Volume 28, issue 9 1 July 2007 ISSN 0167-8655

ELSEVIER

Pattern Recognition
Letters

An official publication of the
International Association for Pattern Recognition

IAPR

Available online at www.sciencedirect.com
ScienceDirect

Pattern Recognition Letters 28 (2007) 1062–1076

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Dynamic agglomerative clustering of gene expression profiles

Faming Liang *, Naisyin Wang

Department of Statistics, Texas A&M University, College Station, TX 77843-3143, USA

Received 18 May 2005; received in revised form 19 December 2006
Available online 3 February 2007

Communicated by A. Fred

Abstract

The increasing use of microarray technologies is generating a large amount of data that must be processed to extract underlying gene expression patterns. Existing clustering methods could suffer from certain drawbacks. Most methods cannot automatically separate scattered, singleton and mini-cluster genes from other genes. Inclusion of these types of genes into regular clustering processes can impede identification of gene expression patterns. In this paper, we propose a general clustering method, namely a dynamic agglomerative clustering (DAC) method. DAC can automatically separate scattered, singleton and mini-cluster genes from other genes and thus avoid possible contamination to the gene expression patterns caused by them. For DAC, the scattered gene filtering step is no longer necessary in data pre-processing. In addition, we propose a criterion for evaluating clustering results for a dataset which contains scattered, singleton and/or mini-cluster genes. DAC has been applied successfully to two real datasets for identification of gene expression patterns. Our numerical results indicate that DAC outperforms other clustering methods, such as the quality-based and model-based clustering methods, in clustering datasets which contain scattered, singleton and/or mini-cluster genes.

Keywords: Dynamic agglomerative clustering; Gene expression profile; Mini-cluster gene; Scattered gene; Silhouette width; Singleton gene

1. Introduction

DNA microarray technology has made it possible to examine the expression of many genes over multiple developmental stages or different experimental conditions. One of the important tasks is to identify groups of genes with similar expression patterns (co-expressed genes) from messy DNA microarray data. Microarray experiments involve many scattered genes whose expression levels do not change much across samples. The scattered genes provide no or little information to the underlying biological processes, and show low correlation to the expression patterns of non-scattered genes. If the scattered genes are forced into a cluster, the average profile of this cluster can be compromised and the composition of this cluster might provide less information for future analyses. We illustrate this phenomenon by Fig. 1, where the genes are from four populations represented by the symbols “1”, “2”, “3”, and “.”, respectively. The scattered genes, which are represented by “.”, share no common patterns with any of the other three populations. Inclusion of the scattered genes means that many of the partition-based clustering methods, such as the K-means (Tavazoie et al., 1999; Tou and Gonzalez, 1979), hierarchical clustering (Carr et al., 1997; Eisen et al., 1998) and self-organizing maps (SOM) (Tamayo et al., 1999), fail to identify true expression patterns of the non-scattered genes. For this example, these methods typically classify the genes into three clusters separated by the dotted lines. The average profile of each cluster is corrupted by the scattered genes. In addition to the scattered genes, the microarray experiments involve some singleton and mini-cluster genes. A gene is called a singleton if its expression pattern is different from the expression patterns of any other genes. Mini-cluster genes refer to the genes which belong to a very small cluster.

[*] Corresponding author. Tel.: +1 979 8453197; fax: +1 979 8453144.
E-mail address: fliang@stat.tamu.edu (F. Liang).

0167-8655/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2007.01.009

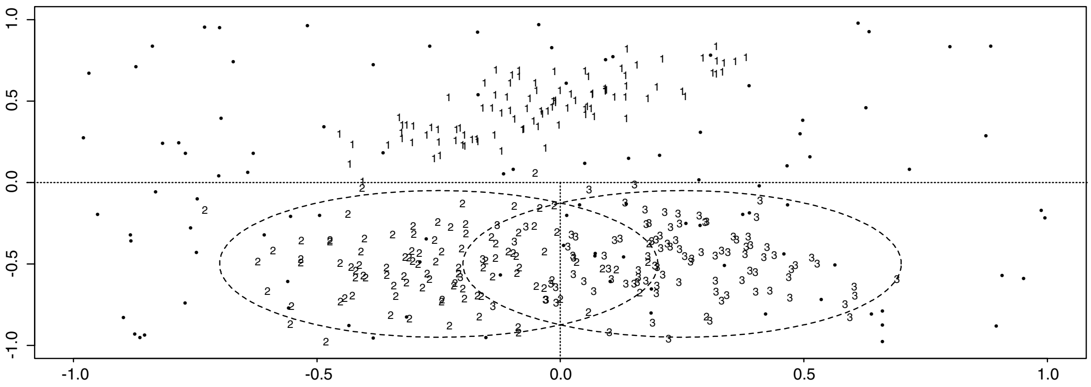

Fig. 1. Illustrative graph for gene expression data. The scattered genes are represented by the symbol ".". The non-scattered genes belong to three clusters represented by the symbols 1, 2 and 3, respectively.

If the singleton or mini-cluster genes are forced into a cluster, the expression pattern represented by this cluster can be contaminated and consequently, be difficult to interpret. In what follows, we call the scattered, singleton and mini-cluster genes the noise genes for simplicity.

A common strategy to deal with scattered genes is to filter them away through a variation filter. The filter is usually set in a way such that only the genes bearing sufficient variation across different treatments will be kept for further analyses. For example, Tamayo et al. (1999) set a filter for the yeast expression data (Cho et al., 1998) to eliminate the genes for which the relative expression change is less than 2 or the absolute expression change is less than 35 units across the samples. It turns out that only about 13 percent (828 out of 6218) of the genes pass the filter. This will inevitably cause some loss of data information. The variation filter can remove genes which do not change sufficiently with different experimental conditions but still follow a certain pattern shared by other genes in a cluster. Although the variation filter can remove scattered genes, in our experience, it does not remove singleton and mini-cluster genes.

To accommodate noise genes, several sequential clustering methods have been proposed in the recent literature, including quality-based clustering (Heyer et al., 1999), adaptive quality-based clustering (AQC) (De Smet et al., 2002), CAST (Ben-Dor et al., 1999), gene shaving (Hastie et al., 2000), CLICK (Sharan and Shamir, 2000), HCS (Hartuv et al., 2000), and tight clustering (Tseng and Wong, 2005). Refer to Shamir and Sharan (2002), Jiang et al. (2004), and Tseng (2005) for overviews of these methods. Taking AQC as an example, it first searches for a cluster center, and then groups the genes around the center into a cluster. Once a cluster is formed, it will be removed from the dataset and the process will be restarted for the remaining genes. This process often results in a sub-optimal separation for the overlapped clusters. This phenomenon can be illustrated by Fig. 1, where clusters 2 and 3 are overlapped. Sequential removal of either cluster 2 or cluster 3 will lead to a sub-optimal separation between them. AQC contains two parameters, minimum cluster size and test significance level. However, no clear criterion is established for choosing the parameters in AQC. Overall, neither of the aforementioned methods have a default option to avoid potential problems caused by singleton and mini-cluster genes.

A clustering method closely related to the sequential clustering methods is the density-based hierarchical clustering (DHC) method (Jiang et al., 2003). DHC considers a cluster as a high-dimensional "dense" area, where genes are "attracted" with each other. At the "core" part of the dense area, genes are crowded closely and, thus, have high density. DHC constructs a hierarchical tree to organize the "dense" and "core" areas. DHC potentially works for a dataset with noise genes by treating the genes lying outside the "core" areas as noise genes. However, to construct the clustering tree, the computational complexity of this step is O(n^2), where n is the number of genes in the dataset. This makes DHC inefficient. Furthermore, DHC requires two global parameters controlling the splitting process of dense areas. Like AQC, DHC lacks a clear criterion for determining its parameters. Because the software is not directly available to public, DHC is not included in evaluations in this paper.

Besides DHC and the aforementioned sequential clustering methods, the model-based clustering methods also contain options that accommodate noise. In these methods, the data are typically modeled by a Gaussian mixture distribution with the noise being handled by adding a term to the mixture. In the context of gene expression profile clustering, the noise refers to the noise genes. By assuming that the noise observations are uniformly distributed in the data region, Banfield and Raftery (1992), Dasgupta and Raftery (1998), Campbell et al. (1997, 1999), McLachlan and Peel (2000), and Fraley and Raftery (2002) modeled the data using the following mixture distribution,

$$f(\boldsymbol{x}|\boldsymbol{\theta}) = \frac{\tau_0}{V} + \sum_{k=1}^{G} \tau_k \phi_k(x_i|\boldsymbol{\theta}_k), \tag{1}$$

where V is the hypervolume of the data region, G is the number of clusters, $\tau_k \geqslant 0$, $\sum_{k=0}^{G}\tau_k = 1$, $\boldsymbol{\theta}_k = (\mu_k, \Sigma_k)$

contains the parameters of component k, $\boldsymbol{\theta} = (\tau_0, \ldots, \tau_G, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_G)$ contains all parameters of the model, and ϕ_k is a multivariate Gaussian density, i.e.,

$$\phi_k(\boldsymbol{x}|\boldsymbol{\theta}_k) = (2\pi)^{-\frac{p}{2}}|\Sigma_k|^{-\frac{1}{2}}\exp\left\{-\frac{1}{2}(\boldsymbol{x}-\mu_k)^T\Sigma_k^{-1}(\boldsymbol{x}-\mu_k)\right\}. \tag{2}$$

This model has been implemented by Fraley and Raftery (2002) in the software MCLUST, which is available at http://www.stat.washington.edu/mclust. In MCLUST, the parameters of the model are estimated using the EM algorithm (Dempster et al., 1977), and the number of clusters and the structure of the covariance matrices are determined according to the BIC criterion. Although the model (1) has been used successfully in many other applications, it may suffer from some difficulties in clustering gene expression profiles. In practice, the normality assumption in (1) is often violated and the violation often leads to sub-optimal clusters. The large number of parameters in the more flexible options of the model could be another concern. This could happen when the dimension of the data is high; see Yeung et al. (2001) for examples. In addition, the default estimates for the hypervolume V may not be suitable for gene expression data. Recently, the model-based methods have been used by a number of authors (Yeung et al., 2001; Medvedovic and Sivaganesan, 2002; McLachlan et al., 2002; Luan and Li, 2003; Wakefield et al., 2003; Medvedovic et al., 2004) to gene expression data. In their analysis, the scattered genes are usually filtered away in advance, and the noise term is not included in the model.

In this paper, we propose a general clustering method, namely a dynamic agglomerative clustering (DAC) method. DAC sets up a special cluster for collecting the noise genes, and groups other genes into informative clusters dynamically and agglomeratively. It avoids potential cluster contamination caused by the noise genes. In addition, we propose a criterion for evaluating clustering results for datasets containing noise genes. The criterion is used to determine the parameter values for DAC. Our numerical results indicate that DAC outperforms other clustering methods, such as AQC and the noise option in MCLUST, in clustering datasets with noise genes.

