Cluster Validity for the Fuzzy c-Means Clustering Algorithm

MICHAEL P. WINDHAM

Abstract—**The uniform data function is a function which assigns to the output of the fuzzy c-means (Fc-M) or fuzzy isodata algorithm a number which measures the quality or validity of the clustering produced by the algorithm.**

For the preselected number of cluster c, the Fc-M algorithm produces c vectors in the space in which the data lie, called cluster centers, which represent points about which the data are concentrated. It also produces for each data point c-membership values, numbers between zero and one which measure the similarity of the data points to each of the cluster centers. It is these membership values which indicate how the point is classified. They also indicate how well the point has been classified, in that values close to one indicate that the point is close to a particular center, but uniformly low memberships indicate that the point has not been classified clearly. The uniform data functional (UDF) combines the memberships in such a way as to indicate how well the data have been classified and is computed as follows. For each data point compute the ratio of its smallest membership to its largest and then compute the probability that one could obtain a smaller ratio (indicating better classification) from a clustering of a standard data set in which there is no cluster structure. These probabilities are then averaged over the data set to obtain the values of the UDF. By constructing the functional in this way one obtains a measure which is insensitive to parameters used to initialize and implement the Fc-M algorithm and respond only to differences in the quality of the clustering produced by the algorithm.

Index Terms—**Cluster validity functional, fuzzy clustering algorithms, fuzzy c-means (Fc-M), fuzzy sets, uniform data functional (UDF).**

I. Introduction

FUZZY CLUSTERING algorithms are mathematical tools for detecting similarities between members of a collection of objects. Perhaps the best known and most widely used member of the family is the fuzzy isodata or fuzzy c-means (Fc-M) algorithm developed by Dunn [1] and extended by Bezdek [2]. Information about the objects to be analyzed is input to the algorithm in the form of d-dimensional vectors. The vector used to represent a particular object has as its components the measurements of d features of the object which have been chosen as a basis for comparing it to other objects. The output of the algorithm can then be used to classify the data into subsets or clusters. Data vectors assigned to the same cluster are in some sense similar to each other, more so than they are to other data vectors not assigned to that cluster.

The primary concern with the use of this algorithm, as with any clustering algorithm, is how well has it identified the structure that is present in the data. This is the "cluster validity problem." We define here a function, the *uniform data functional* (UDF), which assigns to the output of the Fc-M algorithm a number, which measures the effectiveness with which the cluster structure has been properly identified, and show how it can be used to obtain the best clustering of the data.

Manuscript received April 9, 1981; revised January 11, 1982.

The author is with the Department of Mathematics, Utah State University, Logan, UT 84322.

II. Fuzzy Clustering Algorithms

We begin by describing what a fuzzy clustering is and how it is obtained using the Fc-M algorithm. A fuzzy clustering is best understood by contrasting it with the more common hard clustering of a data set. Let $X = \{x_1, \cdots, x_n\}$ be a set of n vectors in R^d, representing the data. For an integer $c \geq 2$, a *hard clustering* of X into c clusters consists of c disjoint subsets of X, $S_1, \cdots, S_c$ whose union is X. For example, if the data set consists of the measurement of d features of n flowers of the same species, a cluster S_i might contain those flowers of the same subspecies. Such a situation is considered in the examples presented below. An equivalent way of defining the clusters is obtained using functions. Namely, for each $i = 1, \cdots, c$ define u_i: $X \to \{0, 1\}$ by $u_i(x) = 1$ if $x \in S_i$ and $u_i(x) = 0$ if $x \notin S_i$. These functions are called membership functions, since they describe in which cluster each data point belongs. The hard clustering could then be described by these membership functions. A fuzzy clustering is a generalization of this point of view.

A *fuzzy clustering* of X into c clusters consist of functions $u_1, \cdots, u_c$ where u_i: $X \to [0, 1]$ and $\Sigma_i u_i(x) = 1$, for all $x \in X$. These functions are also called *membership functions*; however, they do not define subsets in the usual sense, but are, in fact, examples of fuzzy set as introduced by Zadeh [3]. The value of a fuzzy membership function can be any number between 0 and 1, and is meant to be a mathematical characterization of a "set" which may not be precisely defined. For example, consider the set of all people almost six feet tall. How does one determine membership in this set? Certainly, a person whose height is 5 ft 8 in would be more likely to be included in the set than a person who is only 5 ft tall. Zadeh proposed that rather than describe the set by its membership to describe it by a membership function, but allow the function to have values strictly between 0 and 1, if necessary, to indicate ambiguity which might be present in the concept which the set is to represent. So, the person who is 5 ft 8 in might be assigned a membership of 0.8 in the "set" of people almost 6 ft tall, whereas the 5 ft tall person might have a membership of 0.3.

