New Clustering & Boundary Detection Algorithm Based on Adaptive Influence Function

Gleb V. Nosovskiy, Dongquan Liu, Olga Sourina, *Member, IEEE Computer Society*

New Clustering & Boundary Detection Algorithm Based on Adaptive Influence Function

Abstract

Clustering became a classical problem in databases, data warehouses, pattern recognition, artificial intelligence, and computer graphics. Applications in large spatial databases, point-based graphics etc., give rise to new requirements for the clustering algorithms: automatic discovering of arbitrary shaped and/or non-homogeneous clusters, discovering of clusters located in low-dimensional hyperspace, detecting cluster boundaries. On that account, a new clustering and boundary detecting algorithm, ADACLUS, is proposed. It is based on the specially constructed adaptive influence function, and therefore, discovers clusters of arbitrary shapes and diverse densities, adequately captures clusters boundaries, and it is robust to noise. Normally ADACLUS performs clustering purely automatically without any preliminary parameter settings. But it also gives the user an optional possibility to set three parameters with clear meaning in order to adjust clustering for special applications. The algorithm was tested on various $2D$ data sets, and it exhibited its effectiveness in discovering clusters of complex shapes and diverse densities. Linear complexity of the ADACLUS gives it an advantage over some well-known algorithms.

Index Terms

Clustering Algorithms, Data Mining, Density-based Clustering

I. Introduction

Clustering is a classical problem in databases, data warehouses, pattern recognition, and artificial intelligence. In last 10 years, clustering algorithms were significantly improved, and many new features were implemented in them. Recently, due to development of point-based graphics in computer geometry, a new application area for clustering algorithms has emerged [1]–[5]. It brought new requirements to clustering procedures. To fit the geometrical applications, clustering algorithms should be able to determine clusters of arbitrary geometrical shape, clusters of non-uniform density, and clusters belonging to a low-dimensional hyperspace. The algorithms should be robust even in the case when significant amount of noise is present. In some applications it

is necessary to cluster according to the local conditions only. For geometrical applications and classification problems it is often important not only to determine clusters (as subsets of data points) but also to draw an accurate boundary between close neighboring clusters. For real-time applications, it is also very important that clustering algorithms work sufficiently fast, preferably with linear complexity with respect to the number of data points. Therefore, there is a strong demand for development of new efficient and robust clustering algorithms.

Existing clustering approaches proposed in last 15 years include partitioning methods (K-MEANS and K-MEDOIDS) [6], [7], hierarchical methods [8], [9], density-based methods [10]–[12], model-based methods [13], [14], graph-based methods [15], [16], grid-based methods [17], [18], etc., and their various combinations and improvements [19]–[22], etc. Some of these methods became classical. They proved to be successful in detecting certain cluster structures.

Among well-known clustering algorithms there are PAM (partitioning around medoids) [7], CLARANS (partitioning, medoid-based) [23], BIRCH (hierarchical) [9], OPTICS (hierarchical) [24], CHAMELEON (hierarchical) [25], DBSCAN (density-based) [10], DENCLUE (density-based) [11], CLIQUE (density-based and grid-based) [17], CURE (hierarchical) [8], AMOEBA (graph-based) [15], AUTOCLUST (graph-based) [16], MCLUST (model-based) [26], etc.

Most of the clustering methods require setting of the user specified parameters or prior knowledge to produce their best results. For example, partitioning methods such as K-MEANS and K-MEDOIDS face the problem of prescribing the number of clusters in advance. Classic hierarchical methods, such as single-linkage and complete-linkage methods, require setting of the merge/ split conditions to end the clustering process. The representative density-based methods DBSCAN and DENCLUE need to set density threshold parameter. Some other methods require specific data points distribution [27].

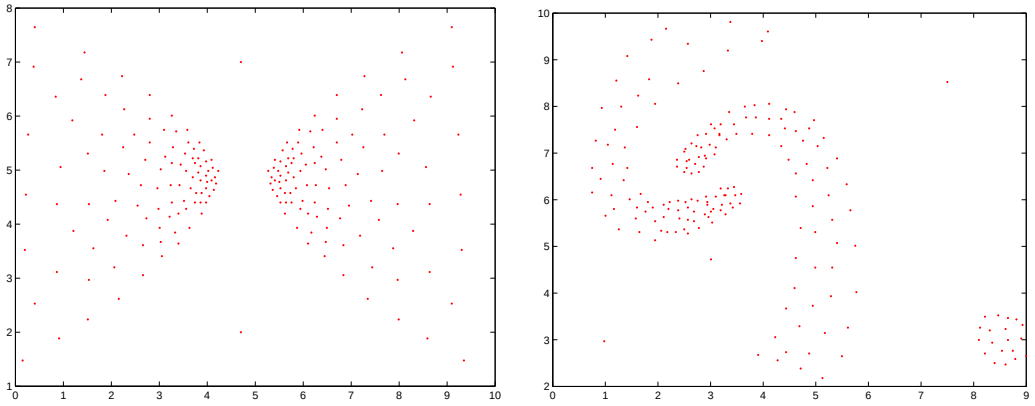
Except for specific applications when we have complete knowledge about the data set to ensure the validity of chosen parameters, these non-automatic methods are quite unstable because of the probability of introducing human-generated bias and are not efficient because of time consuming procedure of parameters tuning. So, it is important to develop a clustering method which can perform data clustering automatically.

Another new requirement to clustering algorithms is discovering of arbitrary shape clusters. Pure partitioning methods absolutely lack the ability to deal with clusters of arbitrary shape. Hierarchical methods can detect clusters of relatively complicated, but still not arbitrary, only

convex shape. Even improved hierarchical algorithms such as CURE cannot detect clusters of very complicated shape. Although hierarchical clustering can be effective in knowledge discovery, the cost of creating dendrogram could be prohibitively expensive since the corresponding algorithms are at least quadratic with respect to the number of data points [11]. Arbitrary shaped clusters can be detected by density-based algorithms such as DBSCAN and DENCLUE. The basic assumption in DBSCAN is that each data point inside cluster must have at least the certain number of data points in it's neighborhood of the given radius. The primary idea here is that density of the clusters should exceed some global threshold [10]. But this global threshold assumption prevents DBSCAN to determine clusters of different density. DENCLUE, contrary to DBSCAN exploits the idea of field function which efficiently measures local density and results in a fast performance. Still, it has the similar difficulty in recognition the clusters of different density. The reason is the same - dependence upon a predefined global threshold.

The distinguishing of different in density clusters is performed (to different extent) in modern graph-based algorithms such as CHAMELEON, AMOEBA and AUTOCLUST. Among them AUTOCLUST seems to be the most effective one. It is able to distinguish not only clusters of different density, but also sparse clusters which are adjacent to high-density clusters. But all algorithms can not recognize clusters with significantly non-uniform internal density which appear in computer graphics applications (for example, in point-based graphics), pattern recognition, and geo-image processing. In Fig.1, two examples of non-uniform density clusters are given. Just by visual inspection, we can easily recognize those clusters whose density are gradually changing inside, since the variation of discrimination according to the variation of density is natural and important. Moreover, the complexity of all these algorithms exceeds $O(N)$ because they require the computation of Delaunay Diagrams of similar graphs. The total complexity is estimated as $O(N \log N)$, where N is the number of data points. Graph-based algorithms are efficient in spatial databases and geographical information systems (GIS), but their application to computer geometry, for example, is limited due to non-linear performance and necessity of total re-computation when the area of clustering is dynamically changing.

Another problem arising from computer geometry clustering is detection of clusters belonging to low-dimensional hyperspaces. This problem becomes even more important due to recent development of point-based graphics. It is associated with the problem of surfels [28] determination for complex-shaped surfaces. When a surface is defined by the cloud of points unrelated to each



(a) (b)

Fig. 1. Examples of clusters with non-uniform density

other (for example, obtained from 3D scanning process), for a given point it is necessary to determine it's neighborhood consisting of such data points, which could be connected to it by a sufficiently short path along the surface. Such determination requires accurate clustering in Euclidean vicinity of a given data point taking in account that such vicinity can contain several different surfels and/or noise.

In common clustering algorithms, boundary detection is usually not incorporated. But, as it was mentioned above, the problem of cluster boundary definition arises, for example, in computer geometry applications where we need to compute (and draw) an accurate boundary of a geometric object determined by the set of points. In work [29], a definition of cluster as a solid described by a set of points endowed with influence functions was introduced. Such definition is capable not only to describe granular property of a cluster but also its boundary.

Classification problems give us another example when boundary detection of clusters is really important. Knowledge of cluster boundaries makes it possible to classify new data without repeating the clustering process [30]. Thus, detection of cluster boundaries is necessary for reasoning about clusters [31].

Furthermore, for clustering problems, detection of noise or outliers is also an important task. In some applications, such as point-based graphics, the definitions of noise and outliers are different. Noise is caused by environmental factors and is not part of data. Outliers are the original data points which do not belong to any cluster. In this paper, we consider outliers the

same as noise. Many traditional methods can detect noise when the density of clusters does not vary greatly. But it becomes quite a challenge to detect noise when the density of clusters changes not only between clusters but also inside clusters.

In this paper, we propose a new algorithm ADACLUS (ADAptive CLUStering) based on local adaptive influence function. It allows to automatically discover clusters of arbitrary shape and different density (density can be different inside cluster as well as between clusters), to detect cluster boundaries, and it is robust to noise. Contrary to many existing algorithms, ADACLUS does not require any parameter pre-setting if it works in pure automatic mode. But it can also work in manual mode when it gives the user a possibility to tune clustering process by setting three parameters all of which have clear meaning reflecting possible preferences in particular applications.

The rest of the paper is organized as follows. In Section 2, we describe new features of ADACLUS and its relation to other methods. In Section 3 the basic ideas and necessary definitions are introduced. Description of our new algorithm ADACLUS is presented by steps in Section 4. Complexity analysis of the algorithm is given in Section 5. Results of the algorithm for several test data sets are described in Section 6. The last Section contains conclusions and future work directions.

II. New Features of ADACLUS and its Relation to Previous Algorithms

ADACLUS is an adaptive density-based clustering algorithm. Similar to the pioneer DENCLUE algorithm, it is based on the general assumption that a "natural" clustering partition of a given data set situated in a metric space can be built through the use of the appropriate "field function" [1](i.e. data points integral influence function) which can be determined at any space point $\vec{X}$ as the sum of influences of all data points at $\vec{X}$. If the influence of any data point is restricted to a limited distance from it, or if it is rapidly decreasing with this distance, then it is enough to take into account only the influences of data points within some limited vicinity of the given point. This approach (first used in DENCLUE) results in a fast algorithm performance: the complexity of the algorithm could be made linear (or close to linear) with respect to the number of data points. It is a very important advantage for computer geometry applications

[1]We called the density function in DENCLUE field function here.

where real-time performance is essential. Another important (especially from geometrical point of view) advantage of field function clustering is that the corresponding algorithms have good capabilities to detect cluster boundaries [29], [32]. One more important advantage is that these algorithms can efficiently eliminate outliers.

ADACLUS possesses new important features and at the same time preserves all listed above advantages of function-based clustering algorithms. The main new feature of ADACLUS is that the parameters of influence function are not global and not predefined. Contrary to the previous density-based algorithms these parameters are not constant in ADACLUS. They are completely adaptive to the local situation in the vicinity of the point in consideration. The control parameters of ADACLUS could be set either automatically or manually. Generally, automatic settings work fine, but the user is provided also with the option to tune ADACLUS clustering rule from purely local-based to local-global-based one. In the latter case the influence function of ADACLUS takes in the account both local and global information combining them in certain proportion. The data points influence is modeled by adaptive piecewise linear function which ensure fast and efficient performance. The reasons of choosing such function and its advantages over other possible choices are described in the next section.

III. **Basic Definitions**

- **Metric choice**.

  ADACLUS can work with any metric which defines the distance between points in space.In ADACLUS we use another metric instead of Euclidean metric, ρ_1. Namely, we use the following definition of the distance between two points:

  Definition 1:

$$\rho_1(\vec{X}_1, \vec{X}_2) = \sum_{i=1}^{d} |x_1^i - x_2^i| \qquad \forall \vec{X}_1, \vec{X}_2 \in \mathbf{R}^d. \tag{1}$$

  From topological point of view the metric ρ_1 is equivalent to Euclidean metric, but results in faster calculations. It is easy to see that ρ_1 depends on the choice of coordinate system, but it is not a disadvantage in our case. If there is no "natural" coordinate system, associated with the data, this dependance will not affect significantly the results of clustering. And if there is such a coordinate system (which is the case for digital images or scanned data

having intrinsic lattice structure), then the above coordinate-dependent form of metric ρ_1 can even result in slightly more accurate performance of the algorithm.

- **Influence function choice**.

