Biometrics 61, 10–16
March 2005

Tight Clustering: A Resampling-Based Approach for Identifying Stable and Tight Patterns in Data

George C. Tseng

Department of Biostatistics and Department of Human Genetics, University of Pittsburgh,
Pittsburgh, Pennsylvania 15261, U.S.A.
email: ctseng@pitt.edu

and

Wing H. Wong

Department of Statistics and Department of Biostatistics, Harvard University,
Cambridge, Massachusetts 02138, U.S.A.
email: wwong@stat.harvard.edu

Summary. In this article, we propose a method for clustering that produces tight and stable clusters without forcing all points into clusters. The methodology is general but was initially motivated from cluster analysis of microarray experiments. Most current algorithms aim to assign all genes into clusters. For many biological studies, however, we are mainly interested in identifying the most informative, tight, and stable clusters of sizes, say, 20–60 genes for further investigation. We want to avoid the contamination of tightly regulated expression patterns of biologically relevant genes due to other genes whose expressions are only loosely compatible with these patterns. "Tight clustering" has been developed specifically to address this problem. It applies *K*-means clustering as an intermediate clustering engine. Early truncation of a hierarchical clustering tree is used to overcome the local minimum problem in *K*-means clustering. The tightest and most stable clusters are identified in a sequential manner through an analysis of the tendency of genes to be grouped together under repeated resampling. We validated this method in a simulated example and applied it to analyze a set of expression profiles in the study of embryonic stem cells.

Key words: Microarray; Resampling; Scattered points; Unsupervised learning.

1. Introduction

Cluster analysis, an unsupervised learning method, is widely used to study the structure of the data when no specific response variable is specified. Our task is to learn the structure of a d-dimensional distribution based on training data of n observations from this distribution. The training data are represented by an $n \times d$ matrix. Given a measure of distance or dissimilarity between any pair of points, the goal is to divide these n points into a number of clusters. Many methods for clustering are now available. These methods roughly fall into two categories, namely heuristic algorithms and model-based analyses. In heuristic algorithms, no probabilistic model is specified. Instead, clustering is obtained either by optimizing a certain target function or iteratively agglomerating (or dividing) nodes to form bottom-up (top-down) trees. Examples of these approaches include K-means clustering and hierarchical clustering. Discussion of many popular methods can be found in Chapter 14 of Hastie, Tibshirani, and Friedman (2001). Another type of popular heuristic approach is to first search for small tight clusters (so-called kernels) and then expand these kernels into a full clustering. An example is the CLICK algorithm by Sharan and Shamir (2000).

In contrast to heuristic methods, model-based clustering methods make inferences based on a probabilistic assumption of the data distribution. Fraley and Raftery (1998) built a Gaussian mixture model for clustering and the EM algorithm was used to maximize the resulting classification likelihood. Then the Bayesian information criterion (BIC) (Schwarz, 1978) is used to select complexity of cluster structure and the number of clusters k; for more references, see also Day (1969), McLachlan and Basford (1988), Yeung et al. (2001), and McLachlan, Peel, and Bean (2003). Another approach of model-based clustering relies on prior specifications of unknown parameters and Bayesian procedures for selecting cluster structure and k, normally via Markov chain Monte Carlo simulation for determining the posterior distribution (see Medvedovic and Sivaganesan, 2002; Liu et al., 2003).

In cluster analyses of microarray experiments, we start with a data matrix $\{\theta_{ij}\}_{n\times d}$, an $n \times d$ matrix representing the expression levels of n genes in d samples. If the goal is to obtain sets of genes with similar expression patterns that are likely to belong to similar functional pathways, we will cluster n points in d-dimensional space under a given distance (dissimilarity) measure. To find groups of samples with similar expression patterns, we cluster samples instead of genes, resulting in d points in the n-dimensional space being clustered. This is useful, for example, in the discovery of subtypes of a disease. Microarray experiments normally have 500–3000 genes after filtering out genes with low information content and 10–500 samples, depending on the study.

