Tight Clustering: a Resampling-based Approach for Identifying Stable and Tight Patterns in Data

George C. Tseng, and Wing Hung Wong

October 6, 2003

Abstract

In this paper 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 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.

keyword: unsupervised learning, resampling, scattered points, microarray.

George C. Tseng is assistant professor, Department of Biostatistics and Human Genetics, University of Pittsburgh, PA 15261 (E-mail: ctseng@pitt.edu). Wing

Hung Wong is Professor, Department of Statistics and Biostatistics, Harvard University, MA 02138 (E-mail: wwong@hsph.harvard.edu). This work was supported by NIH grant P20-CA96470, and 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.

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 a 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 et al. (2001). In the situation of optimization-based clustering algorithm, the optimization is subject to local optimum problems. We will discuss the problem of local minimum of K-means algorithm in Section 2.2 and propose a method to overcome this problem. Another type of popular heuristic approach is to first search for kernels, small tight clusters of about

6-10 points, and then use an algorithm to expand these kernels into a full clustering. One useful algorithm is the CLICK algorithm by Sharan and Shamir (2000), which was specially developed to cluster microarray data.

In contrast to heuristic methods, model-based clustering methods make inferences based on a probabilistic assumption of the data distribution (Fraley and Raftery, 1998; Yeung et al., 2001; Medvedovic and Sivaganesan, 2002). Fraley and Raftery (1998) built a Gaussian mixture model for clustering and the EM algorithm was used to maximize the resulting classification likelihood. In this approach, the complexity of the model varies depending on the parametrization of the covariance matrices in each cluster. An attractive feature of their method is the natural use of Bayesian Information Criterion (BIC) (Schwarz, 1978) to compare different models including estimating the number of clusters and determining the complexity of parametrization of the cluster covariance matrices. Another approach, used by Medvedovic and Sivaganesan (2002), applies a Bayesian infinite mixture model with hyperparameters estimated from the empirical data. The number of clusters and the clustering are then determined by the joint posterior distribution produced from Markov Chain Monte Carlo (MCMC) simulations. These model-based approaches perform well in many cases but the computation becomes intractable when the dimensionality (d) of the data is large. In this case, computation is feasible only under restrictive assumptions on the form of the covariance matrices.

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 which 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 n-dimensional space being clustered. This is useful, for example, in the discovery of subtypes of a disease. Microarray experiments normally have 500 to 3000 genes after filtering out genes with low information content and 10 to 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, model-based clustering has found limited use in large-scale microarray analysis due to the complexity of the computation and the difficulty of specifying distributions in high dimensional situation. 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. Many methods aim to balance the within-cluster dispersion $W(k)$ and the number of clusters k. Among the many published rules in the

literature, none of them 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 paper, 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 paper is structured 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 presented results from 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 monitor of thousands of genes' activities (Brown and Botstein, 1999). In contrast to traditional hypothesis-driven experiments in biological science, it 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 (Hartigan, 1975), K-means (Hartigan and Wong, 1979), K-memoids or Partition Around Medoids (PAM) (Kaufman and Rousseeuw, 1990), and Self-Organizing Map (SOM) (Kohonen, 1997). They usually require the estimation of the number of clusters, k, and then the algorithm assigns all data points into one of 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) is clustered by K-means algorithm with k=10, 15 and 30, and the result is shown as a heat map. Heat map is a useful tool to visualize the clustering result in a high-dimensional data set. Instead of presenting raw data in numbers, it demonstrates the data matrix by gradient colors: red for positive numbers, green for negative, and black for zero. In this paper, heat map figures are plotted with the aid of TreeView, a package for expression profile visualization by Eisen (2000).

It is usually infeasible to estimate the number of clusters, k, in microarray experiments except for some rare situations such as cell cycle experiments (Cho et al., 1998; 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, namely, to suppress the noises and accentuate the pattern (Gilovich et al., 1985). Since 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 finding a good cost-function minimum in the space of partitions. Thus there is a 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

Since 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, namely 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 Hartigan and Wong (1979). In general it is not computationally feasible to search for the global minimum. Instead, the algorithm performs iterative re-allocation 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 3 clusters (in black) and scattered points (in blue) are simulated (Figure 2). Hierarchical clustering with single and complete linkage are 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.

