PERGAMON

Neural Networks 15 (2002) 105–120

Neural Networks

www.elsevier.com/locate/neunet

Contributed article

Projective ART for clustering data sets in high dimensional spaces

Yongqiang Cao, Jianhong Wu*

Department of Mathematics and Statistics, York University, 4700 Keele Street, Toronto, Ontario, Canada M3J 1P3

Received 14 September 2000; revised 20 July 2001; accepted 20 July 2001

Abstract

A new neural network architecture (PART) and the resulting algorithm are proposed to find projected clusters for data sets in high dimensional spaces. The architecture is based on the well known ART developed by Carpenter and Grossberg, and a major modification (selective output signaling) is provided in order to deal with the inherent sparsity in the full space of the data points from many data-mining applications. This selective output signaling mechanism allows the signal generated in a node in the input layer to be transmitted to a node in the clustering layer only when the signal is similar to the top-down weight between the two nodes and, hence, PART focuses on dimensions where information can be found. Illustrative examples are provided, simulations on high dimensional synthetic data and comparisons with Fuzzy ART module and PROCLUS are also reported. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Neural networks; Data mining; Projected clustering; Adaptive resonance theory; Learning; Pattern recognition; ART1; ART2

1. Introduction

Most clustering algorithms do not work efficiently for data sets in high dimensional spaces because of the inherent sparsity of data. In such data sets, for any given pair of points there exist at least a few dimensions on which the points are far apart from one another. Therefore, it is not feasible to find interesting clusters in the original full space of all dimensions. Consequently, a clustering algorithm is often preceded by feature selection whose goal is to find the particular dimensions on which points in the data set are correlated. Unfortunately, a feature selection procedure requires picking up certain dimensions in advance, which can lead to a significant loss of information. As a result, the clustering results preceded by a feature selection procedure may not be reliable. This reliability problem becomes even worse in some data mining applications where some points are correlated with respect to a given set of dimensions and others are correlated with respect to different sets of dimensions and, therefore, it may not always be possible to prune off dimensions without, at the same time, failing to find all interesting features and clusters.

Therefore, we are facing a feasibility–reliability dilemma in clustering data sets of high dimensionality: on the one hand it is feasible to find clusters only in lower dimensional subspaces due to the sparsity of the data in full space; on the other hand, the clustering results by pruning off dimensions in advance are not reliable due to the loss of information. This feasibility–reliability dilemma motivated the concept of projected clustering (Aggarwal, Procopiuc, Wolf, Yu & Park, 1999) whose central goal is to find projected clusters, each of which consists of a subset C of data points together with a subset D of dimensions such that the points in C are closely correlated in the subspace of dimensions D. The idea here is to find desired clusters based not only on points, but also on dimensions. For data sets in high dimensional spaces, projected clustering can lead to significant improvement in the quality of clustering. A fast projected clustering algorithm PROCLUS was proposed by Aggarwal et al. (1999). The algorithm finds the appropriate sets of candidate clusters and dimensions by using the so-called medoids technique which uses points (medoids) in the original data set to serve as surrogate centers for clusters. In addition, the greedy technique and a locality analysis are used to find the set of dimensions associated with each medoid. Unfortunately, PROCLUS requires the number of clusters and the average dimension as input parameters. This seems to impose significant challenge for a user. Moreover, as the illustrative simulations in Section 4 show, PROCLUS is sensitive to the choice of these input parameters.

Here, we present an alternative approach based on a new neural network architecture PART (Projective Adaptive Resonance Theory). The basic architecture of PART is similar to that of ART neural networks which have been shown to be very effective in self-organizing stable recognition

* Corresponding author.
E-mail address: wujh@mathstat.yorku.ca (J. Wu).

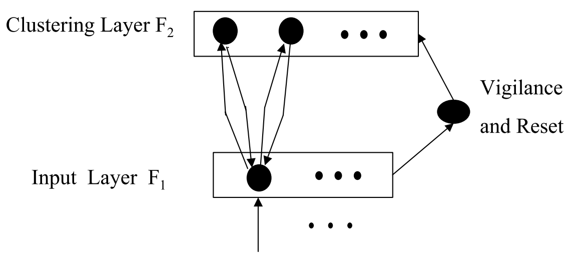

Fig. 1. Simplified configuration of ART architecture consisting of an input layer F_1, a clustering layer F_2 and a reset subsystem.

codes in real time in response to arbitrary sequences of input patterns (for example, Carpenter & Grossberg, 1987a,b, 1990; Carpenter, Grossberg & Reynolds, 1991; Carpenter, Grossberg & Rosen, 1991a,b; Carpenter, Grossberg, Markuzon, Reynolds & Rosen, 1992). Fig. 1 shows the simplified configuration of the ART structure, which involves an input processing field (F_1 layer, also called input layer or comparison layer), a clustering field (F_2 layer, also called clustering layer or recognition layer), and a reset subsystem. There are two sets of connections (each with its own weights) between each node in the F_1 layer and each node in the F_2 layer. The F_1 layer is connected to the F_2 layer by bottom-up weights while the F_2 layer is connected to the F_1 layer by top-down weights. The connection weights between these two layers can be modified according to two different learning rules. The F_2 layer is a competitive layer which follows the winner-take-all paradigm: the node in the F_2 with the largest net input becomes the candidate to learn the input pattern. Whether the candidate will learn the input pattern is decided by the reset mechanism, which controls the degree of similarity of patterns placed in the same node (cluster).

Despite the great success of applying ART to clustering problems, our simulations reported below show that the current ART architecture has to be modified to perform the task of projected clustering. In particular, ART focuses on the similarity of patterns in full space and, therefore, may have difficulty in finding projected clusters in high dimensional data sets due to the inherent sparsity of the data points in the full space. Besides, pre-processing of the input patterns to an ART system is nontrivial. In particular, for analog input patterns, a user faces the problem of meeting the data normalization requirements of ART modules, and simultaneously preserving the original projected clusters in subspaces. Another problem common to all clustering methods is that the subspace of dimensions associated with each projected cluster cannot be identified in advance. One may attempt to find clusters in all possible subspaces and then to compare the results to obtain an optimal partition of the data set, but this is practically not feasible as the number of all possible subspaces $2^m - 1$ is intractably large for a data set with high dimension m.

Similar to ART architecture, the F_2 layer of our proposed PART is a competitive layer which follows the winner-take-all paradigm. Recall that a node of the F_2 layer is called committed if it has learned some input patterns in previous learning traces, and a node is called noncommitted if it has not learned any input pattern. In PART architecture, only the committed nodes of the F_2 layer accept signals from the F_1 layer at the phase of competition, the noncommitted nodes do not take part in the competition. A noncommitted node will automatically become the winner if no proper committed nodes can be chosen.

The principal difference between PART and ART is in the F_1 layer. In PART, the F_1 layer selectively sends signals to nodes in the F_2 layer. In other words, a node in the F_1 layer can be active relative to some F_2 nodes, but inactive relative to other F_2 nodes. To which F_2 node an F_1 node is active is determined by a similarity test between the corresponding top-down weight and the signal generated in the F_1 node. This similarity test plays a key role in the projected clustering of PART.

The remaining part of this paper is organized as follows. Section 2 describes the PART architecture and the STM and LTM equations, as well as the vigilance and reset mechanism. Section 3 describes the projected clustering algorithm according to PART model. Section 4 presents illustrative examples, simulation results on high dimensional synthetic data, and comparisons with Fuzzy ART and PROCLUS. Section 5 provides some concluding remarks.

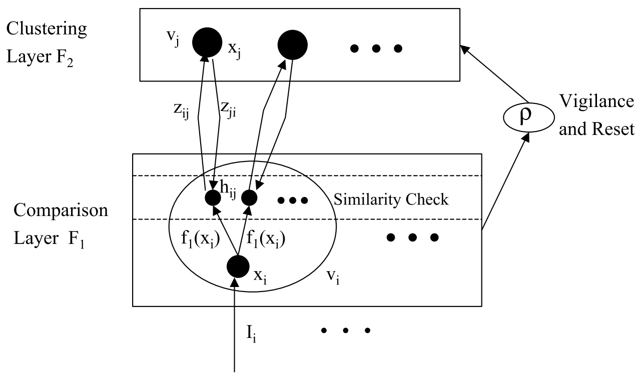

Fig. 2. PART architecture. In addition to the usual F_1 layer (input and comparison), F_2 layer (clustering) and a reset mechanism, there is a hidden layer associated with each F_1 layer node v_i for similarity check to determine whether the node v_i is active relative to an F_2 layer node v_j.

2. Projective adaptive resonance theory

2.1. Basic architecture of PART

Fig. 2 illustrates the basic PART architecture. Here, we denote the nodes in the F_1 layer by v_i, $i = 1, \ldots, m$; nodes in the F_2 layer by v_j, $j = m + 1, \ldots, m + n$; the activation of an F_1 node v_i by x_i, the activation of an F_2 node v_j by x_j; the bottom-up weight from v_i to v_j by z_{ij}, the top-down weight (also called template) from v_j to v_i by z_{ji}. The main difference between PART and ART is the introduction of the selective output signal from a node v_i in F_1 to a committed node v_j in F_2 by a similarity check between the top-down weight z_{ji} and the signal $f_1(x_i)$ generated in v_i, where f_1 is a signal function. More precisely, for an F_1 node v_i and a committed F_2 node v_j, we define the selective output signal of node v_i to node v_j by

$$h_{ij} = h(x_i, z_{ij}, z_{ji}) = h_\sigma(f_1(x_i), z_{ji})l(z_{ij}), \tag{1}$$

where

$$h_\sigma(a,b) = \begin{cases} 1 & \text{if } d(a,b) \le \sigma, \\ 0 & \text{if } d(a,b) > \sigma, \end{cases} \tag{2}$$

with $d(a, b)$ being a quasi-distance function (for example, $d(a, b) = |a - b|$ or $|a - b|/(e + |b|)$), and

$$l(z_{ij}) = \begin{cases} 1 & \text{if } z_{ij} > \theta, \\ 0 & \text{if } z_{ij} \le \theta, \end{cases} \tag{3}$$

with θ (threshold) being 0 or a small number to be specified later, σ is a distance parameter.