Most clustering algorithms assign all points into clusters. However, in microarray experiments, we expect many genes to be unrelated to the biological processes that we are investigating and to show uncorrelated variations with any cluster of genes. These genes should not be assigned into any specific cluster, and are thus called "scattered genes." When analyzing data with scattered genes, if the algorithm is forced to divide all points into clusters, both the estimation of the number of clusters will be problematic and the resulting clusters will be distorted and difficult to interpret. In a model-based approach, Fraley and Raftery (1998) modeled outliers by adding a Poisson process component in the mixture model for clustering. However, it has not been found generally successful in clustering genes as the method heavily relies on the correct model specification, estimation of k, and a good initial value for EM algorithm. To our knowledge, current popular methods are rarely shown to adequately deal with scattered genes.

Another important issue in cluster analysis is the estimation of the number of clusters, k. Among the many published rules in the literature for estimating k, none have enjoyed superior performance over the others in general. Usually, some rules work better than the others only in some special simulated examples. Milligan and Cooper (1985) performed a comprehensive comparison of over 30 published rules and identified several as better rules. Very recently, Tibshirani et al. (2001) introduced a promising method that utilized resampling techniques. They selected k to maximize the prediction rate estimated by resampling (see also Dudoit and Fridlyand, 2002). In this article, we have further developed the resampling approach to identify tight and stable clusters. In our approach, the tight clusters are obtained sequentially, usually in the order of decreasing stability, and the choice of k then becomes secondary. This approach is especially appropriate in the presence of scattered points.

This article is organized as follows. In Section 2.1, we discuss several challenging issues in the use of cluster analysis on microarray data and discuss why current methods are inadequate for these tasks. In Section 2.2, we illustrate the difficulties associated with the initial values and local minimum problem of K-means algorithm. A new method to obtain initial values for K-means algorithm is then proposed to overcome this difficulty. In Section 2.3, a resampling procedure is used to select tight cluster candidates where the exact number of clusters, k, becomes less crucial. A recursive procedure is then applied to produce tight and stable clusters. In Section 3, we present results from a simulation study to illustrate the improvement offered by this approach. We also illustrate the method on a set of expression profiles produced in the study of mouse embryonic development. Finally, we provide conclusions and further discussions in Section 4.

2. Methods

We present our discussion in the framework of microarray experiments, but conceptually this is a general algorithm that can be used to sequentially identify tight clusters in any unsupervised learning situation. For simplicity, we use K-means as the partition engine in the algorithm and assume that the data are in Euclidean space with the usual Euclidean distance as the dissimilarity measure for clustering. As will be discussed in Section 4, K-means can be replaced by other clustering algorithms if needed.

2.1 *Motivation*

Microarray experiments allow simultaneous monitoring of thousands of genes' activities (Brown and Botstein, 1999). In contrast to the traditional hypothesis-driven experiments in biological science, this is a data-driven approach to generate biological hypotheses and models, and to guide further experiments. Many popular clustering algorithms have been used to explore microarray data including hierarchical clustering, K-means, K-memoids or partition around medoids (PAM), and self-organizing map. They usually require the estimation of the number of clusters, k, and then the algorithm assigns all data points into one of the k clusters. Almost all of these algorithms have to assign all genes into clusters. As a result, many genes unrelated to the underlying biological pathway are falsely classified to the tight clusters of interest. These genes may corrupt and dilute the information contained in these clusters. An example of this situation is shown in Figure 1. The microarray data of *Drosophila* life cycle (Arbeitman et al., 2002) are clustered by K-means algorithm with $k = 10$, 15, and 30, and the result is shown as a heat map. A heat map is a useful tool to visualize the clustering result in a high-dimensional dataset. Instead of presenting the raw

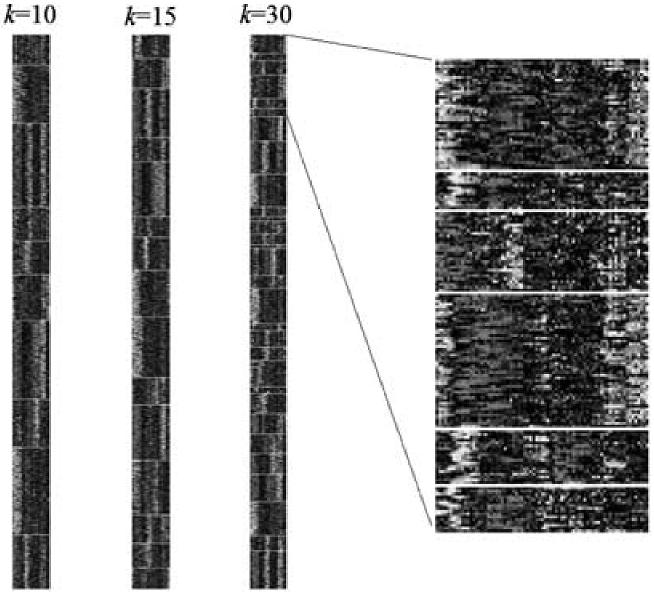

Figure 1. Clustering using K-means algorithm with arbitrary k.

data in numbers, it demonstrates the data matrix by gradient colors.

