Pattern Recognition 43 (2010) 1346–1360

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A robust dynamic niching genetic algorithm with niche migration for automatic clustering problem

Dong-Xia Chang [a,c,*], Xian-Da Zhang [a], Chang-Wen Zheng [b], Dao-Ming Zhang [a]

[a] Tsinghua National Laboratory for Information Science and Technology, State Key Laboratory on Intelligent Technology and Systems, Department of Automation, Tsinghua University, Beijing 100084, China
[b] National Key Lab of Integrated Information System Technology, Institute of Software, Chinese Academy of Sciences, Beijing 100080, China
[c] Institute of Information Science, Beijing Jiaotong University, Beijing 100044, China

ARTICLE INFO

Article history:
Received 16 January 2009
Received in revised form
1 October 2009
Accepted 4 October 2009

Keywords:
Clustering
Genetic algorithms
Niching method
Niche migration
Remote sensing image

ABSTRACT

In this paper, a genetic clustering algorithm based on dynamic niching with niche migration (DNNM-clustering) is proposed. It is an effective and robust approach to clustering on the basis of a similarity function relating to the approximate density shape estimation. In the new algorithm, a dynamic identification of the niches with niche migration is performed at each generation to automatically evolve the optimal number of clusters as well as the cluster centers of the data set without invoking cluster validity functions. The niches can move slowly under the migration operator which makes the dynamic niching method independent of the radius of the niches. Compared to other existing methods, the proposed clustering method exhibits the following robust characteristics: (1) robust to the initialization, (2) robust to clusters volumes (ability to detect different volumes of clusters), and (3) robust to noise. Moreover, it is free of the radius of the niches and does not need to pre-specify the number of clusters. Several data sets with widely varying characteristics are used to demonstrate its superiority. An application of the DNNM-clustering algorithm in unsupervised classification of the multispectral remote sensing image is also provided.

1. Introduction

Clustering analysis [1,2] is a core problem in data mining with innumerable applications spanning many fields. The primary objective of clustering analysis is to partition a given set of data or objects into groups or clusters so that objects in the same cluster are similar in some sense and differentiate from those of other clusters in the same sense. In the past, many clustering methods were proposed [1–5]. Generally, these algorithms can be broadly divided into two classes [3]: hierarchical and partitional. Hierarchical clustering proceeds successively by either merging smaller clusters into larger ones or by splitting larger clusters. Moreover, the hierarchical clustering algorithms can be subdivided into agglomerative methods [6–8], which proceed by series of fusions of the objects into groups, and divisive methods [9,10], which separate objects successively until all clusters are singleton clusters. Partitional clustering attempts to directly decompose the data set into several disjointed clusters based on some criteria. The most common criterion adopted by partitional clustering is minimizing some measure of dissimilarity in the samples within each cluster and maximizing the dissimilarity of different clusters. Among the partitional clustering methods, the K-means [1] algorithm is one of the more widely used algorithms.

However, most hierarchical and partitional clustering methods have a drawback that the number of clusters need to be specified a priori. For hierarchical clustering, the problem of selecting the number of clusters is equivalent to deciding in which level to cut the tree. Partitional clustering algorithms typically require the number of clusters as user input. Since a priori knowledge is generally not always available, estimation of the number of clusters from the data set under review is required under some circumstances. The classical approach of determining the number of clusters involves the use of some validity measures [11–13]. Within a range of the values of cluster number, the evaluation of a certain validity function of the clustering result is performed for each given cluster number and then an optimal number is chosen for the validity measure. The number of clusters searched by this method depends on the selected clustering algorithm, and the performance of the selected algorithm may rely on the initialization. Some other methods of estimating the number of clusters are based on the idea of cluster removal and merging. In progressive clustering [14,15], the number of clusters is over-specified. After convergence, spurious clusters are eliminated and compatible clusters are merged. The thickest challenge of this

* Corresponding author.
E-mail address: chang_dongxia@hotmail.com (D.-X. Chang).

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.10.020

method lies in how to define the spurious and compatible clusters. Moreover, although overspecification of the cluster number can reduce the initial cluster center effects, there is no way to guarantee that all clusters in the data set will be found. An alternative version of the progressive clustering is to seek one cluster at a time until no more “good” clusters can be found [16,17]. The performances of these techniques are also dependent on the validity functions, which are used to evaluate the individual clusters.

Since the global optimum of the validity function would correspond to the most “valid” solutions with respect to the functions, stochastic clustering algorithms based on genetic algorithms (GAs) [18–21] have been reported to be able to optimize the validity functions to determine the number of clusters and partitioning of the data set simultaneously. In these GA-based algorithms, the validity functions are regarded as the fitness function to evaluate the fitness of the individual, which guides the evolution to search for the “valid” solution. In recent years, several clustering algorithms based on simple GA or its variants have been developed [22–36]. These algorithms fall into two broad categories based on the representations for the clustering solutions. The first category uses a straightforward encoding, in which the chromosome is encoded as a string of length n, where n is the number of data points and the element of the chromosome denotes the cluster number that data point belongs to, such as used in Refs. [22–24]. The desired number of clusters should be specified in advance. Moreover, this approach does not reduce the size of the search space and searching the optimal solution can be onerous when the data points proliferate. It is for this reason that some researchers opt to use a relatively indirect approach where the chromosome encodes the centers of the clusters, and each datum is subsequently assigned to the closest cluster center [25–36]. This kind of algorithms can be subdivided into fixed-length encoding algorithms [25–31], which use a fixed-length string to describe the cluster centers and the number of clusters is specified a prior, and variable-length encoding algorithms [32–36], which use a variable-length string to describe the cluster centers and the number of clusters is automatically evolved. Although the number of cluster centers need not to be given in advance in the variable-length encoding algorithms, the initial values of the cluster centers are constrained to be in the range from 2 to $k_{\max}$, and $k_{\max}$ is the upper bound of the number of clusters and should be specified beforehand. Because the traditional GAs are suitable for locating the optimum of unimodal functions as they converge to a single solution of the search space, all these GA-based clustering algorithms consider the clustering problem as a unimodal problem. Each chromosome is described by a sequence of the cluster centers. When every cluster center is contained in the chromosome, then the fitness function reaches its global optimum. However, a simpler way is to consider the clustering problem as a multimodal problem and each cluster center corresponds to a local optimum of the fitness function. In this circumstance, each chromosome represents a cluster center and all the local optima of the fitness function should be found. Algorithms that allow the formation and the maintenance of different solutions can be used to solve this multimodal problem.

In order to preserve the population diversity, which prevents GAs being trapped by a single optimum, niching methods have been developed. The basic idea of the niching methods is based upon the natural ecosystems, which maintain population diversity and permit the GA to investigate many optima in parallel. In nature, an ecosystem is typically composed of different physical niches that exhibit different features and allow both the formation and the maintenance of different types of life (species). It is assumed that a species is made up of individuals with similar biological features capable of interbreeding among themselves, but unable to breed with individuals of other species [37]. By analogy, in artificial systems, a niche corresponds to a local optimum of the fitness function, and the individuals in one niche exhibit similar feature in terms of a given metric. Among niching methods, fitness sharing (FS) and implicit fitness sharing are the best known and the most widely used methods [38–42]. In the former, the fitness represents the resource for which the individuals belonging to the same niche compete [38], while in the latter [40,41], the sharing effects are achieved by means of a sample-and-match procedure.

In FS, the fitness of an individual is reduced if there are many other individuals near it and so the GA is forced to maintain diversity in the population [38]. This method should define a similarity metric on the search space and an appropriate niche radius, representing the maximal distance among individuals to be considered similar and therefore belonging to the same niche. In most circumstance, it is difficult to give an effective value for the niche radius without any a priori knowledge. Deb and Goldberg proposed a criterion for estimating the niche radius given the heights of the peaks and their distances [39]. Since in most of the real applications there is very little prior knowledge about the fitness landscape, it is difficult to estimate the niche radius. In the implicit fitness sharing [40], sharing is accomplished by inducing competition for limited and explicit resources, and there is no specific limitation on the distance between peaks. This method avoids the difficult of appropriately choosing the niche radius and can be used to deal with problems in which the peaks are not equally spaced [40–42]. So, one of the most important limitations of FS seems to be removed. In fact, some other parameters, such as the size of the sample of individuals that compete, the number of competition cycles and the definition of a matching procedure, need to be set. In order to improve the performance of the FS methods, several dynamic niching methods were proposed [46,47]. These methods are based upon a dynamic, explicit identification of species discovered at each generation and the FS mechanism is restricted to individuals belonging to the same species. However, the performance of these algorithms is dependent on the niche radius. When wrong value for the niche radius is selected, the algorithm did not find all the niches perfectly. In Ref. [48], a species conserving genetic algorithm (SCGA) was proposed which does not consider any sharing mechanism. Once a new species is discovered, its fittest individual is retained in the next generations until a fitter individual for that species is generated. Therefore, each species populating a region of the fitness landscape survives during the entire evolution, whether or not it corresponds to an actual niche. Moreover, the performance of this algorithm is also depends on the niche radius. In addition, all these algorithms are not robust to noise. When the data set contains noise points, the performances of these algorithms are poor.

In this paper, a new clustering algorithm based on dynamic niching with niche migration (DNNM-clustering) is proposed which is robust to noise and cluster volumes. Within the DNNM-clustering, a dynamic niching with niche migration is developed to preserve the diversity of the population. A simpler representation is adopted, whereby each individual represents a single cluster center. All the niches presented in the population at each generation are automatically and explicitly identified. Then, the application of FS is limited to individuals belonging to the same niche. In order to overcome the dependence of the niche radius, a niche migration is considered. This makes the algorithm work properly and independent of the niche radius even if some noise points exist and the peaks are not equally spaced and have different cluster volumes.

The rest of this paper is organized as follows. Section 2 provides the fitness function of the clustering problem used in the algorithm. The dynamic niching with niche migration is presented in Section 3. Section 4 describes the evolutionary clustering algorithm. Experimental results on several artificial data sets and remote sensing image are given in Section 5. Experimental results demonstrate the effectiveness of the DNNM-clustering algorithm. Finally, conclusions are drawn in Section 6.

2. The fitness function

Let $\mathbf{X}=\{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_n\}$ be a finite subset of a N-dimensional vector space, K be the number of clusters and $S(\mathbf{x}_j,\mathbf{c}_i)$ denote the similarity measure between $\mathbf{x}_j$ and the i th cluster center $\mathbf{c}_i$. Our clustering goal is to find $\mathbf{c}_i$ to maximize the total similarity measure $J(\mathbf{c})$ with

$$J(\mathbf{c})=\sum_{i=1}^{K}\sum_{j=1}^{n}\left(\exp\left(-\frac{\|\mathbf{x}_j-\mathbf{c}_i\|^2}{\beta}\right)\right)^{\gamma}, \tag{1}$$

where $\mathbf{c}=(\mathbf{c}_1,\mathbf{c}_2,\ldots,\mathbf{c}_K)$ and β can be defined by

$$\beta=\frac{\sum_{j=1}^{n}\|\mathbf{x}_j-\overline{\mathbf{x}}\|^2}{n}\quad\text{where } \overline{\mathbf{x}}=\frac{\sum_{j=1}^{n}\mathbf{x}_j}{n}. \tag{2}$$

According to the analysis of γ in Ref. [49], we know that γ can determine the location of peaks in the objective function $J_s(\mathbf{c})$. And the value of β is no longer sensitive to the peak. Let $\tilde{J}_s(\mathbf{x}_k)$ be the total similarity of the data point $\mathbf{x}_k$ to all data points with

$$\tilde{J}_s(\mathbf{x}_k)=\sum_{j=1}^{n}\left(\exp-\frac{\|\mathbf{x}_j-\mathbf{x}_k\|^2}{\beta}\right)^{\gamma},\quad k=1,2,\ldots,n. \tag{3}$$

This function can be seen closely related to the density shape of the data points in the neighborhood of $\mathbf{x}_k$. A large value for $\tilde{J}_s(\mathbf{x}_k)$ means that the data point $\mathbf{x}_k$ is close to some cluster centers and has many data points around it. A good estimation of γ can give a good estimation of the peak of $\tilde{J}_s(\mathbf{x}_k)$. Here, we use the data set shown in Fig. 1(a) to see the influence of γ on (3) and more detailed explanation can be found in Ref. [49]. Note that the "$*$" in Fig. 1 means the value of $\tilde{J}_s(\mathbf{x}_k)$ with respect to the data point $\mathbf{x}_k$, $k=1,2,\ldots,n$. According to Fig. 1(b), only two clusters will be found when $\gamma=1$ and all the five peaks will be separated when γ increases to 5 and 10 as shown in Figs. 1(c) and (d).