We say that v_i is active to v_j if $h_{ij} = 1$, and inactive to v_j if $h_{ij} = 0$.

2.2. STM equations

The STM equations for nodes in the F_1 layer are simple, and these equations incorporate the internal decay $-x_i$ as well as the input I_i to v_i as follows:

$$\epsilon \frac{\mathrm{d}x_i}{\mathrm{d}t} = -x_i + I_i, \tag{4}$$

where $0 < \epsilon \ll 1$.

The STM equations of the committed nodes in the F_2 layer follow from the on-center off-surround principle in ART (Carpenter & Grossberg, 1987a) and take the form

$$\epsilon \frac{\mathrm{d}x_j}{\mathrm{d}t} = -x_j + (1 - Ax_j)J_j^+ - (B + Cx_j)J_j^-, \tag{5}$$

where

$$J_j^+ = g(x_j) + T_j, \tag{6}$$

$$J_j^- = \sum_{k \ne j, v_k \in F_2} g(x_k). \tag{7}$$

g is a signal function, and in contrast to ART, we have

$$T_j = \sum_{v_i \in F_1} z_{ij}h_{ij} = \sum_{v_i \in F_1} z_{ij}h(x_i, z_{ij}, z_{ji}). \tag{8}$$

The F_2 layer makes a choice by winner-take-all paradigm:

$$f_2(x_j) = \begin{cases} 1 & \text{if node } v_j \text{ is a winner,} \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

The winning node v_j is determined according to the following rule: let $\Gamma = \{T_k : F_2 \text{ node } v_k \text{ is committed and has not been reset on the current trial}\}$, then node v_j is a winner either if $\Gamma \ne \phi$ and $T_j = \max \Gamma$, or if $\Gamma = \phi$ and node v_j is the next noncommitted node in F_2 layer.

According to Eq. (9), a winning F_2 node will become active and all other F_2 nodes will become inactive.

2.3. LTM equations

The LTM trace of the bottom-up pathway from F_1 node v_i to F_2 node v_j and the LTM trace of the top-down pathway from F_2 node v_j to F_1 node v_i obey the following learning equations, respectively:

$$\delta \frac{\mathrm{d}z_{ij}}{\mathrm{d}t} = f_2(x_j)\Big[(1 - z_{ij})Lh(x_i, z_{ij}, z_{ji}) - z_{ij} \sum_{k \ne i, v_k \in F_1} h(x_k, z_{kj}, z_{jk})\Big], \tag{10}$$

$$\frac{\mathrm{d}z_{ji}}{\mathrm{d}t} = f_2(x_j)[-z_{ji} + f_1(x_i)] \text{ if } v_j \text{ is committed,} \tag{11}$$

$$\delta \frac{\mathrm{d}z_{ji}}{\mathrm{d}t} = f_2(x_j)[-z_{ji} + f_1(x_i)] \text{ if } v_j \text{ is noncommitted,} \tag{12}$$

where $0 < \epsilon \ll \delta \ll 1$, which sets up the order of various learning rates: the STM traces are activated much faster than all LTM traces, and the bottom-up LTM traces and the top-down LTM traces for noncommitted nodes are adjusted much faster than the top-down LTM traces for committed nodes. In the above, f_2 is a signal function. Note that if a noncommitted F_2 node v_j becomes the winner, then all F_1 nodes are active to the winner, that is, $h_{ij} = h(x_i, z_{ij}, z_{ji}) = 1$ for all F_1 nodes v_i.

According to Eq. (10), we obtain

$$\delta \frac{\mathrm{d}z_{ij}}{\mathrm{d}t} = \begin{cases} (1 - z_{ij})L - z_{ij}(|X| - 1) & \text{if } v_j \text{ is active and } v_i \text{ is active to } v_j, \\ -|X|z_{ij} & \text{if } v_j \text{ is active, but } v_i \text{ is inactive to } v_j, \\ 0 & \text{if } v_j \text{ is inactive,} \end{cases} \tag{13}$$

where $|X|$ denotes the number of F_1 nodes which are active to the F_2 node v_j. Consequently, the bottom-up LTM traces still obey the Weber Law and the Associative Decay Law (Carpenter & Grossberg, 1987a).

Similarly, from Eqs. (11) and (12), we derive that if an F_2 node is committed, then

$$\frac{\mathrm{d}z_{ji}}{\mathrm{d}t} = \begin{cases} -z_{ji} + f_1(x_i) & \text{if } v_j \text{ is active,} \\ 0 & \text{if } v_j \text{ is inactive;} \end{cases} \tag{14}$$

and if an F_2 node is noncommitted, then

$$\delta \frac{\mathrm{d}z_{ji}}{\mathrm{d}t} = \begin{cases} -z_{ji} + f_1(x_i) & \text{if } v_j \text{ is active,} \\ 0 & \text{if } v_j \text{ is inactive.} \end{cases} \tag{15}$$

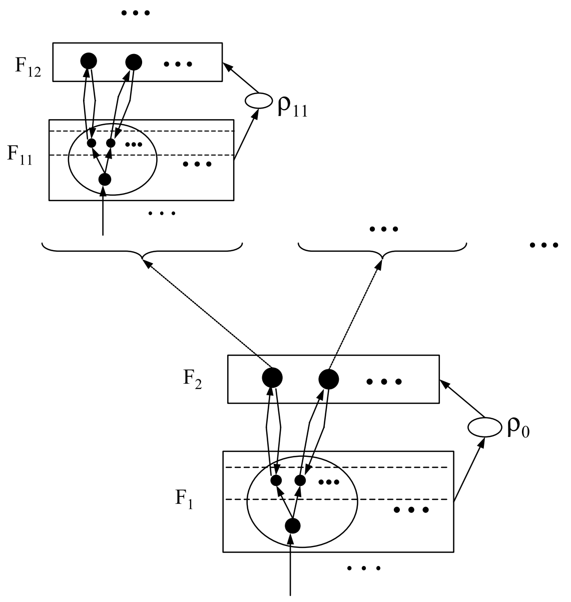

Fig. 3. PART tree architecture with increasing vigilance parameter, $\rho_0 < \rho_{11}, \rho_{21}, \ldots < \ldots$, here all points in a cluster (node) v_j of the F_2 layer can be further classified according to a new vigilance $\rho_{j1} > \rho_0$.

2.4. Vigilance and reset

As in ART, if a winning (active) F_2 node v_j does not satisfy some vigilance conditions, it will be reset so that the node v_j will always be inactive (or cannot become a winner) during the remainder of the current trial.

The vigilance conditions in PART also control the degree of similarity of patterns placed in the same cluster. The major difference between PART and ART2 (Carpenter & Grossberg, 1987b) is that the similarity measurement in PART is closely related to subspaces involved. Namely, for a winning committed F_2 node v_j, define

$$r_j = \sum_i h_{ij}, \tag{16}$$

and we reset the winner v_j if and only if

$$r_j < \rho. \tag{17}$$

Here $\rho \in \{1, 2, \ldots, m\}$ is a vigilance parameter.

Note that in PART no vigilance and reset is assumed for a noncommitted winning node during its learning. The degree of similarity of patterns for a committed node is controlled by not only vigilance parameter ρ, but also distance parameter σ through Eqs. (1) and (2). These two parameters have different roles: the vigilance parameter ρ controls the size of dimensions of the projected subspaces, and the distance parameter σ controls the degree of similarity in a specific dimension involved.

2.5. The extension of PART architecture: PART tree

Fig. 3 illustrates the architecture of a PART tree. By increasing the vigilance parameter, points in a cluster of the F_2 layer can be further classified in terms of a new and larger vigilance parameter. When this is done for each cluster in the F_2 layer, we obtain a new F_2 layer consisting of new sets of projected clusters. This process can be continued for the new F_2 layer and, hence, we will obtain a PART tree. A natural condition to stop this process is when $\rho > m$ or when no new subcluster can be formed. Other practical stopping conditions will be discussed in the next section. PART tree architecture reveals the hierarchical relations of various projected clusters and is useful for different purposes of the same user or for different user groups.

3. Algorithms

This section describes the projected clustering algorithms according to the PART model and PART tree architecture.

F_1 activation and computation of h_{ij}. Here, we take function f_1 as the identical function $f_1(x_i) = x_i$, and by Eq. (4), $x_i = I_i$ at an equilibrium. Therefore, from Eq. (1) we have

$$h_{ij} = h_\sigma(I_i, z_{ji})l(z_{ij}). \tag{18}$$

In what follows, we take the quasi-distance function in Eq. (2) as

$$d(a,b) = |a-b|/(e+|b|). \tag{19}$$

F_2 activation and selection of winner. We compute the input T_j to the committed F_2 node v_j by Eq. (8), and then select the winner according to the rules specified in Section 2.2.

Vigilance and reset. The algorithm uses the vigilance and reset mechanism shown in Eqs. (16) and (17). Namely, winner v_j is reset if and only if

$$\sum_i h_{ij} < \rho, \tag{20}$$

for a vigilance parameter $\rho \in \{1, 2, \ldots, m\}$.