It is usually infeasible to estimate the number of clusters, k, in microarray experiments except for some rare situations such as cell cycle experiments in Cho et al. (1998) and Spellman et al. (1998). At a first glance at Figure 1, all three clustering results seem to show clear and informative cluster patterns no matter which k we use. However, a closer look at even the most homogeneous clusters (those obtained with $k = 30$) shows loose and contaminated patterns due to the assignment of scattered genes into clusters. Research in cognitive science has shown that human visualization has a tendency to overfit, that is, to suppress the noises and accentuate the pattern (Gilovich, Vallone, and Tversky, 1985). Because microarray analysis is often used as an exploratory tool to guide further investigations and these biological experiments are usually costly, the inclusion of false-positive genes is highly undesirable. Moreover, as we will discuss in Section 2.2, scattered genes often hamper the ability of a clustering algorithm to find a good cost-function minimum in the space of partitions. Thus, there is need for a clustering algorithm that can directly identify the most informative, tight, and stable clusters in high-dimensional data.

2.2 *Overcoming the Local Minimum Problem in K-Means Clustering*

Because K-means clustering will be used as an intermediate engine in the tight clustering algorithm in Section 2.3, we will first address an important but often overlooked issue in its implementation, that is, the problem of local minima. The K-means clustering algorithm aims to divide data points into clusters so that the within-cluster dispersion (sum of squares) is minimized (MacQueen, 1965; Hartigan and Wong, 1979). In general, it is not computationally feasible to search for the global minimum. Instead, the algorithm performs iterative reallocation until the within-cluster dispersion stabilizes. Different initial values for the algorithm may result in different clustering results. Often, with a poor initial value, the minimization falls in a local minimum quickly and gives an undesirable clustering result. This problem is especially pronounced when scattered points exist. To illustrate this problem, a total of 90 points from three clusters (in black) and scattered points (in gray) are simulated (Figure 2). Hierarchical clustering with single and complete linkage is performed. The hierarchical trees are then cut to produce three clusters. The cluster centers are denoted by "o" for single linkage and "x" for complete linkage in the first plot of Figure 2. These cluster centers are then used as initial values to perform K-means clustering, respectively. As seen in the second plot of Figure 2, K-means clustering with single linkage initial value (cluster centers indicated as "o") gives an adequate clustering as hoped (within-cluster sum of squares 305.09) while K-means clustering with complete linkage initial value (denoted by "x") falls into a local minimum and results in a very poor clustering (within-cluster sum of squares 965.32). In other replications, single linkage may also perform poorly.

Multiple starting initial values and stochastic methods, such as simulated annealing and genetic algorithms, are often used to overcome this problem. They, however, multiply the computation complexity and the global minimum is often still not obtained. Alternatively, hierarchical clustering can be used to provide an initial value for K-means clustering as in the above example. `S-Plus` adopts this as default when the initial value is not provided by the user while `R` uses a randomly generated initial value. The hierarchical initial values work well when clusters are well separated. However, as our example suggests, this initial value can still fall into a local minimum when cluster boundaries are vague or scattered points are present.