Here, the CCA algorithm [49] is used to estimate γ. For convenience, it is presented in the following:

1. Set $m=1$ and $\varepsilon_1=0.97$.
2. Calculate the correlation of the value of $\tilde{J}_s(\mathbf{x}_k)_{\gamma_m}$ and $\tilde{J}_s(\mathbf{x}_k)_{\gamma_{(m+1)}}$.
3. If the correlation is greater than or equal to the specified ε_1, then choose γ_m to be the estimate of γ, else $m=m+1$ and goto step 2.

After getting the estimation of γ, the function $\tilde{J}_s(\mathbf{x}_k)$ becomes a multimodal function which the number of peaks is equal to the number of clusters. Therefore, the clustering problem can be transformed into a multimodal problem through this objective function. In the following, our new algorithm will be used to estimate all the local optima of $\tilde{J}_s(\mathbf{x}_k)$. The number of the local optima is the same to the number of clusters, and the local optima are the cluster centers.

3. The dynamic niching with niche migration

Niching methods have been developed to minimize the effect of genetic drift resulting from the selection operator in the traditional GA in order to allow the parallel investigation of many solutions in the population. We begin this section by providing a brief overview of the fitness sharing method, which is a representative niche method. In order to overcome the drawbacks of the FS, a dynamic niching with niche migration is proposed.

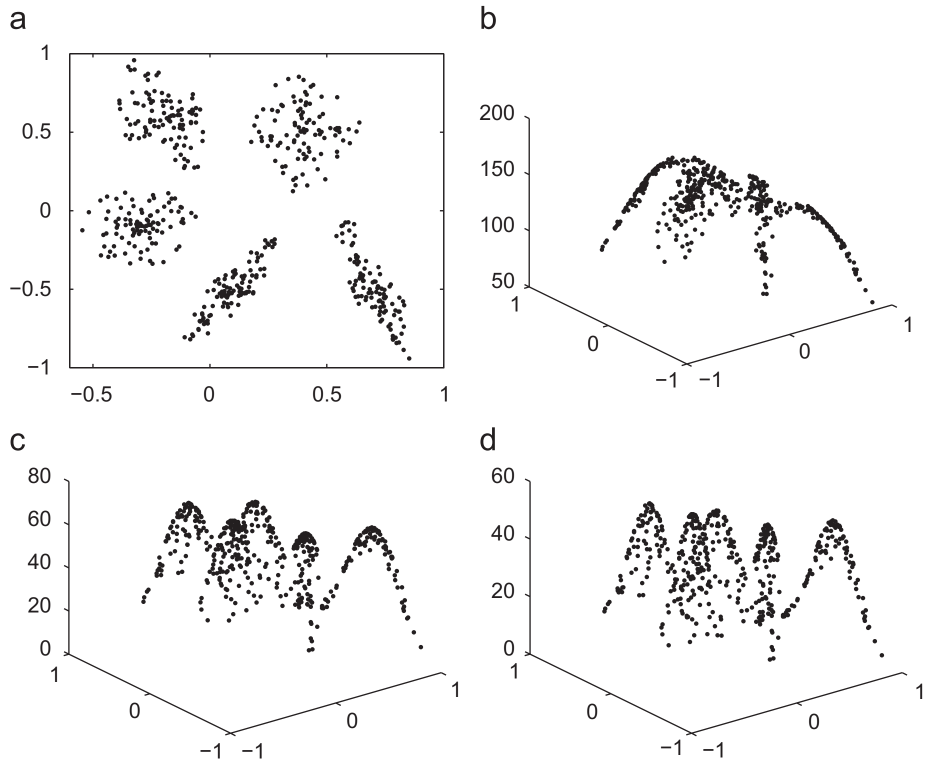

Fig. 1. (a) Five-clusters data set. (b), (c) and (d) are plots of (3) (the approximate density shapes) with $\gamma=1$, 5 and 10, respectively.

3.1. Fitness sharing

Fitness sharing modifies the search landscape by reducing the fitness of an individual in densely populated regions. It works by derating the fitness of each individual by an amount related to the number of similar individuals in the population. Specifically, the shared fitness $f_{sh,t}(i)$ of an individual i at generation t is given by

$$f_{sh,t}(i) = \frac{f_t(i)}{m_t(i)}, \tag{4}$$

where $f_t(i)$ is the raw fitness of the individual, and $m_t(i)$ is the niche count which depends on the number and the relative positions of the individuals within the P population. The niche count is calculated as

$$m_t(i) = \sum_{j=1}^{P} sh(d_{ij}), \tag{5}$$

where P is the population size, d_{ij} is the distance between the individual i and j, and $sh(d_{ij})$ is the sharing function which measures the similarity between two individuals. The most commonly used form of sh is

$$sh(d_{ij}) = \begin{cases} 1\text{-}\left(\dfrac{d_{ij}}{\sigma_{sh}}\right)^{\alpha_{sh}} & \text{if } d_{ij} < \sigma_{sh}, \\ 0 & \text{otherwise}, \end{cases} \tag{6}$$

where σ_{sh} is the niche radius and α_{sh} is a constant parameter which regulates the shape of the sharing function. The value of α_{sh} is commonly set to 1, yielding to a triangular form for the sharing function [50]. The distance d_{ij} between individual i and j is implemented by defining a metric on either the genotypic or the phenotypic space.

It has been proved that when the number of individuals within the population is large enough and the niche radius is properly set, FS provides as many niches in the population as the number of peaks in the fitness landscape [51,52]. But, there are several problems with the fitness sharing approach. In order to ensure that subpopulations are steadily formed and maintained, only the individuals belonging to the same niche should share the resources of the niche. This assumption is not generally true for the FS methods [53], because each individual in the population shares its fitness with all the individuals located at a distance smaller than the niche radius, no matter for the actual peak, i.e., for the niche, to which they belong. As a consequence, individuals belonging to different peaks may share their fitness, while they should not. Moreover, the radius of the niches should be specified and this requires a priori knowledge of how far apart the optima are. However, no information about the search space and the distance between the optima is available in the practical optimization problems. When the niche radius is wrong, the algorithm cannot find all the niches. In order to overcome these drawbacks, a dynamic niching with niche migration is proposed.

3.2. Dynamic niching with niche migration

In this section, we propose a dynamic niching method which is independent of the niche radius. From Refs. [38–48], we can see that the radius of the niches plays a crucial role in the identification of the niches. If the niche radius chosen is too small, many niches may be found in every generation. On the other hand, a large value of the radius will make many solutions indistinguishable. This means that too few niches will be conserved. If the radius is so large that only one niche master is found, the algorithm will degenerate into a simple genetic algorithm and only find one optimum with the largest fitness value.

In our algorithm, two strategies, the initialization of the niche radius and the migration of the niche candidates, are used to achieve the goal of independent of the initial niche radius. After the initialization of the niche radius, the dynamic niching method attempts to find the niches according to this radius at each generation. And the niche candidates identified will change their location under the migration operator. The skeleton of dynamic niching algorithm is presented in Table 1 and the initialization of the niche radius in Table 2.

When a niche radius inputs, a preprocessing of the input by the algorithm shown in Table 2 is conducted to ensure the niche candidates will be sufficiently diverse in the first generation. Here η is a constant and $\eta \in (1,2)$.

In phase II of the algorithm shown in Table 1, a migration operator is introduced. In reality, if a city is prosperous and its citizen lead comfortable lives, then it will attract the people living nearby to migrate to it. For the clustering problem, the effect of this migration operation is to change the relative position of the niches in the entire population. Based on this analogy between our society and a clustering problem, a migration operator is introduced and explained in the following. First, several definitions used in the migration operator are given.

Definition 1 (*Niche attraction*). Suppose $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m$ are m individual in a niche, and the fitness values are $f_1, f_2, \ldots, f_m$, respectively. The attraction one niche acts on another niche is defined as

$$F = (f_1 + f_2 + \cdots + f_m)/m. \tag{7}$$

Definition 2 (*Migration principle*). Let $\mathbf{N}_i$ and $\mathbf{N}_j$ be two niches, and the niche attraction of the two niches are F_i and F_j,

Table 1
The dynamic niching algorithm with niche migration.

```
Input: the population $Pop_t$ at generation $t$, the population size $P$, the niches
  radius $\sigma$( this value is obtained by the algorithm shown in Table 2)

Sort the current population according to the raw fitness
$v(t) = 0$ (the number of actual niches at generation $t$)
$u(t) = 0$ (the number of niche master candidates)
$NC = \emptyset$ (the niche master candidate set)
$DN = \emptyset$ (the dynamic niche set)

Phase I: The niche master candidates identification.

For $i = 1$ to $P$ do
  if the $i$ th individual is not marked then
    $u(t) = u(t)+1$
    $N(u(t)) = 1$ (the number of individuals in the $u(t)$th niche candidate set)
    For $j = i+1$ to $P$ do
      if ($d(i,j) < \sigma$) and ($j$ th individual is not marked)
        insert $j$ th individual into the niche master candidate set $NC$,
        $N(u(t)) = N(u(t))+1$
      end if
    end for
    If ($N(u(t)) > 1$) then
    $v(t) = v(t)+1$
    mark $i$ th individual as the niche master of the $v(t)$th niche
    insert the pair ($i$ th individual, $N(u(t))$) in $DN$
    end if
  end if
End For

Phase II: The migration of the niches.

Calculate the distance between the niche master candidates
For $l = 1$ to $u(t)$
  If $j$ th niche is the nearest neighbor to $l$ th niche, then determine
  the communication edge between these two niches according to Theorem 1.
  If there exits communication between the two niches and $F_l < F_j$, then
  niche $l$ migrates toward niche $j$, otherwise niche $l$ keep station.
End For
```

Table 2
The initialization of the niche radius.

```
Input: $Pop_1$, the population at generation 1 $\sigma_{init}$, an input niche radius

The Phase I described in Table 1 is used to determine the number of master
   candidate $u(1)$ and the number of actual niche masters $v(1)$.
If ($v(1) \le 2$ and $u(1) > P/3$) then
   $\sigma = \eta\sigma_{init}$
Else if ($v(1) = 1$ or ($u(1)$-$v(1)$) $< 1$) then
   $\sigma = \sigma_{init}/\eta$
End
```

respectively. If $F_i > F_j$, then $\mathbf{N}_j$ will migrate to $\mathbf{N}_i$. Otherwise, $\mathbf{N}_i$ will migrate to $\mathbf{N}_j$.