Learning. The learning formulae follow from Eqs. (13)–(15). In particular, for the committed winning F_2 node v_j which has passed the vigilance test, we have

$$z_{ij}^{new} = \begin{cases} L/(L-1+|X|) & \text{if } F_1 \text{ node } v_i \text{ is active to } v_j, \\ 0 & \text{if } F_1 \text{ node } v_i \text{ is inactive to } v_j, \end{cases} \tag{21}$$

Table 1
List of parameters for the PART tree algorithm. (ρ_0 denotes the initial vigilance of the PART tree architecture and ρ_h denotes the vigilance increment)

Parameter	Permissible range	Sample value
L	$L > 1$	2
ρ_0	$1 \le \rho_0 \le m$	NA
ρ_h	$1 \le \rho_h \le m-1$	1
σ	$\sigma \ge 0$	NA
α	$0 \le \alpha \le 1$	NA
θ	$0 \le \theta < L/(L-1+m)$	0
e	$e \ge 0$	1

$$z_{ji}^{new} = (1-\alpha)z_{ji}^{old} + \alpha I_i, \tag{22}$$

where $0 \le \alpha \le 1$ is the learning rate, $|X|$ denotes the number of elements in the set $X = \{i{:}h_{ij} = 1\}$.

For a noncommitted winner v_j, and for every F_1 node v_i, we have

$$z_{ij}^{new} = L/(L-1+m), \tag{23}$$

$$z_{ji}^{new} = I_i. \tag{24}$$

Dimensions of projected clusters. Each committed F_2 node v_j represents a projected cluster C_j. The set D_j of the associated dimensions of the projected cluster C_j is determined by $l(z_{ij})$ according to the following formula

$$\text{The dimension } i \in D_j \text{ if and only if } l(z_{ij}) = 1. \tag{25}$$

Outlier node. Theoretically, the F_2 layer can have arbitrarily many nodes in PART architecture. Therefore, it can classify arbitrarily many input patterns. However, due to the restriction of resource, the number of nodes in the F_2 layer has to be limited. On the other hand, in many clustering problems, there are always some data points (called outliers) which do not cluster well. Therefore, we add a new special node in basic PART module, called outlier node. We simply put into the outlier node all data points that cannot be clustered into F_2 nodes $(v_{m+1}, v_{m+2}, \ldots, v_{m+n})$. The outlier node is treated as a sole container of outliers and it does not learn any new data pattern. Note that PART tree architecture consists of some basic PART modules. Therefore, there is an outlier node for each basic PART module in a PART tree. Although the outlier node in a given module does not learn any new data pattern, in a PART tree an outlier node inherits the pattern of its parent in the lower clustering level. Note also that the data points in an outlier node can be further classified in the next clustering level of a PART tree (see Part tree algorithm below).

While the aforementioned algorithms based on basic PART architecture are sufficient for many clustering problems, we believe that an algorithm based on PART tree architecture is particularly useful when users want to find more information about the hierarchical relations of projected clusters. For this purpose, we provide below an algorithm designed according to PART tree architecture. We should emphasize that this algorithm reduces to the basic PART architecture algorithm if we do not increase the vigilance ρ and if we do Steps 3–7 only one time.

PART tree algorithm.

```
0. Initialization:
   Number $m$ of nodes in $F_1$ layer: = number of dimensions in the input vector
   Number $n$ of nodes in $F_2$ layer: = expected maximum number of clusters that can be formed at each clustering level.
   Initialize parameters $L$, $\rho_0$, $\rho_h$, $\sigma$, $\alpha$, $\theta$, and $e$.
1. Set $\rho = \rho_0$.
2. Repeat Steps 3–7 until the stopping condition is satisfied.
3. Set all $F_2$ nodes as being noncommitted.
4. For each input vector in data set $S$, do Steps 4.1–4.6.
   4.1. Compute $h_{ij}$ for all $F_1$ nodes $v_i$ and committed $F_2$ nodes $v_j$. If all $F_2$ nodes are noncommitted, go to Step 4.3.
   4.2. Compute $T_j$ for all committed $F_2$ nodes $v_j$.
   4.3. Select the winning $F_2$ node $v_J$. If no $F_2$ node can be selected, put the input data into outlier $O$ and then continue to do Step 4.
   4.4. If the winner is a committed node, compute $r_J$, otherwise go to Step 4.6.
   4.5. If $r_J \ge \rho$, go to Step 4.6, otherwise reset the winner $v_J$ and go back to Step 4.3.
   4.6. Set the winner $v_J$ as the committed, and update the bottom-up and top-down weights for winner node $v_J$.
5. Repeat Step 4 $N$ times until stable clusters are formed (i.e. until the difference of output clusters at $N$-th and $(N-1)$-th time becomes sufficiently small).
6. For each cluster $C_j$ in $F_2$ layer, compute the associated dimension set $D_j$. Then, set $S = C_j$ and set $\rho = \rho + \rho_h$ (or $\rho = |D_j| + \rho_h$), go back to Step 2.
7. For the outlier $O$, set $S = O$, go back to Step 2.
```

Table 1 gives the permissible range and the suggested sample values of parameters in the above PART tree algorithm.

For large data sets, some random data points may be correlated in several dimensions. However, it is very unlikely that a large number of random data points are correlated in a large set of dimensions. Therefore, we should choose ρ_0 large enough to eliminate the randomness, but smaller than or equal to the number of dimensions of any

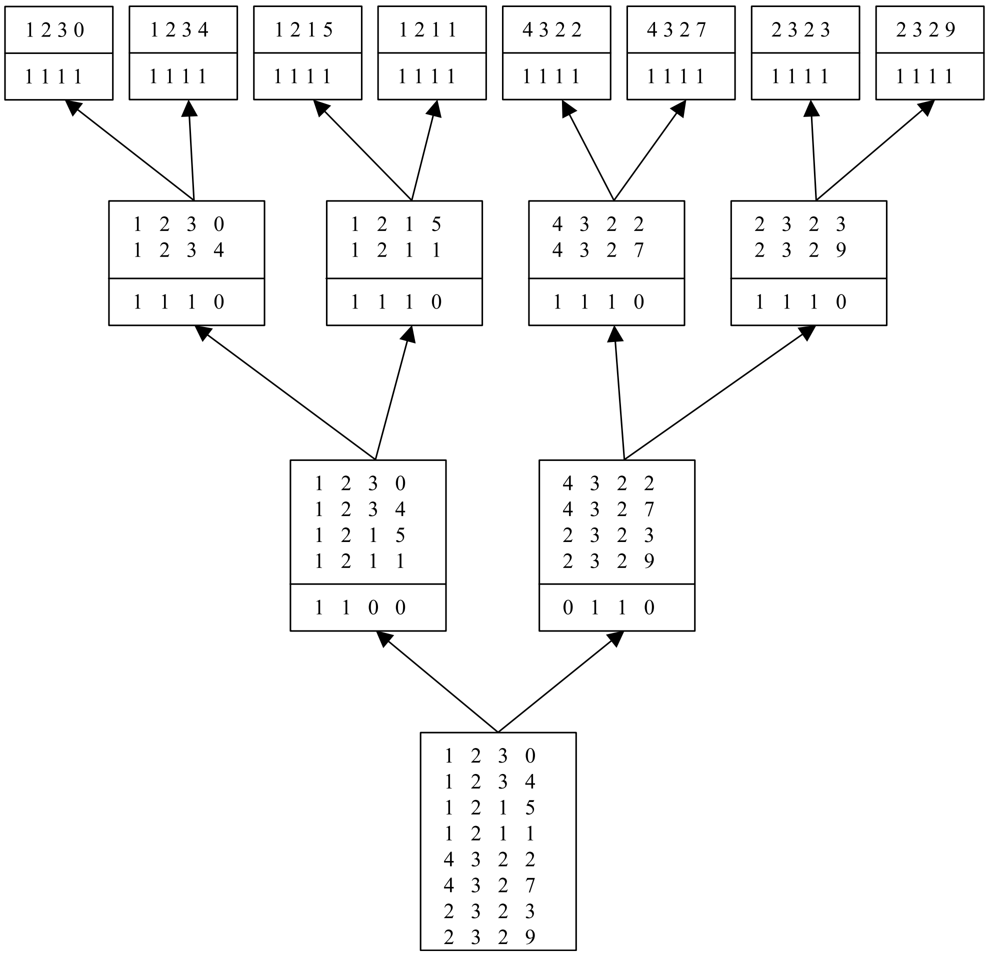

Fig. 4. Part simulation for a data set of eight patterns, where $\rho_0 = 1$, $\rho_h = 1$, $\sigma = 0.1$, $L = 2$, $e = 1$, $\alpha = 0.1$, $\theta = 0$. The data points below the delimitation line in each rectangular box represent the projected subspaces associated with each cluster. 1 denotes the corresponding dimension is in the associated subspace, and 0 denotes the corresponding dimension is not in the associated subspace.

possible projected cluster. As we do not know in advance the numbers of dimensions of projected clusters, we should choose ρ_0 as small as possible, but large enough to eliminate the randomness.

As we mentioned earlier, a natural stopping condition for the above PART tree algorithm is when $\rho > m$ or when no new subcluster can be formed. However, such a condition is usually too restrictive, and a practical stopping condition should be specified in terms of the particular applications. For example, one may choose to specify a threshold for the minimum number of data points in each cluster and stop classifying a cluster further if the number of its data points falls below this threshold. Users may also set the stopping condition as when the vigilance ρ becomes larger than a chosen threshold.

4. Simulations and comparisons

4.1. Two illustrative simulations

For the purpose of illustration, we designed a few small sets of data points which form clusters only in subspaces of lower dimensions. The simulation summarized in Fig. 4 illustrates how the PART tree algorithm classifies a data set with eight input patterns. Clearly, the PART tree algorithm gives a cluster tree which shows each projected cluster and the associated feature, as well as the hierarchical relations of these projected clusters.