  Similar to DENCLUE and other algorithms which use field functions for clustering, ADACLUS also needs a predefined rule for measuring "influence" of certain data point at any point in the space. Once such rule is chosen, field function is obtained by summation of all the influences of data points at the arbitrary space point.

  In principle, ADACLUS, like DENCLUE, can work with any kind of influence function. But in order to create really fast clustering algorithm it is reasonable to choose the simplest (from the computational point of view) influence function which still preserves all necessary features for clustering. It is also reasonable to choose such influence function, which depends on meaningful parameters only. From both points of view, the Gaussian influence function $f_{Gauss}(\vec{P},\vec{X}) = c \cdot e^{-\frac{\rho(\vec{P},\vec{X})^2}{2\sigma^2}}$, which is often used in density-based clustering to measure influence of the data point $\vec{P}$ at the space point $\vec{X}$, seems to be not the best choice. Indeed:

  1) Gaussian function is not quite simple because it requires calculation of the exponent;
  2) Except the case when distribution of data points is Gaussian (which is applicable to special kind of model-based clustering only), Gaussian function is not better than many other monotonic decreasing functions, which could be much simpler than Gaussian one;
  3) The value of parameter σ in Gaussian function does not have any clear meaning if the distribution of data points is not Gaussian.
  4) The long tail of Gaussian distribution could increase the difficulty in the control of field value. It will bring the problem of parameter setting.

  Based on the expressed ideas, we propose the following simple, but flexible function which models the influence of data points in ADACLUS (see Fig.2)

  Definition 2: The influence of the data point $\vec{P}$ at any point $\vec{X}$ in the space is defined as

$$f_{a,b}(\vec{P},\vec{X}) = \begin{cases} 1, & \text{if} \quad 0 \le \rho_1(\vec{P},\vec{X}) \le a, \\ \frac{b-\rho_1(\vec{P},\vec{X})}{b-a}, & \text{if} \quad a < \rho_1(\vec{P},\vec{X}) \le b, \\ 0, & \text{if} \quad \rho_1(\vec{P},\vec{X}) > b, \end{cases} \tag{2}$$

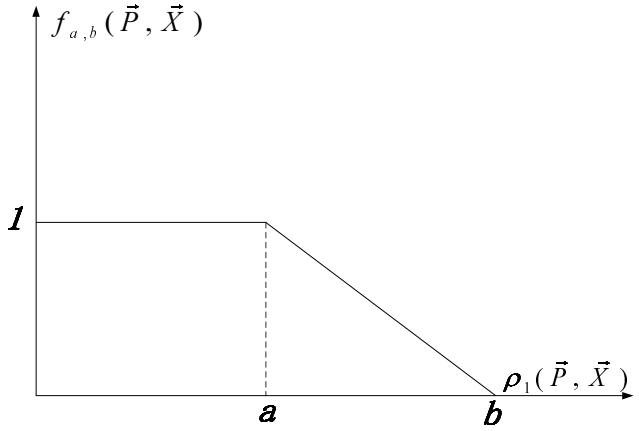


Fig. 2. Parametric influence function used in ADACLUS

where the distance $\rho_1(\vec{P},\vec{X})$ is calculated according to (1). Here parameter a denotes such maximum distance from the data point $\vec{P}$ which is still considered as "near" one. The parameter b denotes such minimum distance from $\vec{P}$ which is already considered as "far" distance.

It is assumed that $b \geq a$. Generally, there exists a non-zero gap between these thresholds, i.e. $b > a$. Both a and b are fully adaptive in ADACLUS and their calculation is based on the mixture of global and local information extracted from the data set.

In the case $b = a$ the function $f_{a,b}(\vec{P},\vec{X})$ transforms to the Square Wave influence function which is used in DBSCAN:

$$f_{a,a}(\vec{P},\vec{X}) = \begin{cases} 1, & \text{if} \quad 0 \leq \rho_1(\vec{P},\vec{X}) \leq a, \\ \\ 0, & \text{if} \quad \rho_1(\vec{P},\vec{X}) > a. \end{cases}$$

Based on the Definition 2 we define the field function at any point $\vec{X}$ in the space as:

$$F(\vec{X}) = \sum_{i=1}^{N} f_{a_i,b_i}(\vec{P_i},\vec{X}), \tag{3}$$

where the values of parameters a_i, b_i which are used to calculate the influence of the data point $\vec{P_i}$ at any space point $\vec{X}$, are calculated for each $\vec{P_i}$ taking in account the mixture of

information about the local distribution of data points in the $R-$neighborhood of $\vec{P}_i$ and the global distribution of data points in the whole screen (area of clustering). Proportion of this mixture is defined by the values of magnitude parameter G and radius R, which are calculated as described below.

By differentiating (3) with respect to all the components of coordinate system, the follows formula can be easily obtained for the gradient grad$F = \left(\frac{\partial F(\vec{X})}{\partial x^1}, \ldots, \frac{\partial F(\vec{X})}{\partial x^d}\right)$. We define here grad$F$ as zero in all points where $F(\vec{X})$ is not differentiable.

$$(\text{grad}F)^k = \frac{\partial F(\vec{X})}{\partial x^k} = \sum_{i=1}^{N} \frac{c_{i,k}(\vec{X})}{b_i - a_i}, \quad (1 \leq k \leq d), \tag{4}$$

where

$$c_{i,k}(\vec{X}) = \begin{cases} 1, & \text{if} \quad a_i < \rho_1(\vec{P}_i, \vec{X}) < b_i \text{ and } x^k > p_i^k, \\ -1, & \text{if} \quad a_i < \rho_1(\vec{P}_i, \vec{X}) < b_i \text{ and } x^k < p_i^k, \\ 0, & \text{in all other cases.} \end{cases} \tag{5}$$

On the basis of a field function and threshold T the ADACLUS algorithm determines *point-based clusters* and, in another version of the algorithm, *boundary-based clusters* which are defined as follows

Definition 3: Given field function F and threshold parameter T, the cluster (or, strictly speaking, point-based cluster) is such a subset C of the set of data points that

1) $F(\vec{P}) \geq T$ for each data point $\vec{P} \in C$;

2) $\forall \vec{P}_i, \vec{P}_j \in C$ there exists a continuous curve $x = x(s), 0 \leq s \leq 1$, such that $x(0) = p_i,\ x(1) = p_j,\ F(x(s)) \geq T\ \forall s \in [0,1]$;

3) it is impossible to add any more data points to C without violating properties 1) or 2).

Definition 4: Given field function F and threshold parameter T, the boundary-based cluster is a connected component of the domain $D = \{\vec{X} :\ F(\vec{X}) \geq T\}$.

Definition 5: The boundary of the boundary-based cluster $D = \{\vec{X} :\ F(\vec{X}) \geq T\}$ is simply it's boundary in the space. It is the domain (curve for 1-dimensional case, surface for 2-dimensional case, and hyper-surface for high dimensional case) $B = \{\vec{X} :\ F(\vec{X}) = T\}$ or a part of it.

- **Control parameters of ADACLUS**.

  ADACLUS depends on 3 scalar parameters which could be either calculated automatically from the number of data points and their distribution on the screen (as described below), or defined by the user. All these parameters have clear meaning which gives the user real possibility to make ADACLUS more adapted for certain area of applications or to tune clustering results in a specific case. These parameters are:

  1) $G \in [0,1]$ - *magnifier parameter.* If $G = 1$ the clustering is purely local-based (i.e. based on the "fully magnified view"). If $G < 1$, then clustering is based on the mixture of global and local information about the data set. Maximum portion of global information is added when $G = 0$, but even in this case the algorithm remains local-adaptive. One can imagine that G describes the "magnifying glass" through which algorithm "looks" at the data while determining clusters. If $G = 1$, then algorithm "looks through a microscope", and therefore performs clustering on a very local level. It results in the tendency to achieve small "micro"-clusters. When G decreases, the "microscope" is being continuously converted to a simple glass without any magnification. The smaller G becomes the more the tendency to glue neighboring small clusters into bigger ones increases. Parameter G must be chosen (automatically or manually) before the algorithm starts.

  2) The second parameter, $T \geq 0$, is a *threshold parameter.* It determines a field function value threshold which divides inner-cluster and outer-cluster domains in the data space. T serves like a hierarchial parameter: big values of T result in more solid, dense inner-cluster domains and in treating relatively sparse areas as outer-cluster domain. Contrary to the first parameter, parameter T may be chosen (or altered) at the end of the clustering process, because changing it's value requires repetition only of the final step of the algorithm.

  3) S, *screen size,* - natural number which gives the upper bound for the "screen" linear size in pixels. The term "screen" is used here for the domain in data space where clustering is performed. Without lost of generality, we assume that this domain is a d-dimensional rectangular parallelogram which longest side does not exceed S pixels.

IV. ADACLUS Algorithm Description

ADACLUS algorithm performs clustering in several steps. Without lost of generality we will assume in this section that the data set consists of $N \quad d-$dimensional vectors: $\{\vec{P}_1, \vec{P}_2, \ldots, \vec{P}_N\}$,

$\vec{P}_i = (p_i^1, \ldots, p_i^d) \in \mathbf{R}^d$, and $N > 1$.

If not specified by the user, the values of parameters T and S are set automatically to $T = 1,\ S = 1000$. Here, the value $T = 1$ means that value of the field function for cluster boundaries is set to 1. The value $S = 1000$ means that the distance between data points which is less than 1/1000 of the side length of the screen is considered as a very small one and such points are glued together. Such value of S is quite natural for visual clustering and for computer geometry purposes.

The parameter G is calculated during STEP 3 if it was not pre-defined by the user.

STEP 1. Quantization and integer-coding.

First, for each of d coordinate axes the minimum and maximum values of corresponding data point component are determined: $m_k = \min\{p_i^k, 1 \le i \le N\}, M_k = \max\{p_i^k, 1 \le i \le N\}, 1 \le k \le d$. Then, for each $1 \le k \le d$ the interval $[m_k, M_k]$ is divided into S intervals ("quanta") $\Delta_j^k (1 \le j \le S)$ of equal length: $\Delta_j^k = [m_k + (j-1)\frac{M_k - m_k}{S}, m_k + j\frac{M_k - m_k}{S}]$, where S is the screen size parameter. Finally, each of d vector components of each data point is coded by a pair of integers (the number of the component and the number of the interval to which this component belongs):

$$p_i^k \to (k, \widetilde{p}_i^k), \quad \text{if } p_i^k \in \Delta_{\widetilde{p}_i^k}^k.$$

All further actions, related to all data points, are performed only with pairs of integers (k, j), such that the corresponding interval Δ_j^k is not empty (i.e. some data points components fall into it). Evidently, the total number of such pairs does not exceed dN.

STEP 2. Minimal distances (MD) calculation.

For each data point $\vec{P}_i$ the corresponding minimal distance $MD(i)$ from this data point to other data points is calculated in the sense of metric ρ_1. Since coordinates of data points were already converted to integers during STEP 1, all the minimal distances will be expressed by positive integers. It immediately follows from the definition of the metric ρ_1 (see Definition 1) and the assumption $N > 1$.

STEP 3. Parameter G automatic determination.

One of the aims of ADACLUS is to serve computer-vision related problems, and therefore the general idea here is to follow a "human's eye approach" in clustering. In automatic parameter setting, the value of magnifier parameter G is chosen according to the number of data points and their distribution on the screen. The exact formulas are given below, but in general, the rule is as follows.

Parameter G is chosen close to 1 (maximum magnification) when there is not too many data points in the screen and/or their distribution is approximately uniform inside the expected clusters. In both cases, the clustering is based on local information only, without taking in account global features of the whole picture. The reason is that if there is such a small amount of data points that one can notice any of the data points "at one glance", then the "natural eye-based" clustering should depend exclusively on the local details of the data points distribution. Moreover, the small number of data points is not enough to generate reliable global features of distribution which could be useful for clustering. Below we assume that number of data points, starting from which a human's eye starts missing individual points and therefore the manner of discrimination starts shifting from local to global one, is approximately 200 points. Another threshold - the number of points, starting from which global features of the data set become fully important for the human's eye, - we choose as 2000. Both numbers are very approximative, but the clustering results are quite robust to their variations.

In the case when the points are distributed uniformly inside clusters, there is also no need to add global information to local one (for any number of points), because the global information in this case tells us nothing new and important about the data point distribution properties. So, in the both cases clustering can be performed purely locally. Vice versa, if there are many data points and/or they are distributed quite irregularly with a lot of small details of distribution, then it is natural to look at the whole picture more globally to unite tiny clusters rather than separate them (if it is not the case in a certain application, user can always set the magnifier parameter G manually). The calculations of G are performed in the following order.