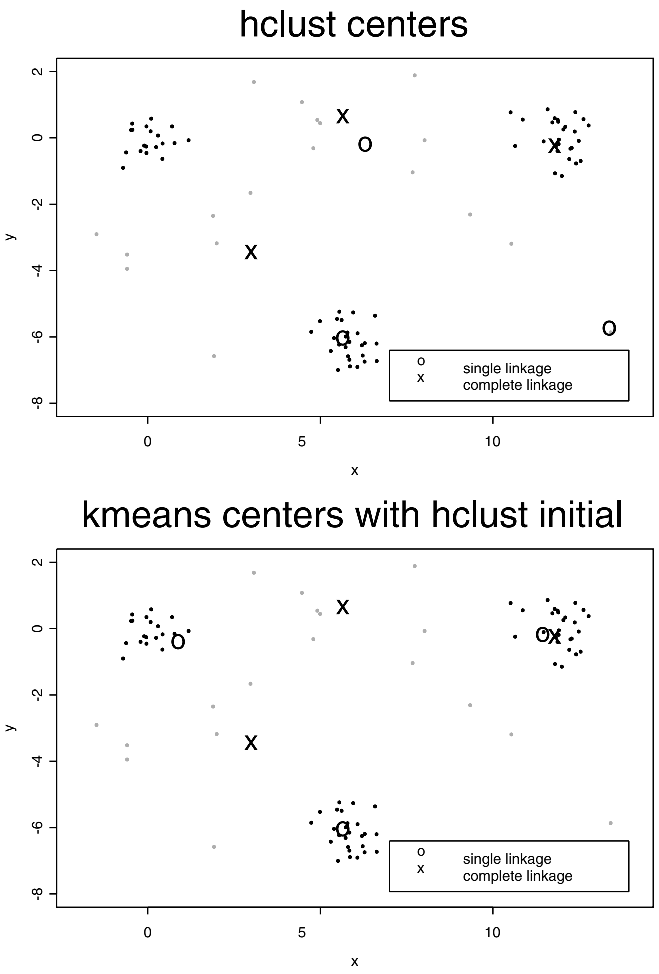

Figure 2. Cluster centers obtained from hierarchical clustering are shown in the upper plot (o: single and x: complete linkage) and are used as K-means initial values. The resulting cluster centers of K-means are shown in the lower plot.

Here, we propose an alternative initial value that is effective in such situations. First, we cut the hierarchical clustering tree to obtain $p \times k$ clusters. Among these $p \times k$ clusters, we choose the k clusters consisting of the largest number of points, and use their cluster centers as the initial value for K-means algorithm.

We test this new approach by a simulation. Three normally distributed clusters centered at (0, 0), (12, 0), and (6, −6) are generated and then scattered points are added. (See Section 3.1 for detailed construction of the simulation.) We

Table 1

Comparing average K-means error rates using different initial values in 100,000 *simulations. For the simulation settings, "*50 × 3 + 10*" means three clusters each containing* 50 *points plus* 10 *scattered points are generated.*

		HS1	HC1	HS3	HC3	HS6	HC6
$p \times k$	R	1×3	1×3	3×3	3×3	6×3	6×3
50 × 3 + 50	0.0366	0.0868	0.0553	0.0019	0.0001	0.0001	0.0049
50 × 3 + 10	0.0699	0.0451	0.0032	0	0.0072	0.0010	0.0446
100 × 3 + 100	0.0260	0.0536	0.0687	0.0008	0	0	0.0001

simulate 100,000 times. In each replication we cluster the data into three clusters using K-means with different choices of initial values: random initial values, initial values from hierarchical single linkage (HS1) and complete linkage (HC1), and early hierarchical tree truncation methods with $p = 3$ (HS3 and HC3) and $p = 6$ (HS6 and HC6). To assess errors in each clustering result, we calculate its R-value, which is defined as the sum of the square of distances from the computed cluster centers to the underlying true cluster centers. It is clear that convergence to local minima can be identified by large R-values. The result in Table 1 shows that simple hierarchical initial values (HS1 and HC1) do not offer an improvement over random initial values (R) in this example due to the existence of scattered points. On the other hand, our early truncation method significantly decreases the chance of converging to a local minimum because it effectively avoids including scattered points in the initial value. We also note that choosing p too large may lead to deterioration in the performance. Unless otherwise specified, we will use early truncation of single-linkage hierarchical tree with $p = 3$ as the initial value for K-means algorithm in all subsequent analyses. We note that this alternative initial value is also useful for other optimization-based clustering algorithms such as PAM.

2.3 *Tight Clustering*

The procedures of "tight clustering" are described below.

2.3.1 *Algorithm A.* This algorithm is used to select candidates of tight clusters when k in the K-means algorithm is prespecified. The subsampling procedure is used to create variabilities so that a pair of points stably clustered together can be distinguished from those clustered by chance.

(a) Take a random subsample X' from the original data X, say with 70% of the original sample size. Apply K-means with the prespecified k on X' to obtain the cluster centers $C(X', k) = (C_1, C_2, \ldots, C_k)$.