0162-8828/82/0700-0357$00.75

So, the "clusters" of a fuzzy clustering are the membership functions themselves. They indicate the substructure of the data in the following sense. If two data points have membership values close to one for the same membership function than they are to be considered similar to each other. The condition $\Sigma_i u_i = 1$ corresponds to the membership of each data point in X. One can use a fuzzy clustering as the basis for a hard clustering by defining for $i = 1, \cdots, c$, $S_i = \{x \in X: u_i(x) \geqslant u_j(x) \text{ for } j = 1, \cdots, c\}$. It should be noted that these sets may not be disjoint. If a data point attains its maximum membership with more than one membership function it will be in more than one of these sets. Moreover, this "hard" clustering does not provide as much information about the structure in the data set as does the fuzzy clustering. Although two points may have their maximum membership for the same membership function, if one has a higher membership than the other, it is better classified by the clustering.

III. The Fuzzy c-Means Algorithm

The Fc-M algorithm is designed to produce a fuzzy clustering of a data set. Again, the Fc-M algorithm is best understood by contrasting it with an algorithm designed to produce hard clusters, the classical c-means algorithm. For a given integer $c \geqslant 2$, this algorithm chooses subsets $S_1, \cdots, S_c$ of the data set X, which minimize

$$\Sigma_i \Sigma_{x \in S_i} |x - v_i|^2$$

where $v_i \in R^d$ is the mean of the data vectors in S_i and $|x|$ denotes the Euclidean norm of a vector x, $(\Sigma_j x_j^2)^{1/2}$. In other words, it chooses the clusters to minimize the distances of the points in the clusters to the "center" of the cluster as represented by the mean. The assumption that is made in using this algorithm is that similarity between objects is measured by the distance between their corresponding data vectors. So, if the distances to the mean of a cluster are small the points in the cluster are also close to each other

The Fc-M algorithm produces a fuzzy clustering in much the same way. For $c \geqslant 2$ and m any real number greater than 1, the algorithm chooses u_i: $X \to [0, 1]$ so that $\Sigma_i u_i = 1$ and $v_i \in R^d$ for $i = 1, \cdots, c$ to minimize the objective function

$$\Sigma_i \Sigma_k (u_{ik})^m |x_k - v_i|^2$$

where u_{ik} is the value of the ith membership function on the kth data point x_k. Here again, the assumption is made that similarity of objects is measured by distance between data vectors. The vectors $v_1, \cdots, v_c$ can be interpreted as prototypes for the clusters represented by the membership functions, and are called *cluster centers*. In order to minimize the objective function, the cluster centers and membership functions are chosen so that high memberships occur for points close to the corresponding cluster centers. The number m is called the *exponent weight*. It can be chosen to "tune out" noise in the data. The higher the value of m used, the less those data points whose memberships are uniformly low contribute to the objective function. Consequently, such points tend to be ignored in determining the centers and membership functions.

The actual construction of the Fc-M algorithm is based on the following set of equations which are a necessary condition for $u_1, \cdots, u_c$ and $v_1, \cdots, v_c$ to produce a local minimum:

$$v_i = \Sigma_k (u_{ik})^m x_k / \Sigma_k (u_{ik})^m \tag{1}$$

$$u_{ik} = (1/|x_k - v_i|^2)^{1/(m-1)} / \Sigma_j (1/|x_k - v_j|^2)^{1/(m-1)}. \tag{2}$$

The necessity is obtained by differentiating the objective function with respect to the components of v_i and u_{ik} for $i = 1, \cdots, c$ and $k = 1, \cdots, n$ subject to the constraint that $\Sigma_i u_{ik} = 1$. If these equations could be solved in closed form, the solution would provide the fuzzy clustering directly. However, no closed form solution has been found, but these equations are the basis for an iterative procedure which converges to a local minimum for the objective function. One chooses a value for c and m and a $c \times n$ matrix U which is an initial guess for the values of the memberships. Using these memberships and (1), one computes cluster centers, then using these cluster centers and (2) recomputes memberships and so forth, iterating back and forth between (1) and (2) until the memberships or cluster centers for successive iteration differ by more than some prescribed value. That this will occur for at least a subsequence of the iterations and that the result will provide approximately a local minimum for the object function has been shown by Bezdek [4]. This iterative procedure is the Fc-M algorithm.