First, based on the minimal distances which have been calculated during STEP 2, the values of mean $m_{MD} = \frac{1}{N}\sum_{i=1}^{N} MD(i)$ and standard deviation $std_{MD} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(MD(i) - m_{MD})^2}$ are calculated. Let us denote

$$t = \frac{std_{MD}}{m_{MD}}.$$

Value of t is used in ADACLUS as an approximative measure of the deviation of data points density inside clusters from the "regular" one (i.e. uniform inside clusters). Let us show that, if the density of data points inside clusters is approximately uniform, then the value of t should be close to $\frac{1}{\sqrt{d}}$. First, we recall the fact that the uniform distribution of data points in clouds appears as a consequence of their Poisson distribution in data space [33], [34]. Next, according to (1), the minimal distance MD can be approximated by a sum of d independent one-dimensional minimal distances, taken along the certain coordinate directions. It is known from probability theory that in the case of Poisson distribution of data points in $\mathbf{R}$, the distance between neighboring points (i.e. minimal distance in our terminology) is distributed by the exponential law. Consequently, the law of distribution of minimal distances MD in the case of uniform density clusters in $\mathbf{R}^d$ can be well approximated by the law of distribution of a sum of d independent one-dimensional variables distributed exponentially with locally the same parameter λ. Of course, λ can vary from one cluster to another, but this does not create a problem because the following statement is valid.

Lemma 1: Assume that $\eta = \xi_1 + \ldots \xi_d$, where $\xi_1, \ldots, \xi_d$ are independent one-dimensional random variables, distributed exponentially $P(\xi_i < x) = 1 - e^{-\lambda x}, \quad 1 \leq i \leq d, \lambda > 0$. Then, for any $\lambda > 0$:

$$t_\eta = \frac{\sqrt{E(\eta - E\eta)^2}}{E\eta} = \frac{1}{\sqrt{d}},$$

where $E\zeta$ denotes the mathematical expectation (mean value) of a random variable ζ.

Proof: For the exponential distribution,

$$E\xi_i = \frac{1}{\lambda}, \quad E(\xi_i - E\xi_i) = \frac{1}{\lambda^2}, \quad 1 \leq i \leq d.$$

Consequently $E\eta = \frac{d}{\lambda}$ and therefore independence of ξ_i implies that $E(\eta - E\eta) = \frac{d}{\lambda^2}$. Therefore, $\frac{\sqrt{E(\eta - E\eta)^2}}{E\eta} = \frac{1}{\sqrt{d}}$. ■

Lemma 1 implies that in the case of uniform (or approximately uniform) density of data points inside each cluster, the value of t will be close to $\frac{1}{\sqrt{d}}$. When the distribution of points inside clusters deviates from the uniform one, the value of t increases and t is a sensitive characteristic. For example, for $d = 2$, the value of t will be about $\frac{1}{\sqrt{2}} \approx 0.7$ in the case of uniform density clusters and about 1.0 for clusters which are already rather irregular in density.

Note, that we do not make here any assumptions about generator of the data set. The proposed method of measuring density irregularity by the value of characteristic t is not model-based. It

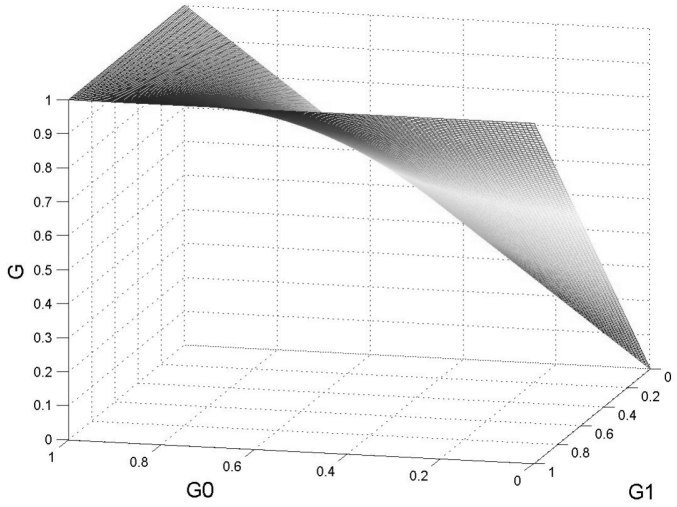


Fig. 3. Parameter G determination based on the values of $G0$ and $G1$

is applicable for any sort of data - stochastic or deterministic (chaotic, for example).

The value of magnifier parameter G is calculated as:

$$G = G_0 + G_1 - G_0 G_1, \tag{6}$$

where

$$G_0 = \begin{cases} 1, & \text{if} \quad N \leq 200, \\ 1 - \frac{N-200}{1800} & \text{if} \quad 200 \leq N \leq 2000, \\ 0, & \text{if} \quad N \geq 2000. \end{cases} \tag{7}$$

$$G_1 = \begin{cases} 1, & \text{if} \quad t \leq 1/\sqrt{d}, \\ 1 - \frac{t-(1/\sqrt{d})}{1-(1/\sqrt{d})} & \text{if} \quad 1/\sqrt{d} \leq t \leq 1, \\ 0, & \text{if} \quad t \geq 1. \end{cases} \tag{8}$$

In (7) and (8), the values of G_0 and G_1 model the "magnifier factors" due to the number of data points N and to the inner-cluster density irregularity characteristic t. The final "magnification

TABLE I

EXTREME G VALUES

$G_1 \backslash G_0$	0	1
0	0	1
1	1	1

rate" G is calculated in (6) by means of bilinear surface, Fig.3 (see, for example [35]) determined by Table.I of extreme G values, where G_0 and G_1 are the entries of the table.

As it was mentioned above, we assume in (7) that 200 data points is a small enough number to observe each of them separately and therefore, "natural" clustering in this case should be purely local-based. It is assumed that 2000 data points is a big enough number to make observation of individual points impossible, forcing therefore the "natural" clustering to be performed from more global point of view. The thresholds $t = (1/\sqrt{d})$ and $t = 1$ used in (8), were already validated above.

STEP 4. Calculation of the localization radius R.

In order to initiate the ADACLUS algorithm we first determine the size of the neighborhood in which the local considerations will be performed for each data point. Taking in account the value of G, which was calculated during STEP 3, the radius R of the "ball" representing this neighborhood is chosen as:

$$R = \min\{GR_{loc} + (1-G)R_{glob},\ 1/2\sqrt[d]{100}\, m_{MD}\}, \tag{9}$$

$$R_{loc} = \min\{x : x \geq R_{glob},\ g^{MD}_{emp}(x) = 0\}, \tag{10}$$

$$R_{glob} = \min\{x : P_{emp}(MD < x) = 0.9\}, \tag{11}$$

where $g^{MD}_{emp}(x)$ denotes the empirical density (frequency histogram) of the MD distribution calculated in STEP 2, R_{glob} represents a 90% of this distribution, and R_{loc} is it's first zero greater than 90%. See Fig.4. Note that according to Definition 1, the "R-ball" in $\mathbf{R}^d$ will be represented by the d-dimensional cube with the side length $2R$ in Euclidean metric. The radius R of this "ball" is restricted to $c \cdot m_{MD}$ (c times average minimal distance between data points) in order to keep the complexity of the calculations linear with respect to N for any kind of

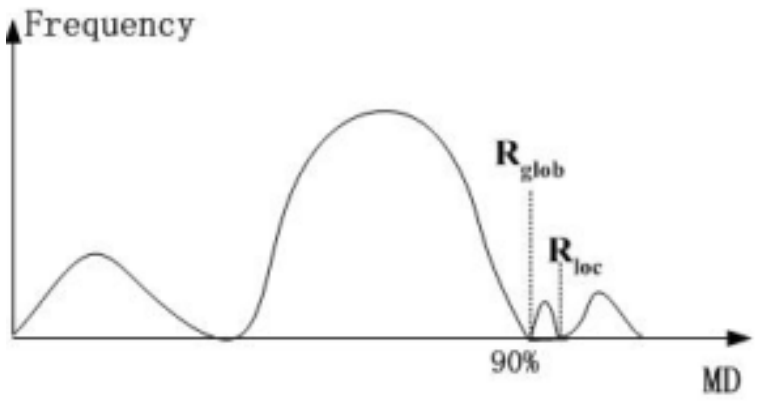


Fig. 4. Localization radius choice in ADACLUS

data set. The value of the coefficient $c = 1/2\sqrt[d]{100}$ is chosen in such a way that the average number of data points in a d-dimensional Euclidean cube with side length $2R$ is restricted to approximate 100 data points maximum.

According to (9)-(11) the localization radius R is computed as the restricted mixture of "global" and "local" values of the neighborhood radius. The mixture is taken in the proportion, determined by the parameter G (which was already calculated in STEP 3 or manually set by the user). The value of R_{glob} represents "10%-cut maximum" of the distribution of MD. 10% margin is cut off taking in account the possibility of the outliers and for algorithm stability. If the distribution of data points is approximately the same in different parts of the data set, then R_{glob} gives a suitable value for the neighborhood radius. Otherwise, it may be not so. One example is the situation when there are several dense parts in the data set which include majority of data points, but there are also sparse areas which are big enough but contain a relatively small amount of data, see Fig.5. If we are performing clustering from the global point of view, then, when we cluster in sparse parts we should remember that they are really sparse. This information could not be extracted locally - it comes from the global distribution of MD and is summarized in the value of R_{glob}. So, for global-based clustering we can use this value. .

But if the aim is to perform clustering in sparse parts of the screen more locally, without taking in account far away density characteristics, then the appropriate value of the neighborhood radius should be determined by sparse areas not based on the whole MD distribution, but on it's right tail only. As we still do not want to be influenced by the outliers in this case, we choose R_{loc}

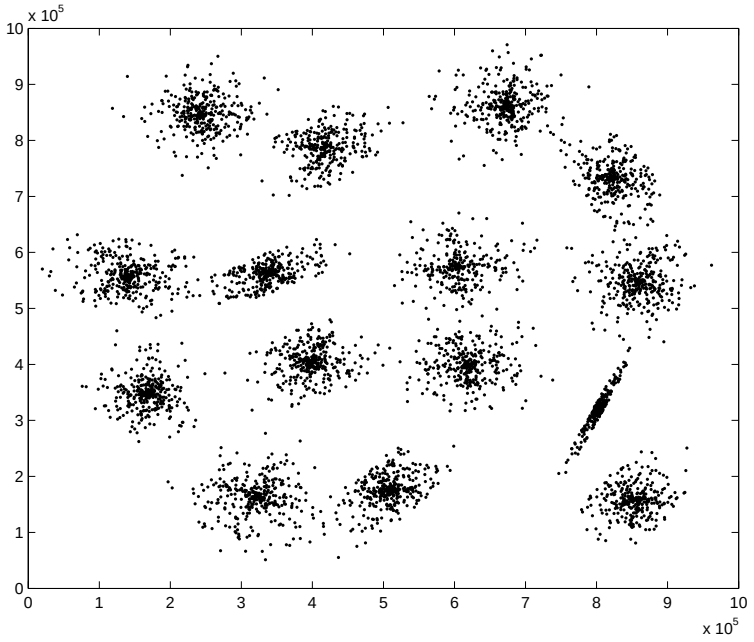


Fig. 5. Example of a data set with non-uniform distribution of data points

- the first zero of MD histogram greater than 90%. It corresponds to the largest value of MD which still belongs to sparse component of MD distribution, cutting the outliers off, see Fig.4. Such value will be good for sparse areas, but it will also work for dense areas, because the final decisions are made based on the microanalysis performed inside the R-neighborhood of the data point. If this neighborhood was chosen too big, it will not spoil the results but rise the amount of calculations. However, in the situation which we consider now, the total area of dense parts is not very big in comparison to the whole screen, so the amount of extra calculations will be reasonable.

The balance between these two situations is calculated by the formula (9) in accordance with the value of parameter G.

STEP 5. Local microanalysis. Determination of a_i and b_i for each data point $\vec{P}_i$.

After localization radius R was determined in STEP 4, for each data point $\vec{P}_i$ $(1 \leq i \leq N)$, we mark all data points belonging to its neighborhood $Ne_R(\vec{P}_i) = \{\vec{P} : \rho_1(\vec{P}, \vec{P}_i) \leq R\}$, including $\vec{P}_i$ itself. And we consider the part of the whole MD distribution (built in STEP 2) which consists of only those minimal distances, which are associated with the data points from $Ne_R(\vec{P}_i)$. Let's denote the corresponding frequency histogram $g_{emp}^{MD;\,i}(x)$, it's mean value m_{MD}^i, and it's standard

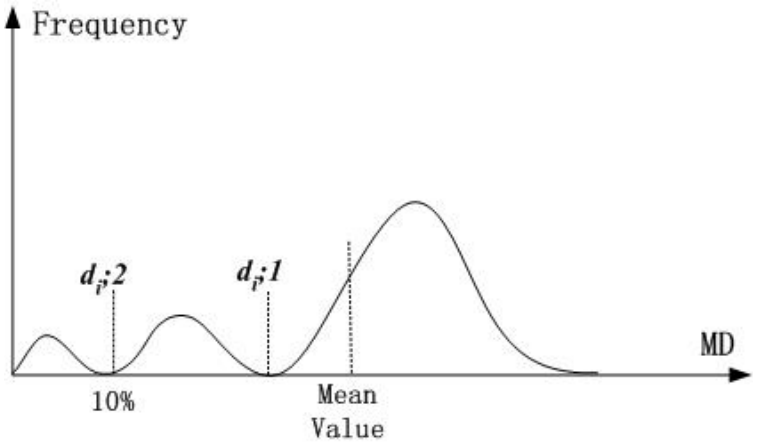


Fig. 6. The choice of local parameters a_i, b_i in ADACLUS

deviation std^i_{MD}.