Table 2 shows the clustering results by applying ART2-A (Carpenter, Grossberg & Rosen, 1991a) to the above data set. Table 3 shows the clustering results by applying ART2-A to the same data set, but with the pre-process of data by the Dayhoff method (Dayhoff, 1990; Fausett, 1994) so that the new data vectors have the same norms by adding an extra component to each data vector. Note that ART2 focuses on similarity of points in the full space and, thus, we failed to use the ART2-A algorithm to find the correct projected clusters in this particular data set where dimension similarity is the unique and essential pattern.

Table 4 shows the clustering results of applying PROCLUS (Aggarwal et al., 1999) to the above data set. Evidently, the clustering results depend on the choice of the number of clusters k and the average dimension l. In most

Table 2
Simulation by ART2-A for a data set of eight patterns, where vigilance $\rho = 0.8$, 0.9, 0.95, 0.99, respectively. The learning rate is 0.1, and the threshold $\theta = 0.05$. The data in the same cell are in a cluster

$\rho = 0.8$				$\rho = 0.9$				$\rho = 0.95$				$\rho = 0.99$			
1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0
1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	2	1	5	1	2	1	5	1	2	1	5	1	2	1	5
4	3	2	7	4	3	2	7	4	3	2	7	2	3	2	9
2	3	2	9	2	3	2	9	2	3	2	9	1	2	1	1
1	2	1	1	1	2	1	1	1	2	1	1	4	3	2	2
4	3	2	2	4	3	2	2	2	3	2	3	4	3	2	7
2	3	2	3	2	3	2	3	4	3	2	2	2	3	2	3

applications, finding the appropriate number of clusters and average dimension in advance imposes significant challenge for the user. Note that in PART there is no need to input these parameters.

Fig. 5 shows another PART simulation which classifies a data set with 12 input patterns.

4.2. Simulations on high dimensional data

This subsection describes our experimental results on high dimensional synthetic data, generated via the method introduced by Aggarwal et al. (1999). For the convenience of readers, we briefly describe this data generation method below and we refer to Aggarwal et al. (1999) for more details.

The data points, either cluster points or outliers, have coordinates in the range [0, 100]. Outliers amount to 5% and are distributed uniformly at random throughout the entire space. The algorithm proceeds by defining the so-called anchor points around which clusters will be distributed, as well as dimensions associated with each anchor point. The anchor points of clusters are obtained by generating k uniformly distributed points in a d-dimensional Euclidean space. The number of dimensions associated with a cluster is given by the realization of a Poisson random variable. The dimensions for each cluster are then chosen using an iterative technique, which is intended to model the fact that different clusters frequently share subsets of correlated dimensions. The number of points in each cluster is determined by first generating k exponential random variables with mean 1 and then assigning to each cluster a number of points proportional to these realizations. The data points for a given cluster i are generated as follows. The coordinates of the points on the noncluster dimensions are generated uniformly at random, the coordinates of the points projected onto a cluster dimension j follow a normal distribution with the mean at the respective coordinate of the anchor point, and with the variance given by $(s_{ij} \cdot r)^2$, where r is a fixed spread parameter and s_{ij} is a scale factor chosen from $[1, s]$ uniformly at random. We use $r = s = 2$ in our data generation.

We design two classes of the experiments. In the first class, we use input data files where all clusters are generated in the same number of dimensions, but in different subspaces. In the second class, we use input data files containing clusters generated in different number of dimensions. We report below our results for one experiment in each class, but we emphasize that similar results hold for other experiments we have carried out. In both experiments reported below, we use input data files with 10,000 data points and number of clusters $k = 5$. The first input file has data points in a 20-dimensional space, and all its five clusters have the same number of associated dimensions, seven, but in different subspaces. The second input file has data points in a 100-dimensional space, and its five clusters were respectively generated in 29, 23, 15, 26, 33-dimensional subspaces. Tables 5 and 6 show the

Table 3
Simulation by ART2-A with Dayhoff pre-process of data for a data set of eight patterns, where vigilance $\rho = 0.8$, 0.9, 0.95, 0.99, respectively. The learning rate is 0.1, and the threshold $\theta = 0.05$. The data in the same cell are in a cluster

$\rho = 0.8$					$\rho = 0.9$					$\rho = 0.95$					$\rho = 0.99$				
1	2	3	0	9.22	1	2	3	0	9.22	1	2	3	0	9.22	1	2	3	0	9.22
1	2	3	4	8.31	1	2	1	14	9.59	1	2	1	1	9.59	1	2	3	4	8.31
1	2	1	5	8.25	4	3	2	2	8.12	1	2	3	4	8.31	1	2	1	5	8.25
1	2	1	1	9.59	1	2	3	4	8.31	1	2	1	5	8.25	1	2	1	1	9.59
4	3	2	2	8.12	1	2	1	5	8.25	2	3	2	3	8.54	4	3	2	2	8.12
2	3	2	3	8.54	2	3	2	3	8.54	4	3	2	2	8.12	4	3	2	7	4.58
4	3	2	7	4.58	4	3	2	7	4.58	4	3	2	7	4.58	2	3	2	3	8.54
2	3	2	9	1.00	2	3	2	9	1.00	2	3	2	9	1.00	2	3	2	9	1.00

Table 4
Simulation by PROCLUS with different numbers of clusters k and average dimensions l for a data set of eight patterns. The points in the same cell are in a cluster, and the data points below the dotted delimitation line in each cell represent the projected subspaces associated with each cluster: 1 denotes the corresponding dimension is in the associated subspace, and 0 denotes the corresponding dimension is not in the associated subspace. Simulation indicates the dependence of clustering results on the choice of input parameters

$k = 2$ $l = 2$				$k = 3$ $l = 3$				$k = 4$ $l = 3$				$k = 5$ $l = 3.4$				$k = 5$ $l = 4$			
1	2	3	0	2	3	2	3	1	2	3	0	1	2	3	0	1	2	3	0
1	2	3	4	2	3	2	9	1	2	3	4	1	2	3	4	1	2	3	4
1	2	1	5	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	1
1	2	1	1	4	3	2	2	4	3	2	2	4	3	2	2	4	3	2	2
1	1	0	0	4	3	2	7	4	3	2	7	4	3	2	7	4	3	2	7
4	3	2	2	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	1
4	3	2	7	1	2	3	0	1	2	1	5	1	2	1	5	1	2	1	5
2	3	2	3	1	2	3	4	1	2	1	1	1	1	1	1	1	2	1	1
2	3	2	9	1	2	1	5	1	1	1	0	1	2	1	1	1	1	1	1
0	1	1	0	1	2	1	1	2	3	2	3	1	1	1	1	2	3	2	3
				1	1	1	0	2	3	2	9	2	3	2	3	1	1	1	1
								1	1	1	0	2	3	2	9	2	3	2	9
												1	1	1	0	1	1	1	1

associated dimensions and the number of data points of each cluster in the first and second input files, respectively.

In both experiments, the data points are presented in random order. In order to test the degree of dependence of our results on the vigilance parameter ρ and the distance parameter σ, we perform our simulations with different values of ρ_0 and σ. Note that we only use the basic PART algorithm in these simulations, namely we do not increase the vigilance ρ and do Steps 3–7 of the PART tree algorithm only one time. Small clusters are treated as outliers because they usually consist of some random points and outliers. We report the results and comparisons, with emphasis on four different aspects: number of clusters found; dimensions found; centers of clusters found; the contingency table (see Hubert & Arabie, 1985) of original clusters (also called input clusters) and clusters found (also called output clusters).

Table 5
Dimensions and numbers of data points of input clusters for a data set in a 20-dimensional space (the first data file)

Input	Dimensions	Points
1	3, 4, 7, 9, 14, 16, 17	2139
2	3, 4, 7, 12, 13, 14, 17	2328
3	4, 6, 11, 13, 14, 17, 19	1824
4	4, 7, 9, 13, 14, 16, 17	1573
5	3, 4, 9, 12, 14, 16, 17	1636
Outliers	–	500

We first consider the experiment on the first input data file. Tables 7 and 8 and Fig. 6 show the simulation results with $\rho_0 = 4$ and $\sigma = 0.18$; Tables 9 and 10 and Fig. 7 show the simulation results with $\rho_0 = 5$ and $\sigma = 0.17$; and Tables 11 and 12 and Fig. 8 show the simulation results with $\rho_0 = 5$ and $\sigma = 0.26$. Note that our PART algorithm succeeds in finding the exact number of original clusters and in finding almost exact centers of all original clusters with negligible errors. The dimensions found are not identical to those of the original clusters, but these found dimensions are contained as subsets of the corresponding dimensions of the original clusters. These subsets are sufficiently large so

Table 6
Dimensions and numbers of data points of input clusters for a data set in a 100-dimensional space (the second data file)

Input	Dimensions	Points
1	13, 16, 21, 22, 32, 35, 36, 39, 41, 42, 47, 52, 54, 59, 61, 63, 65, 68, 74, 76, 77, 79, 83, 87, 92, 94, 96, 98, 99	2139
2	1, 2, 5, 8, 10, 12, 13, 18, 22, 23, 24, 26, 27, 28, 36, 39, 41, 43, 48, 50, 57, 58, 88	2328
3	3, 4, 6, 10, 13, 17, 18, 23, 30, 31, 35, 40, 66, 68, 89	1824
4	2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 25, 39, 46, 48, 61, 62, 75, 79, 80, 95, 99	1573
5	3, 4, 5, 8, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 26, 30, 31, 36, 38, 42, 45, 48, 49, 56, 57, 61, 62, 65, 84, 85, 88, 98	1636
Outliers	–	500

