File Naming

Each file is assigned a unique name that will allow for individual researchers to compare performance on specific images.

Storage Requirements

The number of megabytes of storage for each portion of the database are now listed.

<table>
<thead>
<tr>
<th>Database Component</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grayscale cities</td>
<td>152</td>
<td>16</td>
</tr>
<tr>
<td>Grayscale states</td>
<td>83</td>
<td>9</td>
</tr>
<tr>
<td>Grayscale ZIP Codes</td>
<td>193</td>
<td>9</td>
</tr>
<tr>
<td>Mixed bi-tonal alphabets and numerics</td>
<td>52</td>
<td>6</td>
</tr>
<tr>
<td>Bi-tonal numerics only</td>
<td>38</td>
<td>10</td>
</tr>
</tbody>
</table>

Overall, approximately 600 Mbytes of the CDROM are used. This includes the storage needed for formatting information.

Availability

The database described in this correspondence is available from the Center of Excellence for Document Analysis and Recognition (CEDAR) at the State University of New York at Buffalo.

ACKNOWLEDGMENT

Prof. S. N. Srihari, Director of CEDAR, was the Principal Investigator for the handwritten ZIP Code recognition project. Dr. J. Tan of Arthur D. Little, Inc., helped shape the composition of the database. Dr. E. Cohen assisted in the supervision of the data capture process. J. Giattino directed the formatting of the CDROM.

REFERENCES


Feature-Preserving Clustering of 2-D Data for Two-Class Problems Using Analytical Formulas: An Automatic and Fast Approach

Ja-Chen Lin and Wen-Hsiang Tsai

Abstract—We propose in this correspondence a new method to perform two-class clustering of 2-D data in a quick and automatic way by preserving certain features of the input data. The method is analytical, deterministic, unsupervised, automatic, and noniterative. The computation time is of order \( n \) if the data size is \( n \), and hence much faster than any other method which requires the computation of an \( n \)-by-\( n \) dissimilarity matrix. Furthermore, the proposed method does not have the trouble of guessing initial values. This new approach is thus more suitable for fast automatic hierarchical clustering or any other fields requiring fast automatic two-class clustering of 2-D data. The method can be extended to cluster data in higher dimensional space. A 3-D example is included.

Index Terms—Two-class clustering, cluster representatives, feature-preserving, analytical formulas, decision boundary, automatic fast clustering, \( k \)-means, hierarchical methods.

I. INTRODUCTION

Two-class clustering problems are frequently encountered in real applications. For example, block truncation coding for image compression [1], divisive clustering for hierarchical clustering [2], binary decision tree construction, etc. It is therefore desired to develop a fast automatic method that can be employed to partition an input set \( H \) of \( n \) patterns into two classes. Unfortunately, most of the clustering tools developed so far, such as the \( k \)-means method [3], the divisive method using a dissimilarity matrix [4], etc., are iterative and thus unsuitable for performing fast automatic two-class clustering.

It is desirable to avoid iterative computation by using mathematical formulas to express the decision boundary, which separates the two classes, in terms of the input patterns directly. One way of achieving this goal based on the moment-preserving principle is explained below. When the \( n \) input patterns are one-dimensional, say, forming a set \( H = \{ x_1, \ldots, x_n \} \), the partition of \( H \) into two disjoint clusters \( H_A \) and \( H_B \) is an easy job. We may assume that every pattern in cluster \( H_A \) resembles (in some sense) a single point \( x_A \), and similarly, every pattern in cluster \( H_B \) resembles another single point \( x_B \). The two points \( x_A \) and \( x_B \) are called cluster representatives. Assume further that the fractions of the numbers of patterns in \( H_A \) and \( H_B \) are \( p_A \) and \( p_B \), respectively. It is clear that

\[
p_A + p_B = 1. \quad (1)
\]

By preserving the first three moments, i.e., by requiring that

\[
p_A x_A^k + p_B x_B^k = \bar{x}^k = \frac{1}{n} \sum_{i=1}^{n} x_i^k \quad \text{for } k = 1, 2, \text{ and } 3, \quad (2)
\]

and by the natural requirement (1), we can solve Eqs. (1) and (2) to obtain the four unknowns \( \{ x_A, x_B, p_A, p_B \} \). The solution can be obtained.