Definition 3 (*Distance of niches*). Let $\mathbf{M}_i$ and $\mathbf{M}_j$ be two masters of two niches $\mathbf{N}_i$ and $\mathbf{N}_j$, then the distance of these two niches is defined as

$$d_N(\mathbf{N}_i, \mathbf{N}_j) = d(\mathbf{M}_i, \mathbf{M}_j) = \|\mathbf{M}_i\text{-}\mathbf{M}_j\|^2. \tag{8}$$

For the niche candidate sets identified in phase I of the algorithm shown in Table 1, the nearest neighbor of each niche should be found. Here, a $u(t) \times u(t)$ matrix $\mathbf{D}$ is used to indicate the nearest neighbor of the niche candidates,

$$D_{ij} = \begin{cases} 1 & \text{if } d_N(\mathbf{N}_i, \mathbf{N}_j) = \min\limits_{k \ne j, k = 1,2,\ldots,u(t)} d_N(\mathbf{N}_k, \mathbf{N}_j), \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

where $d_N(\mathbf{N}_i, \mathbf{N}_j)$ is the distance between niche i and niche j. If $D_{ij} = 1$, then given the ability for the two niches to communicate. And the communication topology is specified by a matrix $\mathbf{S}$, where S_{ij} is the number sent from niche i to niche j. Here, $S_{ij} = 1$ means exist communication edge between these two niches, and $S_{ij} = 0$ indicates no communication edge between them. The value of S_{ij} is determined by Theorem 1.

Theorem 1. *Let $\mathbf{N}_i$ and $\mathbf{N}_j$ be two niches, and $\mathbf{M}_i$ and $\mathbf{M}_j$ be the niche masters of these two niches with fitness value f_i and f_j, a line that intersects the two niche masters can be written as*

$$\mathbf{x} = \mathbf{M}_i + k(\mathbf{M}_j\text{-}\mathbf{M}_i), \quad k \in [0, 1]. \tag{10}$$

Then, a series of points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l$ is generated along this line and the fitness of those points is calculated by Eq. (4). If $\exists\ m \in [1, l]$ satisfies

$$f(\mathbf{x}_m) < \min(f_i, f_j), \tag{11}$$

then a valley lies between $\mathbf{N}_i$ and $\mathbf{N}_j$, and at the same time there is no communication between them and $S_{ij} = 0$. Otherwise, the communication exist and $S_{ij} = 1$.

The concept of Theorem 1 is simple. Given two end points in Euclidean space, then choose a number of points along the line in between the two end points and calculate the fitness of those points. In this way, it is possible to determine if a valley lies between the two end points (i.e., $\mathbf{M}_i$ and $\mathbf{M}_j$). If a valley lies between the two niches (i.e., $\exists\ m \in [1, l]$, satisfies the inequality (11)), then the two niches can be seen as two different species and they should not communicate with each other. If no point has lower fitness than either of the endpoints, then no valley lies between the two niches, and they can communicate with each other. The implementation of Theorem 1 is terminated on the first point discovered that had lower fitness than either of the two end points. It is quite obvious how powerful this theorem is and how the decisions of whether the communication of niches exist using it.

In fact, the inequality (11) used in Theorem 1 is not robust to noise. For example, given two end points M_1 and M_4, and two

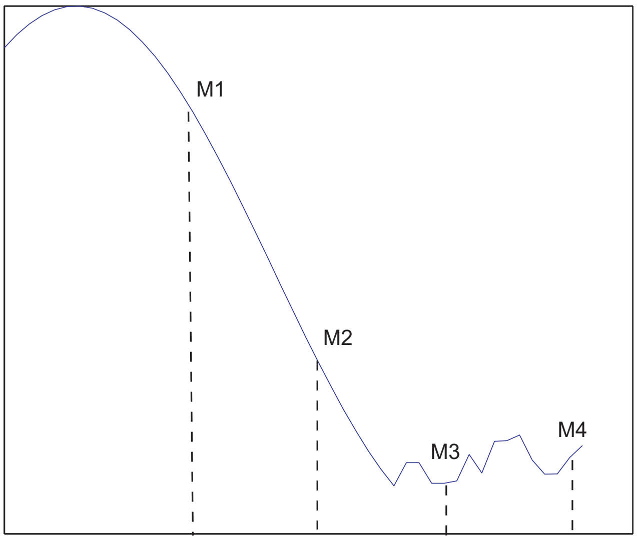

Fig. 2. An example of influence of noise.

samples in between (see Fig. 2). From Fig. 2, we can see $f(M_3) < F(M_4)$. Then according to Theorem 1, there is no communication between M_1 and M_4. However, the slight variance between the function values of M_3 and M_4 can be seen as a result of the noise. In order to overcome the influence of noise, we define a noise tolerance factor ρ $(0.8 \le \rho \le 1)$ and the inequality (11) modified as

$$f(\mathbf{x}_m) < \rho \min(f_i, f_j). \tag{12}$$

Then inequality (12) will be used in the determination of communication between two points.

After the determination of the communication, the magnitude of migration is defined.

Definition 4 (*Migration magnitude*). Let $\mathbf{N}_i$ and $\mathbf{N}_j$ be two niches identified in one generation, and niche attraction of the two niches are F_i and F_j, respectively. The distance of these two niches is $d_N(\mathbf{N}_i, \mathbf{N}_j)$ and the niche masters be $\mathbf{M}_i$ and $\mathbf{M}_j$. If $F_i > F_j$, then the migration magnitude of the individual in $\mathbf{N}_j$ is defined as

$$\Delta l = \delta \frac{F_i}{d_N^2(\mathbf{N}_i, \mathbf{N}_j)} \vec{\mathbf{r}}, \tag{13}$$

where $\vec{\mathbf{r}}$ is the direction vector from the individual in $\mathbf{N}_j$ to $\mathbf{M}_i$, and δ is a small constant greater than 0, called the migrating rate. We imagine here that the niches are migrating with negligible magnitude.

After the dynamic identification of the niche masters of the population Pop_t at generation t, the species belonging to the niche master candidate can be defined as a subset $S_t^i \ne \emptyset$ of individuals in the population Pop_t which have a distance from the master candidate less than the niche radius and do not belong to other species. If the number of the individuals in S_t^i is larger than 1, then this subset is assumed as the actual niche, otherwise, the single individual in the subset is considered as an isolated individual and all the isolated individuals form the subset S_t^*. Then, the population Pop_t at the generation is partitioned into a number $v(t)$ of species, say $S_t^1, S_t^2, \ldots, S_t^{v(t)}$, and a number of isolated individuals

$$Pop_t = \left(\bigcup_{i \in \{1,2,\ldots,v(t)\}} S_t^i \right) \cup S_t^*. \tag{14}$$

After the identification of niches, the sharing fitness of each individual was calculated according to (4). Here, the shared fitness value for an individual within a dynamic niche (identified by the dynamic niching algorithm) is its raw fitness value divided by the niche count. Otherwise, the individual belongs to the isolated category, and its fitness is not modified. The niche count in (5) is modified as

$$m_t(i) = \sum_{p_j \in S_t^i} sh(d_{ij}), \tag{15}$$

and $sh(d_{ij})$ is computed according to (6). Here, only the individuals belonging to the same niche share their fitness and the fitness of the isolated individuals is not modified.

After all the niches have been found, the new population is constructed by applying the usual genetic operators. Since some niche masters may not survive during the evolution, the species elitist strategy is implemented to enable the niche masters to survive. Here, only the actual masters are conserved.

4. The DNNM-clustering algorithm

In this section, we propose the dynamic niching with niche migration clustering algorithm (DNNM-clustering), which can be used to optimize the objective function to automatically evolve the proper number of clusters as well as appropriate partition of the data set.

4.1. Chromosome representation and initialization

For any GA, a chromosome representation is needed to describe each chromosome in the population. The representation method determines how the problem is structured in the algorithm and the genetic operators that are used. Each chromosome is made up of a sequence of genes from certain alphabet. An alphabet can consist of binary digits (0 and 1), floating-point numbers, integers, symbols (i.e., A, B, C, D), etc. In early GAs, the binary digit was used. It has been shown that more natural representations can get more efficient and better solutions. Michalewicz [20] has performed extensive experiments to compare real-valued and binary GAs and shown that the real-valued GA is more efficient in terms of CPU time. Therefore, in this paper, real-valued numbers are used to describe the chromosome.

Here the chromosome is encoded the center of the cluster. Each chromosome is described by a sequence of N real-valued numbers where N is the dimension of the feature space. That is to say, the chromosome of the algorithm is written as

$$\mathbf{c} = [c_1, c_2, \ldots, c_N]. \tag{16}$$

An initial population of size P for DNNM-clustering algorithm is usually chosen at random. In this paper, several P randomly chosen data points from the data set with the exception that no two may be the same are used to initialize the P chromosome.

4.2. Fitness function

The fitness function is used to define a fitness value to each candidate solution. Here, the fitness function of the chromosome, f, is defined as

$$f(\mathbf{c}) = \tilde{J}_s(\mathbf{c}) = \sum_{j=1}^{n} \left(\exp\text{-}\frac{\|\mathbf{x}_j\text{-}\mathbf{c}\|^2}{\beta} \right)^{\gamma}, \tag{17}$$

where $\mathbf{x}_j$, $j = 1, 2, \ldots, n$ are all data points in the data set to be clustered.

4.3. Evolutionary operators

Any combination of standard selection, crossover and mutate operators can be employed by our algorithm. Here intermediate recombination and uniform neighborhood mutation are used.

For two randomly chosen parents $\mathbf{c}_1$ and $\mathbf{c}_2$, the offspring $\mathbf{c}$ of the intermediate recombination crossover (with probability p_c) is

$$\mathbf{c} = \mathbf{c}_1 + r(\mathbf{c}_1\text{-}\mathbf{c}_2), \tag{18}$$

where r is a uniformly distributed random number over [0, 1].

Each chromosome undergoes mutation with a probability p_m. If the minimum and maximum values of the data set along the q th dimension are $c^q_{\min}$ and $c^q_{\max}$, respectively. If the position to be mutated is the q th dimension of a cluster center with value c^q, then after uniform neighborhood mutation the value becomes

$$c_0^q = c^q + r_m R(c^q_{\max}\text{-}c^q_{\min}), \tag{19}$$

where R is a uniformly distributed random number over [-1, 1] and $r_m \in (0, 1)$.

4.4. Description of the algorithm

In our DNNM-clustering algorithm, a chromosome represents one cluster center and is evaluated by using the fitness function described in Section 4.2. The niches are identified by the dynamic niching algorithm at each generation and the fitness sharing is computed in every niches. The evolutionary operators, selected on the basis of probability distribution, can be crossover or mutation, where the former transforms two individuals (parents) into two offspring by combining parts from each parent, and the latter develops on a single individual and creates an offspring by mutating that individual. The elitist strategy [21] is implemented by replacing the worst chromosome of the current population with the niche masters found at each generation. The process

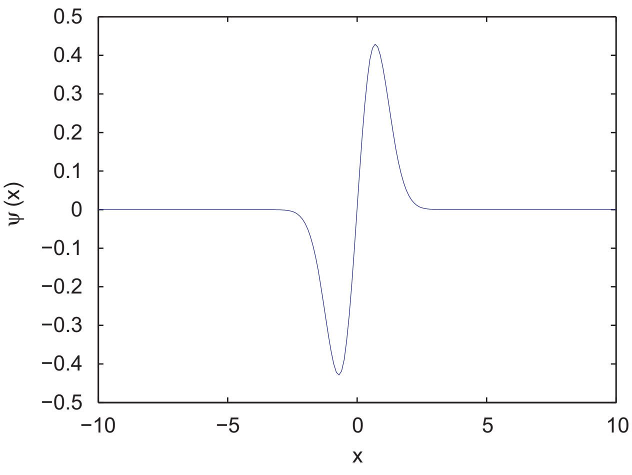

Fig. 3. The ψ function of our estimation.

Table 3
The values of CCA.

Data set	5 and 10	10 and 15	15 and 20	Selected γ
Data 1	0.9000	0.9972	0.9993	10
Data 2	0.9824	0.9925	0.9954	5
Data 3	0.8277	0.9990	0.9996	10
Data 4	0.9779	0.9927	0.9962	5
Data 5	0.9759	0.9948	0.9976	5

terminates after some number of generations, fixed either by the user or determined dynamically by the program itself, and the niche masters obtained is taken to be the solution.