The remainder of this paper is organized as follows. In Section 2, we describe the DAC method. In Section 3, we compare the performance of DAC and other clustering methods through a simulated example and a real dataset taken from the literature. In Section 4, we apply DAC to an avian pineal gland gene expression dataset for which identification of the circadian patterns is the main focus of the study. In Section 5, we conclude the paper with a brief discussion.

2. Dynamic agglomerative clustering

Let the data be arranged in an $n \times p$ matrix denoted by $\boldsymbol{X} = (x_{ij})$, where n is the number of genes and p is the number of samples (treatments or time points). Let x_{ij} be the expression level of gene i in sample j, and $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})$ be the expression profile of gene i.

To let the reader appreciate the idea behind DAC, in what follows we first describe DAC in an intuitive fashion. To cluster a dataset with noise genes, DAC sets up a special cluster denoted by $\mathscr{C}_0$ for collecting the noise genes. Henceforth, this cluster will be called the null cluster even though it is really not a cluster in the sense that genes in it share no common pattern. DAC works by iterating between the following steps: (I) Group the genes into many small tight clusters according to the cluster centers learned in the current iteration; (II) Merge the similar clusters to form larger clusters; (III) Move the small clusters with size less than a threshold value into the null cluster. DAC is different from agglomerative hierarchical clustering in two respects. First, DAC includes an extra step (step III) to collect noise genes. This is based on the observation that the noise genes tend to be grouped into many small clusters when a large number of clusters were imposed in clustering. Second, the clustering process of DAC is dynamic. At each iteration, the genes are re-grouped according to the cluster centers learned in the last iteration, and the cluster centers are then re-adjusted accordingly. The iterative process of grouping and adjusting will drive the cluster centers to the most typical gene expression patterns. We note that some other clustering methods, such as K-means and SOM, also work in an iterative fashion, but the number of clusters in these methods has to be specified *a priori*.

To describe DAC in a formal fashion, we introduce the following notations:

$\mathscr{C}_i$	cluster i formed by DAC, $i = 0, \ldots, c$;
m_i	size of cluster $\mathscr{C}_i$, $i = 0, \ldots, c$;
m_{high}	reference cluster size;
m_{low}	threshold value for the size of non-noise gene clusters;
δ_i	threshold value for assigning a gene to cluster $\mathscr{C}_i$;
δ_{high}	initial value of δ_i at the birth of cluster $\mathscr{C}_i$;
δ_{low}	threshold value for cluster merging;
η	learning rate of DAC;
λ	shrinking factor of η in iterations;
$\phi(\boldsymbol{x})$	normalizing operator which normalizes $\boldsymbol{x}$ to a vector with mean 0 and variance 1;
$\sigma(1, \ldots, n)$	a permutation of the numbers $1, \ldots, n$;
$\sigma(k)$	kth element of the permutation $\sigma(1, \ldots, n)$.

In addition, we define the following similarity functions: the similarity function between two genes:

$$\rho_1(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left\{-\tau\sum_{k=1}^{p}(x_{ik} - x_{jk})^2\right\}; \tag{3}$$

the similarity function between a gene and a cluster:

$$\rho_2(\boldsymbol{x}_i, \mathscr{C}_j) = \exp\left\{-\tau\sum_{k=1}^{p}(x_{ik} - \omega_{jk})^2\right\}; \tag{4}$$

and the similarity function between two clusters:

$$\rho_3(\mathscr{C}_i, \mathscr{C}_j) = \exp\left\{-\tau \sum_{k=1}^{p} (\omega_{ik} - \omega_{jk})^2\right\}. \tag{5}$$

In these similarity functions, τ is a scale parameter, and $\boldsymbol{\omega}_j = (\omega_{j1}, \ldots, \omega_{jp})^{\mathrm{T}}$ is the center of cluster $\mathscr{C}_j$. Note that these similarity functions are equivalent to Pearson's correlation coefficient, as the gene expression profiles have been normalized here. Inclusion of the scale parameter τ makes the similarity measure more flexible. The use of the similarity functions instead of Pearson's correlation coefficient will be justified further at near the end of Section 2. For simplicity, we set $\tau = 1$ for all examples of this paper. The similarity functions and their corresponding threshold values control the pattern learning, cluster merging and noise gene collection steps described above.

In the pattern learning step, a gene $\boldsymbol{x}$ can be assigned to cluster $\mathscr{C}_i$ only if $\rho(\boldsymbol{x}, \mathscr{C}_i) > \delta_i$. The threshold δ_i is a function of cluster size. It is initialized at a large number denoted by δ_{high} at the birth of the cluster, and is then adjusted to a smaller value in iterations according to the updated cluster size. In the cluster merging step, only two clusters with similarity exceeding the threshold value δ_{low} can be merged. In the noise gene collection step, the clusters with size less than the threshold value m_{low} will be moved into the null cluster. The use of the reference cluster size m_{high} is to adjust the value of δ_i's as in (10).

The parameters η and λ control the convergence rate of DAC, and reduce the chance of moving noise genes senselessly from cluster to cluster in the clustering procedure. At each iteration, the genes will be presented to the clustering procedure in the order of a random permutation. This will eliminate the potential bias caused to clustering by the order of data presentation. The DAC algorithm can then be described as follows:

(a) (Initialization) Normalize the expression profile of each gene; that is, set $\boldsymbol{x}_i \leftarrow \phi(\boldsymbol{x}_i)$ for $i = 1, \ldots, n$. Initialize the parameters, δ_{high}, δ_{low}, m_{high}, m_{low}, η, and λ. Set $c = 1$, $m_1 = 1$, $\delta_1 = \delta_{\text{high}}$, and initialize $\boldsymbol{\omega}_1$ by a gene drawn from $\boldsymbol{X}$ at random, and assign all other genes to the cluster $\mathscr{C}_0$.

(b) (Randomization) Generate a random permutation $\sigma(1, \ldots, n)$ of the numbers $1, \ldots, n$.

(c) (Pattern learning) Repeat steps (c.1)–(c.4) for $k = 1, \ldots, n$.

(c.1) (Locate the current cluster) Locate $\mathscr{C}_j$, the current cluster which gene $\boldsymbol{x}_{\sigma(k)}$ belongs to.

(c.2) (Find acceptable clusters) Find $A = \{i : \rho_2(\boldsymbol{x}_{\sigma(k)}, \mathscr{C}_i) > \delta_i, i = 1, \ldots, c\}$, the set of acceptable clusters which the gene $\boldsymbol{x}_{\sigma(k)}$ can be reassigned into.

(c.3) (Find the nearest acceptable cluster and update the related clusters) If A is non-empty, set $j' \leftarrow \arg\max_{i \in A} \rho_2(\boldsymbol{x}_{\sigma(k)}, \mathscr{C}_i)$. Change the cluster membership of gene $\boldsymbol{x}_{\sigma(k)}$ from cluster $\mathscr{C}_j$ to cluster $\mathscr{C}_{j'}$. If $j' \neq j$, update $\boldsymbol{\omega}_{j'}$ to

$$\boldsymbol{\omega}_{j'} \leftarrow \phi\left(\boldsymbol{\omega}_{j'} + \frac{\eta}{m_{j'} + 1}(\boldsymbol{x}_{\sigma(k)} - \boldsymbol{\omega}_{j'})\right). \tag{6}$$

If $j' \neq j$ and $j \neq 0$, update $\boldsymbol{\omega}_j$ to

$$\boldsymbol{\omega}_{j} \leftarrow \phi\left(\boldsymbol{\omega}_{j} - \frac{\eta}{m_{j} - 1}(\boldsymbol{x}_{\sigma(k)} - \boldsymbol{\omega}_{j})\right). \tag{7}$$

Set $m_{j'} \leftarrow m_{j'} + 1$ and $m_j \leftarrow m_j - 1$.

(c.4) (Add a new cluster) If A is empty, add a new cluster. Set $c \leftarrow c + 1$, $\boldsymbol{\omega}_c \leftarrow \boldsymbol{x}_{\sigma(k)}$, and $\delta_c = \delta_{\text{high}}$; change the cluster membership of gene $\sigma(k)$ from cluster $\mathscr{C}_j$ to cluster $\mathscr{C}_c$; and update $\boldsymbol{\omega}_j$ (if $j \neq 0$) as in Eq. (7).

(d) (Cluster merging) Repeat steps (d.1)–(d.2) until $\max_{i,j} \rho_3(\mathscr{C}_i, \mathscr{C}_j) < \delta_{\text{low}}$.

(d.1) (Find the nearest clusters) Set $(i', j') \leftarrow \arg\max_{1 \leqslant i,j \leqslant c} \rho_3(\mathscr{C}_i, \mathscr{C}_j)$.

(d.2) (Merge clusters) If $m_{j'} < m_{i'}$, merge $\mathscr{C}_{j'}$ into $\mathscr{C}_{i'}$, set $m_{i'} \leftarrow m_{i'} + m_{j'}$ and

$$\boldsymbol{\omega}_{i'} \leftarrow \phi\left(\boldsymbol{\omega}_{i'} + \frac{\eta m_{j'}}{m_{i'} + m_{j'}}(\boldsymbol{\omega}_{j'} - \boldsymbol{\omega}_{i'})\right). \tag{8}$$

Otherwise, merge $\mathscr{C}_{i'}$ into $\mathscr{C}_{j'}$, set $m_{j'} \leftarrow m_{i'} + m_{j'}$ and

$$\boldsymbol{\omega}_{j'} \leftarrow \phi\left(\boldsymbol{\omega}_{j'} + \frac{\eta m_{i'}}{m_{i'} + m_{j'}}(\boldsymbol{\omega}_{i'} - \boldsymbol{\omega}_{j'})\right). \tag{9}$$

Set $c \leftarrow c - 1$.

(e) (Null cluster formation) Merge the clusters with size less than m_{low} into the null cluster $\mathscr{C}_0$.

(f) (Threshold updating) Set $\eta \leftarrow \lambda\eta$, and adjust δ_i for $i = 1, \ldots, c$ as follows:

$$\log(\delta_i) \leftarrow \begin{cases} \log(\delta_{\text{high}}), & \text{if } m_i \leqslant m_{\text{low}}, \\ \log(\delta_{\text{high}}) - [\log(m_i) - \log(m_{\text{low}})] & \\ \quad \times \frac{\log(\delta_{\text{high}}) - \log(\delta_{\text{low}})}{\log(m_{\text{high}}) - \log(m_{\text{low}})}, & \text{if } m_{\text{low}} < m_i < m_{\text{high}}, \\ \log(\delta_{\text{low}}), & \text{if } m_i \geqslant m_{\text{high}}. \end{cases} \tag{10}$$

(g) (Termination checking) Checking the termination condition of the procedure. If the condition is satisfied, go to step (i); otherwise, go to step (b).