Manuscript received March 9, 1992; revised January 5, 1993. This work was supported by the National Science Council, Republic of China, under Contract NSC 81-0408-E-009-589. Recommended for acceptance by Associate Editor R. P. W. Duin.

The authors are with the Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 30050, Republic of China.

IEEE Log Number 9214426.
found in [5] or [6]. Having obtained \( x_A, x_B, y_A \) and \( y_B \), one may define, as Tsai did in [6], the decision boundary \( z \) to be the \( p_A \)-tile.

When the input patterns are two-dimensional data, we found that moment-preserving is no more so easy to apply to partition the given data into two clusters \( H_A \) and \( H_B \). To see this, let \( H = \{(x_i, y_i)\}_{i=1}^n \) be the given data to be partitioned, and let the fractions of the numbers of patterns in \( H_A \) and \( H_B \) again be \( p_A \) and \( p_B \), respectively. Assume also that every pattern in cluster \( H_A \) resembles a cluster representative \((x_A, y_A)\), and every pattern in cluster \( H_B \) resembles another cluster representative \((x_B, y_B)\). The general rule to construct a feature-preserving method are both possible to extend the method to higher dimensional data, and to study the feasibility of the solution there is a very good solution. The definition of \( \bar{r}, \bar{\theta}, \phi \) will be given in the next section.

The remainder of this paper is organized as follows. In Section II the formulas to compute \( \{x_A, y_A, x_B, y_B, p_A, p_B\} \) by preserving the set of features (5) are introduced, and to construct the decision boundary to separate the two classes is also discussed. In Section III, we give some experimental results. In Section IV, the method is compared with the k-means and hierarchical methods. Then we discuss in Section V the safe way to apply our method. The possibility to extend the method to higher dimensional data, and the general rule to construct a feature-preserving method are both discussed in Section VI. Finally, concluding remarks are given in Section VII.

II. ANALYTICAL FORMULAS FOR TWO-CLASS CLUSTERING OF 2-D DATA

Without the loss of generality, we assume that \((x, y)\), which is the centroid of the given \( n \)-point system \( H = \{(x_i, y_i)\}_{i=1}^n \), to be the origin \((0, 0)\). If this is not the case, a translation by the amount of \( \Delta x = \bar{x} \) and \( \Delta y = \bar{y} \) should first be done, and then, after \( \{x_A, y_A, x_B, y_B\} \) are generated, these four numbers should be transformed back to the old coordinates by an inverse translation with \( \Delta x = -\bar{x} \) and \( \Delta y = -\bar{y} \). Similarly, we also assume that a preprocessing step of rotation has been performed so that the [1] of \( H \) coincides with the \( y \)-axis. Note that the principal axis is a line going through the centroid \((0, 0)\) of \( H \) with directional angle \( \phi \) characterized by the following two equations [7]:

\[
\tan 2\phi = 2xy/(x^2 - y^2),
\]

\[
(x^2 - y^2)
\cos 2\phi - 2xy \sin 2\phi > 0.
\]

Let \( \{(r_A, \theta_A)\}_{i=1}^n \) be the polar coordinates of the (standardized) points \( \{(x_i, y_i)\}_{i=1}^n \). Similarly, let \( \{(r_B, \theta_B)\} \) be the polar coordinates of the cluster representative \((x_A, y_A)\) and \((x_B, y_B)\), respectively. The preserving of \((x, y, \bar{r}, \bar{\theta}, \phi)\) with \( \bar{r} \) and \( \bar{\theta} \) defined as

\[
\bar{r} = \frac{1}{n} \sum_{i=1}^n r_i \quad \text{and} \quad \bar{\theta} = \frac{1}{n} \sum_{i=1}^n \theta_i,
\]

means that

\[
p_A x_A + p_B x_B = \bar{r} = 0 \quad (6)
\]

\[
p_A y_A + p_B y_B = \bar{\theta} = 0 \quad (7)
\]

\[
p_A x_A + p_B x_B = \bar{r} \quad (8)
\]

\[
p_A \theta_A + p_B \theta_B = \bar{\theta} \quad (9)
\]