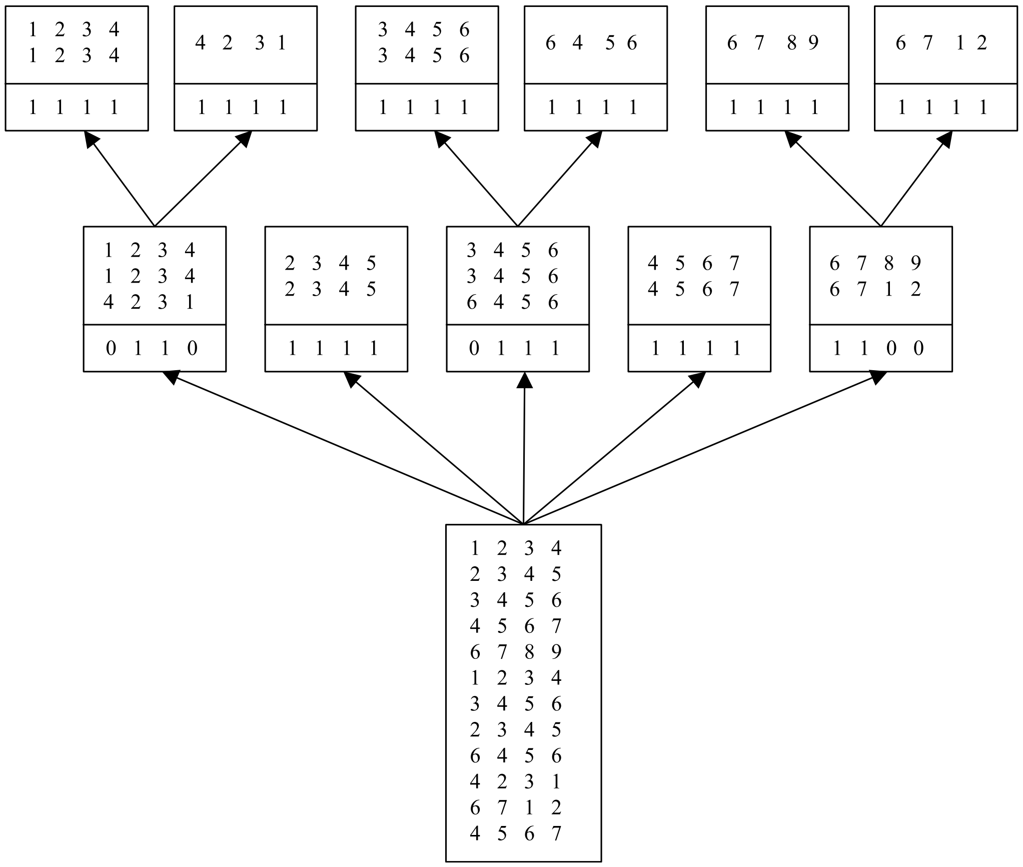

Fig. 5. PART simulation for a data set of 12 patterns, where $\rho_0 = 1$, $\rho_h = 1$, $\sigma = 0.1$, $L = 2$, $e = 1$, $\alpha = 0.1$, $\theta = 0$. The data points below the delimitation line in each rectangular box represent the projected subspaces associated with each cluster. 1 denotes the corresponding dimension is in the associated subspace, and 0 denotes the corresponding dimension is not in the associated subspace.

that, after further reassignment described below, we are able to reproduce the original clusters from the found cluster centers, the found number of clusters and the found dimensions.

Note that when $\rho_0 = 5$ and $\sigma = 0.26$, the output has a relatively large percentage of outliers. However, the found centers coincide with those of the original clusters. Therefore, we can reassign all data points according to their distances to the centers in the subspaces of the dimensions found. Table 13 shows the result of the reassignment. Note that the original input outliers are also reassigned using the same rule. Results of reassignment are similar in other two choices of parameters: ($\rho_0 = 4$, $\sigma = 0.18$) and ($\rho_0 = 5$, $\sigma = 0.17$). It is natural that the set of output outliers becomes large if σ is small. One may wonder, however, why the size of the set of output outliers increases when σ is increased from 0.17 to 0.26, while $\rho_0 = 5$. Our experiments show that when σ is relatively large, some data points can be incorrectly classified to a cluster during a trial, and this changes the templates of the corresponding F_2 node and, therefore, changes the results of subsequent trials.

We also carried out simulations with other values of parameters ρ_0 and σ, and we observed that the number of

Table 7
Dimensions and numbers of the data points of output clusters for the first data file when $\rho_0 = 4$ and $\sigma = 0.18$

Found	Dimensions	Points
1	4, 7, 9, 17	2089
2	3, 12, 13, 14	2315
3	6, 11, 13, 14	1809
4	7, 9, 14, 17	1473
5	9, 12, 14, 16	1576
Outliers	–	738

Table 8
Contingency table of input clusters and output clusters for the first data file when $\rho_0 = 4$, $\sigma = 0.18$. Entry (i, j) denotes the number of data points that are common to output cluster i and input cluster j

Output/input	1	2	3	4	5	Outliers	Sums
1	2089	0	0	0	0	0	2089
2	0	2315	0	0	0	0	2315
3	0	0	1808	0	0	1	1809
4	0	0	0	1473	0	0	1473
5	0	0	0	0	1575	1	1576
Outliers	50	13	16	100	61	498	738
Sums	2139	2328	1824	1573	1636	500	10,000

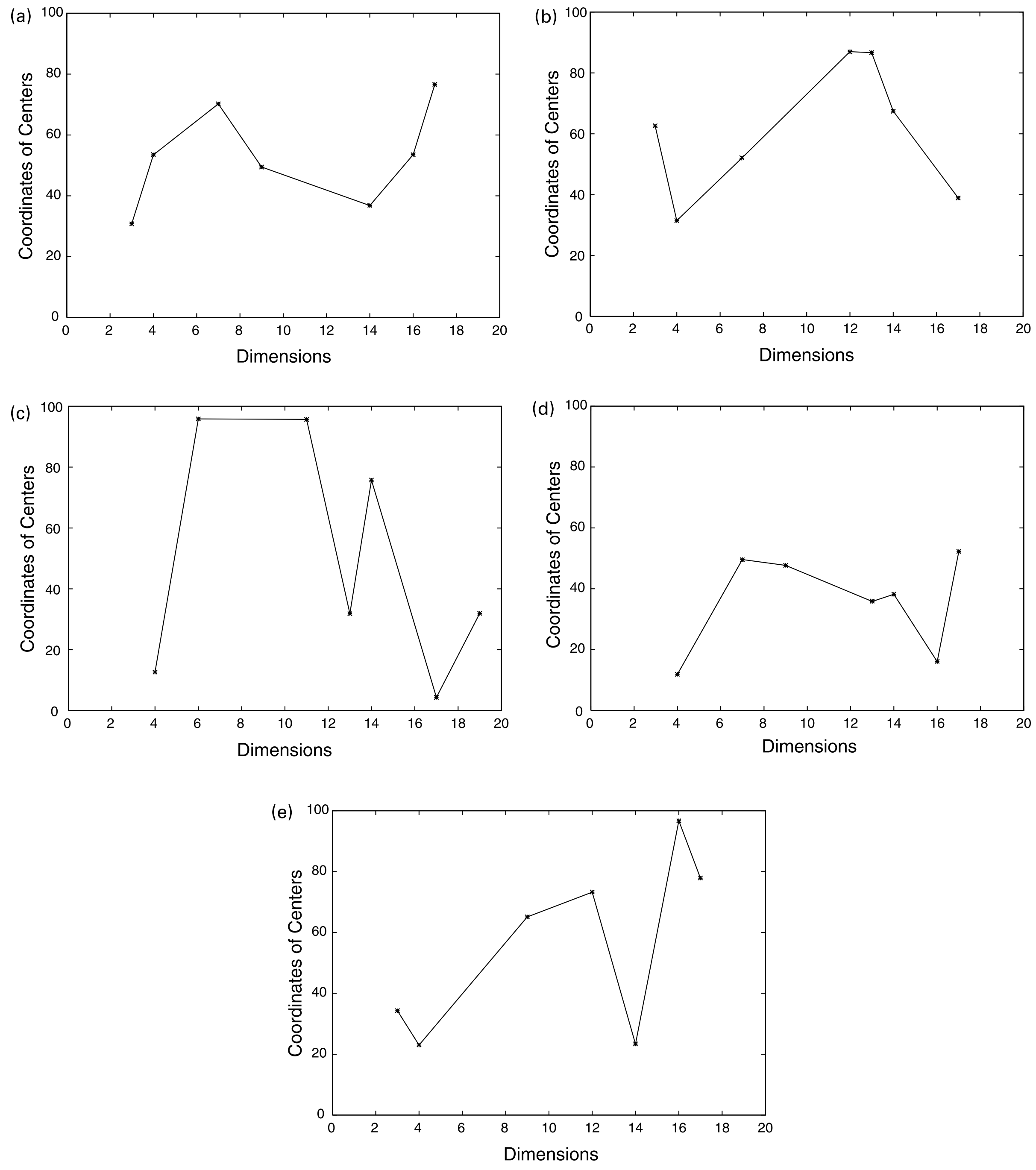

Fig. 6. Comparison of centers of output clusters with original clusters in associated dimensions for the first data file when $\rho_0 = 4$ and $\sigma = 0.18$. + denotes the coordinate of the center of an original cluster, and × denotes the coordinate of the center of an output cluster, in the corresponding dimension.