Based on the histogram $g^{MD;\,i}_{emp}(x)$, we define a_i as the "typical minimum" of MD inside $Ne_R(\vec{P}_i)$ in order to eliminate the effect of local outliers and irregular dense area, which represents "typically small" value of MD in $Ne_R(\vec{P}_i)$. More precisely, we define a_i as following. (see Fig.6)

$$a_i = \frac{d_{i;\,1} + d_{i;\,2}}{2}, \tag{12}$$

where $d_{i;\,1}$ is the biggest zero of $g^{MD;\,i}_{emp}(x)$ which does not exceed the mean:

$$d_{i;\,1} = \max\{x : x \leq m^i_{MD},\ g^{MD;\,i}_{emp}(x) = 0\}, \tag{13}$$

and $d_{i;\,2}$ is 10%-fraction of the distribution $g^{MD;\,i}_{emp}(x)$:

$$d_{i;\,2} = \max\{x : P^{MD;\,i}(MD < x) = 0.1\}. \tag{14}$$

We use $d_{i;\,1}$ to eliminate the potential effect cased by local outliers, which are the outliers detected under local view. One example is the situation when the "R-ball" of a data point in sparse area includes several data points near to a dense cluster. We should not count these data points in, since they are local outliers for the dense cluster. For the same reason, when the "R-ball" of a data point in sparse area includes several data points belonging to a dense cluster, we also need to cut these data points off. This is why we set $d_{i;\,2}$.

Finally, taking in the account "$3\sigma-$rule", we define b_i based on the value of magnifier parameter G as:

$$b_i = a_i + (4 - 2G)std^i_{MD}. \tag{15}$$

So, the gap between a_i and b_i varies around "3σ" ranging from 2 times of standard deviation in the case of local clustering to 4 times of standard deviation in the case of maximum portion of global information added.

STEP 6. Computation of the field function in all data points.

Based on the values a_i, b_i $(1 \leq i \leq N)$ which are determined in STEP 5, we calculate the field function $F(\vec{P}_i)$ for all data points $\vec{P}_1, \ldots, \vec{P}_N$ by means of the formula (3).

STEP 7. Determination of the point-based clusters.

To determine point-based clusters (Definition 3), ADACLUS employs the standard hill-climbing algorithm and applies the similar method for clustering procedure as [11]. The direction of the hill-climbing is determined by the vector of the field function gradient, given in (4). After clustering process, data points which can not be assigned to any clusters are labeled as outliers.

STEP 8. Determination of the boundary-based clusters and boundary detection.

Direct drawing exact cluster boundaries $\{\vec{X} : \ F(\vec{X}) = T\}$ according to the Definition 4 may be time-consuming in the case of big screen, because it requires additional calculation of field function $F(\vec{X})$ in many space points. (Note that in STEP 6 the field function was calculated for data points only.) The total number of space points in the screen after quantization (see STEP 1) equals to S^d, which could be quite big number in comparison with N. But the amount of calculations still can be reduced to $O(N)$ by preliminary marking the space points, such that the field function is greater than zero for them. This can be done together with calculation of a_i, b_i during STEP 5. Namely, after b_i was determined for a certain i, we mark all space points in the ρ_1-ball of b_i-radius with the center in $\vec{P}_i : Ne_{b_i}(\vec{P}_i) = \{\vec{X} : \ \rho_1(\vec{X}, \vec{P}_i) < b_i\}$. This will reduce the amount of calculation to $O(N)$, where N is the number of space points in which the field function is to be calculated during this step.

V. Complexity Analysis: Linear Complexity of ADACLUS

The time complexity of ADACLUS is analyzed theoretically by steps below.

STEP 1. First step required dN operations for determining $m_k, M_k \quad (1 \leq k \leq d)$ and another dN operations for the coding procedure. The total complexity of this step is $O(dN)$.

STEP 2. In order to make the complexity of this step linear with respect to N, the tables of correspondence between quantum numbers (i.e. integers from 1 to S) and the data points which have this number as the code of their $k-$th coordinate, are created for each $1 \leq k \leq d$. These tables are computed simultaneously with the coding procedure during STEP 1, so it requires dN additional operations only. Based on the mentioned tables, the minimal distance (MD) is calculated for each data point $\vec{P}_i$ by searching the nearest data point among only such data points $\vec{P}_j$ that their coordinate codes $\widetilde{p}_j^k$ are neighboring to the coordinate code $\widetilde{p}_i^k$. It requires $O(dN)$ operations, so the total complexity of this step is $O(dN)$, too.

STEP 3. In this step, the calculations are performed according to the formulas (6), (7), and (8). The complexity of these calculations does not depend neither on N nor on d. It is constant.

STEP 4. Calculations in this step are performed according to the formulas (9), (10), and (11). They are based on MD distribution which is obtained together with MD values during STEP 2 (without rising it's complexity). Complexity of the STEP 4 does not depend neither on N nor on d. It linearly depends on S only.

STEP 5. The complexity of this step equals to the complexity of calculation of the histograms $g_{emp}^{MD;\,i}(x)$ for all data points $\vec{P}_i$. All other calculations here consist in direct evaluation of the expressions (12), (13), (14), and (15). In order to calculate the histogram $g_{emp}^{MD;\,i}(x)$ for $\vec{P}_i$, it is enough to determine all data points which belong to the vicinity $Ne_R(\vec{P}_i)$. This could be done based on the tables of correspondence between quantum numbers and the data points (see complexity analysis for STEP 2). Therefore, the total complexity of the STEP 5 equals to $O(cN)$, where c is the average number of data points in the vicinity $Ne_R(\vec{P}_i)$ for an arbitrary data point $\vec{P}_i$. According to the definition of R in (9), there are not more than 100 data points in $Ne_R(\vec{P}_i)$ in average, so the complexity of the step does not exceed $O(100N)$ which is linear with respect to N and can be written as $O(N)$.

STEP 6. According to (3), the complexity of this step in $O(kN)$, where k is the average number of data points influencing the arbitrary data point $\vec{P}_i$ (i.e. such $\vec{P}_j$ that $f_{a_j,b_j}(\vec{P}_i) > 0$). It is easy to see that the same k represents also the average number of data points, which are

influenced by a randomly chosen data point $\vec{P}_i$. The data points, influenced by $\vec{P}_i$ are such and only such data points $\vec{P}_j$, that their distance to $\vec{P}_i$ is less than b_i : $\rho_1(\vec{P}_i, \vec{P}_j) < b_i$. According to (12),(15), b_i does not exceed $a_i + 4std^i_{MD}$, where a_i is the typical minimum of the minimal distance between data points in the R-neighborhood of $\vec{P}_i$ and std^i_{MD} is the standard deviation of this minimal distance in this neighborhood. The worst case, giving the largest number of data points in the neighborhood of $\vec{P}_i$, is evidently the case when typical minimum of MD equals to typical MD in the neighborhood of $\vec{P}_i$, i.e. when the distribution of data points in this neighborhood is uniform (because the more there are MD's larger than minimal ones, the less will be the number of data points in the neighborhood of $\vec{P}_i$). But in the case of approximately uniform data points distribution in the neighborhood of $\vec{P}_i$, according to Lemma 1, the value of std^i_{MD} will equal approximately to $\frac{1}{\sqrt{d}} m^i_{MD}$, and according to (12)-(14), the value of a_i for exponential distribution of MD (corresponding to uniform distribution of data points) will not exceed m^i_{MD}. Consequently in the uniform case we have:

$$b_i \leq \left(1 + \frac{4}{\sqrt{d}}\right) m^i_{MD} < 5m^i_{MD},$$

where m^i_{MD} is average MD in the neighborhood of $\vec{P}_i$. Consequently, in the case of uniform density of data points around $\vec{P}_i$, the average number of them in b_i-neighborhood of $\vec{P}_i$ will be approximately equal to 10^d. For any other distribution, it will be smaller. Consequently,

$$k \leq 10^d.$$

It is theoretical upper bound for k. In real calculations, it is much smaller, because for uniformly distributed data points, the magnifying parameter G is set to 1 (see(6),(8)) and in this case, according to (15), $b_i = a_i + 2std^i_{MD} < 3m^i_{MD}$. So, the inequality for k turns to $k < 6^d$, which is still not a typical value but an upper bound.

We conclude that the complexity of this step is typically lower than $O(6^d N)$, which means $O(36N)$ in the case of $\mathbf{R}^2$ screen. Theoretical upper boundary for the complexity of this step is $O(10^d N)$. This complexity is linear with respect to N.

STEP 7. The hill-climbing algorithm has linear complexity with respect to the number of data points [11], so the complexity of this step is $O(N)$.

STEP 8. The complexity of this step is $O(N)$ as explained in the description of this step.

TABLE II

SHAPE FEATURES OF TESTING DATASETS

Data-set name	Shape feature 1	Shape feature 2	Data points distribution	Number of Outliers
D1	Circular	Non-concave	Non-uniform	3
D2	Non-circular	Concave	Non-uniform	3
D3	Circular	Concave	Non-uniform	3

TABLE III

STATISTIC FEATURES OF TESTING DATASETS

Data-set name	Number of data points	Mean m_{MD}	Standard deviation std_{MD}	$\frac{std_{MD}}{m_{MD}}$
D1	76	0.4714	0.3793	0.8045
D2	191	0.3242	0.3262	1.0062
D3	289	0.2980	0.2842	0.9537

Therefore, ADACLUS has linear complexity. The total complexity is $O(N)$ with respect to the number of data points N. It is $O(dN)$ with respect to both data points number and dimension d of the data space.

VI. **Comparison and Performance Analysis**

The proposed algorithm ADACLUS was implemented and tested on the following data sets. We designed three testing data sets $D1$ to $D3$ with non-uniform distribution of data points inside clusters. Generally, there are clusters of arbitrary shapes in the data sets. We differ two shape features of clusters in our data sets: circular/non-circular, and concave/non-concave. The data sets names, the shape and statistic features of these data sets are listed in Table.II and Table.III.

We applied ADACLUS and classic clustering methods such as K-MEANS, hierarchical Single Linkage and Complete Linkage methods, and DBSCAN on $D1$ - $D3$ data sets. We choose

these classic clustering methods because they are representative in this area, and standard implementations of these methods are available. We can avoid the potential problems causing by different implementations, which could lead us to bias in results comparisons.K-MEANS method is still the most widely used method, and it is based on the optimization principles which form a foundation for many well-known data mining techniques. We employed the squared Euclidean distance and random selection of initial cluster centers for K-MEANS in our testing. Linkage hierarchical methods have a good ability of handling clusters with some complicated shapes. DBSCAN is another well known density-based clustering method for dealing with arbitrary shape clusters and outliers. The implementation of DBSCAN is provided by the author and all parameter settings are based on the instruction given in the original paper ($k = 3$ and $MinCard = 4$ for our 2-dimensional data sets).

By comparing with those methods, we can have a more clear idea of the performance of ADACLUS. The characteristics of ADACLUS such as extracting both global and local information from data set and detecting outliers based on both global view and local view, have been well demonstrated. All results provided here are generated by automatic setting of parameters, which was described in Section IV. For the user who has a very clear expectation of the data set, parameters also can be set based on his knowledge of data.

In Fig.7, the picture of $D1$, statistic information, cluster boundaries and clustering result of $D1$ by ADACLUS are shown. The three clusters are well separated as well as three outliers.

In Fig.8, the clustering results of testing data set $D1$ by four classic clustering algorithms are shown.

Data set $D1$ is a challenging data set with non-uniform density clusters and outliers. There are three outliers and three clusters among which two clusters have relative uniform inner cluster density but different inter cluster density and the biggest cluster has non-uniform inner cluster density. The difficulty here is that the smallest cluster is very close to the biggest one. From the clustering results, the ability of ADACLUS for discovering clusters based on both global and local information is well demonstrated. K-MEANS and Single-linkage method can not separate the clusters correctly. Complete-linkage method fails to detect the outliers. DBSCAN successes in the outliers detection, but fails in separation of the nearby clusters.