IV. Cluster Validity

A *validity functional* is a function which assigns to the output of Fc-M a number which is intended to measure the quality of the clustering provided by the output. By evaluating the functional on the output for a variety of choices of c and m, one hopes to be able to determine the values of these parameters for which the corresponding clustering best identifies the structure in the data.

The quality of a clustering is indicated by how closely the data points are associated to the cluster centers, and it is the membership functions which measure the level of association or classification. If the value of one of the memberships is significantly larger than the others for a particular data point, then that point is identified as being a part of the subset of the data represented by the corresponding cluster center. But, each data point has c memberships; so, it is desirable to summarize the information contained in the memberships by a single number which indicates how well the data point is classified by the clustering. This can be done in a variety of ways; for example, for the data point x_k with memberships $u_{ik}, \cdots, u_{ck}$, one could use any of the following:

$$\Sigma_i (u_{ik})^2$$

$$-\Sigma_i u_{ik} \log u_{ik}$$

$$\max_i u_{ik}$$

$$\min_i u_{ik} / \max_i u_{ik}.$$

In fact, the first three of these have been used as a measure of the quality of clustering and are the basis for the validity functionals *partition coefficient* [5], *classification entropy* [6], and *proportion exponent* [7], respectively.

To illustrate the use of a validity functional, we focus on the

partition coefficient. It is based on using $s_k = \Sigma_i (u_{ik})^2$ as a measure of how well the kth data point has been classified. This is a reasonable indicator because the closer a data point is to a cluster center, the closer s_k is to 1, the maximum value it could have. Conversely, the further away the kth point is from all the cluster centers the closer the value of s_k is to $1/c$, the minimum possible value. The partition coefficient is then the average over the data set of the s_k's. In particular, for a data set $X = \{x_1, \cdots, x_k\}$ and a specific choice of c and m one obtains the output of Fc-M and computes the *partition coefficient* (PC) by $\mathrm{PC} = \Sigma_k (\Sigma_i (u_{ik})^2)/n$. The closer this value is to one the better the data are classified. So, in theory, one computes PC for the outputs of a variety of values of c and m and selects the best clustering as the one corresponding to the highest partition coefficient.

In practice, however, this approach has not worked. The reason for this is that the values of PC are sensitive to the values of c and m independent of any structure in the data. This can be seen by comparing the partition coefficients for different clusterings of the data set pictured in Fig. 1. This is a two-dimensional set consisting of points uniformly distributed over the unit disk. This data set clearly does not have any substructure of the kind identified by Fc-M, but the values of PC vary significantly, as is indicated by Table I. This variation is due entirely to the values of the parameters. Consequently, one cannot select the best clustering by finding the highest partition coefficient.

The classification entropy, which averages $-\Sigma_i u_{ik} \log u_{ik}$ over the data suffers from the same kind of sensitivity to parameters. The proportion exponent attempted to overcome this difficulty by looking not at the values of the measure of quality directly, but by looking at how they compared to a standard. It was constructed using the $\max_i u_{ik}$. The higher the maximum is the better the point is classified. The maximum value itself was not used as an indicator of quality but rather the probability that one could do better by selecting memberships for the data at random. By obtaining such a probability for each of the data points and multiplying them together one obtains the probability that one could have produced a better clustering of the data by simply drawing the memberships "out of a hat." The lower the probability, the better the classification should be. The rationale behind this approach is that if the memberships are selected at random, this probability is expected to be $1/2^n$, no matter what values of c and m are involved. Unfortunately, this measure of validity also fails to be effective in practice. The reason for this is quite simple, in a data set in which there is no structure it is the data points which are randomly or uniformly distributed not the memberships. If one constructs a table of values of the proportion exponent for the data set in Fig. 1, the same kind of variation that occurs for the partition coefficient is present.

Although the proportion exponent failed to provide an effective validity functional, an appropriate modification of the procedure does produce, in theory at least, a functional which should not be sensitive to any of the parameters involved in the Fc-M algorithm. What is meant by a functional not being