Table 9
Dimensions and numbers of data points of output clusters for the first data file when $\rho_0 = 5$ and $\sigma = 0.17$

Found	Dimensions	Points
1	4,7,9,16,17	2130
2	3, 4, 12, 13, 14	2231
3	6, 11, 13, 14, 19	1422
4	7, 9, 13, 14, 17	1273
5	3, 9, 12, 16, 17	1618
Outliers	–	1326

clusters and the center of each cluster can be correctly found with various (ρ_0, σ) in a wide range. Moreover, we found that the final clustering results, after reassignment, are all identical and coincide with the original input clusters, except some differences in the assignment of the original input outliers. Therefore, we believe that PART algorithm is quite robust to the choice of parameters ρ_0 and σ.

We obtained similar observations from our simulations on the second input data file. Tables 14 and 15 and Fig. 9 show the simulation results with $\rho_0 = 10$ and $\sigma = 0.16$, and

Table 10
Contingency table of input clusters and output clusters for the first data file when $\rho_0 = 5$, $\sigma = 0.17$. Entry (i, j) denotes the number of data points that are common to output cluster i and input cluster j

Output/input	1	2	3	4	5	Outliers	Sums
1	2130	0	0	0	0	0	2130
2	0	2231	0	0	0	0	2231
3	0	0	1422	0	0	0	1422
4	0	0	0	1273	0	0	1273
5	0	0	0	0	1618	0	1618
Outliers	9	97	402	300	18	500	1326
Sums	2139	2328	1824	1573	1636	500	10,000

Table 11
Dimensions and numbers of data points of output clusters for the first data file when $\rho_0 = 5$ and $\sigma = 0.26$

Found	Dimensions	Points
1	4,7,14,16,17	1587
2	3, 7, 12, 13, 17	2296
3	4, 6, 11, 13, 14	1391
4	7, 9, 13, 14, 17	1494
5	3, 9, 12, 16, 17	1368
Outliers	–	1864

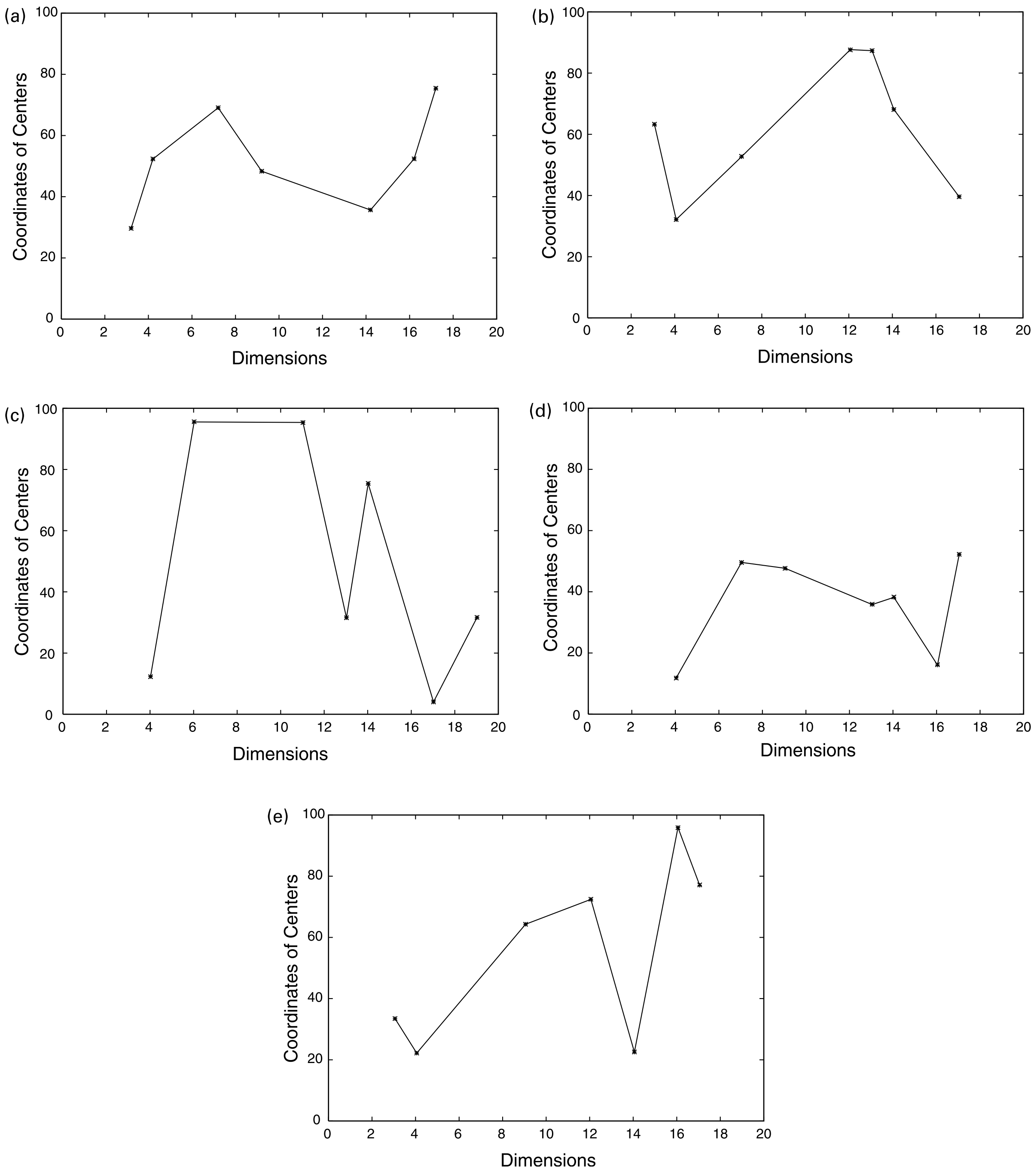

Fig. 7. Comparison of centers of output clusters with original clusters in associated dimensions for the first data file when $\rho_0 = 5$ and $\sigma = 0.17$. + denotes the coordinate of the center of an original cluster, and $\times$ denotes the coordinate of the center of an output cluster, in the corresponding dimension.

Table 12
Contingency table of input clusters and output clusters for the first data file when $\rho_0 = 5$, $\sigma = 0.26$. Entry (i, j) denotes the number of data points that are common to output cluster i and input cluster j

Output/input	1	2	3	4	5	Outliers	Sums
1	1587	0	0	0	0	0	1587
2	0	2296	0	0	0	0	2296
3	0	0	1389	0	0	2	1391
4	0	0	0	1494	0	0	1494
5	0	0	0	0	1368	0	1368
Outliers	552	32	435	79	268	498	1864
Sums	2139	2328	1824	1573	1636	500	10,000

Tables 16 and 17 and Fig. 10 show the simulation results with $\rho_0 = 8$ and $\sigma = 0.15$.

Tables 18 and 19 show the first five largest output clusters by using Fuzzy ART algorithm (Carpenter, Grossberg & Rosen, 1991b) for the first data file with vigilance $\rho = 0.2$ and 0.3, respectively. Clearly, there are heavy overlaps among different clusters. Table 20 shows the first six largest output clusters by using Fuzzy ART algorithm for the first data file with vigilance $\rho = 0.4$. The clustering results are reasonably good. However, the number of data points in each cluster is small and, hence, the set of outliers is