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. If we choose $p = 3$ and in our example where $k = 3$, we obtain 9 clusters from the hierarchical tree. 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 simulate 100,000 times. In each replication we cluster the data into 3 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 clus-

tering result, we calculate its R-value which is defined as the sum of square of distances from the computed cluster centers to the underlying true cluster centers. Figure 3 shows the histogram of R-values from 100,000 simulated data using random initial values. 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 Partitioning Around Medoids (PAM).

2.3 Tight Clustering

The procedures of "Tight Clustering" are described below. We first describe an algorithm (Algorithm A) that identifies candidates of tight cluster by repeated use of K-means with fixed k. We then vary k to identify the largest tight cluster that is stably produced by Algorithm A in consecutive k. This tight cluster is selected and removed from the whole data, and we proceed with the same procedure to find the next tight cluster in the remaining data.

2.3.1 Algorithm A

The following algorithm is used to select candidates of tight clusters when k in the K-means algorithm is pre-specified. 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 pre-specified k on X' to obtain the cluster centers $C(X', k) = (C_1, C_2, \cdots, C_k)$.

(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 co-membership 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 point i and j are in the same cluster and 0 otherwise.

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

(d) Search for a set of points $V = \{v_1, \cdots, v_m\} \subset \{1, \cdots, n\}$ 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 $V_{k1}, V_{k2}, \cdots$. 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 $\{V_{k_0,1}, \cdots, V_{k_0,q}\}$, $\{V_{(k_0+1),1}, \cdots, V_{(k_0+1),q}\}, \cdots$. We use $q = 7$ throughout the paper.

(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 the tight and the most stable cluster. Remove it from the whole data.

(c) Decrease k_0 by 1 and repeat step (a) and (b) to identify the next tight cluster.

Remark 1: Note that $\bar{D}_{ij} = 1$ means that point i and j are always clustered together in each subsampling judgment. Similarly $\bar{D}_{ij} = 0$ when point i and j are never clustered together and $\bar{D}_{ii} = 1, \forall i$.

Remark 2: Intuitively, α controls tightness and β controls stability. When $\alpha = 0$, we are looking for a set, V, such that all points in the set are always clustered together in the B subsampling judgment. When $\beta = 1$, we seek an exactly identical set that is stably chosen for consecutive k.

3 Examples

3.1 Simulated example with scattered points

We simulate 14 two-dimensional normally distributed clusters with covariance matrices $\Sigma = (0.1)^2 I, (0.2)^2 I, \cdots, (1.4)^2 I$ 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. Figure 4 is an example generated from this distribution.

We perform Tight Clustering on this 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 are shown in Table 2. When $k_0 = 10$ the algorithm has to stop when six clusters are identified since the algorithm reduces k_0 by 1 each time a tight cluster is identified and removed. The clustering result with $k_0 = 25$ is shown in the lower plot of Figure 4. We note that 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 and some tight clusters are occasionally torn apart during subsampling judgment.

3.2 An example from functional genomics

We have applied Tight Clustering to a number of large-scale genomic data sets. 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 on 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) containing probe sets for about 10,000 mouse genes. We applied Tight Clustering on this data to obtain about 30 tight clusters showing a variety of distinct regulation patterns. The last plot of Figure 5 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 were 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, 1996) and that disruption of any of the MCM genes results in yeast cells unable to complete the S phase in cell cycle (Labib, 2000). To see whether the tight-regulation of the MCM genes are 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 (Figure 5). For k=100, the MCM genes were distributed in two different clusters (sizes 31 and 15), making it harder to detect the co-regulation 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.

In this paper K-means clustering is used as an intermediate engine for Tight Clustering. We note that K-means can be replaced by any other clustering method provided that the clustering result in subsample can be used as a classifier to cluster the original data (step (b) in Section 2.3.1). For

example, we can use PAM to replace K-means to gain more flexibility on dissimilarity measure since K-means only works on Euclidean space with Euclidean distances as the dissimilarity measure. This modification is helpful in some situations when we want to select a suitable dissimilarity matrix to deal with specific data structures. 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 (Yeung et al., 2003).