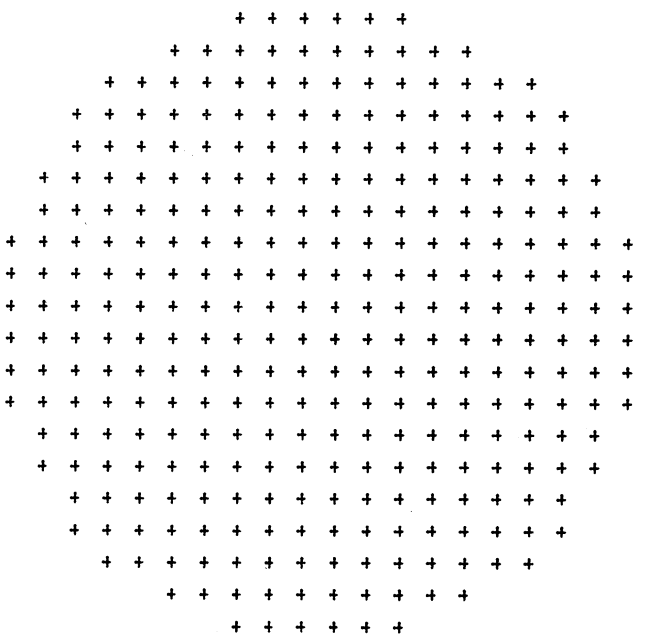


Fig. 1. Two-dimensional uniformly distributed data.

TABLE I
PARTITION COEFFICIENT FOR UNIFORM DATA

	Exponent Weight			
Clusters	1.5	2	3	4
2	.82	.68	.56	.53
3	.81	.61	.43	.37
4	.79	.56	.35	.29
5	.77	.53	.30	.23
6	.75	.49	.26	.20

sensitive to parameters is precisely the following. If one applies the Fc-M algorithm to a data set in which there are no clusters, then computes the value of the validity functional on the output, this value is the same no matter what values of c and m were used. If this functional is then used on any other data set, any deviation of its value away from the constant obtained above would of necessity be due to cluster structure in the data. Furthermore, the amount of deviation should indicate the quality of the clustering. Before attempting to construct a functional it is worthwhile to ask if there is a data set with which to test it. In other words, is there a data set in which there is no cluster structure. The identification of this data set is, in fact, the first step in the construction of the UDF.

A data set in which there is no cluster substructure is one which consists of exactly one cluster of uniformly distributed points. Furthermore, since the Fc-M algorithm tends to identify spherically shaped clusters the entire data set should be spherically shaped. The natural set to choose, then, would be $B^d = \{x \in R^d: |x| \leqslant 1\}$, the unit ball. Of course, this set is infinite and the Fc-M algorithm cannot be used with it. Rather than give up B^d, we construct an extension of the algorithm which can be used for infinite sets. Namely, by an *Fc-M clustering of* B^d, we mean vectors $v_1, \cdots, v_c \in B^d$ and

functions u_i: $B^d \to [0, 1]$ for $i = 1, \cdots, c$ so that $\Sigma_i u_i = 1$ which minimize

$$\phi(v_1, \cdots, v_c, u_1, \cdots, u_c) = \int_{B^d} \Sigma_i (u_i(x))^m \, |x - v_i|^2 \, dx.$$

All we have done is replace summation over the data set in the objective function by integration over the set. Furthermore, by differentiating ϕ with respect to the components of $v_1, \cdots, v_c$ and taking variations with respect to $u_1, \cdots, u_c$ subject to the constraint that $\Sigma u_i = 1$, we obtain the following necessary condition for a local minimum. For $i = 1, \cdots, c$ and $j = 1, \cdots, d$

$$v_{ij} = \int_{B^d} (u_i(x))^m \, x_j dx \Big/ \int_{B^d} (u_i(x))^m \, dx \tag{3}$$

and

$$u_i(x) = (1/|x - v_i|^2)^{1/(m-1)} / \Sigma_j (1/|x - v_j|^2)^{1/m-1}. \tag{4}$$

These equations provide, in theory at least, an iterative procedure for obtaining a clustering of B^d and this procedure is what we mean by the extended Fc-M algorithm.

If such a clustering can be obtained for any choice of d, c, and m, then it can be used to construct a validity function which is not sensitive to these parameters. This can be done as follows. First select a measure of quality, for example, any of the four mentioned in the previous section, except that it should be constructed so that it decreases with increase in the quality of classification. So, one should use, for example, $1 - \Sigma_i (u_i(x))^2$ rather than $\Sigma_i (u_i(x))^2$. Then define ρ: $B^d \to R$ by $\rho(x)$ = the measure of quality of the classification of the point x. Next for each r in the range of ρ define $S_r \subset B^d$ by $S_r = \{x \in B^d: \rho(x) \leqslant r\}$. The set S_r then consists of all points in B^d which have a better quality of classification than the number r. Finally, define Φ: $R \to [0, 1]$ by

$$\Phi(r) = \text{volume of } S_r / \text{volume of } B^d.$$

The function Φ provides a measure of the degree of classification of a point in that $\Phi(\rho(x))$ is the proportion of the data set, B^d which is better classified than x itself. This function also has the property that

$$\int_{B^d} \Phi(\rho(x)) \, dx \Big/ \text{volume of } B^d = \tfrac{1}{2}. \tag{5}$$