(i) (Cluster Assignment) Repeat steps (i.1)–(i.2) for $k = 1, \ldots, n$.

(i.1) (Find acceptable clusters) Find the set $A = \{i : \rho_2(\boldsymbol{x}_{\sigma(k)}, \mathscr{C}_i) > \delta_i, i = 1, \ldots, c\}$.

(i.2) (Assign genes to the nearest acceptable cluster) If A is non-empty, assign the gene to the cluster $\mathscr{C}_{j'}$, where $j' = \arg\max_{i \in A} \rho_2(\boldsymbol{x}_{\sigma(k)}, \mathscr{C}_i)$. Otherwise, assign the gene to the null cluster $\mathscr{C}_0$.

In DAC, the center of the null cluster $\mathscr{C}_0$ is never calculated, as the pattern represented by it is not of interest to us. Besides the similarity function $\rho_3(\cdot,\cdot)$ defined in (5), we have also tried

$$\rho_3(\mathscr{C}_i, \mathscr{C}_j) = \frac{1}{m_i m_j} \sum_{\boldsymbol{x}_{k1} \in \mathscr{C}_i} \sum_{\boldsymbol{x}_{k2} \in \mathscr{C}_j} \rho_1(\boldsymbol{x}_{k1}, \boldsymbol{x}_{k2}), \tag{11}$$

which corresponds to the average-linkage function in the agglomerative hierarchical clustering algorithm (Murtagh, 1983). We note that the function ρ_3 defined in (5) corresponds to the mean-linkage function. These two similarity functions lead to similar results, but the one defined in (11) is more computationally intensive.

In the threshold updating step, $\log(\delta)$ decreases linearly with $\log(m)$. Here, we work on the logarithm of δ, because the similarity functions specified in (3)–(5) are exponential functions of Pearson's correlation coefficient. Working under log-scales allows us to compare the threshold and Pearson's correlation coefficient directly. We measure the cluster size in the logarithmic scale, as we want to avoid a drastic change of δ with a cluster size. Other reduction schemes for δ_i's could also work well but we have satisfactory performances when adopting this choice. The termination condition can be a pre-specified number of iterations, or a criterion for the convergence of cluster centers. For example, we may set the following criterion for measuring the convergence of cluster centers: In the most recent 10 consecutive iterations, the number of clusters does not change and the cluster centers satisfy the following inequality for a pre-specified small number ϵ,

$$\max_{1\leqslant i\leqslant c} \|\boldsymbol{\omega}_i^{(t+1)} - \boldsymbol{\omega}_i^{(t)}\| < \epsilon \quad \text{for } t = T, \ldots, T-9, \tag{12}$$

where T denotes the current iteration number, $\boldsymbol{\omega}_i^{(t)}$ represents the center of cluster i at iteration t, and $\|\cdot\|$ represents a pre-specified distance measure for two cluster centers.

For given similarity functions, DAC has six free parameters to be determined by the user, namely, η, λ, m_{high}, m_{low}, δ_{high}, and δ_{low}. Even though setting six parameters might seem to be a difficult task, we note that the settings for most of the parameters do not significantly affect the outcome. For the parameters which need to be chosen more carefully, we propose a method to determine their values simultaneously by optimizing an objective function. In what follows, we discuss the roles of these parameters and explain how to set them. Through (6)–(9), the learning rate η and the shrinking factor λ determine the convergence rate of DAC. The smaller values of η and λ lead to a faster convergence of DAC, but perhaps also result in a larger chance for DAC to fail to capture the true gene expression patterns. In this paper, we let $\eta = 1$ at the first iteration and let it decrease geometrically with the factor $\lambda = 0.99$ in the following iterations. The learning rate can also be set as a harmonic function of the iteration number as in (Tamayo et al., 1999), for example, $\eta_t = 0.02T/(T + 100t)$ with T being the total number of iterations. The learning rate shrinkage is motivated by the observation that useless steps could be wasted on moving noise genes from cluster to cluster. If η and λ are both fixed to 1, the cluster centers learned in DAC will be reduced to the average profile of the corresponding clusters.

The parameter δ_{high} controls tightness of the clusters formed in the first iteration. The parameter δ_{low} determines tightness of the clusters eventually formed by DAC. A large δ_{low} will result in a large number of small tight clusters, whereas a small δ_{low} will result in a small number of large loose clusters. In our experience, δ_{low} is perhaps the most important parameter of DAC. In practice, we fix δ_{high} and other parameters and vary δ_{low} over a finite set in order to find an acceptable value for δ_{low}. In this paper, we fix $\delta_{\text{high}} = 0.6$ for all examples.

The parameters m_{high} and m_{low} represent our estimate of the maximum and minimum sizes of non-noise gene clusters, respectively. The parameter m_{high} works as a reference value for the cluster size, and its effect on DAC is limited. Our experience shows that a value between $n/10$ and $n/3$ is often appropriate for m_{high}. The parameter m_{low} is more important to DAC than the parameter m_{high}. A large m_{low} will result in that some non-noise genes are grouped as noise genes, whereas a small m_{low} will result in findings of pseudo expression patterns. This suggests that m_{low} should be set to a large value if we are only interested in the major gene expression patterns; and it should be set to a small value otherwise. In practice, m_{low} can be determined as follows. Try a sequence of values for m_{low} from low to high. If some value results in a big gap in the number of clusters, i.e., a drastic decrease of the number of clusters, m_{low} can then be set to that value. The underlying rationale is that the noise genes tend to be grouped into many small clusters.

As discussed above, DAC has essentially only two parameters δ_{low} and m_{low}. The parameters δ_{high} and m_{high} mainly serve as starting values. Inclusion of other parameters makes the algorithm more flexible. In the following we describe one method for determining the parameters of DAC simultaneously by optimizing an objective function. The objective function evaluates the overall quality of the resulting clusters. As reviewed by Chen et al. (2002), there are various criteria in the literature for evaluating the overall quality of a clustering result. However, these criteria are all designed for datasets without noise genes. For example, the average silhouette width (Rousseeuw, 1987) is one such criterion. The average silhouette width is a composite index reflecting the compactness and separation of clusters. For each gene i, its silhouette width is defined as

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}},$$

where $a(i)$ is the average dissimilarity (defined below) of gene i to other genes in the same cluster, and $b(i)$ is the average dissimilarity of gene i to genes in its nearest neighbor cluster. For a dataset without noise genes, the average silhouette width $\bar{s} = \sum_{i=1}^{n} s_i/n$ is a good measure for the overall quality of a clustering result. The larger the value of $\bar{s}$ is, the better is the overall quality of the clustering result. However, for a dataset with noise genes maximizing $\bar{s}$

will force the noise genes to be grouped as non-noise genes. Hence, the relative size of $\bar{s}$ cannot be used as a criterion for evaluating the overall quality of a clustering result produced by DAC.

To accommodate noise genes, in the following we modified $\bar{s}$ to be

$$\widetilde{s} = \left[\sum_{\boldsymbol{x}_i \notin \mathscr{C}_0} s(i) + \sum_{\boldsymbol{x}_i \in \mathscr{C}_0} \max\{0, s(i)\}\right] \Bigg/ n. \tag{13}$$

Because the genes in the null cluster $\mathscr{C}_0$ are not grouped by similarity, members of $\mathscr{C}_0$ tend to have a negative silhouette width. The statistic $\bar{s}$ penalizes the null cluster members with negative silhouette widths, while $\widetilde{s}$ does not. Hence, $\widetilde{s}$ encourages the collection of noise genes into the null cluster. If a non-null cluster gene is wrongly classified into the null cluster, its silhouette width will be changed from a positive value to zero. Hence, $\widetilde{s}$ also penalizes this type of misclassification. The above analysis implies that $\widetilde{s}$ can work as a good criterion for evaluating the overall quality of clusters produced by DAC. Given the statistic $\widetilde{s}$, the problem of parameter determination in DAC is converted into an optimization problem, finding a setting of parameters m_{high}, m_{low}, δ_{high} and δ_{low} such that $\widetilde{s}$ is maximized. In practice, an extensive search for such a setting does not seem necessary. We fix m_{high}, m_{low} and δ_{high}, and then determine δ_{low} by maximizing $\widetilde{s}$ over a finite set of candidate values.

To calculate $\tilde{s}$, we need to define a dissimilarity function for two genes. Considering the similarity function defined in (5), we define the dissimilarity function as

$$\begin{aligned} d_1(\boldsymbol{x}_i, \boldsymbol{x}_j) &= 1 - \rho_1(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &= 1 - \exp\left\{ -\tau \sum_{k=1}^{p} (x_{ik} - x_{jk})^2 \right\}. \end{aligned} \tag{14}$$

Although ρ_1 defined in (5) is equivalent to Pearson's correlation coefficient r and independent of τ essentially, d_1 is not equivalent to $1 - r$ and depends on the parameter τ. When τ is large (the difference of gene expression profiles is amplified), maximizing $\tilde{s}$ tends to classify non-noise genes as noise genes and the resulting clusters tends to be tight. When τ is small, maximizing $\tilde{s}$ tends to classify noise genes as non-noise genes and the resulting clusters tends to be loose. The parameter τ gives us the freedom to adjust the similarity and dissimilarity functions. It should decrease as the dimension p increases. Our experience shows that DAC is not sensitive to the choice of τ. A value of τ taken in the interval [0.5,2] is appropriate for most examples of this paper. As mentioned before, τ is fixed to *1* in this paper. Note that the dissimilarity function is not required to satisfy the triangle inequality (Kaufman and Rousseeuw, 1988, page 16), but it satisfies the other mathematical requirements of a distance function, i.e., $d_1(\boldsymbol{x}_i, \boldsymbol{x}_j) \geqslant 0$, $d_1(\boldsymbol{x}_i, \boldsymbol{x}_j) = d_1(\boldsymbol{x}_j, \boldsymbol{x}_i)$, and $d_1(\boldsymbol{x}_i, \boldsymbol{x}_i) = 0$.

3. Illustrative examples

3.1. A simulated example

This example consists of 10 datasets. Each dataset consists of 1000 genes, among which 400 genes are generated from the 6-dimensional Gaussian distribution $N_6(\boldsymbol{\mu}, 0.2^2 I_6)$, 300 genes from $N_6(-\boldsymbol{\mu}, 0.2^2 I_6)$, and 300 genes from $N(\boldsymbol{0}, 0.2^2 I_6)$, where $\boldsymbol{\mu} = (\frac{3}{4}, \frac{3}{4}, \frac{3}{4}, -\frac{3}{4}, -\frac{3}{4}, -\frac{3}{4})$ and I_6 is the six-dimensional identity matrix. Since, from a biological point of view, we are primarily interested in the relative up/down-regulation of gene expressions instead of the absolute amplitude changes, we normalized the expression profile of each gene to have mean 0 and variance 1. The genes generated from the distribution $N(\boldsymbol{0}, 0.2^2 I_6)$ can be regarded as scattered genes, as they have a flat expression pattern and a large dispersion after normalization. To examine the performance of DAC in clustering a dataset with scattered genes, the scattered genes in these datasets are not filtered away in advance.