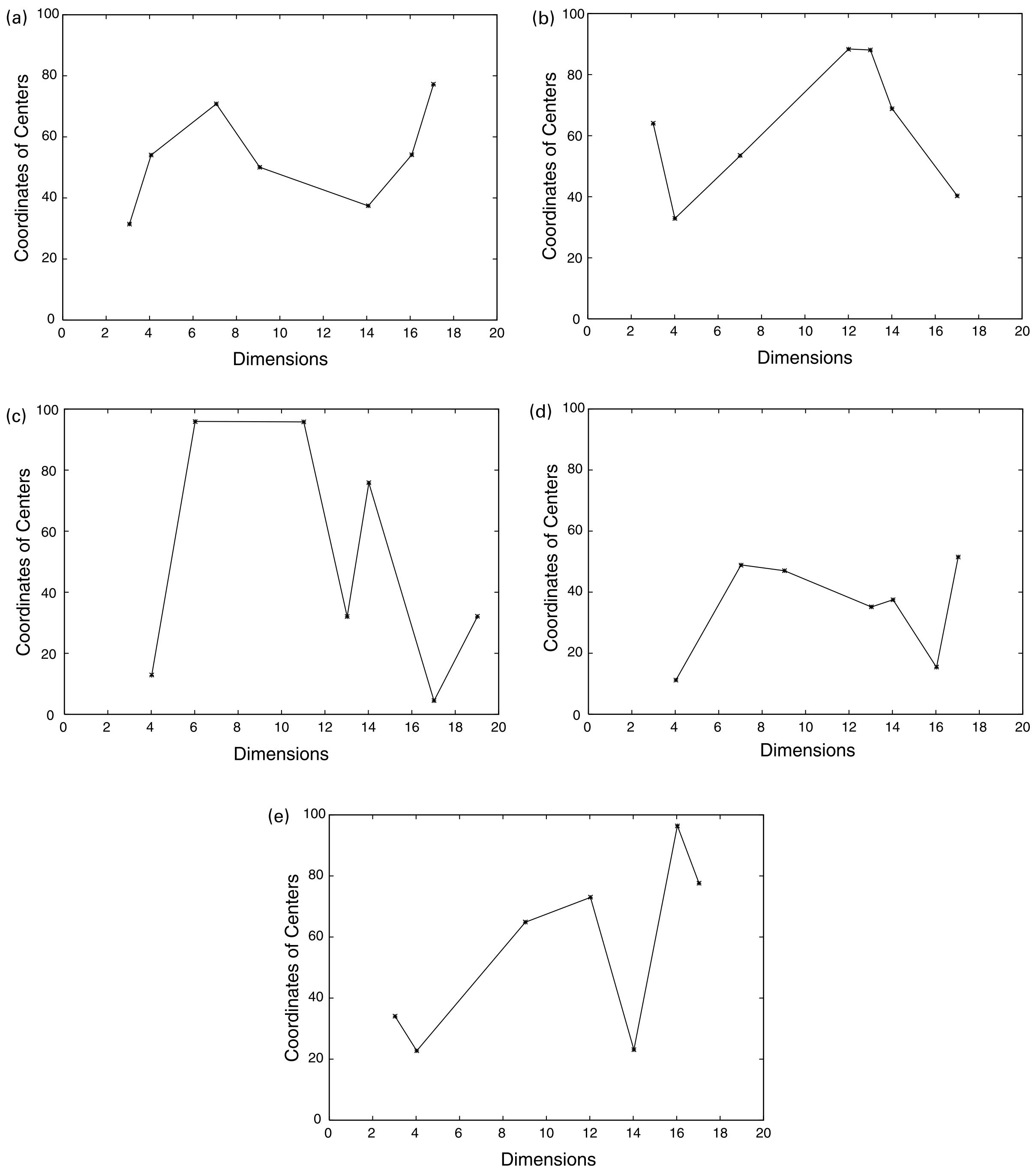

Fig. 8. Comparison of centers of output clusters with original clusters in associated dimensions for the first data file when $\rho_0 = 5$ and $\sigma = 0.26$. + denotes the coordinate of the center of an original cluster, and × denotes the coordinate of the center of an output cluster, in the corresponding dimension.

Table 13
Reassignment according to the found centers of output clusters and the found dimensions for the first data file when $\rho_0 = 5$, $\sigma = 0.26$

Output/input	1	2	3	4	5	Outliers	Sums
1	2139	0	0	0	0	120	2259
2	0	2328	0	0	0	86	2414
3	0	0	1824	0	0	50	1874
4	0	0	0	1573	0	181	1754
5	0	0	0	0	1636	63	1699
Sums	2139	2328	1824	1573	1636	500	10,000

huge. Note also that the first five largest output clusters correspond to only four out of five original input clusters. More precisely, the input cluster 2 is divided into two clusters in the first five largest output clusters. When the vigilance parameter $\rho \geq 0.5$, each output cluster generated by Fuzzy ART algorithm consists of a small number of data points and, hence, no real cluster is formed. For the second input data file, we are unable to find the proper vigilance parameter ρ to make Fuzzy ART work. For example, when $\rho = 0.4$, 0.3 and 0.2, each output cluster consists of very few data points, and when $\rho = 0.1$ the original whole data set is approximately divided into two large clusters. Again, we emphasize that Fuzzy ART and other ART modules have been proved to be very effective in clustering data sets based on similarity of points in the full space. Fuzzy ART does not generate satisfactory clustering results for the data sets here because, for these data sets, dimension similarity is an essential feature.

5. Conclusions

Most clustering algorithms do not work efficiently for data sets in high dimensional spaces. Due to the inherent sparsity of data points, it is not feasible to find interesting clusters in the original full space of all dimensions, but pruning off dimensions in advance, as most feature selection procedures do, may lead to significant loss of information and, thus, render the clustering results unreliable.

We propose a new neural network architecture Projective Adaptive Resonance Theory (PART) in order to provide a solution to this feasibility–reliability dilemma in clustering data sets in high dimensional spaces. The goal of PART is to find projected clusters, each of which consists of a subset C of data points together with a subset D of dimensions such that the points in C are closely correlated in the subspace of dimensions D. This idea of projected clustering and an algorithm PROCLUS were proposed by Aggarwal et al. (1999). Unfortunately, the PROCLUS algorithm seems to be sensitive to the choice of two input parameters: the number of clusters and the average dimension, and selecting these parameters in advance imposes real challenge for the user. This is illustrated by our simulations, reported in Section 4, on even some small data sets.

The basic architecture of the proposed PART is similar to that of ART neural networks which have been shown to be very effective in self-organizing stable recognition codes in real time in response to arbitrary sequences of input patterns. However, ART focuses on similarity of patterns in the full space of all dimensions and, thus, may fail to find patterns in the data sets in high dimensional spaces where dimensional similarity is an essential part of the patterns. This is illustrated in our simulations reported in Section 4. The main new development in PART is the introduction of a selective output signaling mechanism which allows a generated signal at a node in the input layer to be transmitted to a node in the clustering layer only when the corresponding top-down weight is similar to the generated signal. Like ART, the vigilance conditions in PART control the degree of similarity of patterns in the same cluster, but the similarity measurement in PART is closely related to the subspaces involved. In particular, the degree of similarity of patterns

Table 14
Dimensions and numbers of the data points of output clusters for the second data file when $\rho_0 = 10$ and $\sigma = 0.16$

Found	Dimensions	Points
1	13, 41, 65, 68, 74, 76, 83, 94, 98, 99	2006
2	1, 12, 13, 18, 23, 27, 39, 48, 58, 88	2325
3	3, 4, 6, 10, 17, 18, 23, 30, 31, 35	1804
4	2, 5, 15, 16, 39, 46, 61, 75, 79, 80	1265
5	8, 12, 16, 21, 30, 45, 62, 84, 85, 88	1628
Outliers	–	972

Table 15
Contingency table of input clusters and output clusters for the second data file when $\rho_0 = 10$, $\sigma = 0.16$. Entry (i, j) denotes the number of data points that are common to output cluster i and input cluster j

Output/input	1	2	3	4	5	Outliers	Sums
1	2006	0	0	0	0	0	2006
2	0	2325	0	0	0	0	2325
3	0	0	1804	0	0	0	1804
4	0	0	0	1265	0	0	1265
5	0	0	0	0	1628	0	1628
Outliers	133	3	20	308	8	500	972
Sums	2139	2328	1824	1573	1636	500	10,000

Table 16
Dimensions and numbers of the data points of output clusters for the second data file when $\rho_0 = 8$ and $\sigma = 0.15$

Found	Dimensions	Points
1	13, 65, 68, 74, 76, 83,94, 96	2114
2	10, 13, 18, 22, 39, 48, 57, 88	2101
3	3, 4, 6, 10, 18, 30, 31, 35	1801
4	2, 5, 6, 10, 11, 14, 39, 80	1501
5	19, 20, 26, 31, 48, 61, 62, 65	1533
Outliers	–	950

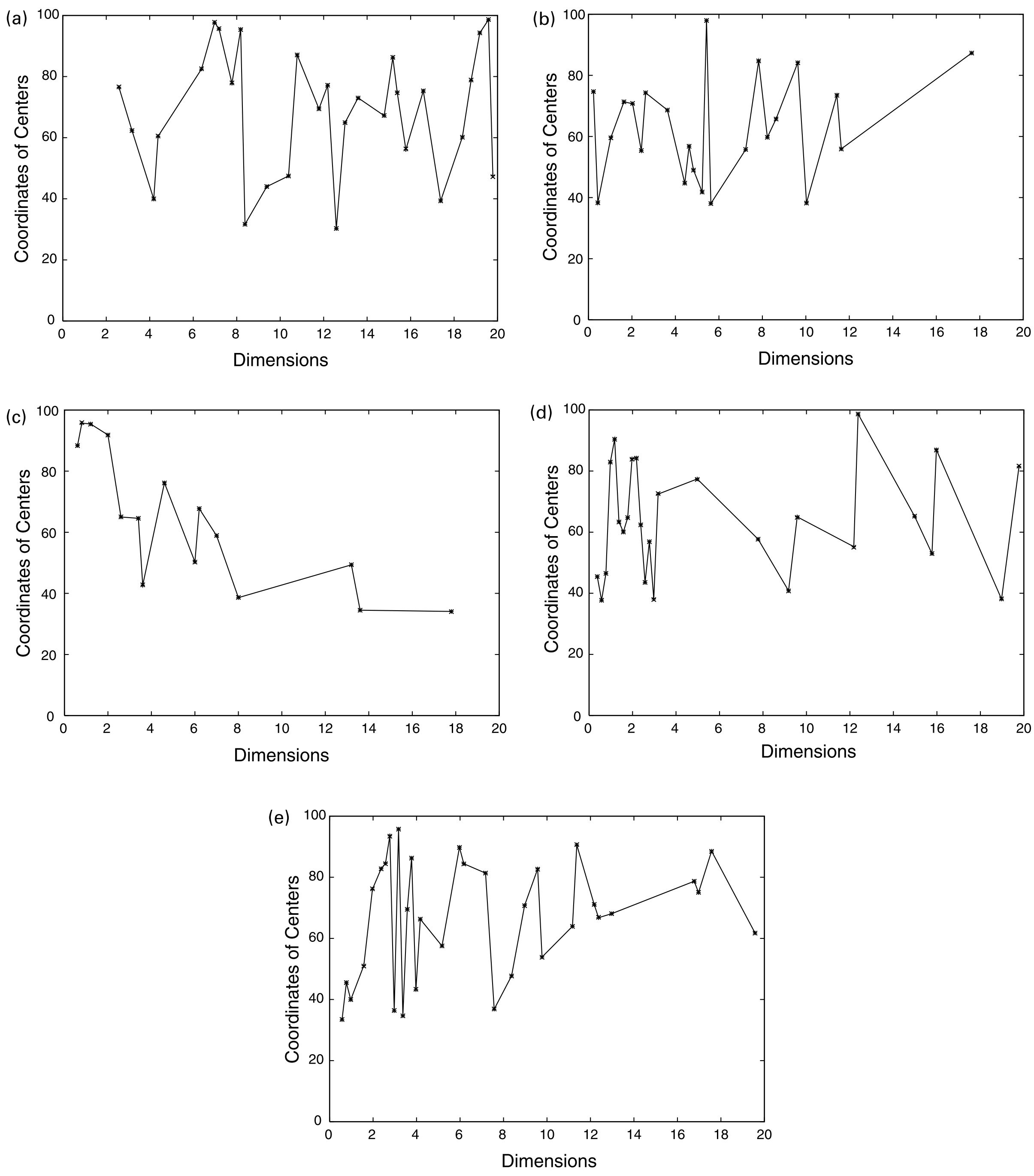

Fig. 9. Comparison of centers of output clusters with original clusters in associated dimensions for the first data file when $\rho_0 = 10$ and $\sigma = 0.16$. + denotes the coordinate of the center of an original cluster, and × denotes the coordinate of the center of an output cluster, in the corresponding dimension.