That is, the average value of $\Phi \cdot \rho$ over B^d is $\frac{1}{2}$ independent of the clustering obtained or the function ρ which is used. This fact can be seen easily as follows. Suppose we let X be a random vector with values in B^d which is uniformly distributed in B^d, then $R = \rho(X)$ is a random variable and $\Phi(r)$ is the probability that $R \leqslant r$. In other words, the function Φ we have defined is the cumulative distribution function of R. So the random variable $\Phi(R)$ is uniformly distributed on the interval $[0, 1]$ and has expected value $\frac{1}{2}$, but this expected value is also given by the expression on the left in (5).

We have, in fact, a family of functions Φ, one for each choice of ρ, c, m, and d. Using these functions we define a UDF as follows. If $X = \{x_1, \cdots, x_n\}$ is a set of d-dimensional vectors and U is a $c \times n$ matrix of memberships obtained by applying the Fc-M algorithm with parameters c and m, then

$$\text{UDF}(U) = \Sigma_k \Phi(\rho(x_k))/n$$

where ρ is a measure of classification and is the function obtained above for this ρ and the same values of c, m, and d. This is just the average over the data set of $\Phi \cdot \rho$ and, consequently, if the value of UDF is close to $\frac{1}{2}$ it can be inferred that the Fc-M has found no cluster structure in the data set. Furthermore, the UDF can be interpreted as the average of the probabilities that the data points would have been better clustered than they were had they been selected at random from the unit ball. Consequently, the closer the value of the UDF is to zero the better the clustering is.

As Dubes and Jains [8] have indicated an effective cluster validity analysis should be able to determine whether or not there is any structure in the data, that is, whether or not the data are random; and whether or not the clusters identified by the algorithm are "real," that is, whether they represent relationships among the data or artifacts of the mathematical procedure. The UDF certainly deals with the first problem in that it compares the clustering output for the data set with the clustering of a set with no structure. As to the second problem, the user has by choosing to use the Fc-M algorithm, determined mathematically what he means by a "real" cluster, in that the algorithm identifies compact, well-separated, spherically shaped clusters. The UDF cannot determine whether or not the Fc-M algorithm is the appropriate one to use, but does indicate how well the clusters meet the three criteria for being "real" as determined by the use of the algorithm. This is the case, because these three properties are all essentially functions of the relationships among distances of the data points to the cluster centers. These relationships are summarized in the membership functions which, in turn, are summarized in the UDF.

As we have previously stated this procedure, in theory at least, produces a validity functional which is not sensitive to parameters. The reason the phrase "in theory" must be used is that there are two difficult, if not impossible, steps in the construction of a working UDF. First, one must know all clusterings of the unit ball, namely, the solutions to (3) and (4). Next one must compute the volumes of the sets S_r. If the cluster centers are known then the membership functions are easy to obtain using (4). Also, although the computation of the volumes is greatly facilitated by the proper choice of ρ, the geometrical relationships among the centers are significant factors in being able to make these computations. So, everything hinges on being able to determine the cluster centers.

In the case $d = 2$, it is possible to guess at least the orientation of the centers. Because of the symmetry of the disk it is reasonable to expect that the centers would form the vertices a regular c-sided polygon centered at the origin. That this happens can be verified by clustering a data set such as the one in Fig. 1. In this case, all that remains is to determine the distance from a cluster center to the origin. Here again, approximation can be obtained empirically, but all that one finds is that this distance increases as c increases and decreases as m

increases. The exact relationship between these values is not known. For dimensions greater than 2, even the orientations of the centers is difficult to predict and empirical results are difficult to obtain. The data set in Fig. 1 contains 316 points; to approximate the three-dimensional ball with the same density requires 4224 points, for the four-dimensional ball, 49 648 points are required, so it rapidly becomes impractical to investigate empirically.