DAC was applied to this example. To determine the parameter values, we first worked on one dataset. Henceforth, this dataset is called the sample dataset. We fix $m_{\text{high}} = 250$ and $m_{\text{low}} = 20$ and let δ_{low} vary over the set $\{0.05, 0.1, \ldots, 0.6\}$. For each value of δ_{low}, DAC was run 10 times. Each run consists of 20 iterations. Fig. 2b suggests that $[0.25, 0.45]$ is an appropriate interval of δ_{low}. Fig. 2a shows that DAC is fairly robust to the value of δ_{low}. It can identify the true number of clusters for the non-noise genes with any value of δ_{low} in the set $\{0.05, \ldots, 0.6\}$. DAC was then run for each of the 10 datasets with $\delta_{\text{low}} = 0.4$. Each run also consists of 20 iterations. Fig. 3 shows the clusters identified by DAC for the sample dataset. DAC has successfully separated the non-noise genes from the noise genes and grouped them into two clusters. A comparison with the true clusters shows that there are only 4 (out of 1000) genes being misclustered. The results for the other 9 datasets are similar. The overall misclustering rate for the 10 datasets is 0.53%, and the average of $\widetilde{s}$ is 0.471. These results together with those achieved by AQC and MCLUST with the noise option are summarized in Table 1.

For comparison, we also applied AQC and MCLUST with the noise option to this example. The software for AQC was developed by De Smet et al. (2002) and is available at http://www.esat.kuleuven.ac.be/~thijs/Work/Clustering.html. For this example, we set the minimum cluster size to 20 and the test significance level to 0.95. The latter one is the default value given in the software. The overall misclustering rate is 2.63%, which is significantly higher than that of DAC. The average of $\widetilde{s}$ is 0.455. In calculating $\widetilde{s}$, the unclustered genes are grouped as the null cluster. AQC was also run with other settings, for example, the minimum cluster size equals to 5, 10, and 50. The results are all similar.

To apply MCLUST (model (1)) to this example, an initial estimate for the noise is required (Fraley and Raftery,

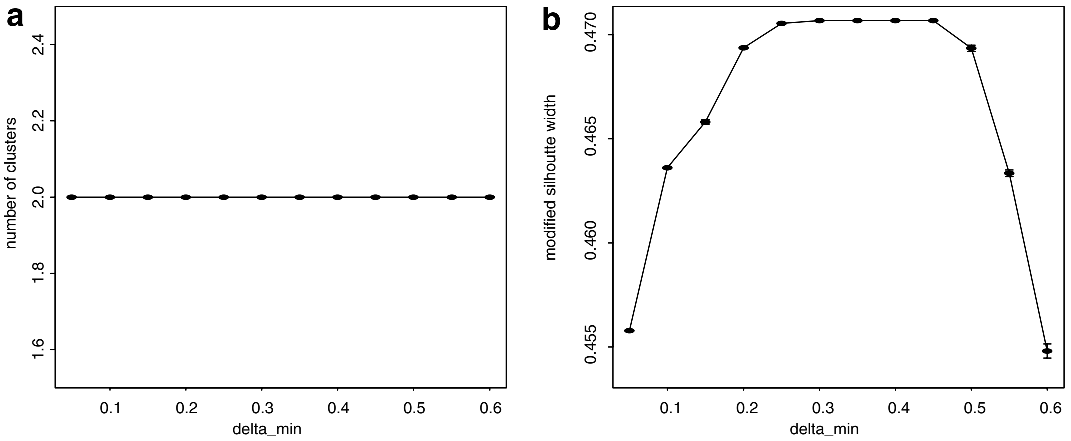

Fig. 2. Clustering results of DAC for the sample dataset with various choices of δ_{low}. (a) The average number of clusters (except for the null cluster). (b) The average modified silhouette width. The averages are calculated based on 10 independent runs, and the vertical segments in the plots show the respective 95% confidence intervals of the averages.

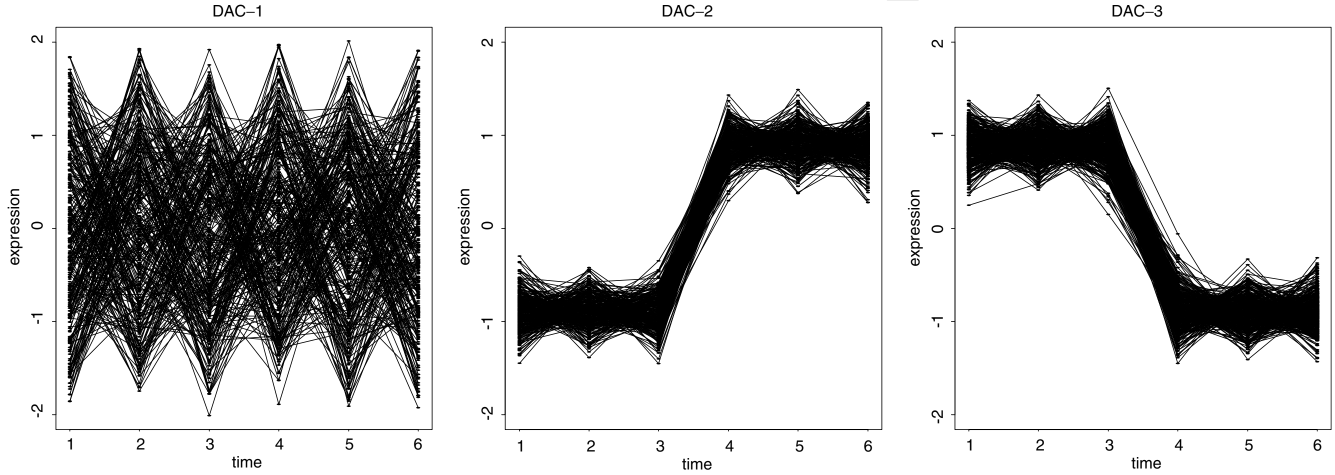

Fig. 3. Clusters identified by DAC for the sample dataset.

Table 1
Comparison of clustering results produced by AQC, MCLUST and DAC for the simulated example

Algorithm	AQC	MCLUST	DAC
Error rate (%)	2.63	1.32	0.53
SD	0.12	0.10	0.06
$\widetilde{s}$	0.455	0.460	0.471
SD	0.001	0.002	0.000

Error rate and $\widetilde{s}$ are calculated by averaging over 10 datasets. SD: standard deviation.

2002). For simplicity, we used the true noise as a surrogate for the estimate; that is, specifying that the last 300 genes of each dataset are noise genes. The BIC analysis suggests that the EEE model (ellipsoidal and equal variance) is appropriate for this example. The overall misclustering rate is 1.32% and the average of $\widetilde{s}$ is 0.46. In calculating $\widetilde{s}$, the noise cluster is treated as the null cluster. This is true for the other examples of this paper. Because the data are generated from a Gaussian mixture distribution and the sample size of each cluster is large, this is an ideal example for MCLUST with noise. However, even for this example, DAC still outperforms MCLUST with noise.

It is encouraging to note that $\widetilde{s}$ shows a consistent (reverse) order with "error rate" in Table 1. This implies that when the error rate is not calculable, $\widetilde{s}$ can potentially work as an evaluation criterion for the clustering results of a dataset with noise genes. In each of the real examples of this paper, $\widetilde{s}$ is reported for the three methods as a measure for the overall quality of the clustering results.

Since DAC and AQC are both model-free methods and AQC works fairly well for this example, we make a further examination for their performance. Fig. 4 shows the noise genes (in the non-normalization scale) identified by DAC and AQC from the sample dataset. For this dataset, AQC classifies some non-noise genes as noise genes, while DAC avoids this mistake.

To assess the sensitivity of DAC to its parameter values, we conducted the following experiment: fix $\delta_{\text{low}} = 0.4$ and tried different values of m_{high}, m_{low} and δ_{high}. The results

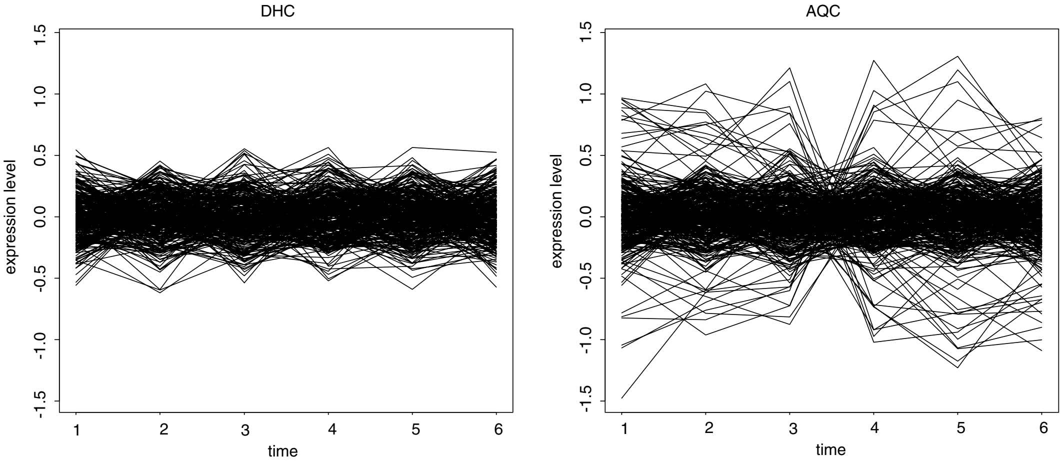

Fig. 4. The left panel shows the gene expression profiles clustered into the null cluster by DAC. The right panel shows the gene expression profiles unclustered by AQC.

are shown in Table 2. They indicate that DAC is not sensitive to the choice of the parameters m_{high}, m_{low} and δ_{high}. No error rate presented in Table 2 is significantly different from 0.54 (0.06), the error rate obtained with $(m_{\text{high}}, m_{\text{low}}, \delta_{\text{high}}) = (250, 20, 0.6)$.

