was 19.4 with a standard deviation of 3.3. These results are promising. In particular, the best evolved solution had a fitness of 84.7245 and 21 clusters. As was the case whenever an evolved solution had more than 20 clusters, the 'extra' cluster(s) in this case was redundant having no assigned data members. Thus, when the redundant cluster is removed, the best fitness is in fact lower than the 'true' fitness of 84.6963.

The results are marred by a single evolved solution that performed significantly worse than the others. The fitness of this individual was 414.602 and the number of clusters was 10.

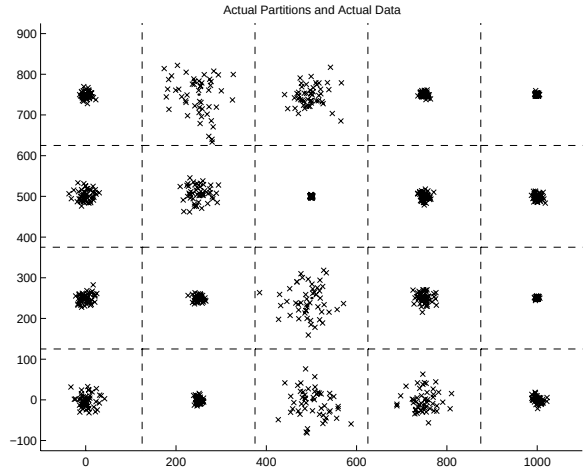

Figure 1: Actual data and partitions for non-touching case.

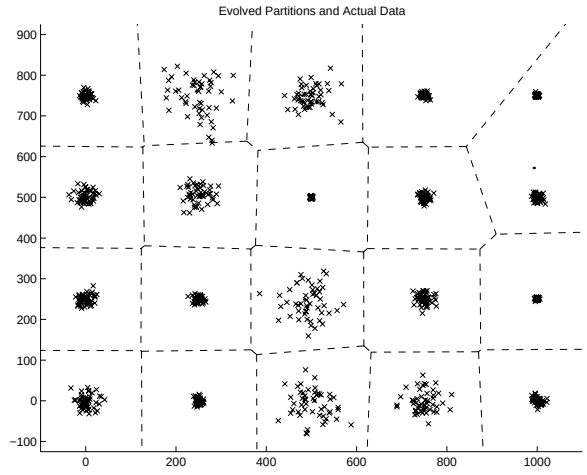

Figure 2: Actual data and typical evolved partitions for non-touching case.

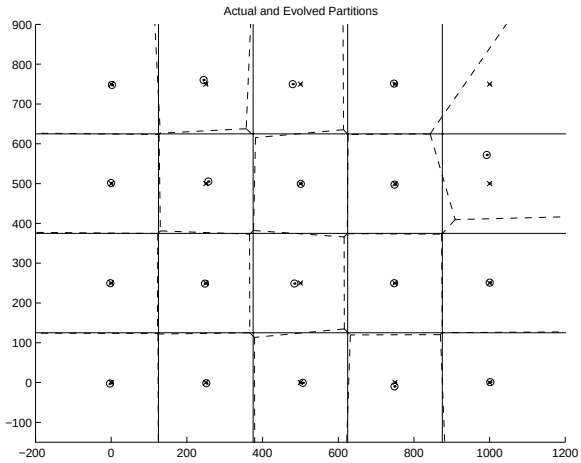

Figure 3: Actual (–) and typical evolved partitions (- -) for non-touching case. Actual (×) and typical evolved (⊙) cluster centers.

4.2 Touching clusters

Parallel to the results and figures shown for the non-touching test case, Figures 4–6 are the touching equivalents to Figures 1–3.

The fitness measure of the test data using the actual cluster centers is 245.927. The average evolved clustering fitness was 251.855, only marginally worse than the 'true' value, with a standard deviation of 18.3938. However, the average number of clusters was 13.6, significantly lower than the actual cluster number, with a standard deviation of 3.2. This is expected since several of the actual clusters overlap, leading to more ambiguous clusters. In any event, for more than half of the ES runs, the evolved solution was able to better the actual clustering's fitness. As compared to the non-touching case, evolved clusterings for the touching data never resulted in redundant clusters nor were there any outliers in the evolved clusterings. The best evolved clustering had 17 clusters and a fitness of 234.463. Conversely, the worst evolved solution had 7 clusters and a fitness of 285.52.

5 Conclusion and Future Work

A novel evolution strategy implementing variable length genomes has been developed for the application of dynamic partitioning. Results for 2-D spatial data are promising, as the evolved clusterings often surpassed the fitness measures of the 'true' clusterings for both the non-touching and touching test cases.