Equations (6) and (7) imply that

\[
x_A = \frac{p_B}{p_A} x_B \quad (10)
\]

\[
y_A = \frac{p_B}{p_A} y_B \quad (11)
\]

As a result,

\[
\tau_A = \sqrt{x_A^2 + y_A^2} = \frac{p_B}{p_A} \sqrt{x_B^2 + y_B^2} = \frac{p_B}{p_A} \tau_B
\]

which in turn means that

\[
\bar{r} = \frac{\bar{\theta}}{2p_B}
\]

because

\[
\bar{r} = p_A r_A + p_B r_B = p_A \left( \frac{p_B}{p_A} \tau_B \right) + p_B r_B = 2p_B r_B \quad (14)
\]

can be derived from (8).

On the other hand, the straight line connecting the system centroid \((x, y) = (0, 0)\) and \((x_A, y_A)\) is identical to the straight line connecting \((x, y)\) and \((x_B, y_B)\), for both lines go through \((0, 0)\) and have the same slope

\[
s = \frac{y_A - y}{x_A - x} = \frac{y_B - y}{x_B - x}
\]

by (10) and (11).

Since the three points \((x_A, y_A), (0, 0), (x_B, y_B)\) are on the same straight line, and \((0, 0)\) is between \((x_A, y_A)\) and \((x_B, y_B)\), by (10) and (11), we see that the polar angles \( \theta_A \) and \( \theta_B \) of the two points \((x_A, y_A)\) and \((x_B, y_B)\) must differ from each other by \( \pi \). Without the loss of generality, let \( \theta_B = \theta_A + \pi \). Equation (9) thus becomes

\[
\bar{r} = p_A \theta_A + p_B (\theta_A + \pi) = (p_A + p_B) \theta_A + p_B \pi = \theta_A + p_B \pi.
\]

We then have

\[
p_B = (\bar{\theta} - \theta_A) / \pi. \quad (15)
\]

If we can obtain the value of \( \theta_A \), then the values of all six unknowns \( \{x_A, y_A, x_B, y_B, p_A, p_B\} \) can be computed easily because we can first compute \( p_B \) and \( p_A \) by (15) and

\[
p_A = 1 - p_B. \quad (16)
\]

respectively. Then we can evaluate \( \tau_B \) and \( \tau_A \) by the formulas

\[
\tau_B = \bar{r}/(2p_B) \quad \text{and} \quad \tau_A = \tau_B p_A/p_B, \quad \text{as stated in (13) and (12)}.
\]

Finally, \( x_A = r_A \cos \theta_A, y_A = r_A \sin \theta_A, x_B = r_B \cos \theta_B, y_B = r_B \cos (\theta_A + \pi) = -r_B \cos \theta_A, y_B = r_B \sin \theta_B = r_B \sin (\theta_A + \pi) = -r_B \sin \theta_A \). Therefore, all we have to do now is to assign a suitable value to \( \theta_A \) which is defined earlier to be the polar angle of \((x_A, y_A)\), or equivalently, the directional angle of the straight line going through the three points \((x_A, y_A), (0, 0), \text{and } (x_B, y_B)\).
just estimated values, instead of the exact values of the population
distribution. Containing (ZA, YA) has been defined to be the straight line I which is perpendicular to the line
I. The proposed approach is finally applied to H. The computed cluster representatives are marked by two crosses, and the computed decision boundaries I and I' are shown as a dotted straight line and a solid straight line, respectively, in each figure. As stated at the end of the last section, both I and I' are perpendicular to the line segment connecting the two generated crosses. The only difference is that I' bisects the line segment connecting the two generated crosses while I guarantees that the fractions of the populations on the two sides of I match the values PA and PB computed by (16) and (15). In both Figs. 1 and 2, H contains 2100 points. In Fig. 1, each H is designed to be the mergence of two disjoint subsets, and the number of points in each subset is approximately 1050 points. In Fig. 2, however, one of the subsets contains approximately 1400 points and the other subset contains about 700 points. It is observed that the values of PA and PB computed by formulas (16) and (15) do get close to the expected ratio of 50% vs. 50% in Fig. 1, and 67% vs. 33% in Fig. 2. It is also observed that the computation time taken to generate \{ZA, YA, ZB, YB, PA, PB\} is only about two seconds for a 2100-point set H using an IBM PC with an 80386 processor.