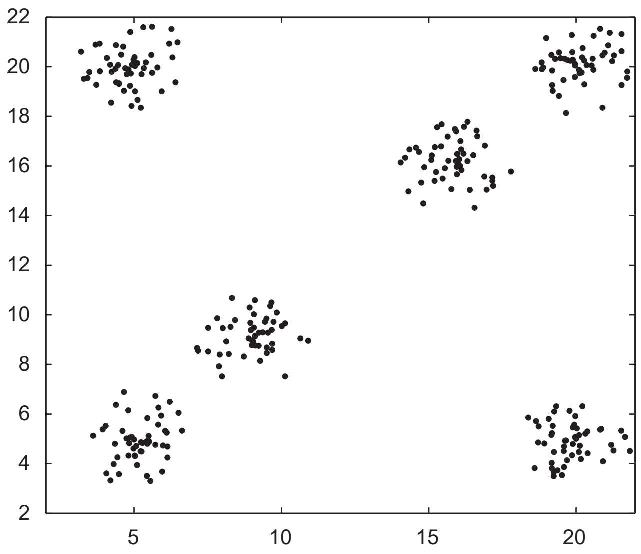

Fig. 4. Data 1.

The DNNM-clustering algorithm is described as follows:

1. Initialize a group of cluster centers with size of P.
2. Evaluate each chromosome.
3. Apply the dynamic niching algorithm and apply the fitness sharing among the individuals belonging to the same niche.
4. If the termination condition is not reached, go to Step 5. Otherwise, select the niche master from the population as the final cluster centers.
5. Apply the selection operator.
6. Apply crossover operator to the selected individuals based on the crossover probability.
7. Apply mutation operator to the selected individuals based on the mutation probability.
8. Evaluate the newly generated candidates.
9. Apply the elitist strategy.
10. Go back to Step 3.

5. The robust property to noise

A good clustering method should be robust that it can determine good clusters for noisy data set. Several different classes of robust methods (such as the M, R, L estimators) exist [43–45]. Here, the influence function [45] is used to show that our method is robust to noise. Let $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be an observed data

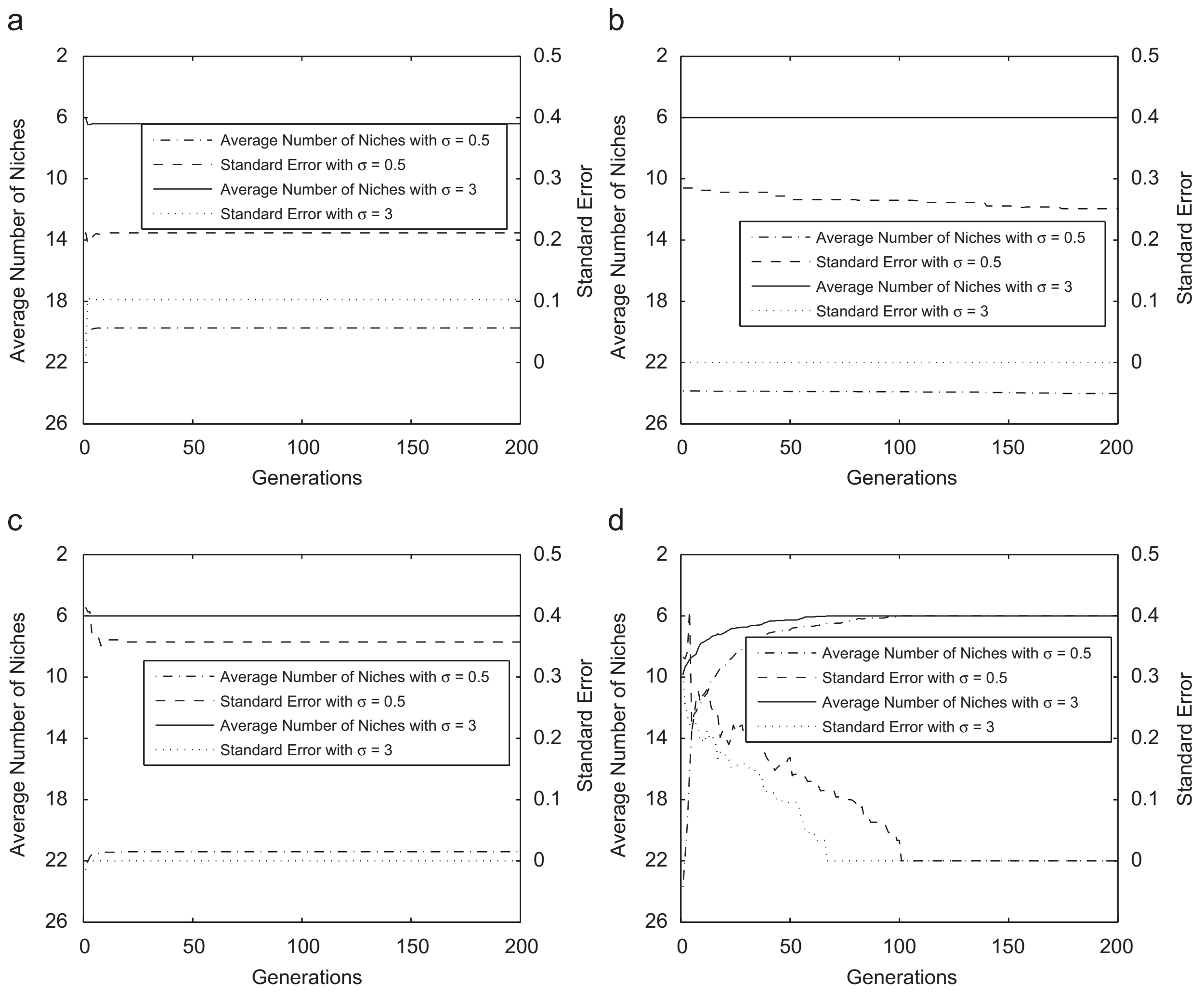

Fig. 5. The average number of clusters and its standard error by (a) DNS, (b) SCGA, (c) DFS, (d) DNNM-clustering. In all the experiments $P = 100$.

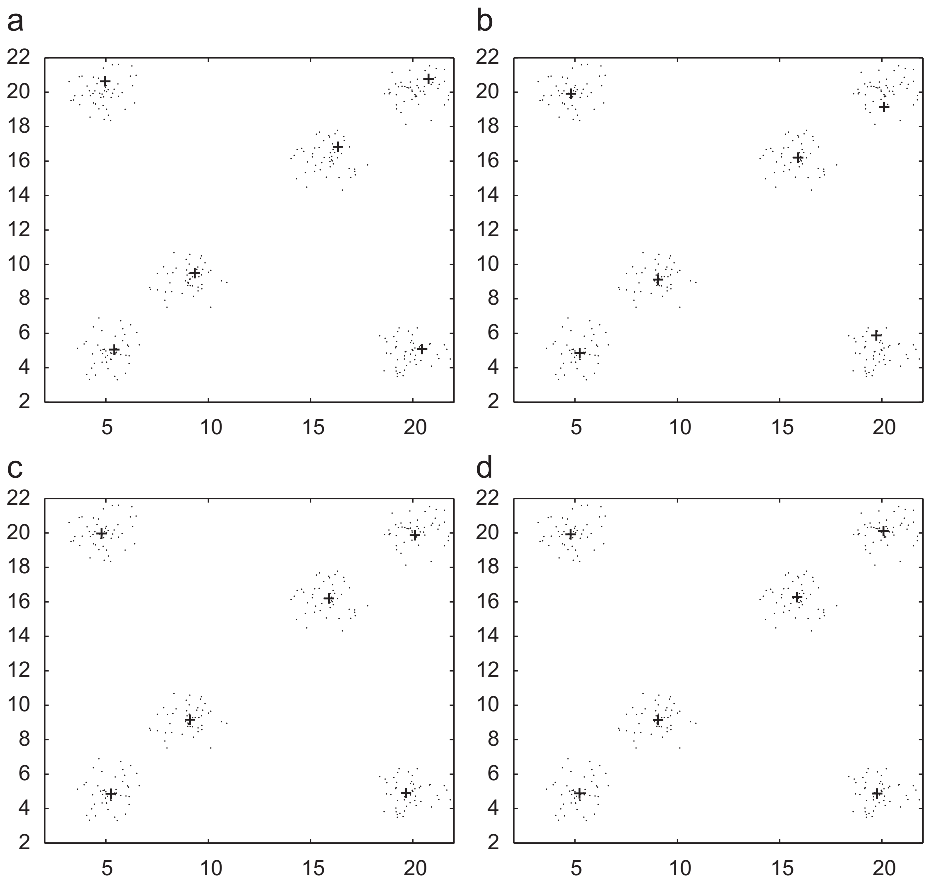

Fig. 6. The cluster centers obtained by using (a) DNS, (b) SCGA, (c) DFS, (d) DNNM-clustering. In all the experiments $P=100$.

set of real numbers and θ is an unknown parameter to be estimated. We consider $\mathbf{x}_i$ and θ to be scalars and an M estimator [45] is generated by minimizing the form

$$\hat{\theta}=\underset{\theta}{\operatorname{argmin}}\sum_{i=1}^{n}\rho(x_i\text{-}\theta), \tag{20}$$

where ρ is a function that can measure the loss of x_i and θ. If we let $\psi(x_i\text{-}\theta)$ be the derivative of $\rho(x_i\text{-}\theta)$, i.e., $\psi(x_i\text{-}\theta)=\partial\rho(x_i\text{-}\theta)/\partial\theta$, then the M estimator is obtained by

$$\sum_{i=1}^{n}\psi(x_i\text{-}\theta)=0. \tag{21}$$

The influence function (IF) can help us to assess the relative influence of individual observations toward the estimation value. It has been shown that influence function of an M estimator is proportional to its ψ function [45]. For a location M estimator, we have the influence function

$$\mathrm{IF}_{\hat{\theta}}(x;F,\theta)=\frac{\psi(x\text{-}\theta)}{\mathrm{E}\psi'(x\text{-}\theta)}, \tag{22}$$

where F is the distribution of x. If the influence function of an estimator is unbounded, a noise might case trouble. In the following, we will show the boundedness of the IF function. Let

$$\rho(x_i\text{-}c)=1\text{-}\exp(\text{-}\beta^{-1}(x_i\text{-}c)^2)^{\gamma}. \tag{23}$$

Minimize (20) with (23) is equivalent to minimize

$$n\text{-}\sum_{i=1}^{n}\exp(\text{-}\beta^{-1}(x_i\text{-}c)^2)^{\gamma} \tag{24}$$

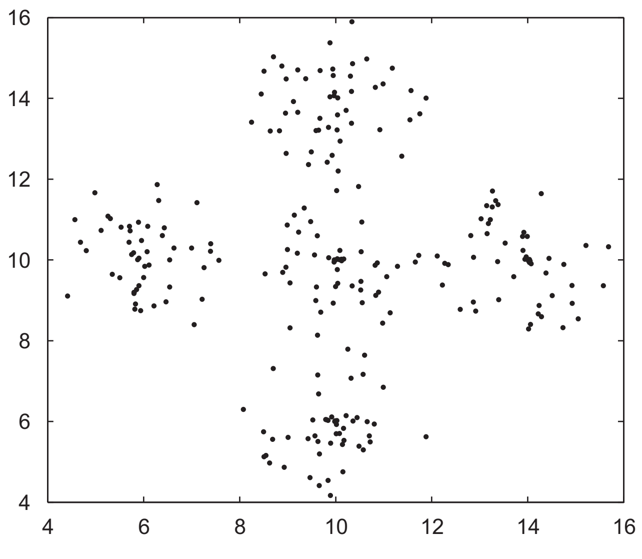

Fig. 7. Data 2.