Until further information is obtained, the only way to construct a usable UDF is to assume the location of the cluster centers in a reasonable if not accurate way, then proceed with the remainder of the construction as described above. This is done as follows. For given $d \geq 2$ and $c \geq 2$ let for $k = 1, \cdots, c$, $v_k = D(\cos(2\pi k/c), \sin(2\pi k/c), 0, \cdots, 0)$ where

$$D = c \sin(\pi/c)\, \Gamma(d/2 + 1)/(2\pi^{1/2}\Gamma((d+1)/2 + 1).$$

These points are the centers of gravity of the c congruent sets

$$\Omega_k = \{(r \cos\Theta, r \sin\Theta, x_3, \cdots, x_d) B^d:\ 0 \leq r \leq 1 \text{ and } (2k-1)\pi/c \leq \Theta \leq (2k+1)\pi/c\}.$$

The case $d = 2$ is just a pie sliced into c equal pieces and for $d = 3$ these sets are analogous to the sections of an orange. The membership functions are then determined by (4) and all that remains is to choose ρ and describe the computation of the volumes of the sets S_r.

The function ρ is chosen to be $\rho(x) = \min_i u_i(x)/\max_i u_i(x)$.

First of all, the value $\rho(x) = |x - v_j|^{2/(m-1)}/|x - v_i|^{2/(m-1)}$ where the jth cluster center is the one closest to x and the ith center is the one furthest from x. Consequently, the closer x is to a cluster center the smaller the value of ρ is. Because of this and the symmetry in the location of the cluster centers, the volume of S_r is equal to c times the volume of the intersection of S_r with any one of the Ω_k's, in particular Ω_c. So, it suffices to consider the values of ρ on Ω_c and to compute the volume of $S_r \cap \Omega_c$. If c is even then for $x \in \Omega_c$ the cluster center closest to x is $v_c = (D, 0, \cdots, 0)$ and the one furthest from x is $v_* = (-D, 0, \cdots, 0)$. It follows from the nature of ρ that $S_r \cap \Omega_c$ is the intersection of Ω_c with a ball centered at $t(v_c - v_*) + v_c$ and radius $(t(t+1))^{1/2}|v_c - v_*|$ where $t = r^{m-1}/(1 - r^{m-1})$. If c is odd then for $x \in \Omega_c$ the closest center is, again, v_c, but the center furthest from x depends on which half of Ω_c x is in. That is, for $x = (r \cos\Theta, r \sin\Theta, x_3, \cdots, x_d)$, x is furthest from $v_{(c+1)/2}$ if $0 \leq \Theta \leq \pi/c$ and furthest from $v_{(c-1)/2}$ if $-\pi/c \leq \Theta \leq 0$. However, again because of symmetry, the volume of $S_r \cap \Omega_c$ is twice the volume of $S_r \cap \Omega_{c,+}$ where $\Omega_{c,+}$ is the set of points in Ω_c with $\Theta \geq 0$. Letting $v_* = v_{(c+1)/2}$, the set $S_r \cap \Omega_{c,+}$ is the intersection of $\Omega_{c,+}$ and a ball with the center and radius given above. So, the geometry of the sets S_r is such that their volume can be computed by integration. The necessary integration is tedious but possible, so the function Φ can be obtained and its computation implemented by a subroutine attached to the Fc-M algorithm program.

The uniform data functional can then be computed as described above:

$$\text{UDF} = \Sigma_k \Phi(\min_i u_{ik}/\max_i u_{ik})/n.$$

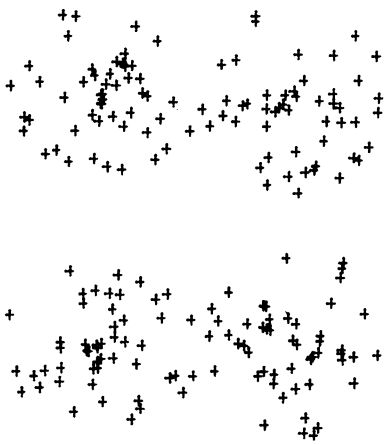


Fig. 2. Data set X_1.

However, it may be convenient and even necessary to transform the values of Φ before averaging. For example, if the values of Φ are spread over a wide range, roundoff error may produce inaccuracies in the average. One way to avoid this is to average the values of $-\log \Phi$ (using the natural logarithm). High values of the corresponding UDF indicate good clustering and one expects to see a value of one on an unstructured data set. It should be noted that if a data point coincides with a cluster center, this transformation is undefined since Φ has value zero.

So it is possible to construct a usable UDF based on the choice of centers given above. It is not the most desirable UDF, since it is not based on an actual fuzzy clustering of the standard data set, but it does provide a validity functional whose effectiveness is illustrated by the examples which follow.

V. Examples

Cluster validity functionals are often used to determine the number of clusters present in a data set. The Fc-M algorithm is applied for various values of c and the best clustering, as indicated by the value of the functional, serves to identify the number of clusters present. In order to illustrate the effectiveness of the UDF, we use it in this way to identify the number of clusters present in data sets where the answer is known.