To show that the performance of the popular clustering methods could suffer from the existence of noise genes, we applied agglomerative hierarchical clustering (AHC), K-means and SOM methods to this example. For each method, the genes were forced to be grouped into three clusters. The software for AHC and K-means are available in S-PLUS. The software for SOM was developed by Tamayo et al. (1999) and is available at http://www-genome.wi.mit.edu/software/genecluster2/gc2.html. AHC was applied this example with the Euclidean distance and the average linkage. The overall misclustering rate for the ten datasets is 24.98% with standard deviation 0.53%, which is very far from that of DAC. AHC was also tried with various options of linkage and dissimilarity functions, such as single-linkage, complete-linkage, and the dissimilarity function defined in (14). The results are all similar. Like AHC, K-means also fails to capture the true expression patterns of the non-noise genes. The overall misclustering rate for the ten datasets is 10.74% with standard deviation 0.28%. In SOM, a grid of size 1×3 was used for each dataset, and other parameters were set to the default values given in the software. The overall misclustering rate is 10.75% with standard deviation 0.31%. For these three methods, $\tilde{s}$ is not calculated, as the null cluster is not well defined among the clusters found by them.

3.2. Leukemia cell line HL-60 data

The myeloid leukemia cell line HL-60 undergoes macrophage differentiation on treatment with the phorbol ester PMA. Nearly 100% of HL-60 cells become adherent and exit the cell cycle with 24 h of PMA treatment. To monitor the process, expression levels of more than 6000 human genes were measured at four time points 0, 0.5, 4 and 24 h after PMA stimulation. This dataset is available at http://www-genome.wi.mit.edu/software/genecluster2/gc2.html and has been used by Tamayo et al. (1999) as an example to support the use of SOM.

In this paper, we use this dataset to demonstrate that DAC can make a further improvement in a dataset with scattered genes being removed in advance by a variation filter. In particular, DAC can further remove singletons and mini-clusters so that the patterns in tighter large clusters can be better identified. The variation filter used here is the same as that used in (Tamayo et al., 1999). Totally there are 590 genes left after filtration. This number is slightly different from the number (567) reported in (Tamayo et al., 1999). The 590 genes were then normalized such that the expression profile of each gene has mean 0 and variance 1.

SOM was applied to the pre-processed data. As in (Tamayo et al., 1999), a grid of size 4×3 was specified for the dataset. The genes were clustered into 12 clusters, which are shown in Fig. 5.

SOM performs very well for this example. It is only in its comparison to outcomes of DAC that we see the

Table 2
Clustering error rates of DAC with various choices of $(m_{\text{high}}, m_{\text{low}}, \delta_{\text{high}})$

Setting	(250, 20, 0.5)	(250, 20, 0.7)	(250, 10, 0.6)	(250, 30, 0.6)	(200, 20, 0.6)	(300, 20, 0.6)
Error rate (%)	0.56	0.55	0.57	0.54	0.58	0.54
SD	0.06	0.08	0.07	0.07	0.08	0.05

The error rates are calculated by averaging over the 10 simulated datasets. SD: standard deviation of the error rate.

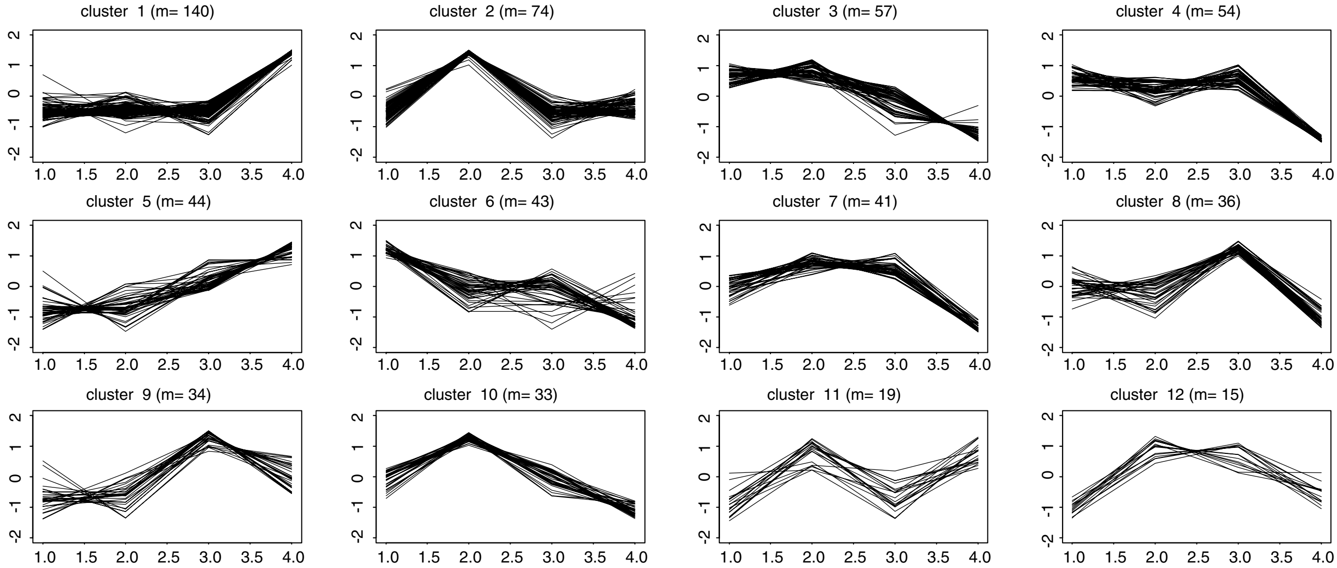

Fig. 5. Clustering result of SOM for the pre-processed HL-60 data.

outcome could be further improved by removing singleton and mini-cluster genes.

AQC was applied to the pre-processed data with the minimum cluster size 2 and the test significance level 0.725. AQC finds 11 clusters which are shown in Fig. 6. The corresponding value of $\widetilde{s}$ is 0.320. Here we set a very low test significance level, because AQC finds no clusters for this example when the test significance level is greater than 0.85. Fig. 6 indicates that AQC tends to produce tight clusters and leave too many genes unclustered. This finding is consistent with the results presented in Fig. 4 for the simulated example.

MCLUST with the noise option was applied to this example. The unclustered genes by AQC were used as the initial estimate for the noise genes. The BIC analysis (Fig. 7) suggests that an 11-component VVV model (excluding the noise component) is appropriate for this dataset. The clusters are shown in Fig. 8, and the corresponding value of $\widetilde{s}$ is 0.425. The VVV model is the most general model in MCLUST, in which the covariance matrices are unrestricted across the Gaussian components of the mixture model. For this example, MCLUST has similar performance to AQC: it clusters too many genes (162 out of 590 genes) as noise genes. Note that this dataset does not include scattered genes, and the percentage of singleton and mini-cluster genes should be low.

Finally, DAC was applied to this example with $m_{\text{high}} = 200$, $m_{\text{low}} = 5$, and $\delta_{\text{low}} \in \{0.05, 0.1, \ldots, 0.6\}$. Here

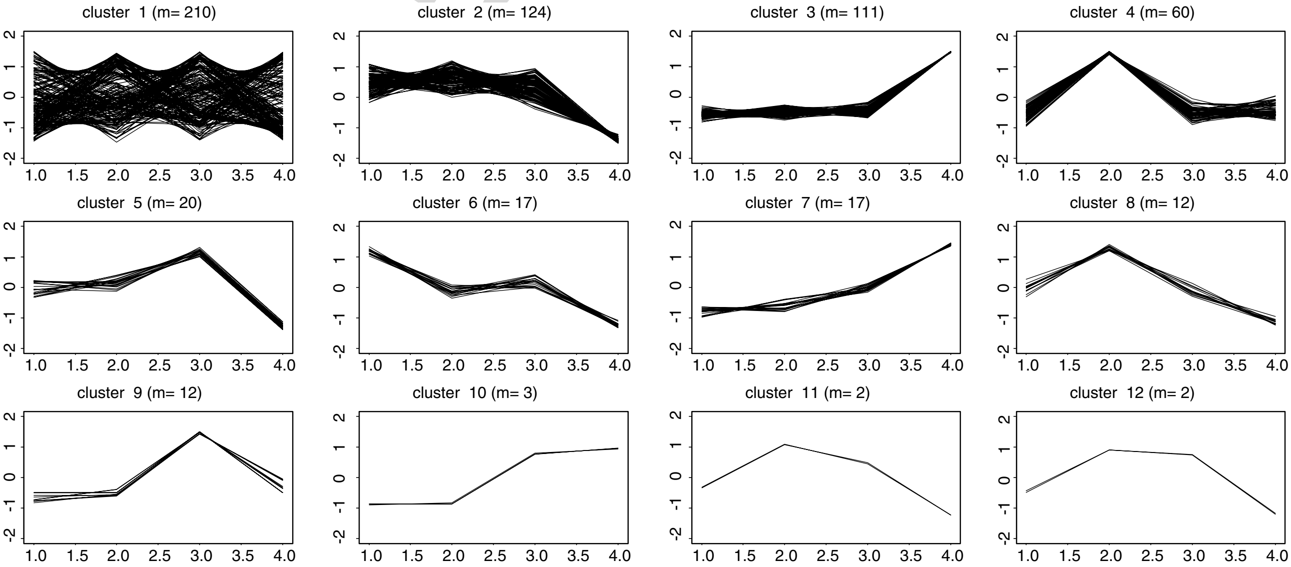

Fig. 6. Clustering result of AQC for the pre-processed HL-60 data. Cluster 1: collection of the genes unclustered by AQC. Clusters 2–12: clusters identified by AQC.

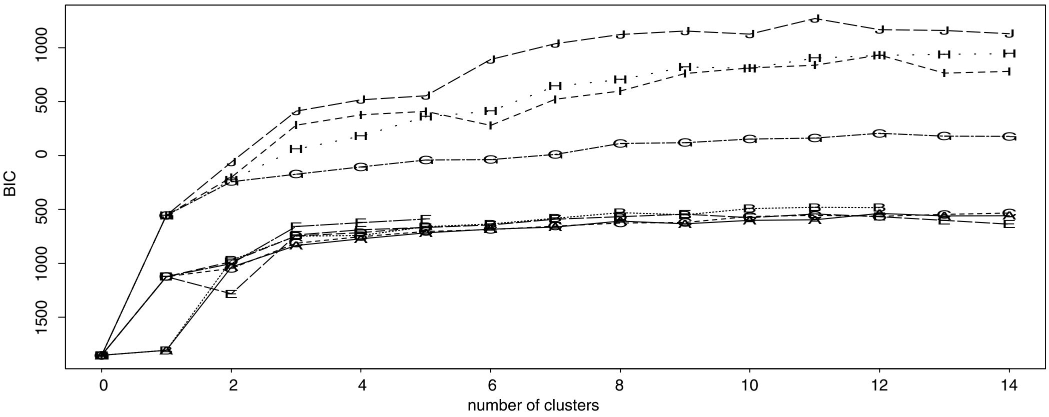

Fig. 7. BIC values from MCLUST with noise for the pre-processed HL-60 data. The models considered include B: VII, J: VVV, etc.