One potential problem of Tight Clustering is that the result is not deterministic due to the involvement of resampling procedures. Figure 6 shows two repeated Tight Clustering results when $\alpha = 0.1, \beta = 0.6, B = 10$ and $k_0 = 35$ but with different resampling random seeds. The clustering results are similar in most of the larger tight clusters but not identical for the remaining ones. Methods for combining Tight Clustering results from different random seeds can be developed in the future to pursue a better clustering result.

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 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 Brieman (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.

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. O., 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.

Dudoit, S. and Fridlyand, J. (2002). A prediction-based resampling method for estimating the number of clusters in a dataset. *Genome Biology.* 3, 0036:1-21.

Eisen, M. B. (2000). TreeView. (version 1.5)
Software download: http://rana.lbl.gov/.

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

Gilovich, T., Vallone, R., and Tversky, A. (1995). The hot hand in basketball: On the miperception of random sequences. *Cognitive Psychology.* 17:295-314.

Hartigan, J. A. (1975). *Clustering algorithms.* New York: Wiley.

Hartigan, J. A. and Wong, M. A. (1979). A K-means clustering algorithm. *Applied Statistics.* 28:100-108.

Hastie, T., Tibshirani, R. and Friedman, J. (2001). *The elements of statistical learning.* Springer.

Kaufman, L. and Rousseeuw, P. J. (1990). *Finding groups in data: an introduction to cluster analysis.* New York: Wiley.

Kohonen, T. (1997). *Self-organizing maps.* Springer-Verlag.

Kimura H., Ohtomo T., Yamaguchi M., Ishii A., Sugimoto K. (1996). Mouse MCM proteins: complex formation and transportation to the nucleus. *Genes Cells*, 1 (11): 977-93

Labib K., Tercero J. A., Diffley J. F. (2000). Uninterrupted MCM2-7 function required for DNA replication fork progression. *Science*, 288: 1643-7.

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. *Proc. 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.

Yeung, K. Y., Medvedovic, M. and Bumgarner, R. E. (2003). Clustering gene expression data with repeated measurements. *Genome Biology.* 4:R34.

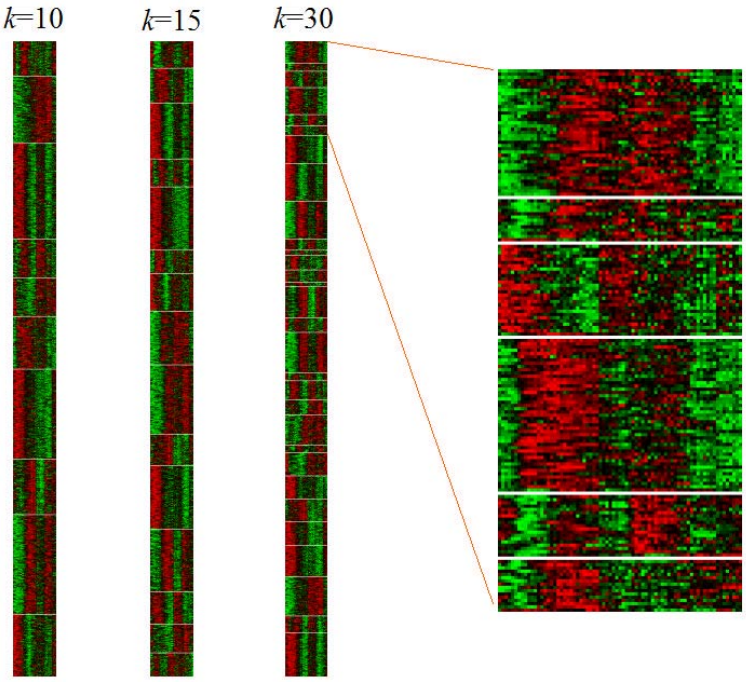

Figure 1: Clustering using K-means algorithm with arbitrary k.

Table 1: Comparing average K-means error rates using different initial values in 100,000 simulations. For the simulation settings, "$50 \times 3 + 10$" means 3 clusters each containing 50 points plus 10 scattered points are generated.