The UDF was evaluated on clusterings of three data sets. Two of the data sets were artifically generated and the third is the famous Iris data of Anderson [9].

The first data set X_1 consists of the two-dimensional data pictured in Fig. 2. It was obtained by choosing 50 points at random in each of the disks of radius one centered at the points (2, 1), (2, -1), (-2, 1), and (-2, -1), respectively. As the figure indicates, it would be reasonable to expect that a clustering algorithm would identify the presence of four clusters. The second data set X_2 was obtained in exactly the same way, except that the radii of the disks were 1.5 rather than 1. As can be seen in Fig. 3, because of the overlap of the disks, this data set appears to have two clusters rather than four. Finally, the Iris data consists of 150 four-dimensional vectors. The components of a vector are the measurements of the petal length, petal width, sepal length, and sepal width of a particular Iris plant. There are 50 flowers in each of

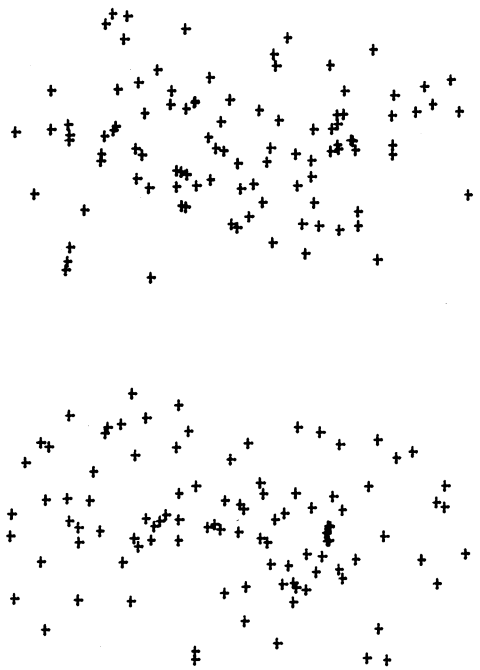


Fig. 3. Data set X_2.

TABLE II
UNIFORM DATA FUNCTIONAL AND PARTITION COEFFICIENT
(EXPONENT WEIGHT = 2)

Data set	X_1		X_2		Iris	
Clusters	PC	UDF	PC	UDF	PC	UDF
2	.87	2.4	.84	2.2	.89	7.4
3	.79	2.5	.74	2.0	.78	7.8
4	.80	2.8	.69	2.0	.70	7.2
5	.72	2.7	.63	2.0	.59	7.0
6	.66	2.6	.58	2.0	.44	6.8

three subspecies of Iris represented in the data, so it was assumed that an effective validity functional should indicate the presence of three clusters.

The results of the analysis are shown in Table II. The Fc-M was applied for the indicated values of c and m; and the UDF (using - log transformation) and the partition coefficient (PC) were evaluated for each of the outputs. Both of these functionals should indicate the best clustering by their highest value. As can be seen the UDF correctly identified the number of clusters in each case.

A closer examination of the values of the UDF for data set X_1 indicates results which appear to be counterintuitive. The UDF assumes its lowest value for two clusters, but the human observer may consider two clusters as a natural preference over three or perhaps even four clusters. In order to resolve this dilemma one must recall that the Fc-M algorithm tends to identify compact, well-separated, spherically shaped clusters, and the UDF measures how well this has been done. Although the two clusters are well-separated they are not compact, in that there are large concentrations of data away from the cluster center, and are not spherically shaped. So, the UDF has properly evaluated this clustering relative to all three criteria, but the preference of the human observer is influenced more by separation than by the density and shape of the individual clusters. So the dilemma has been produced by the choice of algorithms rather than the results obtained.

VI. SUMMARY

The uniform data functional is based on comparing the fuzzy clustering of a data set with an analogous clustering of a data set with no cluster substructure, in a way that is not sensitive to the parameters of the Fc-M algorithm. Consequently, variations in the value of the UDF are due to variations in the quality of the fuzzy clustering. Used in conjunction with Fc-M, it promises to make this fuzzy clustering algorithm an even more powerful tool for the analysis of data.

REFERENCES

[1] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters," *J. Cybern.*, vol. 3, pp. 32-57, 1974.

[2] J. C. Bezdek, "Fuzzy mathematics in pattern classification," Ph.D. dissertation, Appl. Math., Cornell Univ., Ithaca, NY, 1973.