In Fig.9, the picture of data set $D2$, statistic information, cluster boundaries and clustering result of $D2$ by ADACLUS are shown.

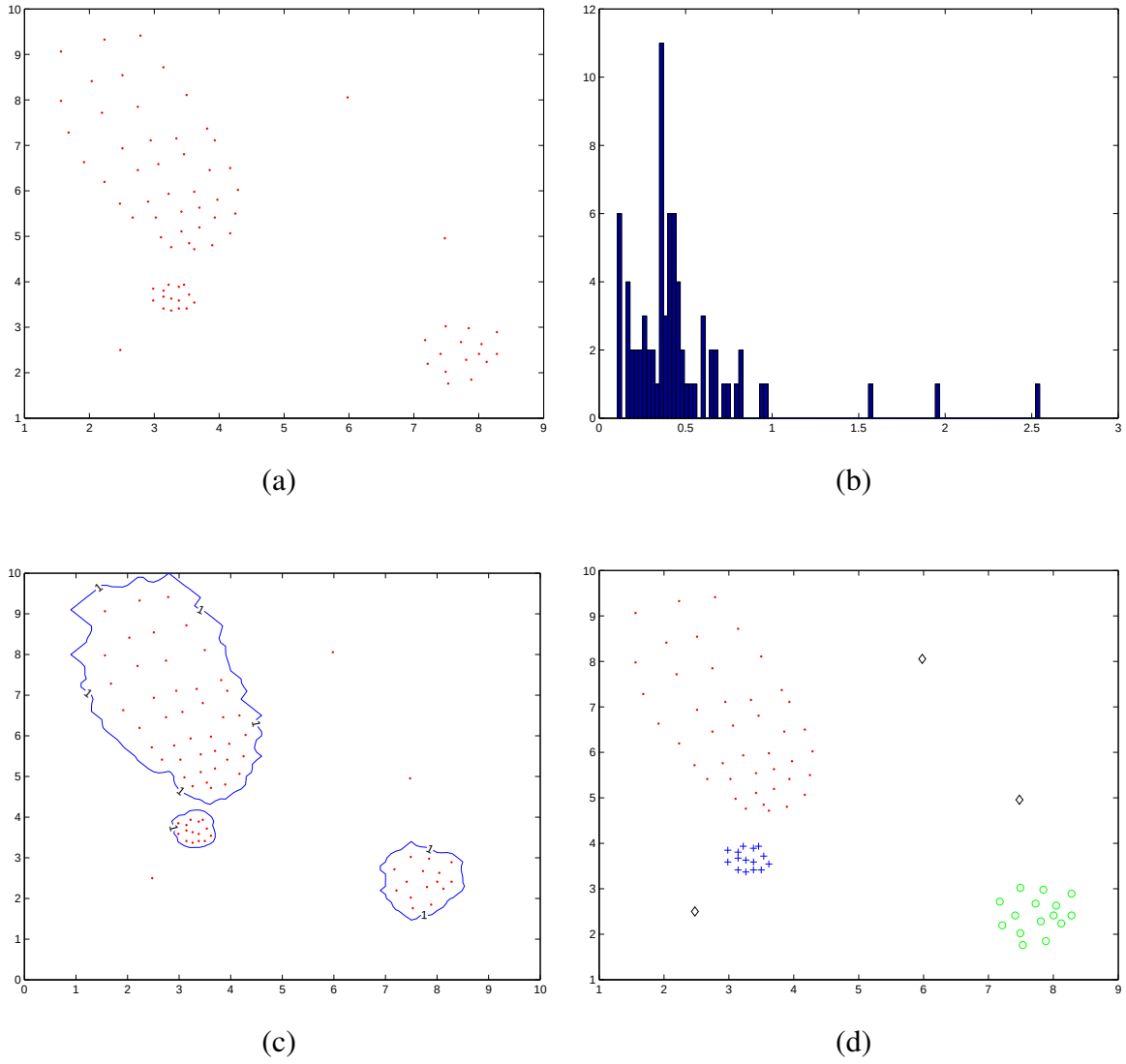


Fig. 7. The testing on data set $D1$ of ADACLUS: (a)The picture of data set $D1$; (b)The distribution of all minimal distances of data set $D1$; (c)The boundary of data set $D1$ built by ADACLUS, $T = 1$; (d)Clustering result of $D1$ by ADACLUS

In Fig.10, the clustering results of testing data set $D2$ by four classic clustering algorithms are shown.

Data set $D2$ is a very challenging one for most clustering algorithms. There are three clusters together with three outliers among which one is very close to one cluster. There are two clusters with non-uniform inner density. Based on visual inspection, we can easily notice the density differences between clusters and outliers, although the smallest distance between one outlier to the nearby cluster is smaller than the biggest distance between data points inside that cluster.

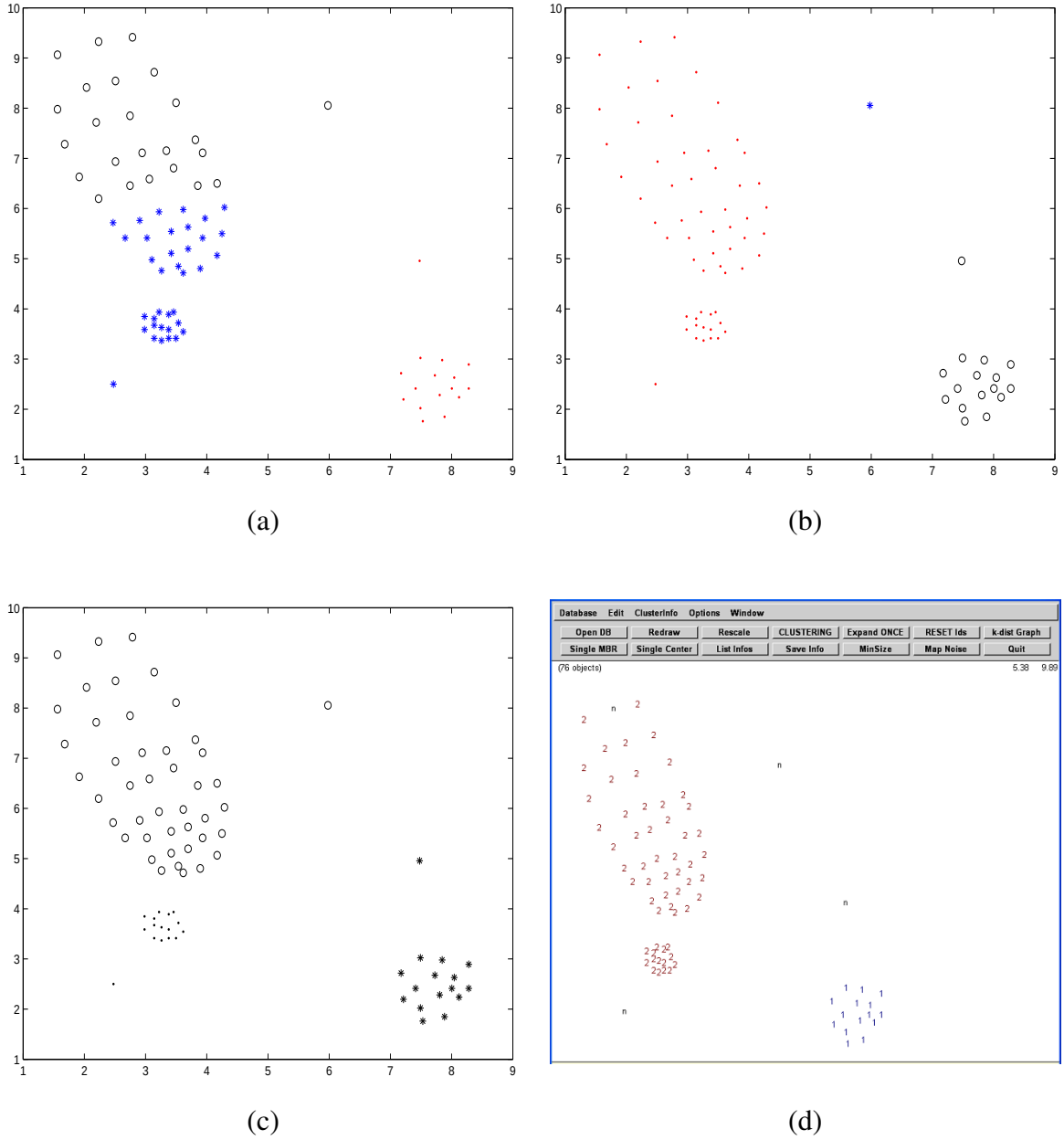


Fig. 8. Clustering results of $D1$ by comparison methods: (a)Clustering result of testing data set $D1$ by K-MEANS; (b)Clustering result of $D1$ by Single-linkage algorithm; (c)Clustering result of $D1$ by Complete-linkage algorithm; (d)Clustering result of testing data set $D1$ generated by DBSCAN, where $EPS = 0.8095$

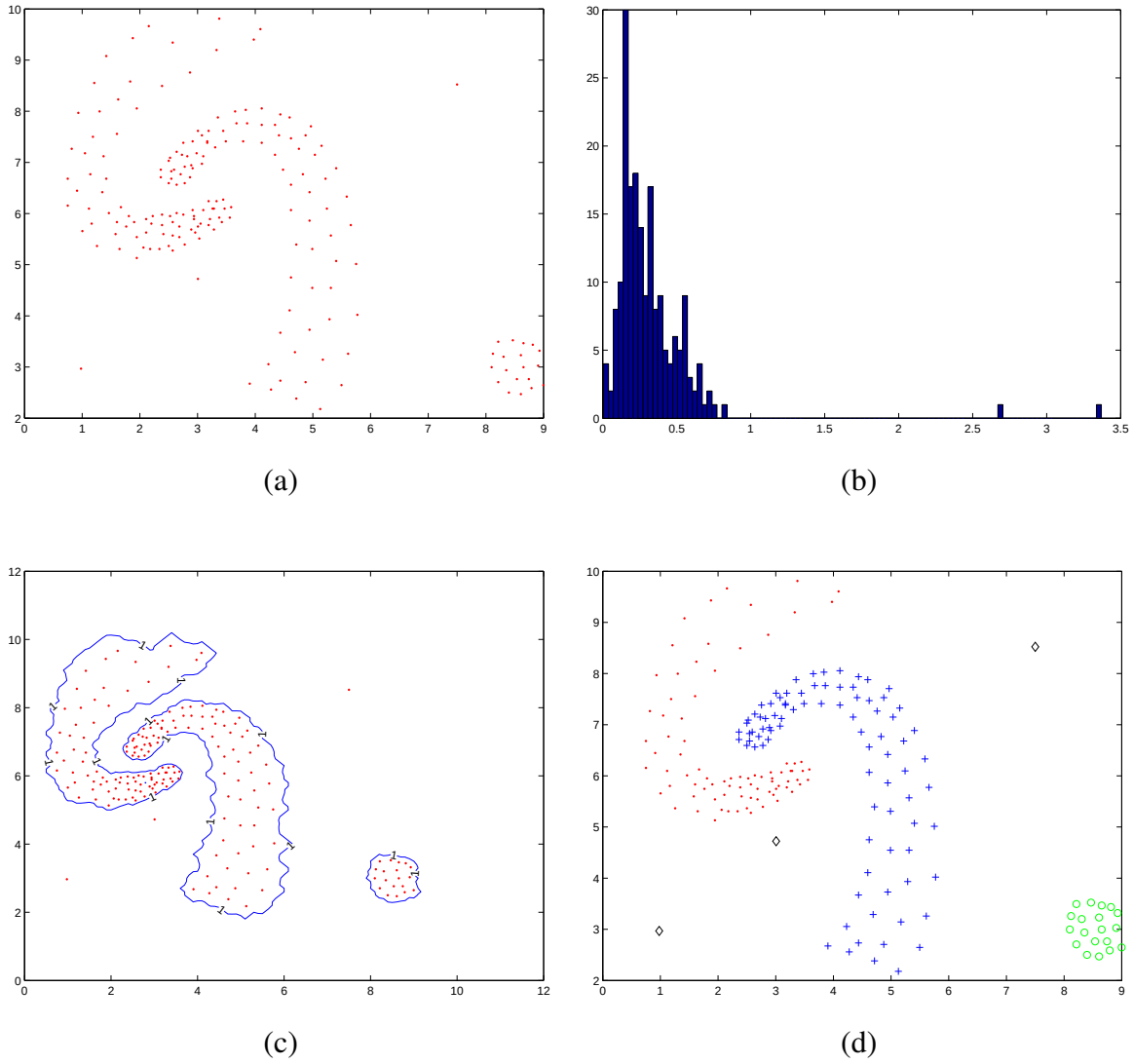


Fig. 9. The testing on data set $D2$ of ADACLUS: (a)The picture of data set $D2$; (b)The distribution of all minimal distances of data set $D2$; (c)The boundary of data set $D2$ built ADACLUS, $T = 1$; (d)The clustering result of data set $D2$ by ADACLUS

Except ADACLUS, no method can separate all clusters and detect all outlier correctly at the same time.