and this also equivalent to maximize

$$\sum_{i=1}^{n}\exp(\text{-}\beta^{-1}(x_i\text{-}c)^2)^{\gamma} \tag{25}$$

which is the objective function of our genetic algorithm with one cluster. Our estimation of the cluster center c (by (1)) is equivalent to an M estimator with the $\sum_{i=1}^{n}\exp(\text{-}\beta^{-1}(x_i\text{-}c)^2)^{\gamma}$ in (20) replaced

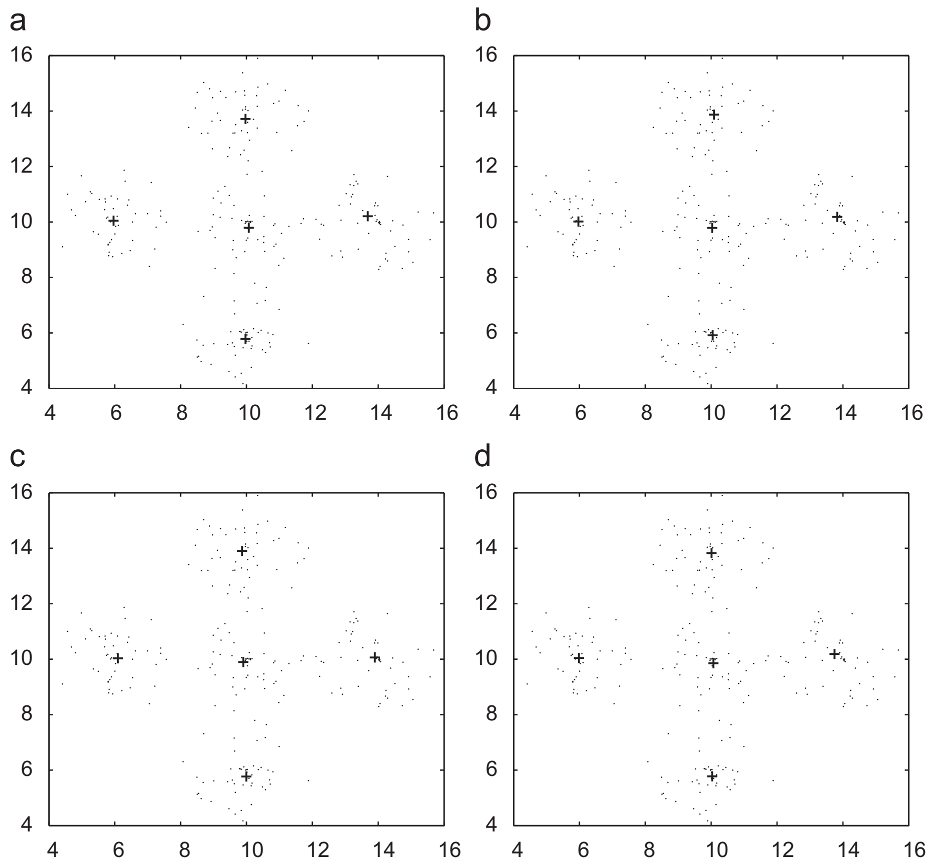

Fig. 8. The cluster centers obtained by using (a) DNS, (b) SCGA, (c) DFS, (d) DNNM-clustering. In all the experiments $P=100$.

by a dissimilarity measure (23) and the ψ function of our estimation is

$$\psi(x_i\text{-}c)=\frac{-2\beta^{-1}(x_i\text{-}c)}{\exp(\beta^{-1}(x_i\text{-}c)^2)}. \tag{26}$$

Since the influence function $\mathrm{IF}_{\hat{\theta}}(x;F,\theta)$ is proportion to $\psi(x_i\text{-}c)$ according to (22), we need only to analyze the term $\psi(x_i\text{-}c)$. By applying the L'Hospital's rule, we have

$$\lim_{x_i\to+\infty}\psi(x_i\text{-}c)=\lim_{x_i\to-\infty}\psi(x_i\text{-}c)=0. \tag{27}$$

Thus, we have $\mathrm{IF}(x_i;F,c)=0$ when x_i tends to positive or negative infinity. From Eq. (27), $\exists\ M>0$, when $|x_i|\geq M$, we have $|\psi(x_i\text{-}c)|<1$. $|\psi(x_i\text{-}c)|<1$ is continuous on interval $[-M,M]$, thus, $\exists\ K>0$, for $\forall\ x\in[-M,M]$, it holds that $|\psi(x_i\text{-}c)|\leq K$. Let $G=\max\{1,K\}$, then for all $x\in(-\infty,+\infty)$, we have $|\psi(x_i\text{-}c)|\leq G$. According to above, the function $\psi(x_i\text{-}c)$ with (23) is bounded and continuous, as shown in Fig. 3. Therefore, the influence of an extremely large or small x_i on our estimator is very small according to (27). In fact, (27) also shows that an extremely large or small x_i can be thought of a new observation that have no influence (i.e., $\mathrm{IF}(x;F,\theta)=0$) on our estimator.

From the analysis above, we can deduce that our estimator has a bounded and continuous influence function. Hence, it is robust to noise from the robust statistical point view.

6. Experiments results

In order to validate the proposed algorithm, we have performed a set of experiments with several data sets with widely varying characteristics and multispectral remote sensing image. The experiments show that DNNM-clustering has high performance and flexibility.

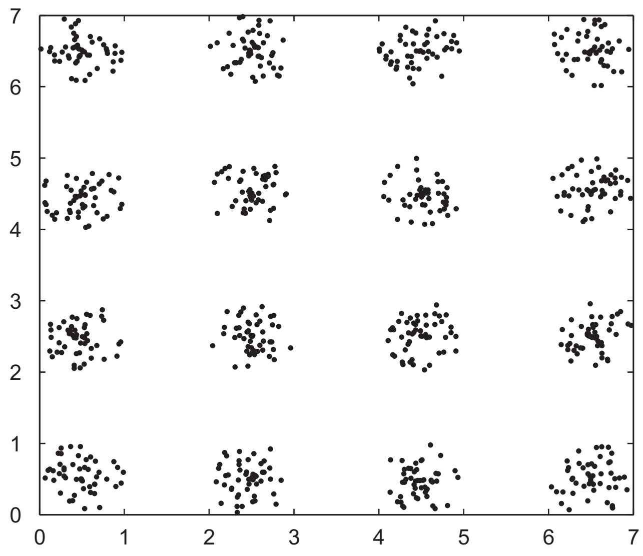

Fig. 9. Data 3.

6.1. Experiments on artificial data sets

In this section, the performances of the DNS [46], SCGA [48], DFS [47] and DNNM-clustering are compared through the

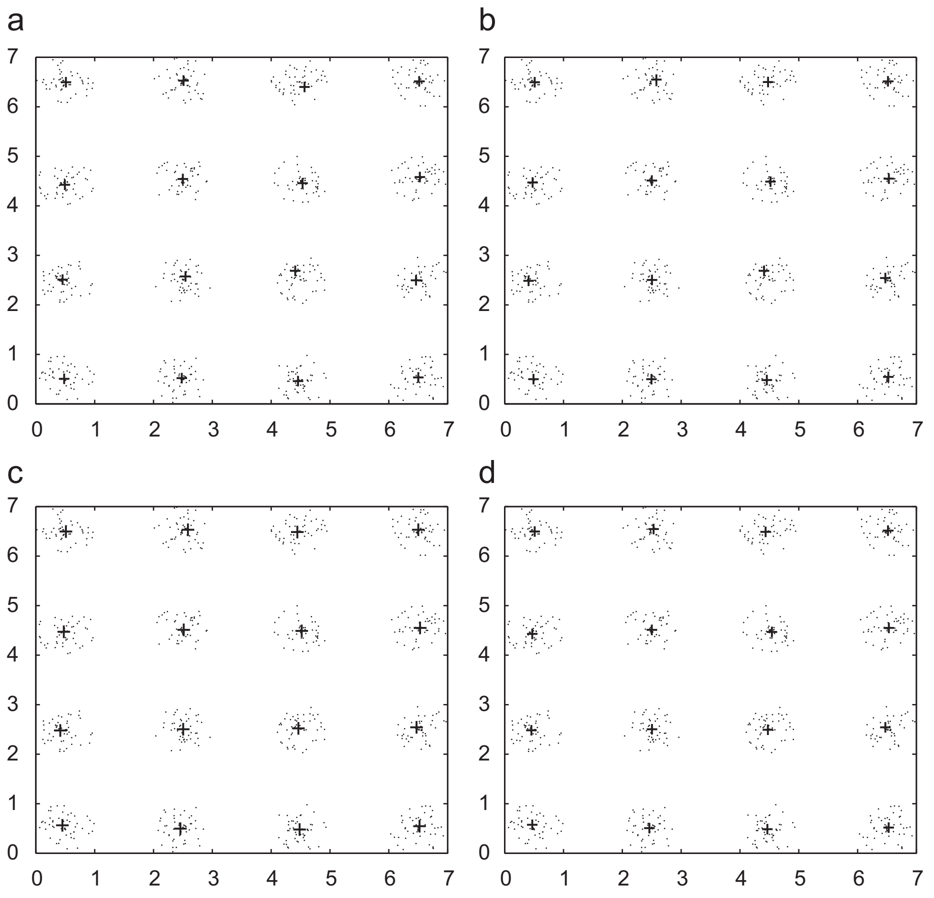

Fig. 10. The cluster centers obtained by using (a) DNS, (b) SCGA, (c) DFS, (d) DNNM-clustering. In all the experiments $P=150$.

experiments. In the experiments, the crossover and mutation probabilities used by all algorithms are $p_c=0.8$ and $p_m=0.005$, respectively. The population size is taken to be 150 for Data 3 since it has more clusters than the other data sets, while it is taken to be 100 for the other data sets. The parameter used in mutation operator is $r_m=0.2$. The solution acceptance threshold r_f used by SCCG is $r_f=0.95$. The total number of generations G is equal to 200. For all the experiments, the entire evolution of G generations, i.e., the run, has been repeated $R=30$ times, with different initial populations in order to reduce the well-known effects of randomness embedded in the GAs.

In order to evaluate the performance, the average number of niches and the standard errors [47] can be used. For each evolution, we compute at each generation the number of niches $v(t)$ (the population size P and niche radius σ being fixed) and store it in a $R\times G$ matrix W, where one row for each evolution and one column for each generation. At the end of each experiment consisting in R runs, we compute the average number of niches discovered at each generation by averaging the R values $v(t)$ in all the columns

$$\langle v(t)\rangle=\frac{1}{R}\sum_{i=1}^{R}W_{it},\quad t=1,2,\ldots,G. \tag{28}$$

Then, the values $\langle v(1)\rangle,\ldots,\langle v(G)\rangle$ represent the average behavior of the algorithm for the assigned values of P and σ. Finally, we compute the standard errors

$$\varepsilon(\langle v(t)\rangle)=\sqrt{\frac{1}{R}\left(\frac{\sum_{i=1}^{R}(W_{it}\text{-}\langle v(t)\rangle)^2}{R\text{-}1}\right)} \tag{29}$$

of $\langle v(t)\rangle$, $\forall t\in\{1,2,\ldots,G\}$.

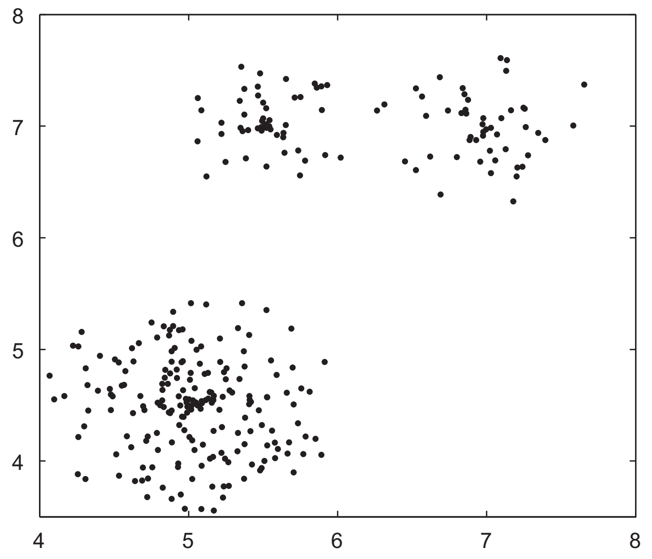

Fig. 11. Data 4.