IV. COMPARISON WITH K-MEANS AND HIERARCHICAL METHODS

In this section, we compare our method with some clustering methods available in the IMSL [9] package, namely, the hierarchical agglomerative methods [10] and the k-means method. The k-means method used in IMSL is based on an algorithm written by Hartigan and Wong [11], and this program tries to reduce the total sum E of the within-cluster squares. When k = 2, i.e., when it is applied to the 2-class problem discussed here, E is expressed as

\[
E = E(H_A, H_B) = \sum_{a \in H_A} d(a, \hat{a})^2 + \sum_{b \in H_B} d(b, \hat{b})^2
\]

for any two-class partition \{H_A, H_B\} of the given input set H. As usual, \(\hat{a}\) and \(\hat{b}\) denote the centroids of clusters \(H_A\) and \(H_B\), respectively. In Fig. 3, we compare the clustering results of ours with those of the k-means program provided by IMSL. It can be seen that our method yields clustering results similar to those of the k-means method. The two methods give identical results for (a), (c)-(f). Only two points in (b) and one point in (g) are clustered differently. The data shown in Fig. 3 were those proposed by Nagy [12]. We use these data sets because they illustrate cluster distributions of typical clustering problems such as "neck," unequal cluster populations, etc.

Similar typical clustering problems were also pointed out by Zahn [13]. The hierarchical agglomerative methods available in the IMSL package were also tested. It was observed that none of the proposed, the k-means, and the hierarchical agglomerative methods can cluster well the nonlinear data set (g) and the nonspherical data set (d) sketched in Fig. 3. When there are necks in the data, like in (b) and (c), our method cut necks, but not exactly (i.e., a little portion of the patterns is misclustered). Similar troubles also exist for the k-means method (see (c) of Fig. 3), the complete linkage method, and Ward's method. As for the single linkage method, it is terribly inapplicable to data sets with "necks" because of the chaining effect.

The computation time used is also an important index to evaluate distinct methods. In general, the time for the k-means method is about two times longer than ours, but the time needed for either of

III. ILLUSTRATIVE EXAMPLES

In this section, we show the results of using our formulas to compute \(\{x_A, y_A, x_B, y_B, p_A, p_B\}\) and construct both kinds of the decision boundaries I and I' for some randomly generated data set \(H = \{(x_i, y_i)\}_{i=1}^N\). The results are shown in Figs. 1 and 2. We first use an algorithm for generating random numbers to create a 2-D set \(S_A\), and use the same algorithm to create another 2-D set \(S_B\). These two sets are then merged together to form H. The proposed method used in IMSL is based on an algorithm written by Hartigan and Wong [11], and this program tries to reduce the total sum E of the within-cluster squares. When k = 2, i.e., when it is applied to the 2-class problem discussed here, E is expressed as

\[
E = E(H_A, H_B) = \sum_{a \in H_A} d(a, \hat{a})^2 + \sum_{b \in H_B} d(b, \hat{b})^2
\]

for any two-class partition \{H_A, H_B\} of the given input set H. As usual, \(\hat{a}\) and \(\hat{b}\) denote the centroids of clusters \(H_A\) and \(H_B\), respectively. In Fig. 3, we compare the clustering results of ours with those of the k-means program provided by IMSL. It can be seen that our method yields clustering results similar to those of the k-means method. The two methods give identical results for (a), (c)-(f). Only two points in (b) and one point in (g) are clustered differently. The data shown in Fig. 3 were those proposed by Nagy [12]. We use these data sets because they illustrate cluster distributions of typical clustering problems such as "neck," unequal cluster populations, etc. Similar typical clustering problems were also pointed out by Zahn [13].