(b) Use the clustering result $C(X', k)$ as a classifier to cluster the original data X according to the distances from each point to the cluster centers. Following the convention of Tibshirani et al. (2001), the resulting clustering is represented by a comembership matrix $D[C(X', k), X]$ where $D[C(X', k), X]_{ij}$, the element of the matrix in row i and column j, takes value 1 if points i and j are in the same cluster and 0 otherwise.

(c) Repeat independent random subsampling B times to obtain subsamples $X^{(1)}, X^{(2)}, \ldots, X^{(B)}$. The average comembership matrix is defined as $\bar{D} = \text{mean}(D[C(X^{(1)}, k), X], \ldots, D[C(X^{(B)}, k), X])$.

(d) Search for a set of points $V = \{v_1, \ldots, v_m\} \subset \{1, \ldots, n\}$ such that $\bar{D}_{v_i v_j} \geq 1 - \alpha, \forall i, j$ where α is a constant close to 0. Order sets with this property by size to obtain V_{k1}, $V_{k2}, \ldots$ These V sets are candidates of tight clusters.

2.3.2 *Sequential identification of tight and stable clusters.* The following algorithm is used to identify a tight cluster that is stably chosen by consecutive k. After a tight and stable cluster is identified, it is removed from the whole data and the same procedure is repeated to identify the next tightest cluster. We first define a similarity measure of two sets V_i and V_j to be $s(V_i, V_j) = |V_i \cap V_j|/|V_i \cup V_j|$ where $|V|$ is the size of set V. Therefore, $s(V_i, V_j) = 1$ if and only if sets V_i and V_j are identical.

(a) Start with a suitable k_0. Apply algorithm A on consecutive k starting from k_0. Choose the top q tight cluster candidates for each k, namely $\{V_{k_0,1}, \ldots, V_{k_0,q}\}$, $\{V_{(k_0+1),1}, \ldots, V_{(k_0+1),q}\}, \ldots$ We use $q = 7$ throughout the article.

(b) Stop when $s(V_{k',l}, V_{(k'+1),m}) \geq \beta$. Here β is a constant close to 1, $k' \geq k_0$, and $1 \leq l, m \leq q$. Identify $V_{(k'+1),m}$ as a tight and stable cluster. Remove it from the whole data.

(c) Decrease k_0 by 1 and repeat steps (a) and (b) to identify the next tight cluster. The cluster selection terminates when k_0 is decreased to five or a user-specified target number of clusters is achieved.

Remark 1. Note that $\bar{D}_{ij}$ is an estimate of the probability of point i and j to be clustered together in each subsampling judgment.

Remark 2. Intuitively, α controls tightness and β controls stableness of selected clusters.

3. Examples

3.1 *Simulated Example with Scattered Points*

We simulate 14 two-dimensional normally distributed clusters with covariance matrices $\Sigma = (0.1)^2 I, (0.2)^2 I, \ldots, (1.4)^2 I$ each containing 50 points, where I is the identity matrix. Another 175 noise points are then uniformly added in the space. In each cluster, each point is generated within two standard deviations to its cluster center; otherwise, a new point is generated to replace it. The scattered points (noise samples) are uniformly distributed in the space that is more than three standard deviations away from each cluster center. This restriction eliminates any confusion of the definition of cluster points and scattered points.

Table 2
Tight clustering results on simulated data

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Noise
Truth	50	50	50	50	50	50	50	50	50	50	50	50	50	50	175
$k_0 = 10$	58	59	59	78	72	60									489
$k_0 = 20$	59	56	55	53	57	53	53	52	52	52	56	51	51	51	112
$k_0 = 25$	55	56	53	56	53	53	52	55	51	51	51	50	50	50	130
$k_0 = 40$	52	51	51	52	51	51	51	50	26	25	22	50	18	17	278

We perform tight clustering on these simulated data with parameters $\alpha = 0$, $\beta = 0.7$, $B = 10$, and $k_0 = 10$, 20, 25, and 40. The number of points in each cluster identified is shown in Table 2. When $k_0 = 10$ the algorithm has to stop when six clusters are identified because the algorithm reduces k_0 by 1 each time a tight cluster is identified and removed. Both $k_0 = 20$ and $k_0 = 25$ give all 14 correct tight clusters plus few surrounding scattered points. This suggests some robustness property on the selection of k_0 in this algorithm. However, when k_0 is too large ($k_0 = 40$) the algorithm splits tight clusters because we assigned the points into too many clusters. Larger k_0 usually results in smaller tight clusters. We suggest to set k_0 roughly within one- to twofold of underlying true k. In our experience, $B = 10$ is enough to produce satisfying clustering and setting stringent parameters such as $\alpha = 0$, $\beta = 0.8$ helps to produce smaller and tighter clusters but may miss some loose while interesting clusters.