In the experiments, five artificial data sets with widely varying characteristics are used for comparison. All the algorithms run with two different radii. The number of niches (i.e., the number of clusters) and the cluster centers obtained are given. Here, we only give the number of niches of the first data set as example. In the experiments, the value of γ in (3) should be determined by the CCA algorithm. The correlations for the five data sets are shown in Table 3.

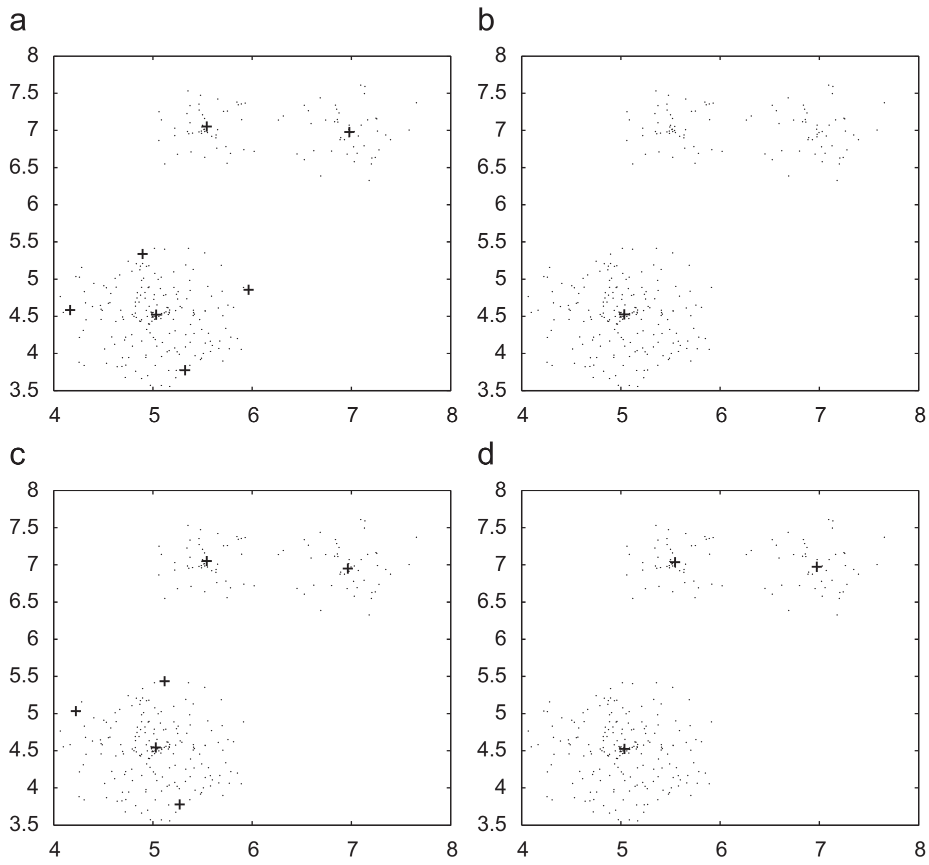
Fig. 12. The cluster centers obtained by using (a) DNS, (b) SCGA, (c) DFS, (d) DNNM-clustering. In all the experiments $P=100$.

Data 1: This data set consists of 300 two dimensional data points distributed over six disjoint clusters where each cluster contains 50 data points. This data set is shown in Fig. 4. The CCA result is shown in the first row of Table 3. $\gamma=10$ will be a good estimate. The final number of clusters and the standard errors obtained by DNS, SCGA, DFS and DNNM-clustering are given in Figs. 5(a), (b), (c) and (d), respectively. From Fig. 5, we can see all the algorithms are able to find out the number of optimal of the fitness function (i.e., the cluster centers) when the radius is properly selected. But for a small niching radius, $\sigma=0.5$, only DNNM-clustering algorithm works properly. The cluster centers obtained by all algorithms are shown in Fig. 6.

Data 2: This data set consists of 250 two dimensional data points distributed over five spherically shaped clusters as shown in Fig. 7. The clusters present here are highly overlapping, each consisting of 50 data points. The CCA result is shown in the second row of Table 3. $\gamma=5$ will be a good estimate. As is evident, all algorithms succeed in providing the number of clusters as well as the cluster centers with $\sigma=2$ while DNS, SCGA and DFS fail in doing so with a small radius. The cluster centers obtained by all algorithms are shown in Fig. 8.

Data 3: This data set is consists of 16 clusters as shown in Fig. 9. The CCA result is shown in the third row of Table 3. $\gamma=10$ will be a good estimate. As earlier, DNS, SCGA and DFS succeed with a proper radius and fail with a small radius. Only DNNM-clustering algorithm is insensitive to the choice of the initial radius. The cluster centers obtained by all algorithms are shown in Fig. 10.

Data 4: This is a two dimensional data set with different volume as shown in Fig. 11. There is one large cluster and two small clusters. The CCA result is shown in the fourth row of Table 3. $\gamma=5$ will be a good estimate. The cluster centers obtained by all algorithms are shown in Fig. 12. For this data set, the volumes of the clusters are different, so the peaks of the fitness function are not identical. In fact, the three peaks are 106.3, 39.3 and 34.9, respectively. For DNS and DFS, there are some false optimal values around the local optimal with the larger cluster with a smaller radius and succeed with a larger radius. For SCGA, the two small peaks are discarded during the global optima identification phase due to their small values compared with the largest one. Moreover, this result is also show that the DNNM-clustering algorithm is robust to cluster volumes.

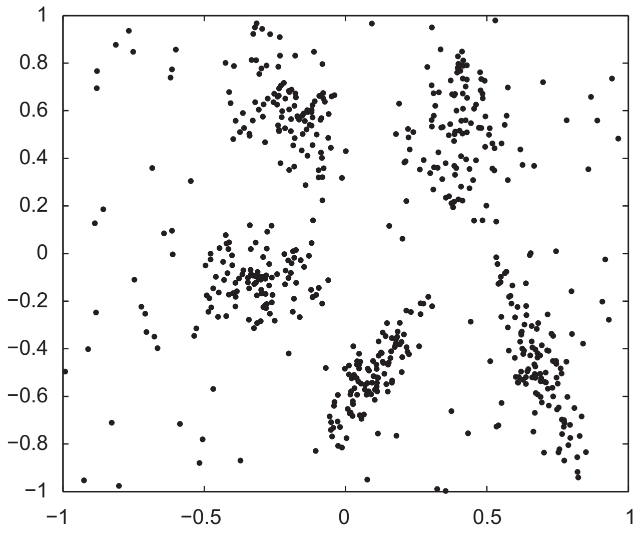
Fig. 13. Data 5.

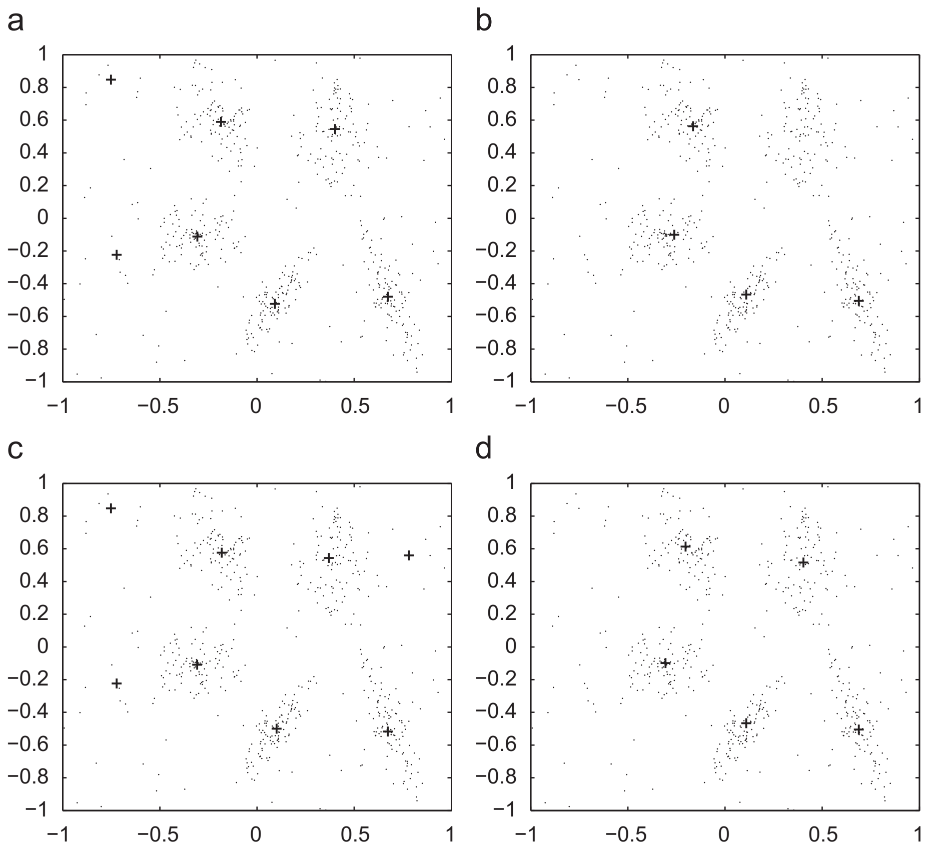

Fig. 14. The cluster centers obtained by using (a) DNS, (b) SCGA, (c) DFS, (d) DNNM-clustering. In all the experiments $P=100$.

Table 4
The mean of the number of clusters obtained by DNS, SCGA, DFS and DNNM-clustering applied to the two real-world data sets, here AC denotes the actual number of clusters present in the data set.

Data set	AC	DNS	SCGA	DFS	DNNM-clustering
Iris	3	9.65	1.85	9.50	2
Breast cancer	2	11.80	5.65	11.95	2

Data 5: This data set consists of 500 two dimensional data points distributed over five clusters. A very noisy background consisting 100 data points uniformly distributed within the region defined by $[-1,1]\times[-1,1]$ is added to this data set. This data set is shown in Fig. 13. The CCA result is shown in the fifth row of Table 3. $\gamma=5$ will be a good estimate. The cluster centers obtained by all algorithms are shown in Fig. 14. From Fig. 14, it is seen that only the DNNM-clustering succeed in providing the number of clusters as well as the cluster centers while DNS, SCGA and DFS fail miserably in doing so. For SCGA, the false optima and the optima with smallest fitness were discard through the principle of identify global optima used by SCGA. But for DNS and DFS, it is difficult to discard these values. The false optima obtained by DNS and DFS are dependent of the initialization of the algorithm.