The hierarchical agglomerative methods available in the IMSL package were also tested. It was observed that none of the proposed, the k-means, and the hierarchical agglomerative methods can cluster well the nonlinear data set (g) and the nonspherical data set (d) sketched in Fig. 3. When there are necks in the data, like in (b) and (c), our method cut necks, but not exactly (i.e., a little portion of the patterns is misclustered). Similar troubles also exist for the k-means method (see (c) of Fig. 3), the complete linkage method, and Ward's method. As for the single linkage method, it is terribly inapplicable to data sets with "necks" because of the chaining effect.

The computation time used is also an important index to evaluate distinct methods. In general, the time for the k-means method is about two times longer than ours, but the time needed for either of
the two hierarchical agglomerative methods becomes several hundred times longer than ours if the data size is 1000. We have also tried a hierarchical divisive method given in [4]. The clustering results are similar to ours (when applied to Nagy’s data sets), but the computation time also becomes several hundred times longer than ours when \( n = 1000 \) although it is faster than the hierarchical agglomerative methods. A reason to explain this fact is that our method has the work load of order \( n \) because all we need is to compute the average values \( \{\bar{x}, \bar{y}, \bar{\bar{x}}, \bar{\bar{y}}, \bar{\bar{\bar{x}}}, \bar{\bar{\bar{y}}}\} \) of \( n \) patterns, where \( \{x^1, y^1, \ldots, x^n, y^n\} \) are used to obtain the principal axis. On the other hand, it is also easy to see that each iteration of the \( k \)-means method has the computation load of order \( n \). However, each of the hierarchical methods, no matter agglomerative or divisive, has the work load of at least \( n^2 \) because of the construction of an \( n \)-by-\( n \) dissimilarity matrix [4]. Also notice that the computer storage problem for this \( n \)-by-\( n \) matrix. This makes the hierarchical methods unsuitable for personal computers when the sample size is \( n = 1000 \).

As for the \( k \)-means method, although the computation time and storage are of no trouble, outliers far away from the rest of the data will cause unexpected clustering results when any of them is taken as one of the initial guess points. The clustering result might be that one point forms a cluster and the other \( n - 1 \) points form the other. On the other hand, outliers only affect our method a little because the average functions \( \bar{x}, \bar{y}, \bar{\bar{x}}, \bar{\bar{y}}, \bar{\bar{\bar{x}}}, \bar{\bar{\bar{y}}} \), etc., will “smooth” their impact.

Therefore, our method is quick and storage-saving, and has no worry about the choice of the initial guess, the number of iterations needed, the problem of being convergent or not, etc. As long as the quality of the clustering result is not too far away from those obtained by other methods, our method deserves a try, especially for data of large sizes. Therefore, we try to find in the next section the limitation of our method and discuss the situations in which it is safe to use our method.

V. APPLYING PROPOSED METHOD IN A SAFE WAY

Since our method uses a straight line \( l' \) to split data into two clusters, we concentrate our discussion on linearly separable problems only. When the two clusters are circular-like (hollow or not), our method works well even if the two clusters touch each other (see (a)-(c) of Fig. 4 for illustrations). However, when there are well-elongated shapes in the data, our method may fail if the two clusters are too close to each other. For example, the data in (a) and (m) of Fig. 4 can be clustered well by our method (even though the two clusters touch each other), but the data in (d), (f), (h), (and maybe (j)) are not suitable to our method. Note that in (f) and (j) our cluster representatives are close to the means of the given clusters, and in (d) and (h) although our cluster representatives are far away from the means of the given (left and right) clusters, splitting the data into upper and lower halves, as shown in (d) and (h), is not worse than splitting the data into left and right halves, as in the direction shown in (e), in the sense that they give a smaller \( E \) defined in (17).

In short, all the clustering results shown in Fig. 4 have small \( E \). Moreover, the \( k \)-means method also have trouble in handling (d), (f), (h), and (j) if the policy of minimizing \( E \) is used. In fact, with the goal of minimizing \( E \), the clustering results of the \( k \)-means method are very similar to ours for all the data sets shown in Fig. 4 with the only exception (d), of which the decision boundary generated by the \( k \)-means method is neither vertical nor horizontal, but slanted.