3.2 *An Example from Functional Genomics*

We have applied tight clustering to a number of large-scale genomic datasets. The findings support the notion that smaller clusters of genes showing tight regulation of expression is biologically more relevant than larger clusters with loose patterns. In other words, exclusion of scattered genes from being clustered allows us to obtain the underlying biological implication of the clusters in a more reliable manner. We present here just one example that concerns the analysis of the gene expression profiles of 126 cell samples of a laboratory mouse. About half of the samples are from different stages of mouse embryonic development, and the remaining half is a diverse collection of samples from various tissues, including several types of adult stem cells. The samples were profiled using an oligonucleotide array (U74Av2 mouse array from Affymetrix, Santa Clara, CA) containing probe sets for about 10,000 mouse genes. We applied tight clustering on these data to obtain about 30 tight clusters showing a variety of distinct regulation patterns. The last plot of Figure 3 shows one of these tight clusters. A majority of the 26 genes in this cluster appeared to be involved in DNA replication. The expression of these genes was high mainly in several embryonic stages and in some adult stem cells, but not in differentiated tissue types. Particularly striking was the fact that 7 of the 26 probe sets in this cluster map to mammalian homologs of the mini-chromosome maintenance (MCM)-deficient genes in budding yeast. It is known that in mouse the six MCM proteins form a complex (Kimura et al., 1996) and that the disruption of any of the MCM genes results in yeast cells being unable to complete the

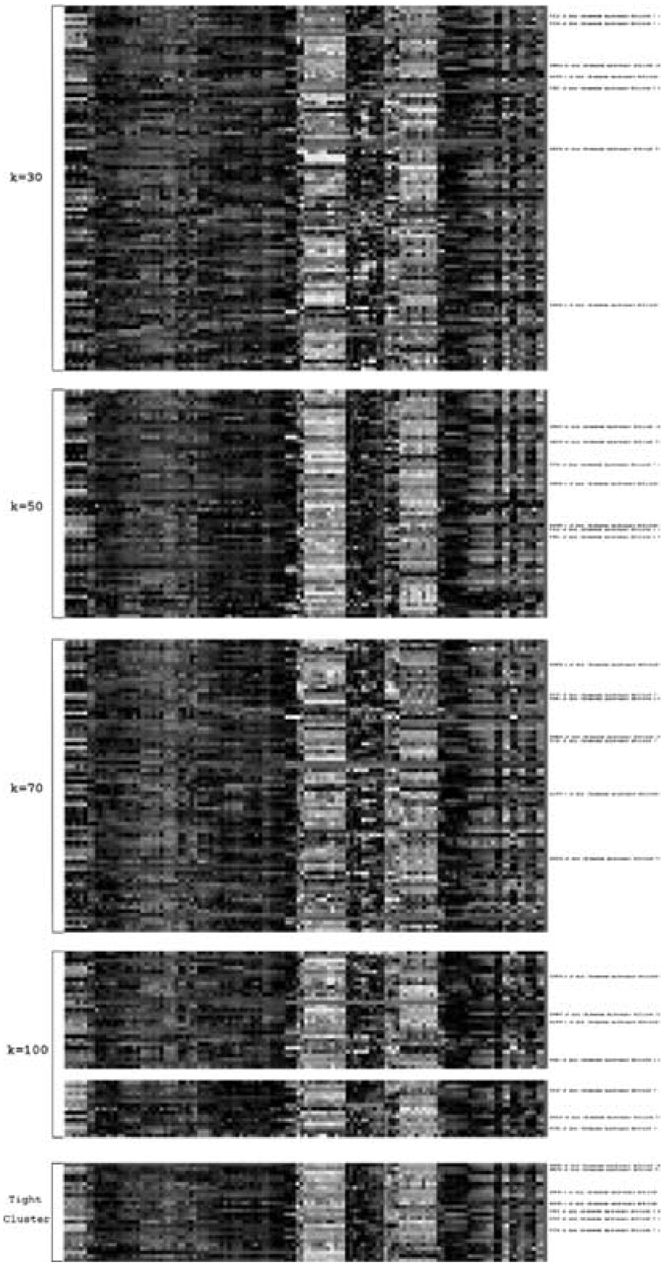

Figure 3. Selected clusters of tight clustering and K-means clustering with $k = 30$, 50, 70, and 100 containing seven MCM genes. The MCM genes are indicated on the right.