In Fig.11, the picture of data set $D3$, statistic information, cluster boundaries and clustering result of $D3$ by ADACLUS are shown.

In Fig.12, the clustering results of testing data set $D3$ by four classic clustering algorithms are shown.

Data set $D3$ which is another very challenging data set to all well-known clustering methods.

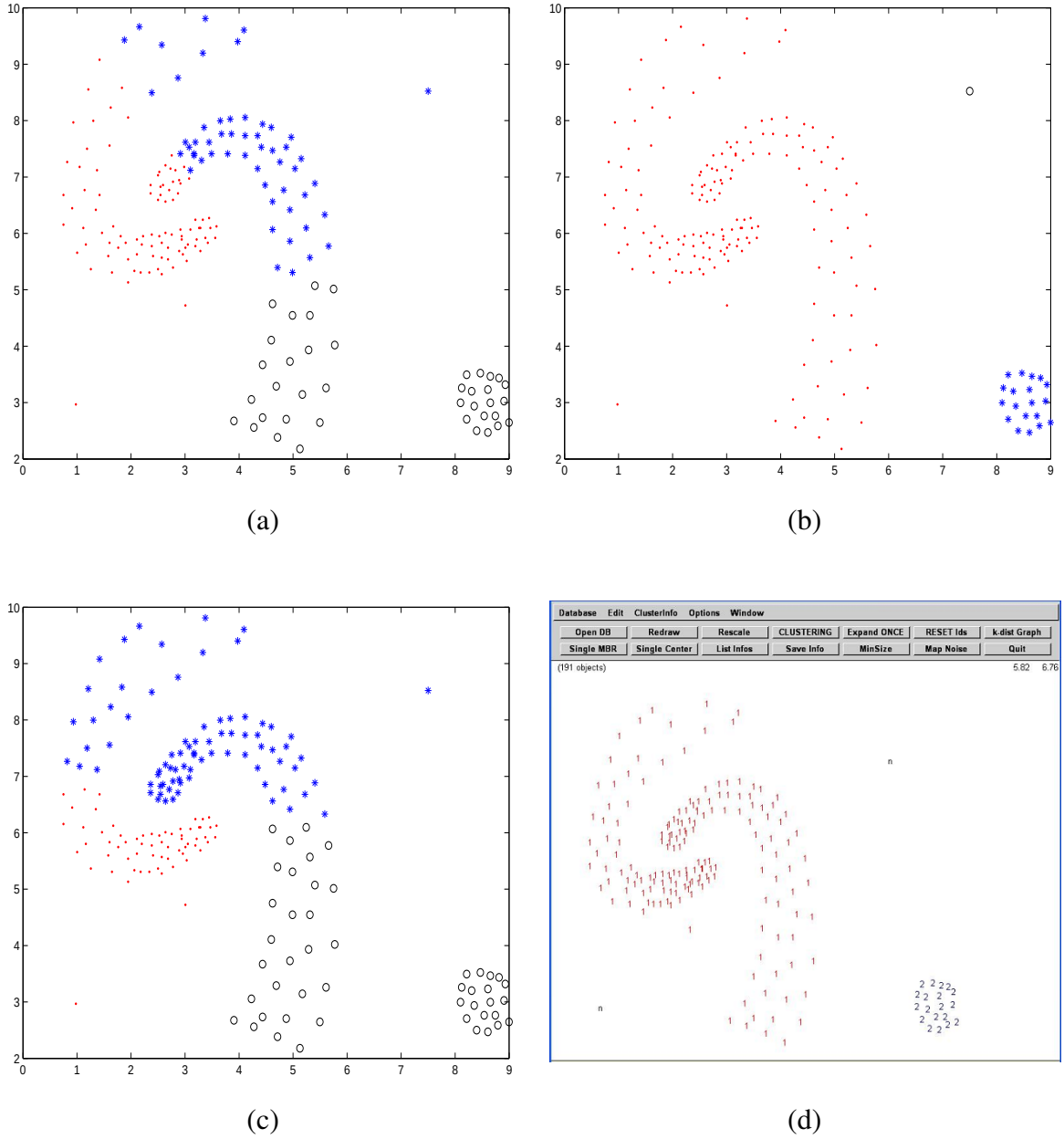


Fig. 10. Clustering results of data set $D2$ by comparison methods: (a)Clustering result of testing data set $D2$ by K-MEANS; (b)Clustering result of $D2$ generated by Single-linkage algorithm; (c)Clustering result of $D2$ generated by Complete-linkage algorithm; (d)Clustering result of testing data set $D2$ generated by DBSCAN, where $EPS = 0.7432$

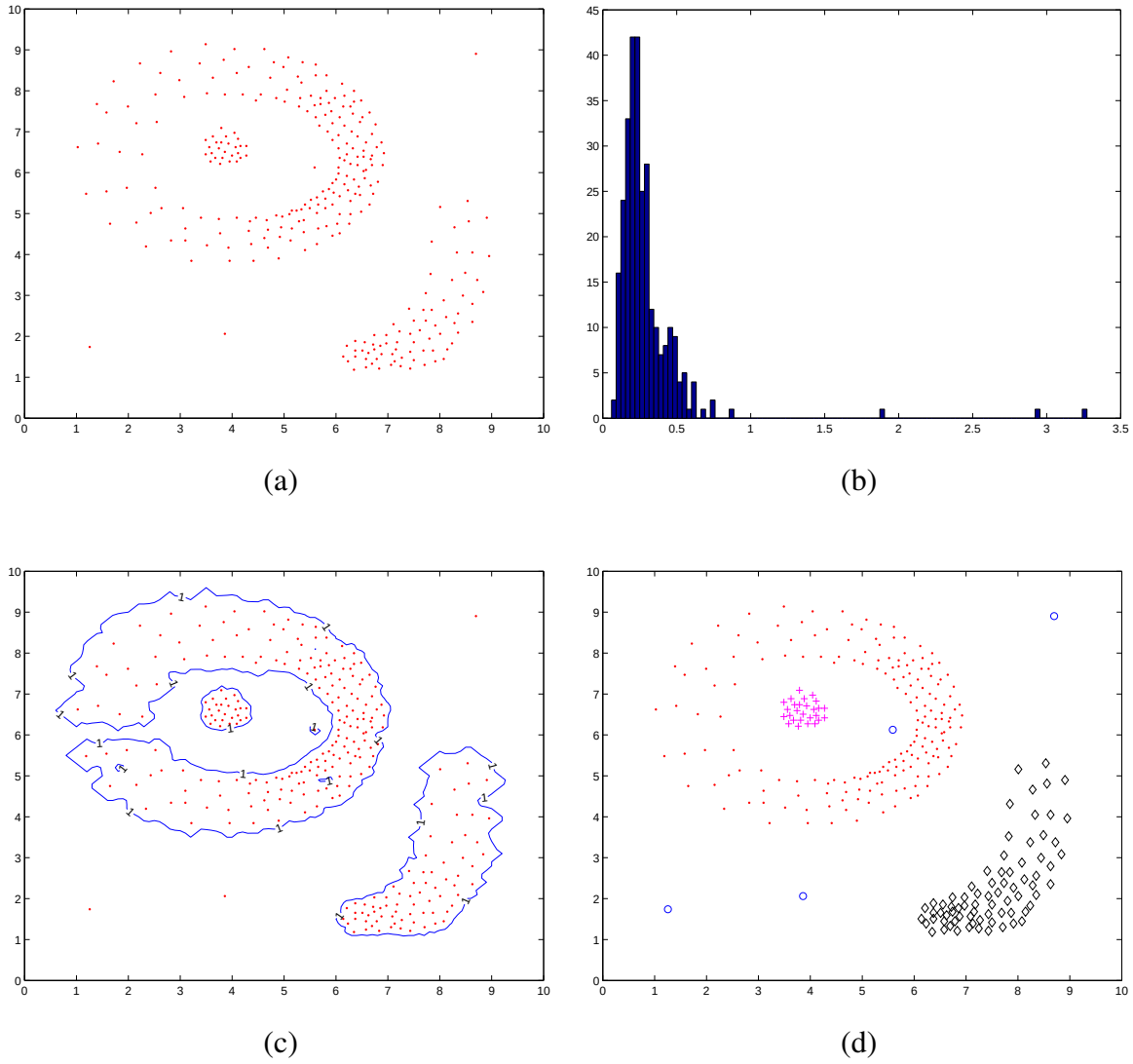


Fig. 11. The testing on data set $D3$ of ADACLUS: (a)The picture of data set $D3$; (b)The distribution of all minimal distances of data set $D3$; (c)The boundary of data set $D3$ built ADACLUS, $T = 1$; (d)The clustering result of data set $D3$ ADACLUS

In the data set $D3$, there is one small cluster inside another cluster which is a big ring shape cluster with non-uniform density. There is an outlier located inside the ring as well. It can be observed from the results that the ADACLUS detects natural clusters even in this complicated case. As it is seen from the Fig.12, other algorithms can not cluster this data set in the expected way. K-MEANS method and Complete-linkage method can not discover the ring shape cluster since they lack the ability to handle clusters with concave shapes. Single-linkage method suffers from correct separation of clusters. DBSCAN fails in keeping the integrity of the ring shape

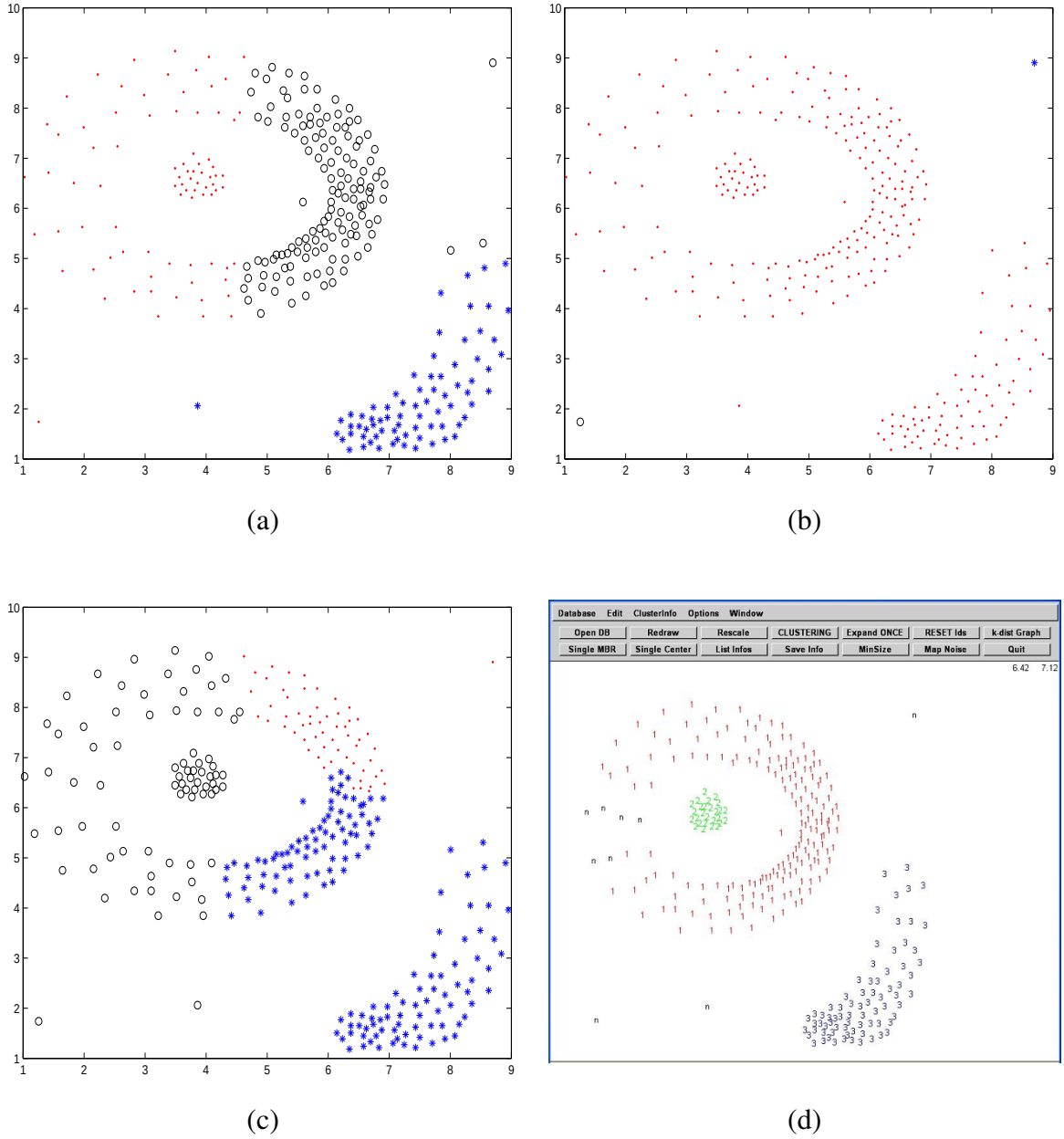


Fig. 12. Clustering results of data set $D3$ by comparison methods: (a)Clustering result of testing data set $D3$ by K-MEANS; (b)Clustering result of $D3$ generated by Single-linkage algorithm; (c)Clustering result of $D3$ generated by Complete-linkage algorithm; (d)Clustering result of testing data set $D4$ generated by DBSCAN, where $EPS = 0.6737$

cluster and detecting one local outlier.