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

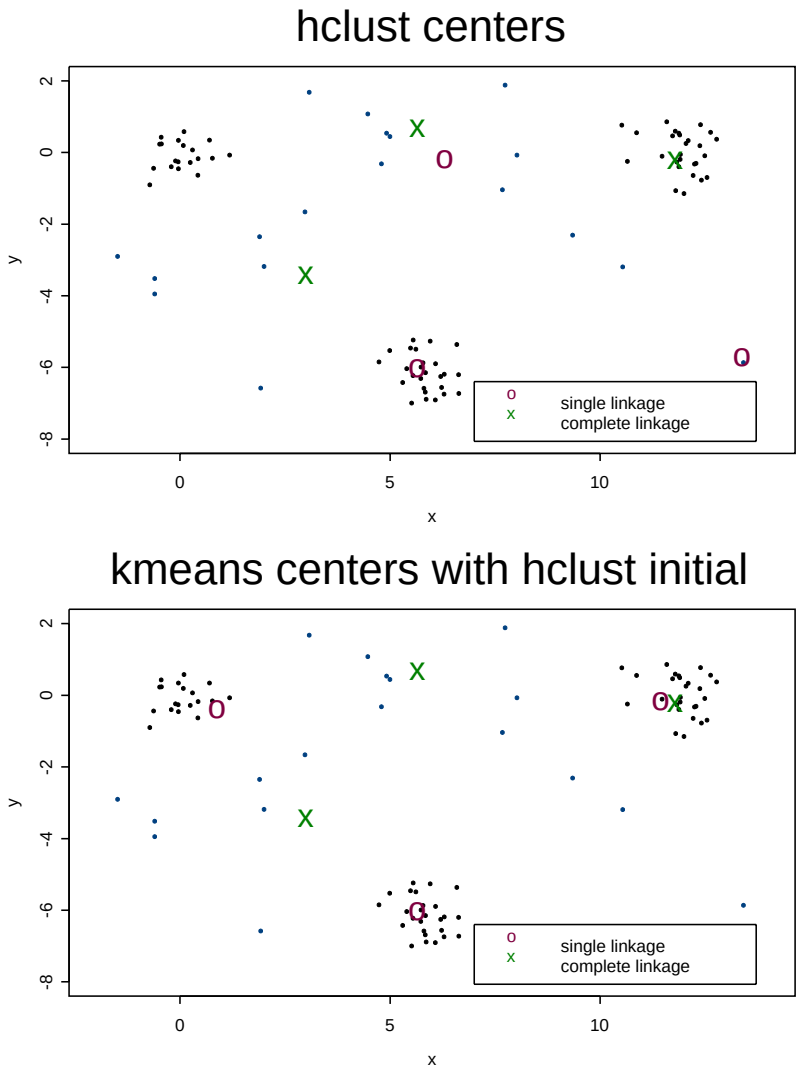

Figure 2: Cluster centers obtained from hierarchical clustering is shown in the upper plot (o:single and x:complete linkage) and are used as K-means initial values. The resulting cluster centers of K-means are shown in the lower plot.

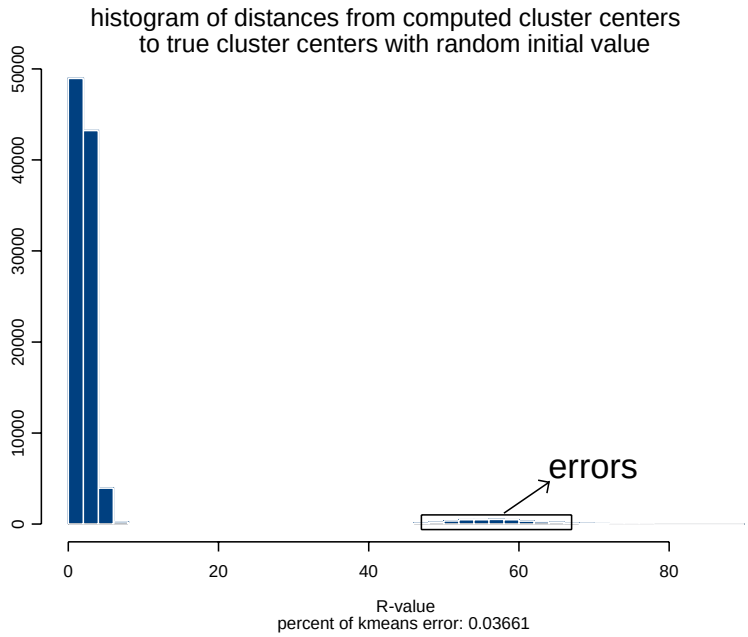

Figure 3: Histogram of R-values in 100,000 simulations using random initial values in K-means algorithm.