S phase in the cell cycle. To see whether the tight regulation of the MCM genes is easily detected using standard *K*-means clustering, we performed the *K*-means algorithm on this dataset several times, each with a different value of k ($k = 30, 50, 70, 100$). In each run, we selected the clusters that contained any of the MCM genes. For $k = 30$, 50, and 70, all MCM genes fell into one cluster but the cluster sizes (96, 60, and 77, respectively) were much larger than the one in tight clustering (the first three parts of Figure 3). For $k = 100$, the MCM genes were distributed in two different clusters (sizes 31 and 15), making it harder to detect the coregulation of the MCM genes.

4. Conclusion and Discussion

Tight clustering contains a novel concept that does not necessitate the estimation of the number of clusters and the assignment of all points into clusters. An immediate advantage is that it reduces the chance of including false-positive genes into the clusters. As a result, it allows us to concentrate on the more informative and biologically relevant genes. Current algorithms are problematic in situations where data are chaotic and have large numbers of scattered points. The resulting clusters are usually skewed or misleading due to the assignment of all points into clusters. Tight clustering alleviates this problem by only focusing on the core patterns and the result becomes more interpretable. Once the core patterns are learned, scattered genes with relatively loose correlations can be added to the cluster if necessary.

A mixture model-based approach has been used in cluster analysis. Through explicit modeling it provides a firm mathematical basis on estimation and statistical inference, and hence it represents a very attractive alternative to classical hierarchical or *K*-means clustering. On the other hand, in application to gene clustering based on microarray data, there can be several thousands of genes and the fitting of these models can fail to converge to the global optimum. In fact, we find the algorithm often finds undesirable local minimum even in the simple simulating setting in Section 3.1. The many strong assumptions underlying these models, such as Gaussian distribution, equal variances across clusters, or sphericity of the dispersion matrices, are also difficult to validate. This is especially true in large studies when the vector of expression values of a gene can be of high dimension (e.g., 100). Furthermore, the determination of the number of clusters remains a difficult issue in practice. The BIC for model selection is approximate and the convergence to local minimum makes it even more unstable. For these reasons, current model-based approaches may not conclude to a satisfying clustering result in the analysis of microarray data. In this article, we decide to explore the extension of classical *K*-means algorithms through resampling-based assessment and sequential identification of tight clusters. The careful comparison of these approaches awaits further investigations.

In this article, *K*-means clustering is used as an intermediate engine for tight clustering. We note that conceptually *K*-means can be replaced by any other clustering method including a model-based approach provided that the clustering result in subsamples can be used as a classifier to cluster the original data (step [b] in Section 2.3.1). For example, we can replace *K*-means with *K*-memoids so that a suitable dissimilarity measure other than Euclidean distance can be used for a specific data structure. An example is when we have information on the measurement variability of each sample (variable). In this case, an inverse weighting of these measurement variabilities in the distance calculation (so-called variability-weighted similarity) is more proper.

An alternative approach for tight clustering is to identify and exclude scattered points so that the remaining data form a tighter clustering. We have tried a similar resampling procedure in this approach. However, our experience shows that it is less effective than the method we propose here.

Resampling procedures have been widely used in supervised learning to improve classification performance (e.g., bagging, committee algorithm, and random forests in Breiman [2001]), but have not been widely applied in cluster analysis except for estimating the number of clusters in Tibshirani et al. (2001). Further methodological exploration and studies of theoretical foundation of resampling methods in clustering are worth pursuing in the future.

A `C` library and a stand-alone package for implementing the method and data visualization can be downloaded from `http://www.pitt.edu/∼ctseng/`.

Acknowledgements

This work was supported by NSF grant DMS-0090166. The authors thank C. Li, R. S. Kim, X. Zhou, and H. Huang for applying methods on their projects and comments on the algorithm and software package. They also thank the associate editor for comments and providing references.