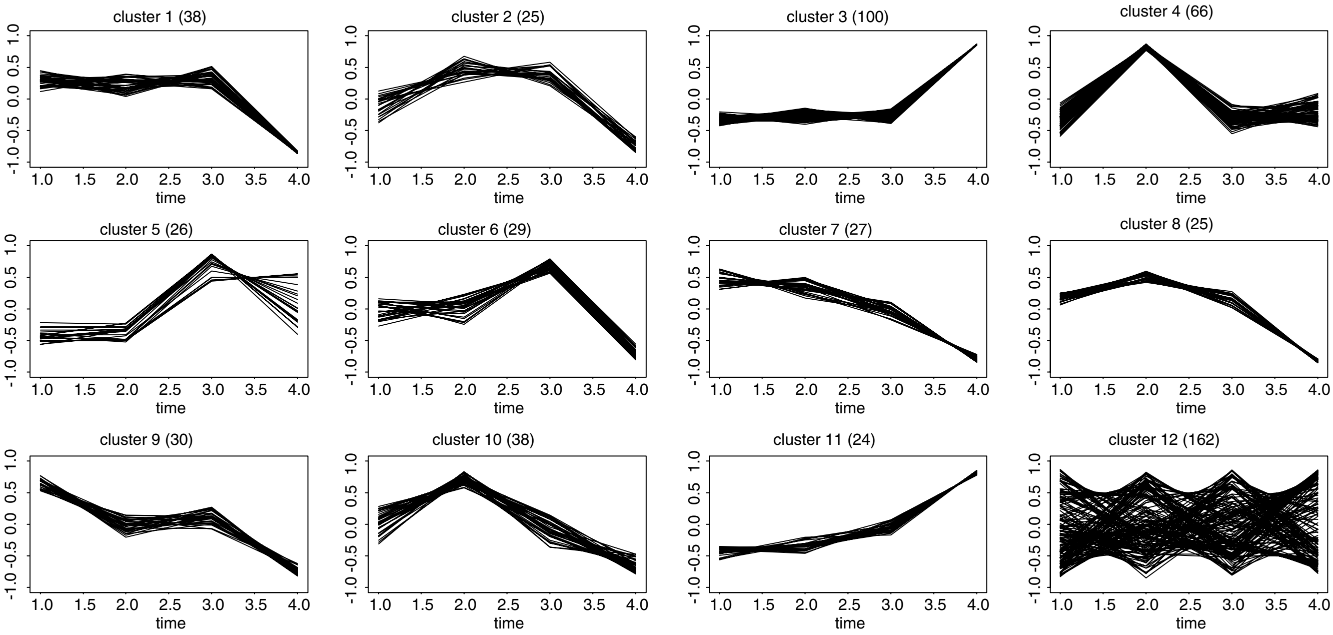

Fig. 8. Clustering result of MCLUST with noise for the pre-processed HL-60 data. Cluster 12 corresponds to the noise cluster.

m_{low} was set to a small value as the scattered genes have been eliminated from the dataset. For each value of δ_{low}, DAC was run 50 iterations. Fig. 9b suggests that $\delta_{\text{low}} = 0.25$ is appropriate for this example. Fig. 10 shows the 12 clusters (including the null cluster) obtained in a run with $\delta_{\text{low}} = 0.25$. The corresponding value of $\widetilde{s}$ is 0.493. In Fig. 9b, the curve of $\widetilde{s}$ appears to be bimodal. It indicates that the null cluster (cluster 9) may contain one or more mini-clusters. It is easy to see that cluster 9 contains two separable subclusters and some singleton genes. Except for the null cluster, each of the other 11 clusters represents a different pattern.

In summary, the major gene expression patterns contained in the HL-60 data can be identified by any of the four clustering methods: SOM, MCLUST with noise, AQC and DAC. For example, the largest 8 DAC clusters correspond to SOM clusters 1, 3, 2, 4, 6, 8, 12, and 5, respectively. Here we refer the clusters identified by DAC as the DAC clusters, and refer the clusters identified by other methods equivalently. Since this is an illustrative example for SOM, it is no surprise that SOM works well. To complete the comparison, we note that the largest 6 DAC clusters correspond to AQC clusters 3, 2, 4, 5, 6, and 9, respectively; and they also correspond to MCLUST clusters 3, 8, 4, 6, 9 and 5, respectively. As a reference quantity for the overall quality of the clustering results, the values of $\widetilde{s}$ produced by AQC, MCLUST and DAC are summarized in Table 3. The comparison indicates again the superiority of DAC for this example.

4. Avian pineal gland gene expressions data

The avian pineal gland contains both circadian oscillators and photoreceptors to produce rhythms in

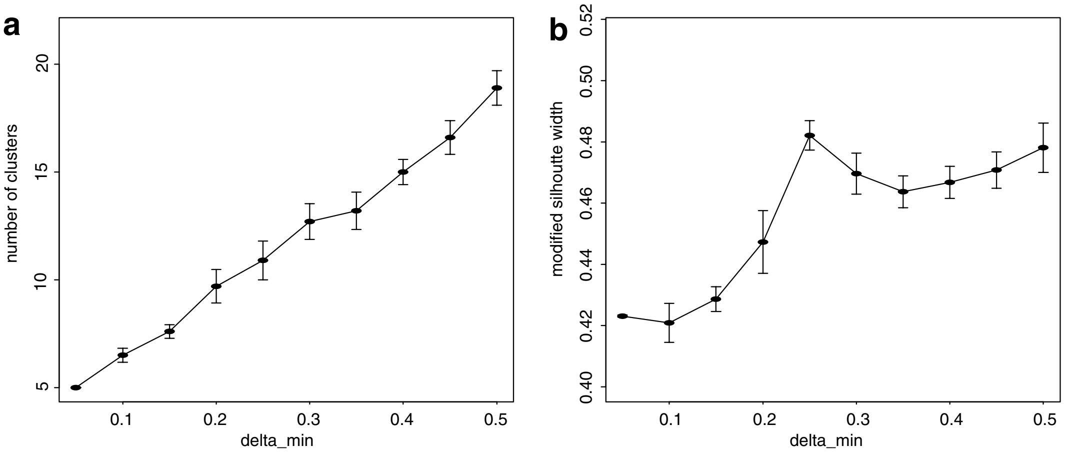

Fig. 9. Clustering results of DAC for the pre-processed HL-60 data with $\delta_{\text{low}} \in \{0.05, 0.1, \ldots, 0.6\}$. (a) The average number of clusters. (b) The average of modified silhouette widths. The notations are the same as in Fig. 2.

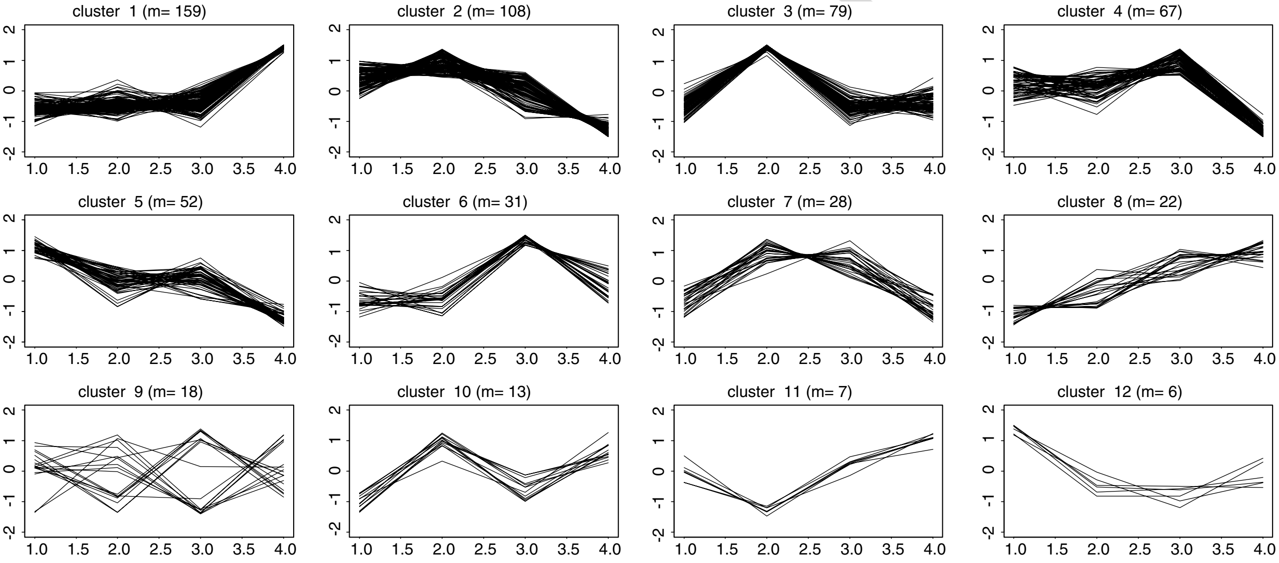

Fig. 10. Clustering result of DAC for the pre-processed HL-60 data. Cluster 9: the null cluster which comprises singleton and mini-cluster genes. Other Clusters: the clusters identified by DAC.

Table 3
Comparison of $\tilde{s}$ produced by different methods for the HL-60 dataset and the avian dataset

Examples	AQC	MCLUST	DAC	
			Average[a]	Maximum[b]
HL-60	0.320	0.425	0.482 (0.002)	0.493
avian	0.166	0.132	0.175 (0.002)	0.186

The numbers in the parentheses are the standard deviations of the averages.

[a] Obtained by averaging over 10 runs.

[b] Maximum value of $\tilde{s}$ found in the 10 runs.

biosynthesis of the hormone melatonin in vivo and in vitro. It is of great interest to understand the genetic mechanisms driving the rhythms. For this purpose, a sequence of cDNA microarrays of birds' pineal gland transcripts under the light–dark (LD) condition were generated. The birds were euthanized at 2, 6, 10, 14, 18, 22 h Zeitgeber time (ZT) to obtain mRNA to produce adequate cDNA libraries. Four microarray chips per time point were produced. Throughout the experiment, samples from LD ZT18 were used as controls. Four observations at each time point were log-transformed and averaged. This produces a data matrix of size 7730 × 6. Each row represents the expression profile of a particular gene at six time points, and it was then normalized to have mean 0 and variance 1. To examine the performance of DAC in clustering a large dataset with noise genes, no filtration is performed in data pre-processing. All genes are kept for the clustering analysis.