[3] L. A. Zadeh, "Fuzzy sets," *Inform. Contr.*, vol. 8, pp. 338-353, 1965.

[4] J. B. Bezdek, "A convergence theorem for the fuzzy ISODATA clustering algorithms," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-2, pp. 1-8, 1980.

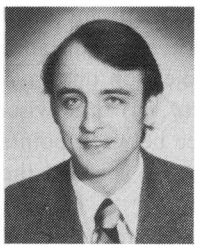
[5] —, "Cluster validity with fuzzy sets," *J. Cybern.*, vol. 3, pp. 58-73, 1973.

[6] —, "Mathematical models for systematics and taxonomy," in *Proc. 8th Int. Conf. on Numerical Taxonomy*, G. Estabrook, Ed. San Francisco, CA: Freeman, 1975, pp. 143-166.

[7] M. P. Windham, "Cluster validity for fuzzy clustering algorithms," *Fuzzy Sets Syst.*, vol. 5, pp. 177-185, 1981.

[8] R. Dubes and A. Jain, "Validity studies in clustering methodologies," *Pattern Recognition*, vol. 11, pp. 235-253, 1979.

[9] E. Anderson, "The iris of the Gaspe peninsula," *Bull. Amer. Iris Soc.*, vol. 59, pp. 2-5, 1935.



Michael P. Windham was born in Houston, TX, on September 23, 1944. He received the M.A. and Ph.D. degrees in mathematics from Rice University, Houston, TX, in 1970.

He was Instructor in Mathematics at University of Miami in 1970. Since 1971 he has been with the Department of Mathematics, Utah State University, Logan, where he is an Associate Professor. He is engaged in teaching and research in stochastic differential equations and cluster analysis.

Medial Axis Transformation of a Planar Shape

D. T. LEE, MEMBER, IEEE

Abstract—**The medial axis transformation is a means first proposed by Blum to describe a shape. In this paper we present a $O(n \log n)$ algorithm for computing the medial axis of a planar shape represented by an n-edge simple polygon. The algorithm is an improvement over most previously known results in terms of both efficiency and exactness and has been implemented in Fortran. Some computer-plotted output of the program are also shown in the paper.**

Index Terms—**Analysis of algorithm, computational complexity, continuous skeleton, divide-and-conquer, medical axis transformation, simple polygon, Voronoi diagram.**

I. INTRODUCTION

THE MEDIAL axis transformation is a technique first proposed by Blum [2] as a means to describe a figure. It is formally defined as follows: given an object represented, say by a simple polygon G, the medial axis $M(G)$ is the set of points $\{q\}$ internal to G such that there are at least two points on the object's boundary that are equidistant from $\{q\}$ and are closest to $\{q\}$. Because of its shape, the medial axis of a figure is also called the skeleton or the symmetric axis of the figure. Associated with the medial axis is a radius function R, which defines for each point on the axis its distance to the boundary of the object. With the axis and the radius function one can reconstruct the figure by taking the union of all circles centered on the points comprising the axis, each with a radius given by the radius function.

Since the introduction of the notion of medial axis, there has been a great deal of work involving the computation of the medial axis reported [1], [3], [4]-[8], [10]-[13]. Most of the previously known results take time proportional to n^2 where n is the number of boundary edges of the figure. Recently, Lee and Drysdale [8] and Kirkpatrick [7] have presented a general algorithm for finding continuous skeletons of a set of disjoint objects. Lee-Drysdale's algorithm runs in $O(n \log^2 n)$ time whereas Kirkpatrick's runs in $O(n \log n)$ time. The $O(n \log n)$ time algorithm by Kirkpatrick [7] is asymptotically optimal but is very tedious to implement. In this paper we shall give an algorithm which is simpler to implement and computes the medial axis of a simple polygon in $O(n \log n)$ time. The output of the algorithm would be precisely the medial axis of the polygon if the computer had arithmetic with infinite precision. The computer-plotted diagrams shown in the paper are exact to within the precision of the computer used. The medial axis of a simple polygon is a tree-like planar graph composed of straight-line segments and portions of parabolic curves. Before we give the description of the algorithm in the next section we shall introduce a few definitions and present some preliminary results.

Definition 1: A *closed line segment* $\overline{a, b}$ consists of two endpoints a and b, and a straight-line portion which is denoted by

Manuscript received December 12, 1980; revised January 20, 1982. This work was supported in part by the National Science Foundation under Grant MCS-7916847.

The author is with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60201.

0162-8828/82/0700-0363$00.75 © 1982 IEEE