When the two given clusters in each of (d), (f), (h), and (j) of Fig. 4 are far away enough from each other, as sketched in (e), (g), (i) and (k), respectively, our method yields “visually” good results again. Our experience is that, if well-elongated clusters are among the data to be clustered, it is usually safe to apply our method if the two expected clusters \( A \) and \( B \) satisfy \( \min \{d(a, b) | a \in A, b \in B\} \geq \frac{1}{2} \max \{\text{Diam}(A), \text{Diam}(B)\} \).

Here, the diameter \( \text{Diam}(C) \) of a point set \( C \) is defined as \( \text{Diam}(C) = \max \{d(c, c') | c, c' \in C\} \). Note that the inequality should be interpreted as a sufficient condition instead of a necessary condition. For example, both (l) and (m) of Fig. 4 can be partitioned well by our method although the sets are close to each other in each case.

At the end of this section, we give in Fig. 5 some examples of partitioning input data into two classes when the data are in fact formed of more than two clusters. In some cases, our method might improperly cut one of the clusters into two halves (see (b) and (d)). Since improper cutting is possible (a trouble also occurs to the \( k \)-
Fig. 4. Applying our procedure to some linear-separable data sets of which the two clusters are hollow circular-like vs. circular-like, circular-like vs. well-elongated, well-elongated vs. well-elongated, etc. As usual, the two crosses are the two detected cluster representatives, and the solid line is the detected decision boundary \( Z' \). Note that (d), (f), (h), and (i) are examples where applying our method is not suitable although these partitions have small total sums \( E \) of within-cluster squares. Also note that the crosses in (i) and (j) have been very close to the means of the expected clusters.

With the \( k \)-means (which minimizes \( E \)) and our methods both applied to the data shown in Figs. 3-5, we observed that these two methods have similar clustering results and requires approximately the same amounts of computation time. We therefore classify our method as an automatic fast clustering method with performance similar to that of the \( k \)-means method which minimizes the \( E \). Note that the \( k \)-means method has the trouble of choosing an initial guess, however.

VI. HIGHER DIMENSIONAL PROBLEMS AND FEATURES TO BE PRESERVED

In this section, we discuss the possibility of extending our method to higher dimensional problems, and the general principle to choose the features to be preserved.

Without the loss of generality, we discuss the 3-D case only. However, the method discussed below can be generalized to any higher dimension. Assume that a given set \( H = \{(x_i, y_i, z_i)\}_{i=1}^{n} \) is to be split into two clusters \( H_A \) and \( H_B \) with cluster representatives being \((x_A, y_A, z_A)\) and \((x_B, y_B, z_B)\), respectively. The goal is again to obtain some formulas to compute easily. Notice that there are two more unknowns \( \{z_A, z_B\} \) other than the six unknowns \( \{x_A, y_A, x_B, y_B, p_A, p_B\} \) appearing in (3) for the 2-D case. In general, if the dimensionality increases from \( d \) to \( d + 1 \),
The clustering results are perfect, while in (c) there is only one misjudgment. Considered to see what happens when the two clusters are away, the 3-D radii. Note that the principal axis of a set in d-dim space to solve the eight unknowns in (18) is the equation \( p_A + p_B = 1 \) and the equations obtained by preserving \( \{x, y, z, r, \theta, \varphi, p_A, \varphi_A\} \) with "P.A." denoting "principal axis" of \( H \) and \( r \) the average of the 3-D radii. Note that the principal axis of a set in d-dim space is the eigenvector corresponding to the largest eigenvalue of the d-by-d correlation matrix. After certain derivations, the solution is \( p_H = (\theta - \theta_A)/\pi; p_A = 1 - p_B; r_A = \ell(2p_A); \theta_A = \theta_{P.A.}; \varphi_A = \varphi_{P.A.}; x_A = (r_A \sin \varphi_A) \cos \theta_{P.A.}; y_A = (r_A \sin \varphi_A) \sin \theta_{P.A.}; z_A = r_A \cos \varphi_A; z_H = -(p_A/p_B)x_A; y_H = -(p_A/p_B)y_A; \) and \( z_H = -(p_A/p_B)z_A \). The decision boundary is then a plane bisecting and perpendicular to the line segment connecting the two 3-D cluster representatives.