DAC was first applied to cluster the data with $m_{\text{high}} = 1000$, $m_{\text{low}} = 20$ and $\delta_{\text{low}} = 0.05$, 0.1, 0.15, 0.2, 0.25. For each value of δ_{low}, DAC was run 2000 iterations. The value of $\tilde{s}$ was calculated at the end of each run. Fig. 11a shows that the resulting number of clusters

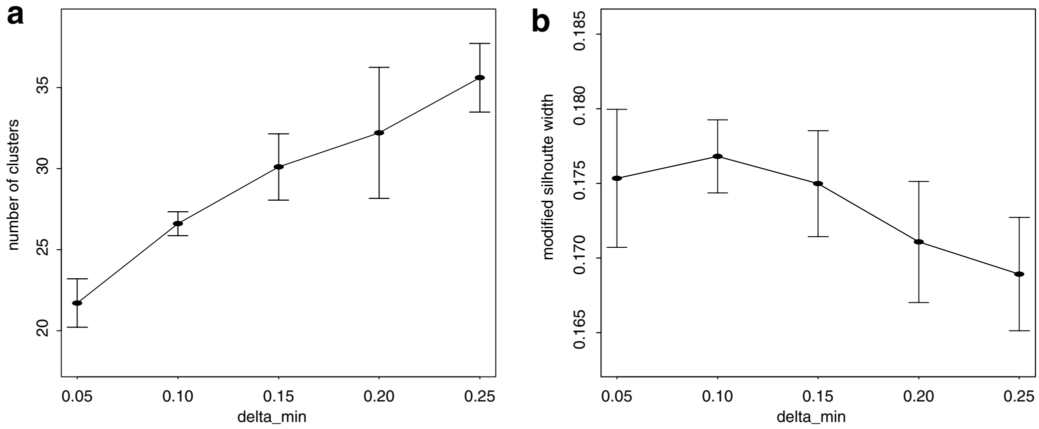

Fig. 11. Clustering results of DAC for the avian data. (a) The average number of clusters. (b) The average of modified silhouette widths. The notations are the same as in Fig. 2.

increases as $\delta_{\min}$ increases. Fig. 11b shows the curve of $\tilde{s}$ versus $\delta_{\min}$. It suggests that 0.05 is a suitable value for $\delta_{\min}$, as we prefer a clustering result with a small number of clusters and a large value of $\tilde{s}$. The values of $\tilde{s}$ resulted from the choice $\delta_{\min} = 0.1$ is not significantly larger than that from $\delta_{\min} = 0.05$, while the number of clusters resulted from $\delta_{\min} = 0.05$ is significantly smaller than that from $\delta_{\min} = 0.1$. Fig. 12 shows the clusters obtained in a run of DAC with $\delta_{\text{low}} = 0.05$. In the run, 555 genes (about 7.2% of 7730 genes) are grouped into the null cluster, and other genes are grouped into 18 clusters which are tight and reasonably separated. The corresponding value of $\tilde{s}$ is 0.186. This value together with the values of $\tilde{s}$ produced by AQC and MCLUST are summarized in Table 3. Fig. 13a shows the expression profiles of all genes in the dataset, and Fig. 13b shows the expression profiles of the genes grouped into the null cluster. The plots indicate that DAC has successfully separated the noise genes from non-noise genes for this dataset. DAC was also run with other parameter settings. The results are similar. The major gene expression patterns, say, those represented by clusters $1, 2, \ldots, 7$, 10, and 11 (in Fig. 11), can be found in all runs.

AQC was run for this dataset with the minimum cluster size 20 and the test significance level 0.95. It leaves 4946 genes (about 64% of total genes) unclustered, and groups other genes into three clusters, which correspond to clusters 1, 2, and 3 as shown in Fig. 12, respectively. The respective cluster sizes are 1345, 1249 and 190. It misses several other

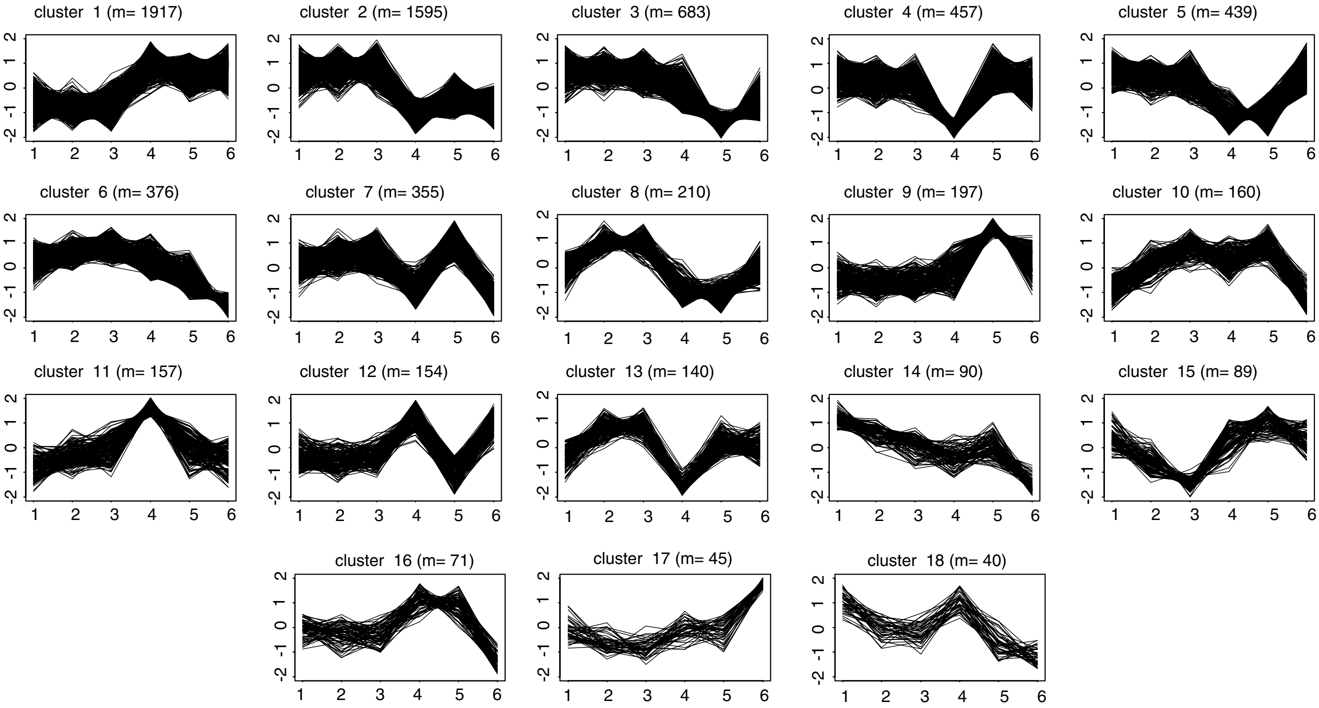

Fig. 12. Clusters identified by DAC for the avian data. The clusters are shown in the normalization scale.

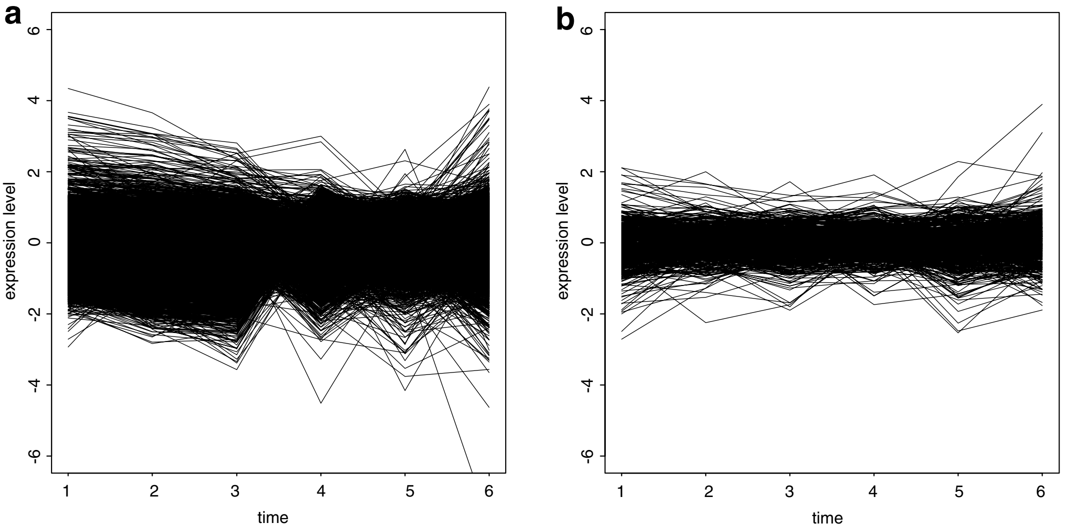

Fig. 13. (a) Expression profiles of all avian pineal gland genes. (b) Expression profiles of the genes grouped into the null cluster in a run of DAC with $\delta_{\rm low} = 0.05$.

typical gene expression patterns identified by DAC. The corresponding value of $\tilde{s}$ is 0.166.

Finally, MCLUST with noise was also applied to this dataset. The unclustered genes by AQC were used as the initial estimate of the noise genes. The BIC analysis (Fig. 14) suggests that an 18-component VVV model (excluding the noise component) is appropriate for the data. The resulting clusters are shown in Fig. 15, and the corresponding value of $\tilde{s}$ is 0.132. MCLUST works less well for this example. The normality assumption for the non-noise genes could be of a problem for this dataset. For example, clusters 1, 3, 9, 16 and 18 represent very similar patterns and perhaps should be combined into one cluster. Due to the imposition of normality, MCLUST groups them into several clusters.

5. Discussion

In this paper, we have proposed a new clustering method—the DAC method. DAC can automatically separate noise genes from other genes and thus avoid possible contamination to gene expression patterns caused by the noise genes. For DAC, the scattered gene filtering step is no longer necessary in data pre-processing. In addition, we have proposed a criterion for evaluating clustering results of a dataset which contains noise genes. DAC has been applied successfully to two real datasets containing noise genes. DAC has also been applied successfully to some conventional datasets which do not contain noise observations (the results are not reported in the paper). Our numerical results indicate that DAC can work as a general clustering method for a dataset with or without noise observations.

DAC is closely related to the noise modeling method in MCLUST. In the latter, it is assumed explicitly that the noise genes are uniformly distributed in the data region. In DAC, this is done implicitly. The agglomerative hierarchical clustering method tends to cluster the noise genes into many small clusters. Based on this observation, we merge small clusters into the null cluster in step (e) of

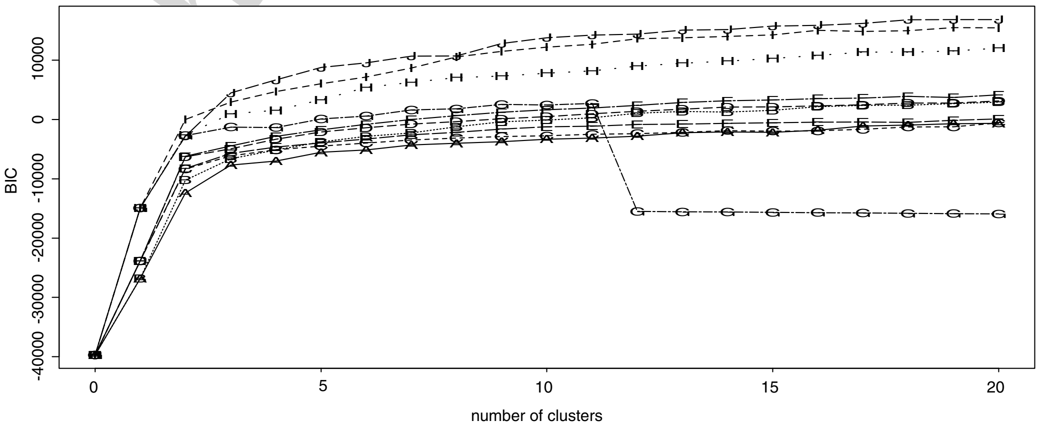

Fig. 14. BIC values from MCLUST with noise for the avian data. The models considered include B: VII, J: VVV, etc.