In the following, two real-world high-dimensional data sets (Iris and Breast Cancer) from UCI Machine Learning Repository [54] are used to test whether the DNNM-clustering algorithm works well for the high-dimensional real data sets. Table 4 shows cluster numbers found on the two real data sets. It can be seen from Table 4 that the DNNM-clustering algorithm can be used to determine the number of clusters in a data set. For Iris data, all algorithms cannot provide the correct number of clusters, but only the DNNM-clustering algorithm can find two clusters. The

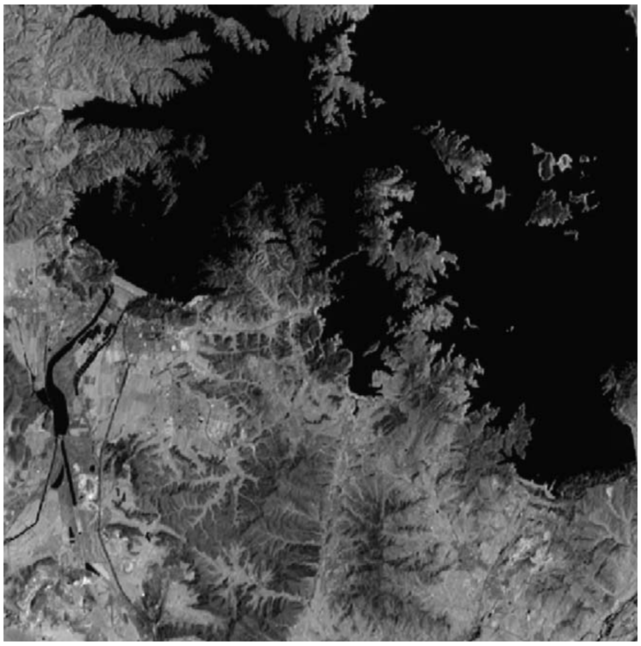

Fig. 15. The pseudocolor image of a part of MiYun obtained from Landsat-7 multispectral scanner composite by displaying band 5 as red, band 4 as green, and band 3 as blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

DNNM-clustering algorithm has separated the first class-Setosa-from the others. It is known that two classes (Versicolor and virginica) have a large amount of overlap, while the class Setosa is linearly separable from the other two. In fact, some researchers think that the Iris data set can be classed into tow classes [55,56]. For breast data, only DNNM-clustering algorithm can provide the correct number of clusters. And the classification error of DNNM-clustering algorithm is 3.53 percent. DNA, DFS and SCGA are also misled in the clustering because of the aforementioned problem in the selection of the niche radius.

From the experiment results of these data sets, it is seen that the DNNM-clustering algorithm is robust to the initializations. Since all the experiments have been repeated $R=30$ times with different initializations and in all the cases the correct estimation of the cluster centers is derived.

6.2. *Experiment on remote sensing image clustering*

Remote sensing image analysis is attracting a growing interest in real-world applications. The design of robust and efficient clustering algorithms becomes one of the most important issues addressed by the remote sensing community. In this section, we will apply DNS, SCGA, DFS and DNNM-clustering to the clustering of multispectral remote sensing image based on the spectral data of pixels. Although the remote sensing images usually have a large number of overlapping clusters, the experimental results show that the multispectral image can be effectively grouped into several clusters by the proposed method.

In this experiment, the algorithms are used to partition different landcover regions in the remote sensing image. A 512×512 remote sensing image of a part of MiYun obtained from Landsat-7 have been chosen. The image considered has three bands in the multispectral mode: band 3-red band, wavelength $0.63 \sim 0.69\,\mu\text{m}$; band 4-near-infrared band, wavelength $0.76 \sim 0.94\,\mu\text{m}$; band 5-shortwave infrared band, wavelength $1.55 \sim 1.75\,\mu\text{m}$. The pseudocolor images are shown in Fig. 15.

From the pseudocolor images, it can be seen that the landcovers of the images mainly contain five classes: water, vegetation (Veg), mountain (Moun), residential areas (RA) and blank regions (BR). In the experiment, we expect that the four algorithms can partition the remote sensing images into visually distinct clusters automatically. The number of population is set to 600 and the maximum generation 200. The crossover and mutation probabilities are the same as those used in the first experiment.

The clustering results for the image are shown in Fig. 16 with gray scale. The number of clusters identified by DNS, SCGA, DFS and DNNM-clustering are 3, 4, 3 and 6, respectively. As seen from Fig. 16,

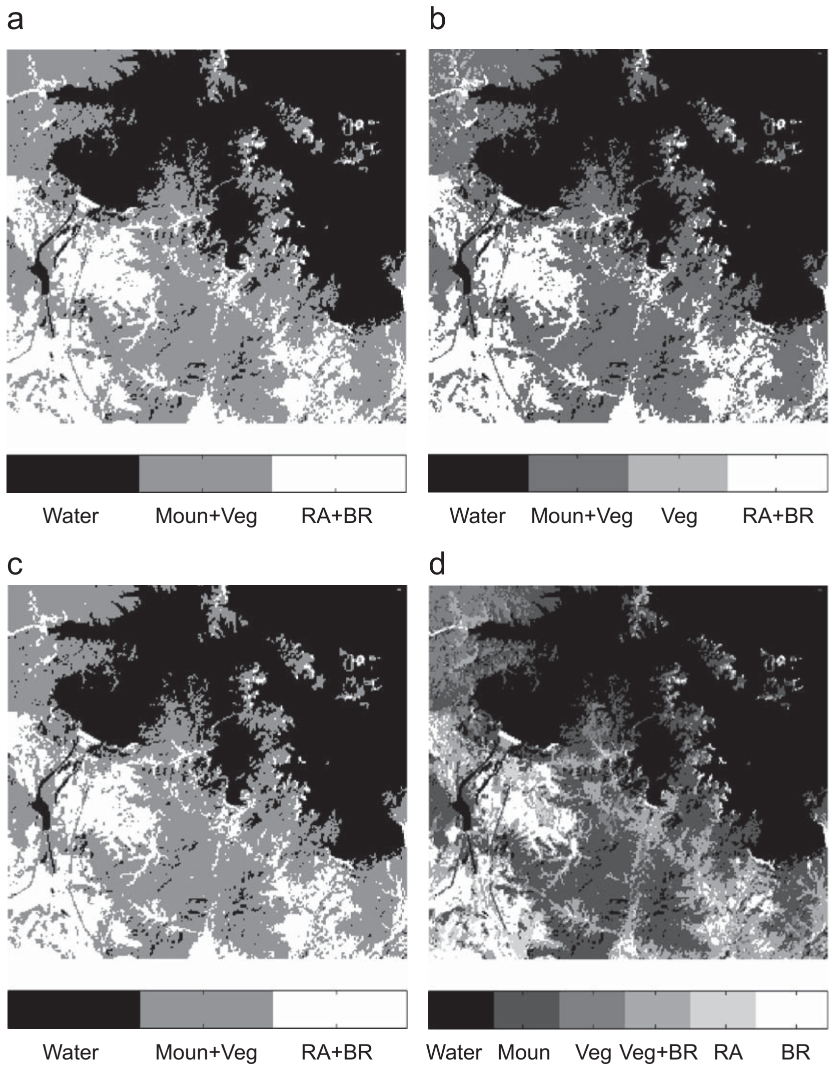

Fig. 16. The clustering results of the remote sensing image using: (a) DNS; (b) SCGA; (c) DFS; (d) DNNM-clustering.

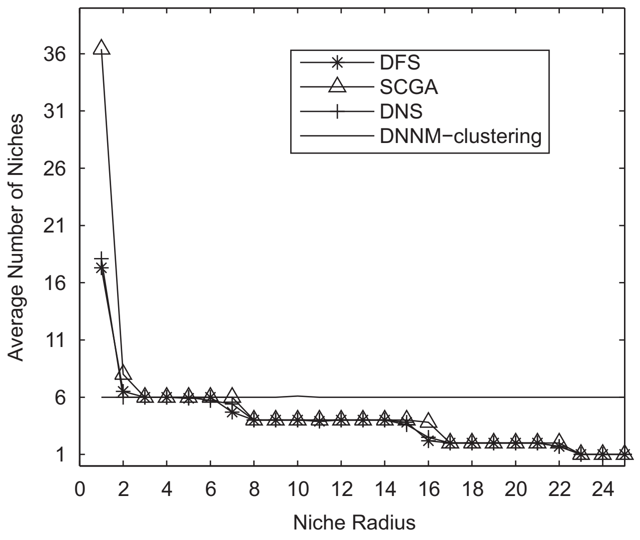

Fig. 17. The number of clusters obtained by using DNS, SCGA, DFS and DNNM-clustering algorithms for Data 1.

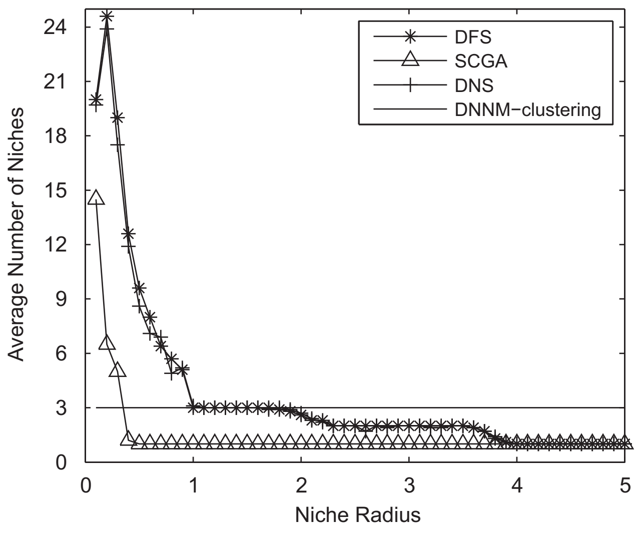

Fig. 18. The number of clusters obtained by using DNS, SCGA, DFS and DNNM-clustering algorithms for Data 4.

the water and the rivers in the residential areas are distinctly demarcated from the rest by all the four algorithms. For DNS, SCGA and DFS, there are some confusion between the residential areas and blank regions and between the mountain and the vegetation. For the DNNM-clustering algorithm, most of the landcover categories have been correctly distinguished. For example, the vegetation on the top left of the image, the residential areas and many other structures are identified by the DNNM-clustering algorithm. So we can conclude that DNNM-clustering algorithm is an efficient clustering algorithm for differentiating the various landover types present in the image.

6.3. *Effect of niche radius*

As mentioned earlier, the performance of the DNNM-clustering algorithm is independent of the initial niche radius. To examine this claim, we conduct a series of experiments, in which we vary the value of niche radius σ and count the number of niches found. For these runs we use $p_c = 0.8$, $p_m = 0.005$, set the population size $P = 100$, and set the number of generations $G = 200$. The results

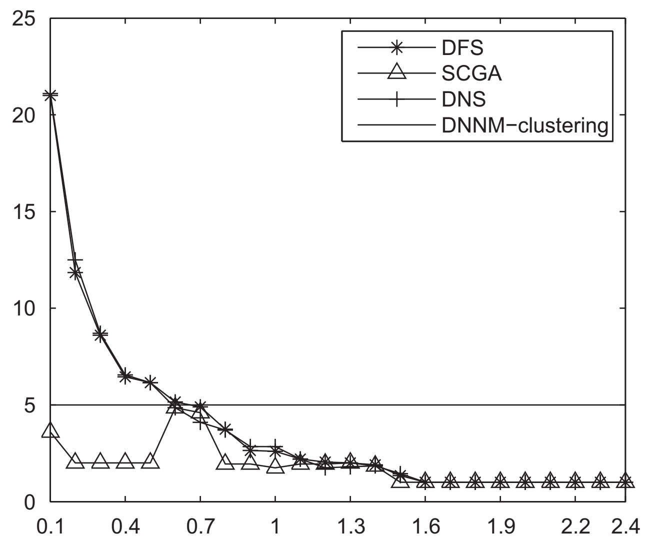

Fig. 19. The number of clusters obtained by using DNS, SCGA, DFS and DNNM-clustering algorithms for Data 5.

are averaged over 30 runs for each value of σ. The results obtained for Data 1, Data 4 and Data 5 are shown in Figs. 17, 18 and 19, respectively. In the experiments, the maximum value of the niche radius is set to the largest distance between the data points.

The figures show that, as expected, as the niche radius is increased, the numbers of niches found by DNNM-clustering remain same to the numbers of the clusters. The DNNM-clustering algorithm can consistently find all global optima of the data sets while other three algorithms success only for some values of radius.