Fig. 6 gives an illustrative example showing the clustering results of the 3-D version of our method, in which three subcases are considered to see what happens when the two clusters are away, near, or mixed. Notice that in the first two subcases (a) and (b) the clustering results are perfect, while in (c) there is only one misjudgment.

We now give a short discussion about the general principle to choose the variables to be preserved. In general, when it is dimensional, there are \( 2d + 2 \) unknowns \( \{p_A, p_B; x_A, x_B; y_A, y_B; z_A, z_B, \ldots\} \). Besides \( p_A + p_B = 1 \) and the preserving of the centroid \( \{x, y, z, \ldots\} \), we still need \( d + 1 \) features to be preserved. It seems that a lot of features might be used, such as the squared moments \( \{x^2, y^2, z^2, \ldots\} \), the correlated moments \( \{xy, xz, yz, \ldots\} \), the absolute moments \( \{|x|, |y|, \ldots\} \), the angles \( \{\theta, \phi, \ldots\} \), the lengths \( \{|x^2 + y^2, \sqrt{x^2 + y^2}, \sqrt{x^2 + y^2 + z^2}, \ldots\} \), the special-property axes such as the principal axis or any other axis that can be used to define shape orientations, and so on. The question is that certain combinations might form contradictive equation sets. To see whether a set of equations is contradictive or not, one can utilize well-known inequalities such as Schwartz's inequality, Holder's inequality, Minkowski's inequality, etc. When one has proved that the equation set has a solution and derived the formula to generate that solution, experiments using distinct types of data to test the clustering performance of this formula are important. Many combinations have been observed in this study to have poor clustering results. In general, maximizing the information contained in the d+1 selected quantities will usually be a way leading to success, if they do not form a contradictive equation set.

VII. CONCLUDING REMARKS

In this correspondence, we have proposed a new fast method to perform two-class clustering for 2-D data. The method is analytical, automatic, deterministic, unsupervised and noniterative. We have derived some simple analytical formulas to compute the two cluster representatives and the decision boundary by preserving the centroid, principal axis orientation, average polar radius and average polar angle of the input data set. The clustering result is satisfactory, and for an input set of several thousand points, the clustering procedure to generate the two desired cluster representatives and the fractions of the numbers of patterns in the two classes takes only a few seconds using a microcomputer. The computation speed is several hundred times faster than many hierarchical methods like the single linkage, complete linkage, Ward’s methods, and so on when the number of patterns is about 1000. The clustering result is, roughly speaking, not worse than those of the hierarchical methods. Unlike hierarchical
clustering methods or any graph-theoretical method using nearest neighbors, our method does not compute pairwise distances between input patterns; therefore, a lot of time is saved. But, since we do not use "local information" such as nearest neighbors, the global information that we use, such as \(\{\bar{e}, \bar{y}, \bar{r}, \bar{t}, \bar{d}\}\), can only give us a "rough" partition; it should be of no surprise if our method misassigns one or two points while some hierarchical methods do not. In a word, a clustering method using global information is fast, but the clustering result is usually not the best (see, for example, part (b) or (e) of Fig. 3). We have also compared the proposed method with the k-means method. Although the k-means method which minimizes the total sum of the within-clustersquares has similar clustering results and computation time as ours, the former has the trouble of choosing a safe initial guess (more specifically, it has the problem of avoiding taking an outlier as the initial guess of a cluster representative). Because of the weakness of the other methods mentioned above, the proposed method becomes very attractive for fast automatic hierarchical clustering or any other fields requiring fast automatic two-class clustering.

**ACKNOWLEDGMENT**

The authors wish to thank the referees for their valuable comments which lead to significant improvement on the writing of the correspondence. The authors also thank the editors, especially, Dr. A. K. Jain and Dr. R. P. W. Duin, for prompt processing of the correspondence.

**REFERENCES**