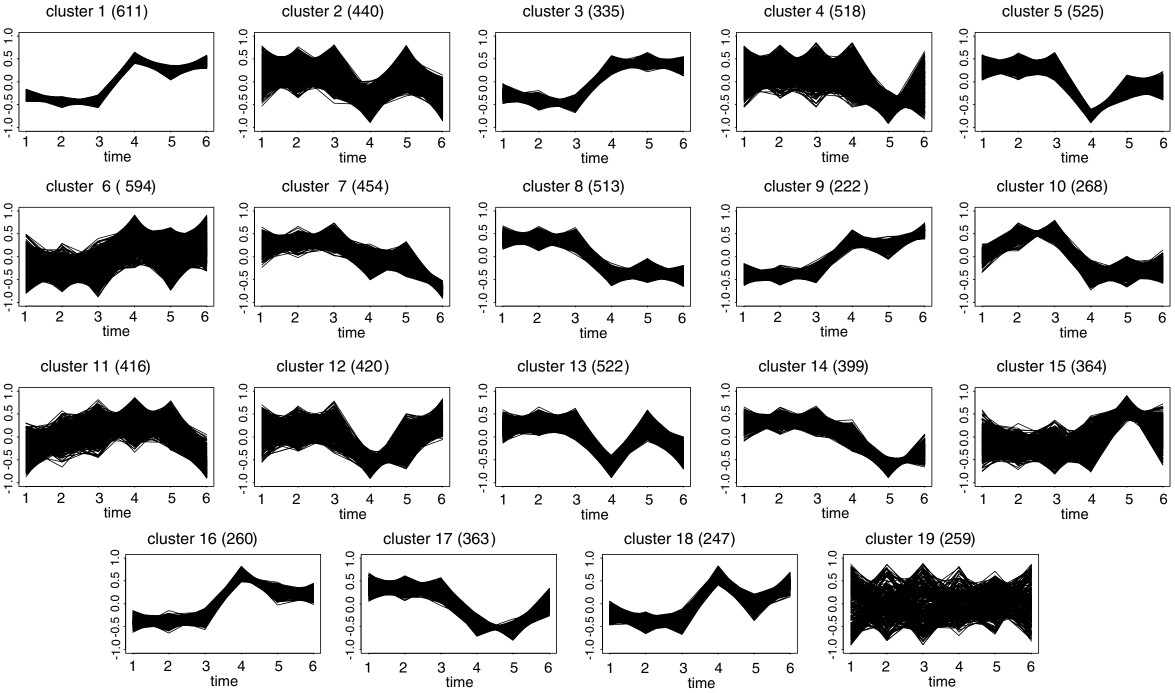

Fig. 15. Clusters identified by MCLUST with noise for the avian data.

DAC. The difference between these two methods is obvious. The noise modeling method is model-based and requires the non-noise data to satisfy some distribution assumptions. When the assumptions are violated, the clustering results may be sub-optimal. Typically, similar patterns will be clustered into several clusters. DAC is a model-free method. It is robust to the distribution of the data. In addition, DAC sets up a parameter (δ_{low}) to control the similarity of clusters. This leads to the phenomenon that the DAC clusters are more separable than the clusters produced by other methods. For example, in Fig. 5, SOM clusters 8 and 9 are almost non-separable; in Fig. 6, AQC clusters 5 and 9 are very similar; and in Fig. 15, MCLUST clusters 1, 3, 9, 16 and 18 are very similar. Highly similar clusters are not observed in the DAC clusters.

Acknowledgements

The authors thank the editor, associate editor and two referees for their constructive comments which have led to a significantly improvement of the paper. Liang's research is partially supported by grants from the National Science Foundation (DMS-0405748) and the National Cancer Institute (CA104620). Wang's research is partially supported by a grant from the National Cancer Institute (CA74552).

References

Banfield, J.D., Raftery, A.E., 1992. Model-based Gaussian and non-Gaussian clustering. Biometrics 49, 803–821.

Ben-Dor, A., Shamir, R., Yakhini, Z., 1999. Clustering gene expression patterns. J. Comput. Biol. 6, 281–297.

Campbell, J.G., Fraley, C., Murtagh, F., Raftery, A.E., 1997. Linear flaw detection in woven textiles using model-based clustering. Pattern Recognition Lett. 18, 1539–1548.

Campbell, J.G., Fraley, C., Stanford, D., Murtagh, F., Raftery, A.E., 1999. Model-based methods for real-time textile fault detection. Internat. J. Imaging Syst. Technol. 10, 339–346.

Carr, D.B., Somogyi, R., Michaels, G., 1997. Templates for looking at gene expression clustering. Statist. Comput. Statist. Graphics Newslett. 8, 20–29.

Chen, G., Jaradat, S.A., Banerjee, Tanaka, T.S., Ko, M.S., Zhang, M.Q., 2002. Evaluation and comparison of clustering algorithms in analyzing ES cell gene expression data. Statist. Sinica 12, 241–262.

Cho, R.J., Campbell, M.J., Winzeler, E.A., Steinmetz, L., Wodicka, L., Wolfsberg, T.G., Gabrielian, A.E., Landsman, D., Lockhart, D.J., Davis, R.W., 1998. A genome wide transcriptional analysis of the mitotic cell cycle. Mol. Cell 2, 65–73.

Dasgupta, A., Raftery, A.E., 1998. Detecting features in spatial point processes with clutter via model-based clustering. J. Amer. Statist. Assoc. 93, 294–302.

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood for incomplete data via the EM algorithm (with discussion). J. Roy. Statist. Soc. B 39, 1–38.

De Smet, F., Mathys, J., Marchal, K., Thijs, G., Moor, B.D., Moreau, Y., 2002. Adaptive quality-based clustering of gene expression profiles. Bioinformatics 18, 735–746.

Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D., 1998. Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. USA 95, 14863–14868.

Fraley, C., Raftery, A.E., 2002. Model-based clustering, discriminant analysis, and density estimation. J. Amer. Statist. Assoc. 97, 611–631.

Hartuv, E., Schmitt, A., Lange, J., Meier-Ewert, S., Lehrach, H., Shamir, R., 2000. An algorithm for clustering cDNA fingerprints. Genomics 66, 249–256.

Hastie, T., Tibshirani, R., Eisen, M.B., Alizadeh, A., Levy, R., Staudt, L., Chan, W.C., Botstein, D., Brown, P., 2000. 'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. Genome Biol. 1, research 0003.1–0003.21.

Heyer, L.J., Kruglyak, S., Yooseph, S., 1999. Exploring expression data: Identification and analysis of coexpressed genes. Genome Res. 9, 1106–1115.

Jiang, D., Pei, J., Zhang, A., 2003. DHC: A density-based hierarchical clustering method for time series gene expression data. In: Proc. BIBE 2003: Third IEEE Internat. Symp. Bioinformatics and Bioeng.

Jiang, D., Tang, C., Zhang, A., 2004. Cluster analysis for gene expression data: A survey. IEEE Trans. Knowledge Data Eng. 16, 1370–1386.

Kaufman, L., Rousseeuw, P.J., 1988. Finding Groups in Data. Wiley, New York.

Luan, Y., Li, H., 2003. Clustering of time-course gene expression data using a mixed-effects model with B-splines. Bioinformatics 19, 474–482.

McLachlan, G., Peel, D., 2000. Finite Mixture Models. John Wiley & Sons, New York.

McLachlan, G., Bean, R.W., Peel, D., 2002. A mixture model-based approach to the clustering of microarray expression data. Bioinformatics 18, 413–422.

Medvedovic, M., Sivaganesan, S., 2002. Bayesian infinite mixture model based clustering of gene expression profiles. Bioinformatics 18, 1194–1206.

Medvedovic, M., Yeung, K.Y., Bumgarner, R.E., 2004. Bayesian mixture model based clustering of replicated microarray data. Bioinformatics 20, 1222–1232.

Murtagh, F., 1983. A survey of recent advances in hierarchical clustering algorithms. Comput. J. 26, 354–359.

Rousseeuw, P.J., 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J. Computat. Appl. Math. 20, 53–65.

Shamir, R., Sharan, R., 2002. Algorithmic approaches to clustering gene expression data. In: Jiang, T., Xu, Y., Zhang, M.Q. (Eds.), Current Topics in Computational Molecular Biology. The MIT Press, pp. 269–299.

Sharan, R., Shamir, R., 2000. CLICK: A clustering algorithm with applications to gene expression analysis. In: Mewes, H.W., Seidel, H., Weiss, B. (Eds.), Proc. 38th Erns Schering workshop on Bioinformatics and Genome Analysis. Springer-Verlag, pp. 83–108.

Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E.S., Golub, T.R., 1999. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. Proc. Natl. Acad. Sci. USA 96, 2907–2912.

Tavazoie, S., Hughes, J.D., Campbell, M.J., Cho, R.J., Church, G.M., 1999. Systematic determination of genetic network architecture. Nat. Genet. 22, 281–285.

Tou, J.T., Gonzalez, R.C., 1979. Pattern classification by distance functions. In: Tou, J.T., Gonzalez, R.C. (Eds.), Pattern Recognition Principles. Addison-Wesley, Reading, MA, pp. 75–109.

Tseng, G.C., 2005. A comparative review of gene clustering in expression profile. In: The 8th Internat. Conf. on Control, Automation, Robotics and Vision (ICARCV), pp. 1320–1324.

Tseng, G.C., Wong, W.H., 2005. Tight clustering: A resampling-based approach for identification stable and tight patterns in data. Biometrics 61, 10–16.

Wakefield, J., Zhou, C., Self, S., 2003. Modeling gene expression over time: Curve clustering with informative prior distributions. In: Bernardo, J.M., Bayarri, M.J., O, B.J., Dawid, A.P., Heckerman, D., Smith, A.F.M., West, M. (Eds.), Bayesian Statistics 7. Clarendon Press, Oxford.

Yeung, K.Y., Fraley, C., Murua, A., Raftery, A.E., Ruzzo, W.L., 2001. Model-based clustering and data transformations for gene expression data. Bioinformatics 17, 977–987.