A number of issues regarding the proposed ES re-

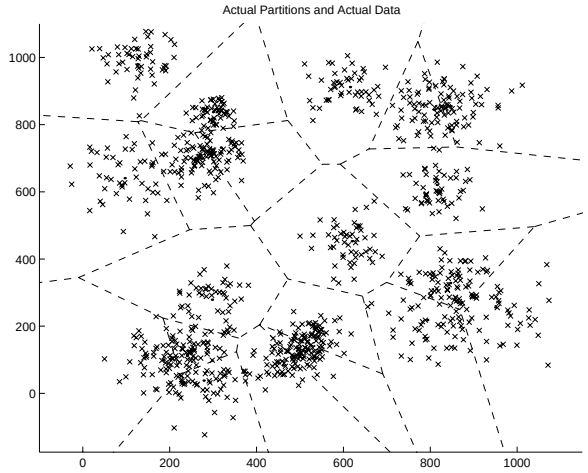

Figure 4: Actual data and partitions for touching case.

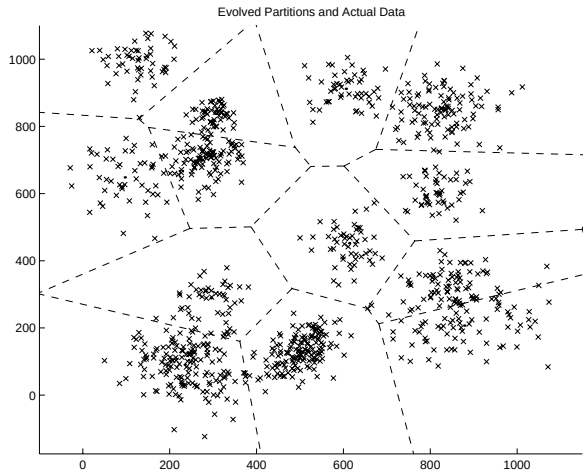

Figure 5: Actual data and typical evolved partitions for touching case.

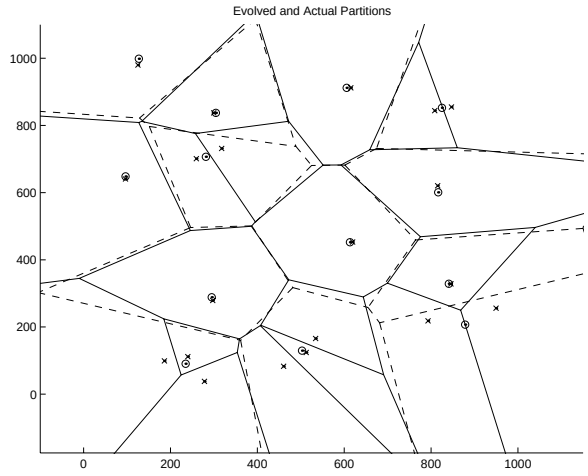

Figure 6: Actual (–) and evolved partitions (- -) for touching case. Actual ($\times$) and typical evolved ($\odot$) cluster centers.

main. In particular, the effectiveness of the proposed ES for dynamic clustering in dimensions greater than two needs to be addressed. Since the ES implementation is easily generalizable to any dimension, it is expected that the ES should remain effective as data dimension increases. However, the use of two point crossover may limit algorithm performance. The reason is that epistasis, or linkage, between good cluster centers quite likely will not be between adjacent genes in the genome. Thus, research on uniform crossover operators that can sidestep this problem is necessary.

A further issue is that of the fitness function. Since evolutionary algorithms are highly modular, any fitness function can be substituted in place of the MSE heuristic fitness measure used. Modification of the fitness function will result in new types of partitionings such as hyper-ellipsoids (as opposed to Voronoi diagrams). Since the proposed ES is developed as a general framework for dynamic partitioning, it would be interesting to observe its performance using different fitness criteria. Also, and perhaps most importantly, a comparison of the dynamic partitioning approach taken here with other approaches is necessary.

References

[1] L.O. Hall, B. Ozyurt, and J.C. Bezdek, "Clustering with a genetically optimized approach," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 2, pp. 103–112, 1999.

[2] A. Ghozeil and D.B. Fogel, "Discovering pat-

terns in spatial data using evolutionary programming," in *Genetic Programming 1996: Proceedings of the first annual conference*, Cambridge, MA, 1996, MIT Press, pp. 512–520.

[3] C.-Y. Lee and E.K. Antonsson, "Variable length genomes for evolutionary algorithms," in *Proceedings of the Genetic and Evolutionary Computation Conference*, Las Vegas, NV, 2000, Morgan Kaufmann, p. 806.

[4] H.-P. Schwefel, *Evolution and Optimum Seeking*, John Wiley, New York, 1995.

[5] V. Cherkassky and F. Mulier, *Learning from Data: Concepts, Theory, and Methods*, John Wiley, 1998.

[6] D.L. Davies and D.W. Bouldin, "A cluster separation measure," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 224–227, 1979.