Résumé

Dans ce papier, nous proposons une méthode de classification qui produit des classes étroites et stables sans forcer tous les points au sein des classes. La méthodologie présentée à une portée générale mais fut initialement développée pour l'analyse des classes dans les études de puces. La plupart des algorithmes disponibles visent à assigner tous les gènes au sein des clusters. Dans beaucoup d'études biologiques, cependant, nous sommes plutôt intéressés par identifier les classes les plus stables et les plus informatives de telle façon que, par exemple, de 20 à 60 gènes seulement soit l'objet d'investigations complémentaires. Nous souhaitons éviter la contamination de profils d'expression finement régulés d'un petit nombre de gènes biologiquement pertinent par des gènes dont le profil d'expression n'est que faiblement compatible avec le profil précédent. La 'Classification Etroite' a été développée pour répondre spécifiquement à ce problème. Elle utilise la classification de type 'K-means' comme un moteur de classification intermédiaire. La troncature précoce des arbres hiérarchiques est utilisée pour surmonter le problème de minima locaux rencontrés avec l'approche K-means. La classe la plus stable et le et la plus serrée est identifiée de façon séquentielle par l'analyse de la tendance de certains gènes à être groupés ensemble lors de rééchantillonnages répétés. Nous validons cette méthode à l'aide d'un exemple simulé et nous l'appliquons à l'analyse d'un jeu de profils d'expression dans le cadre d'une études sur les cellules souches embryonnaires.

References

Arbeitman, M., Furlong, E., Imam, F., Johnson, E., Baker, B., Davis, R., and White, K. (2002). Gene expression during the life cycle of *Drosophila melanogaster*. *Science* **297,** 2270–2275.

Breiman, L. (2001). Random forests. *Machine Learning* **45,** 5–32.

Brown, P. T. and Botstein, D. (1999). Exploring the new world of the genome with DNA microarrays. *Nature Genetics* **21** (1 suppl.), 33–37.

Cho, R. J., Campbell, M. J., Winzeler, E. A., Steinmetz, L., and Davis, R. W. (1998). A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell* **2,** 65–73.

Day, N. E. (1969). Estimating the components of a mixture of two normal distributions. *Biometrika* **56,** 463–474.

Dudoit, S. and Fridlyand, J. (2002). A prediction-based resampling method for estimating the number of clusters in a dataset. *Genome Biology* **3,** 0036.1–0036.21.

Fraley, C. and Raftery, A. E. (1998). How many clusters? Which cluster method? Answers via model-based cluster analysis. *Computer Journal* **41,** 578–588.

Gilovich, T., Vallone, R., and Tversky, A. (1985). The hot hand in basketball: On the misperception of random sequences. *Cognitive Psychology* **17,** 295–314.

Hartigan, J. A. and Wong, M. A. (1979). A *K*-means clustering algorithm. *Applied Statistics* **28,** 126–130.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning.* New York: Springer-Verlag.

Kimura, H., Ohtomo, T., Yamaguchi, M., Ishii, A., and Sugimoto, K. (1996). Mouse MCM proteins: Complex formation and transportation to the nucleus. *Genes Cells* **1,** 977–993.

Liu, J. S., Zhang, J. L., Palumbo, M. J., and Lawrence, C. E. (2003). Bayesian clustering with variable and transformation selections. *Bayesian Statistics* **7,** 249–275.

MacQueen, J. (1965). On convergence of *k*-means and partitions with minimum average variance. *Annals of Mathematical Statistics* **36,** 1084.

McLachlan, G. J. and Basford, K. E. (1988). *Mixture Models: Inference and Applications to Clustering.* New York: Marcel Dekker.

McLachlan, G. J., Peel, D., and Bean, R. W. (2003). Modelling high-dimensional data by mixtures of factor analyzers. *Computational Statistics & Data Analysis* **41,** 379–388.

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

Milligan, G. W. and Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika* **50,** 159–179.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics* **6,** 461–464.

Sharan, R. and Shamir, R. (2000). Click: A clustering algorithm with applications to gene expression analysis. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 307–316.

Spellman, P. T., Sherlock, G., Zhang, M. Q., Brown, P. O., Botstein, D., and Futcher, B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Molecular Biology of the Cell* **9,** 3273–3297.

Tibshirani, R., Walther, G., Bostein, D., and Brown, P. O. (2001). *Cluster Validation by Prediction Strength.* Technical report, Department of Statistics, Stanford University.

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

Received October 2003. *Revised April* 2004.
Accepted April 2004.