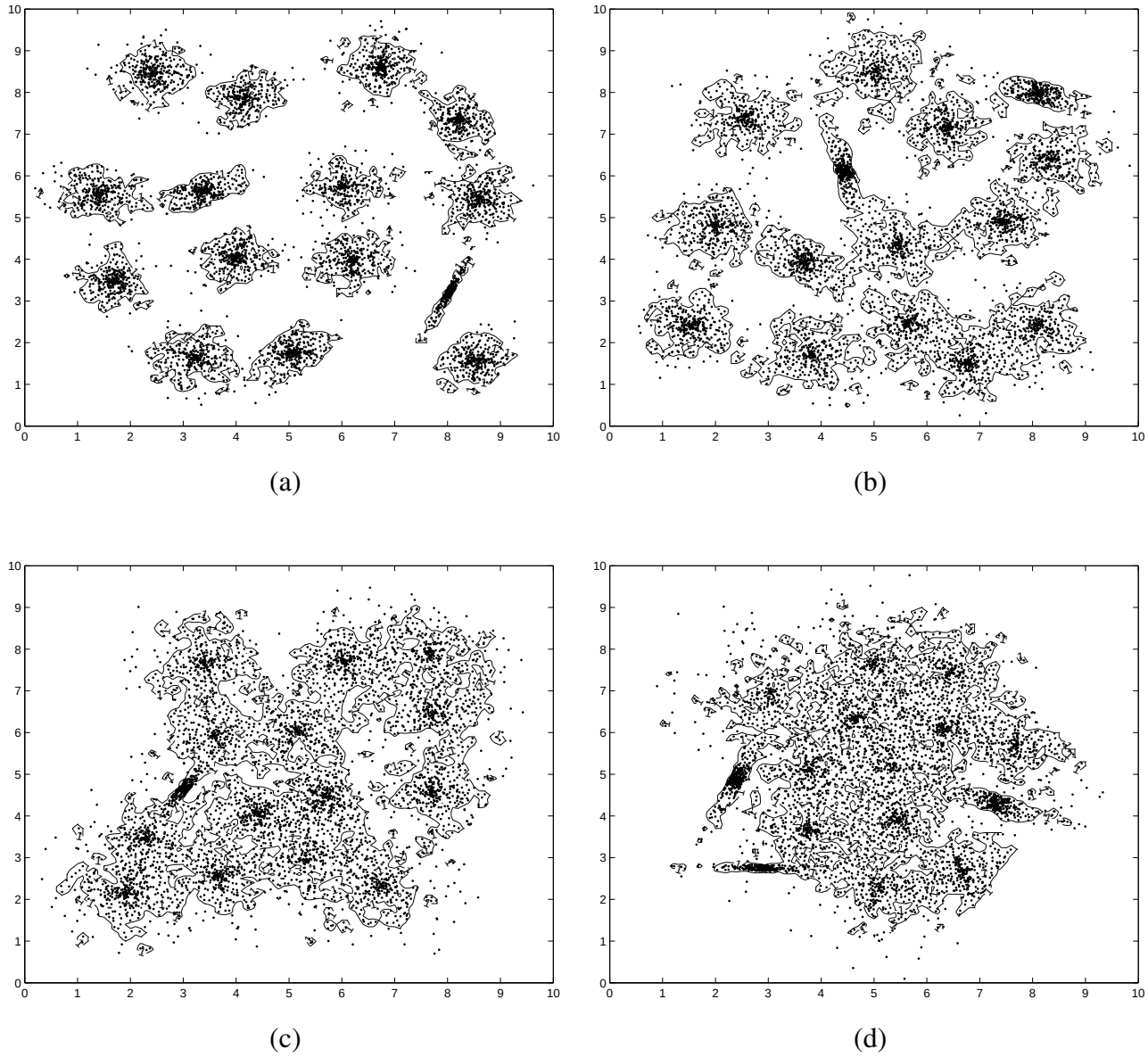


Fig. 13. Clustering results of ADACLUS for data sets $S1 - S4$, from (a) to (d) correspondingly

We also tested ADACLUS with some large data sets $S1 - S4$ proposed in work [36]. We showed the results of clustering on the data sets with clusters boundaries. Fig.13 shows the results of ADACLUS for testing data sets $S1 - S4$. The details of the data sets are listed in Table.IV.

The $S1 - S4$ data sets are two-dimensional data sets with 15 predefined clusters. The distances between the clusters keep decreasing from data set $S1$ to $S4$. Fig.13 shows the results of

TABLE IV

STATISTIC FEATURES OF DATA SETS $S1 - S4$

Data-set name	Number of data points	Mean m_{MD}	Standard deviation std_{MD}	$\frac{std_{MD}}{m_{MD}}$
S1	5000	0.0459	0.0497	1.0818
S2	5000	0.0547	0.0530	0.9677
S3	5000	0.0591	0.0550	0.9311
S4	5000	0.0534	0.0557	1.0426

ADACLUS. For data set $S1$, ADACLUS separated 15 clusters very well. Since the information of local distribution of each cluster is used, outliers are detected by ADACLUS as well. Parts of the data set inside boundaries are formed by data points with continuously changing density. The areas where the field density changes suddenly are cut by ADACLUS. From data set $S1$ to $S4$, the number of clusters keeps decreasing because of the decreasing distances between clusters. For data set $S4$, we have only one big cluster for the whole data set. But based on the local information, there are still many small isolated parts surrounding the big cluster which can be regarded as outliers.

To test the capacity of ADACLUS in handling large data set with irregular shape and non-uniform density distribution. We applied ADACLUS on two large data sets $LD1$ and $LD2$. The statistic information of those data sets are list in Table.V. From the result, we can draw the conclusion that ADACLUS is able to deal with quite large and complex data set, especially, when data density inside clusters are changing as long as local outliers existing.

VII. **Real World Application of ADACLUS**

To illustrate the practical value of ADACLUS, we apply it on the real world data set collected from European Topic Center on Air and Climate Change (ETC/ACC), which focuses on the EU-ECCP (European Climate Change Programme), CAFE (Clean Air for Europe) and related legislation, such as the Greenhouse Gas Monitoring Mechanism and the Air Quality Framework Directive. To achieve those objectives, ETC/ACC established the European air quality database

TABLE V

STATISTIC FEATURES OF LARGE DATA SETS $LD1 - LD2$

Data-set name	Number of data points	Mean m_{MD}	Standard deviation std_{MD}	$\frac{std_{MD}}{m_{MD}}$
LD1	4760	1.8511	1.2844	0.6939
LD2	8170	1.5252	0.7191	0.4714

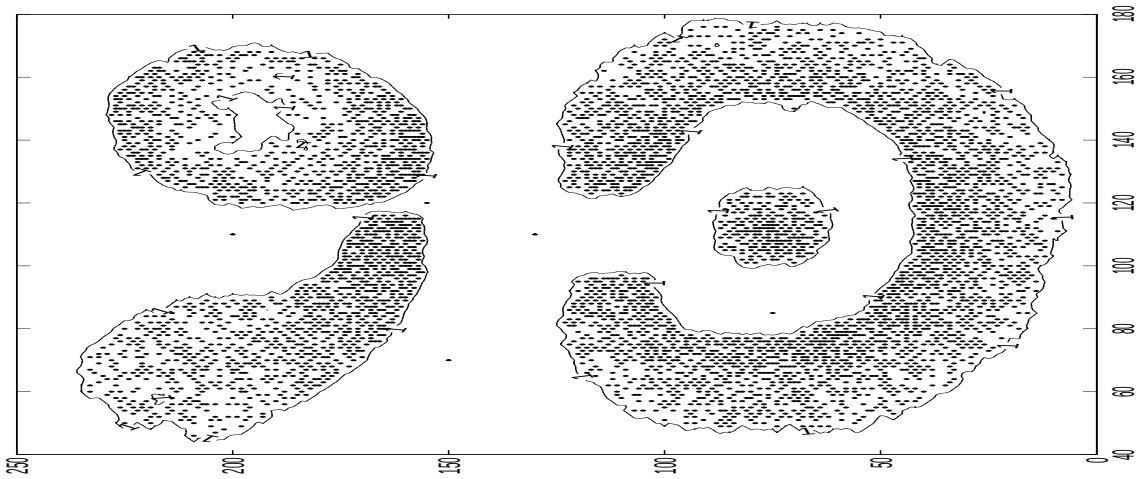


Fig. 14. Clustering result of large data set $LD1$ by ADACLUS

system which contains next to multi-annual time series of measurement data and their statistics for a representative selection of stations throughout Europe. We chose the data set of the locations of stations. From our clustering result shown in Fig.16 and Fig.17, the coverage of the stations over Europe and the relationship between stations can be discovered especially by the building cluster boundaries. People who work in this area can conduct further investigation in the distribution of station locations based on our result. This result is generated automatically by ADACLUS. We can also set the parameters according to the knowledge of data set, such as the coverage of each station, to fit the specific application requirements. The statistic features of this data set is shown in Table.VI.

ADACLUS allows automatically to discover clusters of arbitrary shape, different in density

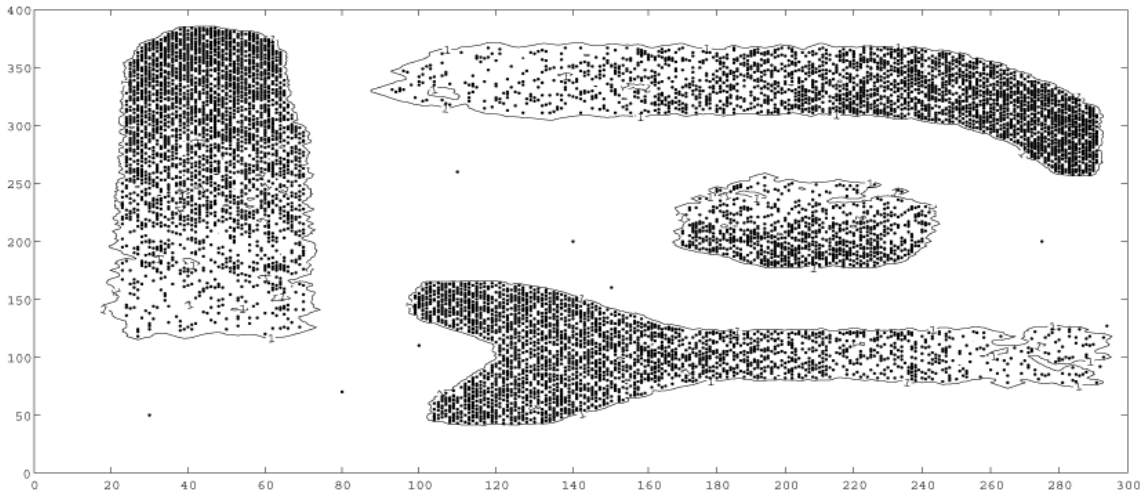


Fig. 15. Clustering result of large data set $LD2$ by ADACLUS

TABLE VI

STATISTIC FEATURES OF OUR REAL WORLD DATA SET

Data-set name	Number of data points	Mean m_{MD}	Standard deviation std_{MD}	$\frac{std_{MD}}{m_{MD}}$
Real world data	5456	0.1061	0.2959	2.7903

and clusters of non-uniform density, to detect boundary of the clusters, and it is robust to outliers. It was demonstrated that the algorithm has linear performance that is very important for real-time applications. Although the proposed algorithm is generally automatic, the user is given the possibility to tune the algorithm to his/her application by choosing values of three parameters that have clear meaning. ADACLUS is also able to be applied in high dimensional applications by extending the concepts of surface to hyper-surface and solid to hyper-solid.