Table 17
Contingency table of input clusters and output clusters for the second data file when $\rho_0 = 8$, $\sigma = 0.15$. Entry (i, j) denotes the number of data points that are common to output cluster i and input cluster j

Output/input	1	2	3	4	5	Outliers	Sums
1	2114	0	0	0	0	0	2114
2	0	2081	0	0	20	0	2101
3	0	0	1801	0	0	0	1801
4	0	0	0	1501	0	0	1501
5	0	0	0	0	1533	0	1533
Outliers	25	247	23	72	83	500	950
Sums	2139	2328	1824	1573	1636	500	10,000

Table 18
Fuzzy ART simulation for the first data file, where the choice parameter $\alpha = 0.1$, the learning rate $\beta = 0.1$, and vigilance parameter $\rho = 0.2$. Entry (i, j) denotes the number of data points that are common to output cluster i and input cluster j

Output/input	1	2	3	4	5	Outliers	Sums
1	894	940	365	575	455	84	3313
2	333	360	366	170	328	25	1582
3	111	199	163	71	0	9	553
4	44	72	126	118	45	7	412
5	74	92	47	50	96	9	368

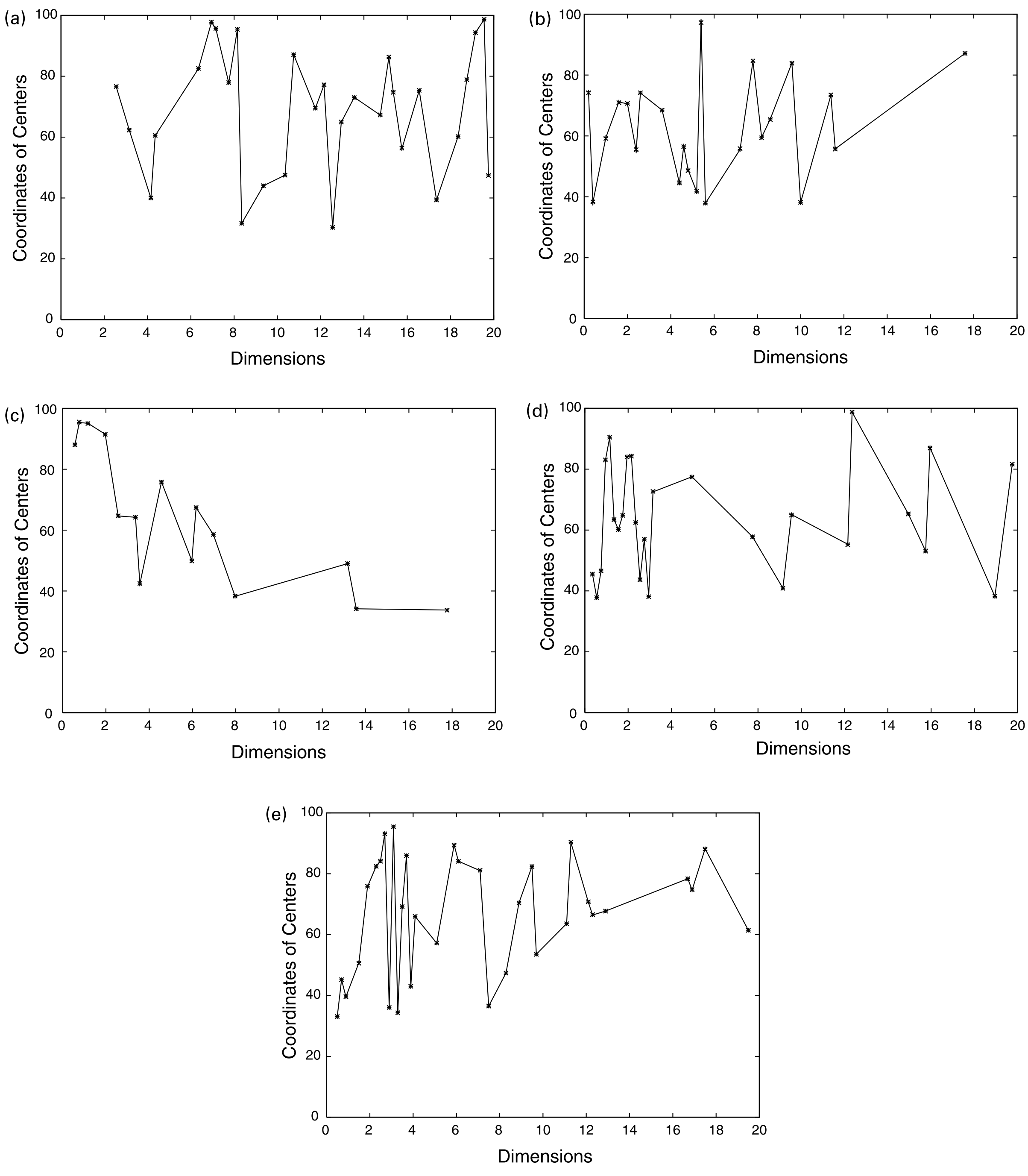

Fig. 10. Comparison of centers of output clusters with original clusters in associated dimensions for the second data file when $\rho_0 = 8$ and $\sigma = 0.15$. $+$ denotes the coordinate of the center of an original cluster, and $\times$ denotes the coordinate of the center of an output cluster, in the corresponding dimension.

Table 19
Fuzzy ART simulation for the first data file, where the choice parameter $\alpha = 0.1$, the learning rate $\beta = 0.1$, and vigilance parameter $\rho = 0.3$. Entry (i, j) denotes the number of data points that are common to output cluster i and input cluster j

Output/input	1	2	3	4	5	Outliers	Sums
1	0	0	807	47	0	0	854
2	342	207	14	164	52	5	784
3	6	718	0	11	0	0	735
4	0	0	0	348	260	0	608
5	212	26	0	0	0	0	238

for a committed node is controlled by both vigilance parameter and distance parameter which control the size of dimensions of the projected subspaces and the degree of similarity in a specific dimension involved, respectively.

These vigilance and distance parameters are the only required input parameters for the PART algorithm, and our simulations on high dimensional synthetic data show that the clustering results obtained by the PART algorithm are very robust to the choice of these input parameters. In particular, we observed that the PART algorithm, with a

Table 20
Fuzzy ART simulation for the first data file, where the choice parameter $\alpha = 0.1$, the learning rate $\beta = 0.1$, and vigilance parameter $\rho = 0.4$. Entry (i, j) denotes the number of data points that are common to output cluster i and input cluster j. Note that the number of data points in each of the first six largest output clusters is already small. The remaining clusters are all so small that all their points should be classified as outliers. This generates a huge set of outliers

Output/input	1	2	3	4	5	Outliers	Sums
1	0	0	950	0	0	0	950
2	0	544	0	0	0	0	544
3	0	0	0	0	540	0	540
4	0	0	0	509	0	0	509
5	0	460	0	0	0	0	460
6	412	0	0	0	0	0	412

wide range of input parameters, enables us to find the correct number of clusters, the correct centers of the clusters and the sufficiently large subsets of dimensions where clusters are formed, so that we are able to fully reproduce the original input clusters after a reassignment procedure.

Acknowledgements

We would like to thank Professor S. Grossberg, two anonymous referees and an anonymous action editor for their invaluable comments and suggestions about the preliminary version of this work. Their comments and suggestions inspired some additional simulations on high dimensional synthetic data sets, and additional comparisons with existing clustering methods and algorithms. This research was partially supported by Natural Sciences and Engineering Research Council of Canada, by Network of Centers of Excellence Program MITACS (Mathematics for Information Technology and Complex Systems), and by Canada Research Chairs Program.

References

Aggarwal, C. C., Procopiuc, C., Wolf, J. L., Yu, P. S., & Park, J. S. (1999). Fast algorithms for projected clustering. In *SIGMOD '99* (pp. 61–72).

Carpenter, G. A., & Grossberg, S. (1987a). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, *37*, 54–115.

Carpenter, G. A., & Grossberg, S. (1987b). ART2: self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, *26*, 4919–4930.

Carpenter, G. A., & Grossberg, S. (1990). ART3: hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks*, *3*, 129–152.

Carpenter, G. A., Grossberg, S., & Reynolds, J. H. (1991). ARTMAP: supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, *4*, 565–588.

Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991a). ART2-A: an adaptive resonance algorithm for rapid category learning and recognition. *Neural Networks*, *4*, 493–504.

Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991b). Fuzzy ART: fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, *4*, 759–771.

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., & Rosen, D. B. (1992). Fuzzy ARTMAP: a neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, *3*, 698–713.

Dayhoff, J. E. (1990). *Neural network architectures: an introduction*, New York: Van Nostrand Reinhold.

Fausett, L. (1994). *Fundamentals of neural networks: architectures, algorithms, and applications*, Englewood Cliffs, NJ: Prentice-Hall.

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*, 193–218.