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
$\alpha = 0, \beta = 0.7$															
$k_0 = 10$	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

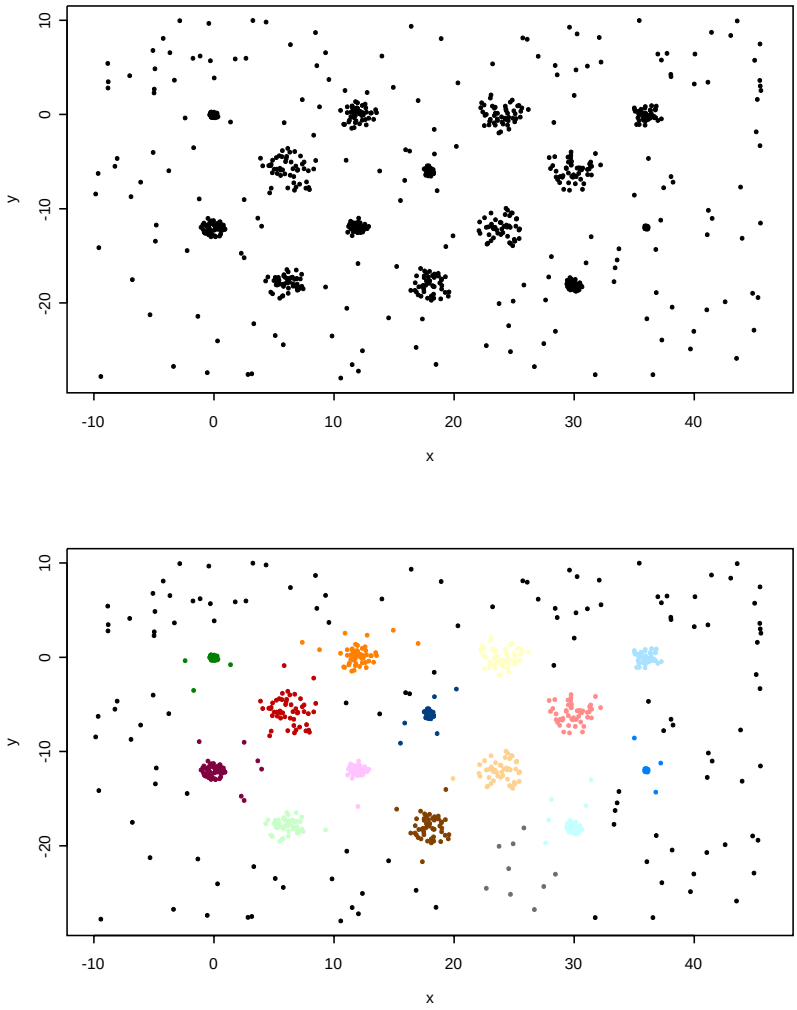

Figure 4: Upper: An example of 14 clusters with 175 scattered points. Lower: Different colors represent clusters identified by Tight Clustering.