VIII. **Conclusion and Future work**

In this paper, we proposed and described new clustering and boundary detection algorithm ADACLUS. The algorithm is based on the introduced adaptive influence function. It allows automatically to discover clusters of arbitrary shape, different in density and clusters of non-

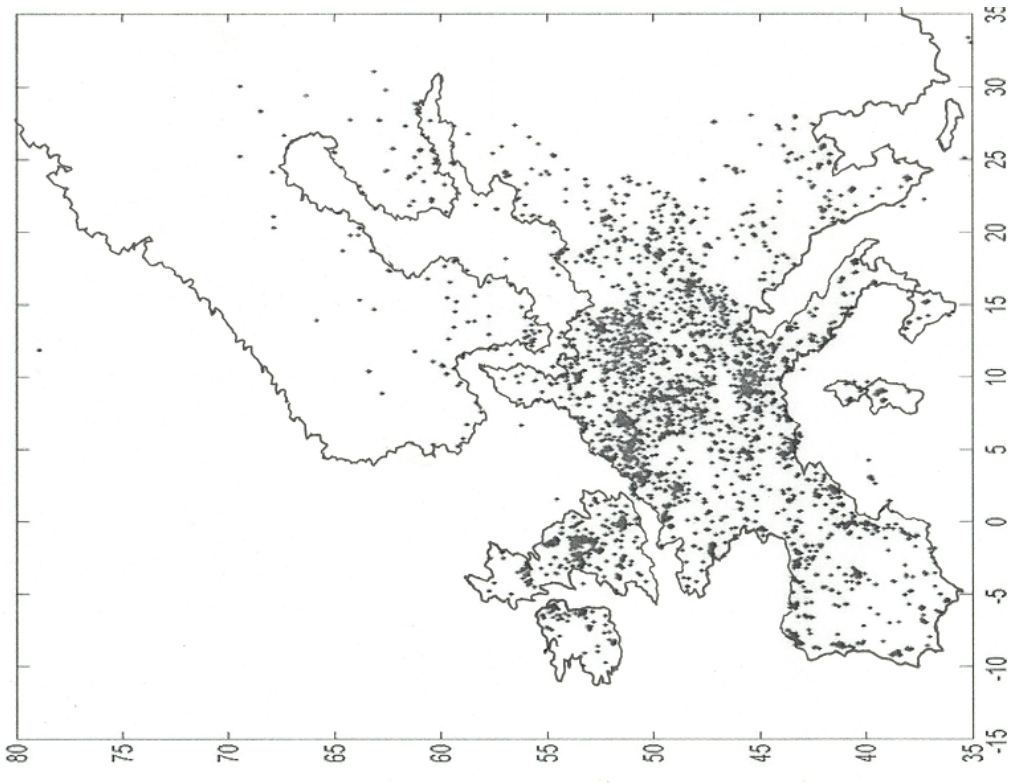


Fig. 16. The locations of stations throughout Europe

uniform density, to detect boundary of the clusters, and it is robust to noise. It was demonstrated that the algorithm has linear performance that is very important for real-time applications. Although the proposed algorithm is generally automatic the user is given possibility to tune the algorithm to his/her application by choosing values of three parameters that have very clear meaning.

The algorithm was tested on 2-dimensional data sets, and performance was compared with performance of other well-known algorithms. As the next step in our research, we plan to apply our algorithm to the real application data including the one in computer geometry for detection of clusters belonging to a low-dimensional hyperspace.

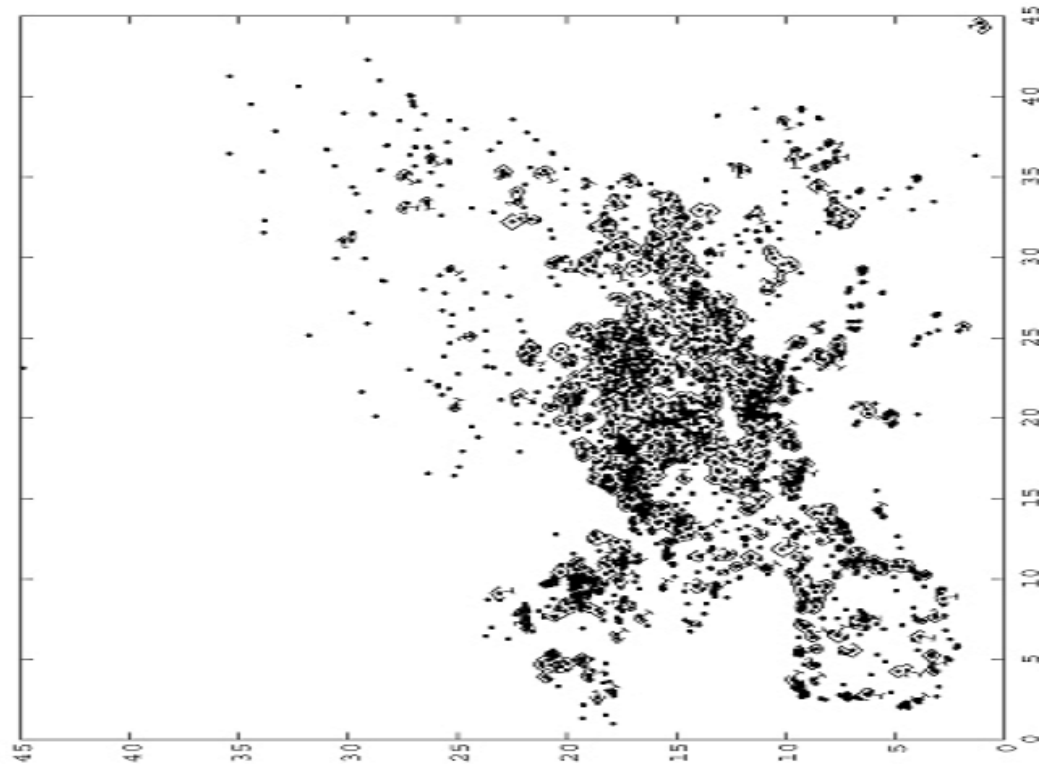


Fig. 17. Clustering result of ADACLUS

REFERENCES

[1] A. Adamson and M. Alexa, "Approximating Bounded, Non-orientable Surfaces from Points," in *Proc. of Shape Modeling International 2004*, 2004, pp. 243–252.

[2] H. Pfister and M. Gross, "Point-Based Computer Graphics," *IEEE Computer Graphics and Applications*, vol. 4, pp. 22–23, 2004.

[3] M. Andersson, J. Giesen, M. Pauly, and B. Speckmann, "Bounds on the k-Neighborhood for Locally Uniformly Sampled Surfaces," in *Proc. of the 1st Symposium on Point-Based Graphics*, 2004, pp. 167–171.

[4] M. Pauly, M. Gross, and L. Kobbelt, "Efficient Simplification of Point-Sampled Surfaces," in *Proc. of the conference on Visualization '02*, 2002, pp. 163–170.

[5] E. Gobbetti and F. Marton, "Layered Point Clouds," in *Eurographics Symposium on Point-Based Graphics*, Switzerland, Jun 2004, pp. 113–120.

[6] J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," in *Proc. 5th Berkeley Symp. Math. Statist. Prob.*, 1967, pp. 281–297.

[7] L. Kaufman and P.J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, Wiley-Interscience, 2rev ed edition, 2005.

[8] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim, "CURE: An Efficient Clustering Algorithm for Large Databases," in

Proc. of the 1998 ACM SIGMOD International Conference on Management of Data, New York, NY, USA, 1998, pp. 73–84, ACM Press.

[9] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an Efficient Data clustering method for very large databases," in *Proc. of the 1996 ACM SIGMOD international conference on Management of data*, 1996, pp. 103–114.

[10] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proc. of the Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226–231.

[11] A. Hinneburg and D.A. Keim, "An Efficient Approach to Clustering in Large Multimedia Databases with Noise," in *Proc. of the Fourth International Conference on Knowledge Discovery and Data Mining*, 1998, pp. 58–65.

[12] Xiaowei Xu, Martin Ester, Hans-Peter Kriegel, and Jörg Sander, "A Distribution-Based Clustering Algorithm for Mining in Large Spatial Databases," in *Proc. of the Fourteenth International Conference on Data Engineering*, Washington, DC, USA, 1998, pp. 324–331, IEEE Computer Society.

[13] C. Fraley and A.E. Raftery, "Model-Based Clustering, Discriminant Analysis, and Density Estimation," *Journal of the American Statistical Association*, vol. 97, pp. 611–631, Jun 2002.

[14] C. Fraley and A.E. Raftery, "MCLUST: Software for Model-Based Clustering, Density Estimation and Discriminant Analysis," Tech. Rep. 415, Department of Statistics, University of Washington, 2002.

[15] V. Estivill-Castro and I. Lee, "AMOEBA: Hierarchical Clustering Based on Spatial Proximity Using Delaunay Diagram," in *Proc. of the 9th International Symposium on Spatial Data Handling*, 2000, pp. 7a.26–7a.41.

[16] V. Estivill-Castro and I. Lee, "AUTOCLUST: Automatic Clustering via Boundary Extraction for Mining Massive Point-Data Sets," in *Proc. of the 5th International Conference on Geocomputation*, 2000.

[17] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications," in *Proc. of the 1998 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA, 1998, pp. 94–105, ACM Press.

[18] W. Wang, J. Yang, and R. Muntz, "STING: A Statistical Information Grid Approach to Spatial Data Mining," in *Proc. of the 23rd VLDB Conference*, 1997, pp. 186–195.

[19] M. Xu and P. Fränti, "A Heuristic k-means Clustering Algorithm by Kernel PCA," in *Proc. of IEEE Int. Conf. on Image Processing (ICIP'04)*, 2004, pp. 3503–3506.

[20] Pasi Fränti and Olli Virmajoki, "Iterative Shrinking Method for Clustering Problems," *Pattern Recognition*, vol. 39, pp. 761 C 775, 2006.

[21] Miin-Shen Yang and Kuo-Lung Wu, "A Similarity-Based Robust Clustering Method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 4, pp. 434–448, 2004.

[22] Cheng-Ru Lin and Ming-Syan Chen, "Combining Partitional and Hierarchical Algorithms for Robust and Efficient Data Clustering with Cohesion Self-Merging," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 2, pp. 145–159, 2005.

[23] R. T. Ng and J. Han, "Efficient and Effective Clustering Methods for Spatial Data Mining," in *Proc. of the 20th International Conference on Very Large Data Bases VLDB 94*, Santiago, Chile, Sept 1994, pp. 144–155.

[24] M. Ankerst, Markus M.Breunig, H.-P. Kriegel, and J. Sander, "OPTICS: Ordering Points to Identify the Clustering Structure," in *Proc. ACM SIGMOD Int. Conf. on Management of Data SIGMOD'99*, 1999, pp. 49–60.

[25] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: Hierarchical Clustering Using Dynamic Modeling," *IEEE Computer*, vol. 32, no. 8.

[26] C. Fraley and A. Raftery, "Mclust: Software for Model-based Cluster Analysis," *Journal of Classification*, vol. 16, pp. 297–306, 1999.

[27] Rui Xu and II. D.Wunsch, "Survey of Clustering Algorithms," *IEEE Transactions on Neural Networks*, vol. 16, pp. 645 – 678, 2005.

[28] H. Pfister, and J. van Baar M. Zwicker, and M.H. GrossSurfels, "Surface Elements as Rendering Primitives Computer Graphics," in *Proc. of SIGGRAPH 2000*, Jul 2000, pp. 335–342.

[29] O. Sourina and Dongquan Liu, "Geometric Approach to Clustering and Querying in Databases and Warehouses," in *Proc. of Cyberworlds 2003*, Dec 2003, pp. 326–333.

[30] Jiawei Han and Micheline Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 1st edition, 2000.

[31] I. Lee and V. Estivill-Castro, "Polygonization of Point Clusters through Cluster Boundary Extraction for Geographical Data Mining," in *Proceedings of the 10th International Symposium on Spatial Data Handling SDH*, Ottawa, Canada, Jul 2002, pp. 27–40.

[32] O. Sourina and D. Liu, "Visual Interactive 3-Dimensional Clustering with Implicit Functions," in *Proc. of IEEE CIS 2004*, 2004, pp. 382–386.

[33] S.Karlin, *A First Course in Stochastic Processes*, NY:London, Academic Press, 1968.

[34] K. Matthes, J. Kerstan, and J. Mecke, *Infinitely Divisible Point Processes*, NY: Akademie-Verlag/John Willey & Sons, 1978.

[35] N.N. Golovanov, A.T. Fomenko, D.P. Il'ytko, and G.V. Nosovskiy, *Computer Geometry*, Moscow: ACADEMIA, 2006.

[36] Pasi Fränti and J.Kivijärvi, "Randomized Local Search Algorithm for the Clustering Problem," *Pattern Analysis and Applications*, vol. 3, pp. 358–369, 2000.