7. Conclusion

In this paper, a robust clustering algorithm based on dynamic niching with niche migration (DNNM-clustering) has been developed for solving clustering problems with unknown cluster number. The DNNM-clustering algorithm can find the optimal number of clusters as well as the cluster centers automatically. As the number of clusters is not known a priori in most practical circumstance, DNNM-clustering algorithm can be used more widely. In the DNNM-clustering algorithm, each chromosome is encoded a center of a cluster by a sequence of real-valued numbers. This is more natural and simple than the presentation used by other clustering algorithms based on GA. The dynamic niching is accomplished without assuming any prior knowledge on the number of niches and the niche radius. The introduction of the niche migration makes the DNNM-clustering algorithm is insensitive to the choice of the initial radius. The superiority of the DNNM-clustering algorithm over DNS, SCGA and DFS algorithm has demonstrated by the experiments. All the experiment results described in this paper have shown that our algorithm is effective, because it provides all the actual cluster centers. Moreover, the DNNM-clustering has been applied to the multispectral remote sensing image for clustering the pixels into several classes, which also illustrated its effectiveness and superiority.

Although the results presented here are extremely encouraging, there is an issue that deserves in-depth study in the future. The population size is undoubtedly crucial to the performance of the algorithm. In order to steadily maintain the actual number of cluster, we should estimate the minimum population size needed by our method.

References

[1] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Englewood Cliffs, NJ, 1988.
[2] J.T. Tou, R.C. Gonzalez, Pattern Recognition Principles, Addison-Wesley, Reading, MA, 1974.
[3] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.
[4] R. Xu, D. Wunsch, Survey of clustering algorithm, IEEE Trans. Neural Networks 16 (3) (2005) 645–678.
[5] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, Wiley, New York, 1990.
[6] P.H.A. Sneath, The application of computers to taxonomy, J. Gen. Microbiol. 17 (1957) 201–226.
[7] S. Hands, B.S. Everitt, A Monte Carlo study of the recovery of cluster structure in binary data by hierarchical clustering techniques, Multivar. Behav. Res. 22 (1987) 235–243.
[8] J.H. Ward, Hierarchical groupings to optimize an objective function, J. Am. Stat. Assoc. 58 (1963) 236–244.
[9] R.W. Payne, D.A. Preece, Identification keys and diagnostic tables: a review, J. R. Stat. Soc. A 143 (1980) 253–292.
[10] Z. Hubálek, Coefficients of association and similarity based on binary (presence–absence) data: an evaluation, Biol. Rev. 57 (1982) 669–689.
[11] G.W. Milligan, M.C. Cooper, An examination of procedures for determining the number of clusters in a data set, Psychometrika 50 (1985) 159–179.
[12] X.L. Xie, G. Beni, A validity measure for fuzzy clustering, IEEE Trans. Pattern Anal. Mach. Intell. 13 (8) (1991) 841–847.
[13] N.R. Pal, J.C. Bezdek, On cluster validity for fuzzy c-means model, IEEE Trans. Fuzzy Syst. 3 (3) (1995) 370–379.
[14] R. Krishnapuram, C.P. Freg, Fitting an unknown number of lines and planes to image data through compatible cluster merging, Pattern Recognition 25 (4) (1992) 385–400.
[15] R. Krishnapuram, H. Frigui, O. Nasraoui, Fuzzy and possibilistic shell clustering algorithms and their application to boundary detection and surface approximation, IEEE Trans. Fuzzy Syst. 3 (1) (1995) 29–60.
[16] X. Zhuang, Y. Huang, K. Palaniappan, Y. Zhao, Gaussian mixture density modeling, decomposition and applications, IEEE Trans. Image Process. 5 (9) (1996) 1293–1302.
[17] J.M. Jolion, P. Meer, S. Bataouche, Robust clustering with applications in computer vision, IEEE Trans. Pattern Anal. Mach. Intell. 13 (8) (1991) 791–802.
[18] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, 1975.
[19] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[20] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, AI Series, Springer, New York, 1994.
[21] K.A. De Jong, An analysis of the behavior of a class of genetic adaptive systems, Doctoral Dissertation, University of Michigan, Ann Arbor, Michigan, 1975.
[22] C.A. Murthy, N. Chowdhury, In search of optimal clusters using genetic algorithms, Pattern Recognition Lett. 17 (1996) 825–832.
[23] U. Maulik, S. Bandyopadhyay, Genetic algorithm based clustering technique, Pattern Recognition 33 (9) (2000) 1455–1465.
[24] A. Tucker, J. Crampton, S. Swift, RGFGA: an efficient representation and crossover for grouping genetic algorithms, Evol. Comput. 13 (4) (2005) 477–499.
[25] S. Bandyopdhyay, U. Maulik, An evolutionary technique based on K-means algorithm for optimal clustering in RN, Inf. Sci. 146 (2002) 221–237.
[26] L.O. Hall, I.B. Özyurt, J.C. Bezdek, Clustering with a genetically optimized approach, IEEE Trans. Evol. Comput. 3 (2) (1999) 103–112.
[27] K. Krishna, M.N. Murty, Genetic K-means algorithm, IEEE Trans. Syst. Man Cybern. Part B Cybern. 29 (1999) 433–439.
[28] S. Bandyopdhyay, S. Saha, GAPS: a clustering method using a new point symmetry-based distance measure, Pattern Recognition 40 (12) (2007) 3430–3451.
[29] M. Laszlo, S. Mukherjee, A genetic algorithm using hyper-quadtrees for lowdimensional K-means clustering, IEEE Trans. Pattern Anal. Mach. Intell. 28 (4) (2006) 533–543.
[30] M. Laszlo, S. Mukherjee, A genetic algorithm that exchanges neighboring centers for K-means clustering, Pattern Recognition Lett. 28 (2007) 2359–2366.
[31] D.X. Chang, X.D. Zhang, C.W. Zheng, A genetic algorithm with gene rearrangement for K-means clustering, Pattern Recognition 42 (7) (2009) 1210–1222.
[32] R. Srikanth, R. George, N. Warsi, et al., A variable-length genetic algorithm for clustering and classification, Pattern Recognition Lett. 16 (1995) 789–800.
[33] P. Scheunders, A genetic c-means clustering algorithm applied to color image quantization, Pattern Recognition 30 (6) (1997) 859–866.
[34] L.Y. Tseng, S.B. Yang, A genetic approach to the automatic clustering problem, Pattern Recognition 34 (2) (2001) 415–424.
[35] S. Bandyopadhyay, U. Maulik, Genetic clustering for automatic evolution of clusters and application to image classification, Pattern Recognition 35 (6) (2002) 1197–1208.
[36] S. Bandyopadhyay, S. Saha, A point symmetry-based clustering technique for automatic evolution of clusters, IEEE Trans. Knowl. Data Eng. 20 (11) (2008) 1441–1457.
[37] E. Mayr, Systematics and the Origin of Species from the Viewpoint of a Zoologist, Columbia University Press, New York, 1942.
[38] D.E. Goldberg, J. Richardson, Genetic algorithms with sharing for multimodal function optimization, in: J.J. Grefenstette (Ed.), Genetic Algorithms and Their Applications, Lawrence Erlbaum, Hillsdale, NJ, 1987, pp. 41–49.
[39] K. Deb, D.E. Goldberg, An investigation of niche and species-formation in genetic function optimization, in: J.D. Schaffer (Ed.), Proceedings of the 3rd International Conference on Genetic Algorithms, San Mateo, CA, 1989, pp. 42–50.
[40] R.E. Smith, S. Forrest, A.S. Perelson, Searching for diverse, cooperative populations with genetic algorithms, Evol. Comput. 1 (2) (1992) 127–149.
[41] S. Forrest, R.E. Smith, B. Javornik, A.S. Perelson, Using genetic algorithms to explore pattern recognition in the immune system, Evol. Comput. 1 (1993) 191–211.
[42] P. Darwen, X. Yao, Every niche method has its niche: fitness sharing and implicit sharing compared, in: H.-M. Voigt, W. Ebeling, I. Rechenberg, H.-P. Schwefel (Eds.), Parallel Problem Solving from Nature-PPSN IV, Lecture Notes in Computer Science, vol. 1141, Springer, Berlin, Germany, 1996, pp. 398–407.
[43] C. Goodall, M-estimator of location: an outline of the theory, in: D.C. Hoaglin, F. Mosteller, J.W. Tukey (Eds.), Understanding Robust and Exploratory Data Analysis, New York, 1983, pp. 339–403.
[44] F.R. Hampel, E.M. Ponchotti, P.J. Rousseeuw, W.A. Stahel, Robust Statistics: The Approach based on Influence Functions, Wiley, New York, 1986.
[45] R.A. Maronna, R.D. Martin, V.J. Yohai, Robust Statistics: Theory and Methods, Wiley, New York, 2006.
[46] B.L. Miller, M.J. Shaw, Genetic algorithms with dynamic niche sharing for multimodal function optimization, in: Proceedings of the 1996 IEEE Transactions on Evolutionary Computation, 1996, pp. 786–791.
[47] C.D. Antonio, S.D. Claudio, M. Angelo, Where are the niches? Dynamic fitness sharing, IEEE Trans. Evol. Comput. 11 (4) (2007) 453–465.
[48] J.P. Li, M.E. Balazs, G.T. Parks, P.J. Clarkson, A species conserving genetic algorithm for multimodal function optimization, Evol. Comput. 10 (3) (2002) 207–234.
[49] M.S. Yang, K.L. Wu, A similarity-based robust clustering method, IEEE Trans. Pattern Anal. Mach. Intell. 26 (4) (2004) 434–448.
[50] D. Beasley, D.R. Bull, R.R. Martin, A sequential niche technique for multimodal function optimization, Evol. Comput. 1 (2) (1993) 101–125.
[51] S.W. Mahfoud, Genetic drift in sharing methods, in: Proceedings of the 1st IEEE Conference on Evolutionary Computation, 1994, pp. 67–72.
[52] S.W. Mahfoud, Population size and genetic drift in fitness sharing, in: L.D. Whitley, M.D. Vose (Eds.), Proceedings of the Foundations Genetic Algorithms, 1995, pp. 185–223.
[53] W.M. Spears, Simple subpopulation schemes, in: A.V. Sebald, L.J. Fogel (Eds.), Proceedings of the 4th Annual Conference on Evolutionary Programming, 1994, pp. 296–307.
[54] 〈http://www.ics.uci.edu/mlearn/MLRepository.html〉.
[55] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.
[56] R. Kothari, D. Pitts, On finding the number of clusters, Pattern Recognition Lett. 20 (4) (1999) 405–416.

About the Author—DONGXIA CHANG received the B.S. and M.S. degrees in mathematics from Xi Xidian University, in 2000 and 2003, respectively. She is currently pursuing the Ph.D. degree at the Department of Automation, Tsinghua University. Her current research interests include evolutionary computation, clustering and intelligent signal processing.

About the Author—XIANDA ZHANG received the B.S. degree in radar engineering from Xidian University, in 1969, the M.S. degree in instrument engineering from Harbin Institute of Technology in 1982, and the Ph.D. degree in electrical engineering from Tohoku University, Sendai, Japan, in 1987. Since 1992, he has been with the Department of Automation, Tsinghua University. His current research interests are signal processing with applications in radar and communications and intelligent signal processing.

About the Author—CHANGWEN ZHENG received the B.S. degree in mathematics and Ph.D. degree in control science and engineering from Huazhong Normal University in 1992 and 2003, respectively. He was with the General Software Laboratory, Institute of Software, Chinese Academy of Sciences since 2003. His current research interests include route planning, evolutionary computation and neural networks.

About the Author—DAOMING ZHANG received the B.S. and M.S. degrees in physics from National University of Defense Technology, in 2000 and 2003, respectively. He is currently pursuing the Ph.D. degree at the Department of Automation, Tsinghua University. His current research interests include image fusion and intelligent signal processing.