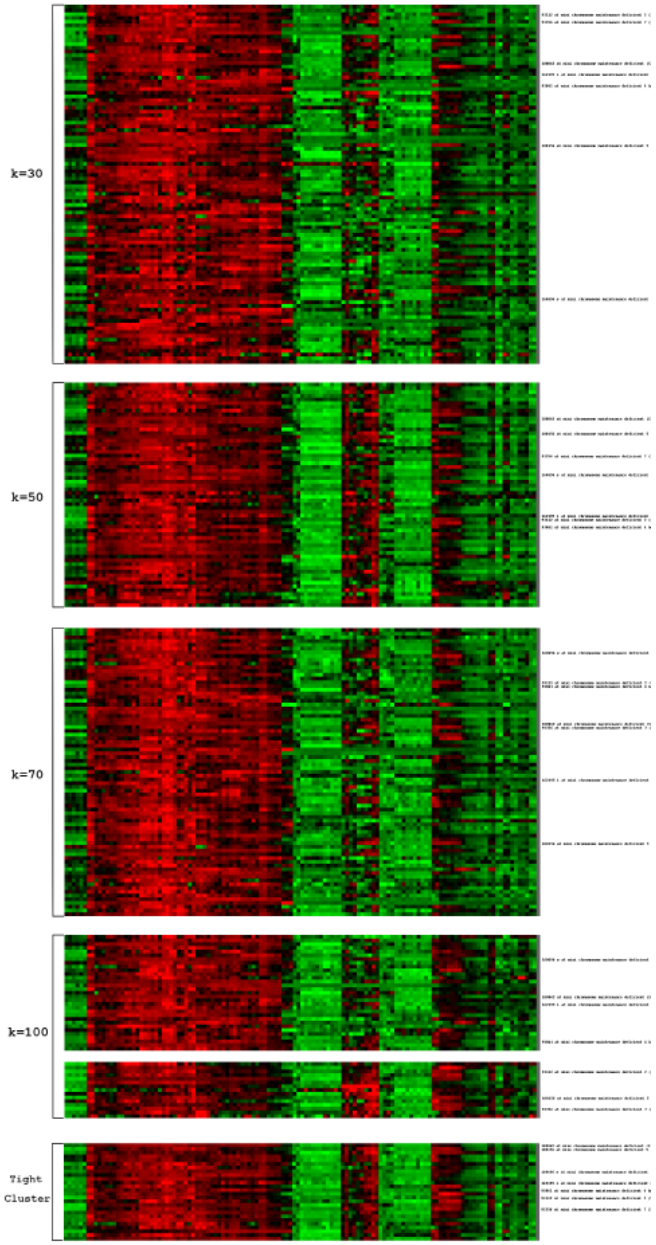

Figure 5: Selected clusters of Tight Clustering and K-means clustering with k=30, 50, 70 and 100 containing seven MCM genes. The MCM genes are indicated on the right.

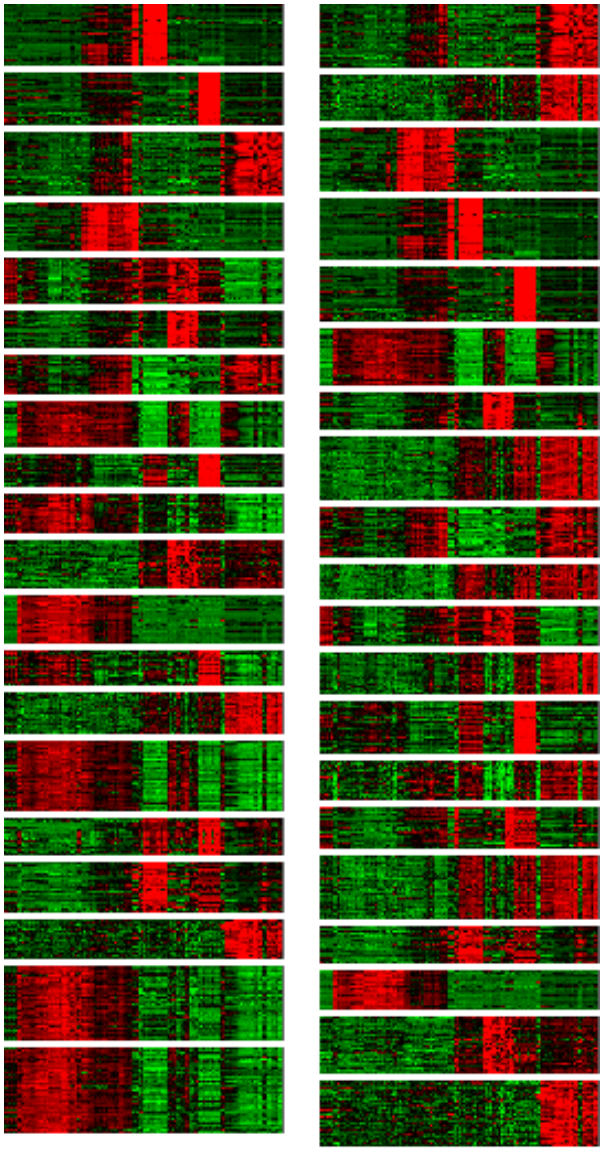

Figure 6: Two Tight Clustering results with k_0=35 but different resampling random seeds.