University of California, Berkeley
U.C. Berkeley Division of Biostatistics Working Paper Series

Year 2002 *Paper* 105

A New Partitioning Around Medoids Algorithm

Mark J. van der Laan [*] Katherine S. Pollard [†]

Jennifer Bryan [‡]

[*]Division of Biostatistics, School of Public Health, University of California, Berkeley, laan@stat.berkeley.edu

[†]Division of Biostatistics, School of Public Health, University of California, Berkeley, kpollard@wald.ucdavis.edu

[‡]Department of Statistics, University of British Columbia, jenny@stat.ubc.ca

This working paper is hosted by The Berkeley Electronic Press (bepress) and may not be commercially reproduced without the permission of the copyright holder.

http://www.bepress.com/ucbbiostat/paper105

Copyright ©2002 by the authors.

A New Partitioning Around Medoids Algorithm

Mark J. van der Laan, Katherine S. Pollard, and Jennifer Bryan

Abstract

Kaufman & Rousseeuw (1990) proposed a clustering algorithm Partitioning Around Medoids (PAM) which maps a distance matrix into a specified number of clusters. A particularly nice property is that PAM allows clustering with respect to any specified distance metric. In addition, the medoids are robust representations of the cluster centers, which is particularly important in the common context that many elements do not belong well to any cluster. Based on our experience in clustering gene expression data, we have noticed that PAM does have problems recognizing relatively small clusters in situations where good partitions around medoids clearly exist. In this note, we propose to partition around medoids by maximizing a criteria "Average Silhouette" defined by Kaufman & Rousseeuw. We also propose a fast-to-compute approximation of "Average Silhouette". We implement these two new partitioning around medoids algorithms and illustrate their performance relative to existing partitioning methods in simulations.

1 A new partitioning around medoids algorithm

Suppose that one is interested in clustering p elements $\mathbf{x}_j$, $j \in \{1, \ldots, p\}$ and that each element $\mathbf{x}_j$ is an n dimensional vector $(x_{1j}, \ldots, x_{nj})^T$. We have encountered this problem in the gene expression context, where each element is a gene whose relative expression has been measured across a variety of experiments or patients. Other contexts in which clustering has been applied include environmental studies, astronomy, and digit recognition. Let $d(\mathbf{x}_j, \mathbf{x}_{j'})$ denote the dissimilarity between elements j and j' and let $\mathbf{D}$ be the $p \times p$ symmetric matrix of dissimilarities. Typical choices of dissimilarity include Euclidean distance, 1 minus correlation, 1 minus absolute correlation and 1 minus cosine-angle. For example, the cosine-angle distance between two vectors was used in Eisen et al. [1998] to cluster genes based on gene expression data across a variety of cell lines. It is of interest to note that the cosine-angle distance equals 0.5 times the squared Euclidean distance standardized to have Euclidean norm 1.

The clustering procedure PAM [Kaufman and Rousseeuw, 1990, chap. 2] takes as input such a dissimilarity matrix $\mathbf{D}$ and produces as output a set of cluster centers or "medoids". Let K be the number of clusters and let $\mathbf{M} = (M_1, \ldots, M_K)$ denote any size K collection of the n elements $\mathbf{x}_j$. Given $\mathbf{M}$, we can calculate the dissimilarity $d(\mathbf{x}_j\ , M_k)$ of each element and each member of $\mathbf{M}$. For each element $\mathbf{x}_j$, we denote the minimum and minimizer by $\min_{k=1,\ldots,K} d(\mathbf{x}_j\ , M_k) = d_1(\mathbf{x}_j\ , \mathbf{M})$ and $\min^{-1}_{k=1,\ldots,K} d(\mathbf{x}_j\ , M_k) = l_1(\mathbf{x}_j\ , \mathbf{M})$. PAM selects the medoids $\mathbf{M}^*$ by minimizing the sum of such distances $\mathbf{M}^* = \min^{-1}_{\mathcal{M}} \sum_j d_1(\mathbf{x}_j\ , M)$. Each medoid M^*_k identifies a cluster, defined as the elements which are closer to this medoid than to any other. This clustering is captured by a vector of labels $l(\mathbf{X}, \mathbf{M}^*) = (l_1(\mathbf{x}_1, \mathbf{M}^*), \ldots, l_1(\mathbf{x}_p, \mathbf{M}^*))$.

One can consider K as given or it can be data-adaptively selected, for example, by maximizing the average silhouette as recommended by Kaufman and Rousseeuw. The silhouette for a given element is calculated as follows. For each gene j, calculate a_j which is the average dissimilarity of gene j with other elements of its cluster:

$$a_j = \operatorname{avg} d(\mathbf{x}_j\ , \mathbf{x}_{j'}), j' \in \{i : l_1(\mathbf{x}_i, M) = l_1(\mathbf{x}_j\ , M)\}.$$

For each gene j and each cluster k to which it does not belong (that is, $k \neq l_1(\mathbf{x}_j\ , M)$), calculate b_{jk}, which is the average dissimilarity of gene j with the members of cluster k:

$$b_{jk} = \operatorname{avg} d(\mathbf{x}_j\ , \mathbf{x}_{j'}), j' \in \{i : l_1(\mathbf{x}_i, M) = k\}.$$

Let $b_j = \min_k b_{jk}$. The silhouette of gene j is defined by the formula:

$$S_j(\mathbf{M}) = \frac{b_j - a_j}{\max(a_j, b_j)}. \tag{1}$$

Note that the largest possible silhouette is 1, which occurs only if there is no dissimilarity within gene j's cluster (*i.e.*: $a_j = 0$). The other extreme is -1. Heuristically, the silhouette measures how well matched an object is to the other objects in its own cluster versus how well matched it would be if it were moved to another cluster. It has also been our experience, based on simulated and real gene expression data, that the average silhouette is actually a very good measure of the strength of clustering results:

Hosted by The Berkeley Electronic Press

see also [Fridlyand, 2001] for a favorable performance of average silhouette relative to other validation functionals.

PAM has several favorable properties. Since PAM performs clustering with respect to any specified distance metric, it allows a flexible definition of what it means for two elements to be "close". We have found that this flexibility is particularly important in biological applications where researchers may be interested, for example, in grouping correlated or possibly also anti-correlated elements. Many clustering algorithms do not allow for a flexible definition of similarity. KMEANS, for example, could be performed with respect to any metric, but allows only Euclidean and Manhattan distance in current implementations of which we are aware. In addition to allowing a flexible distance metric, PAM has the advantage of identifying clusters by the medoids. Medoids are robust representations of the cluster centers that are less sensitive to outliers than other cluster profiles, such as the cluster means of KMEANS. This robustness is particularly important in the common context that many elements do not belong well to any cluster.

We have found some cases, in both real and simulated data, where existing clustering routines fail to find the main clusters. The problem of finding relatively small clusters in the presence of one or more larger clusters is particularly hard. In this situation, PAM often does not succeed in finding a sensible set of medoids. This criticism is just as valid (if not more so) for KMEANS and Self-Organizing Maps (SOMs). Inspired by this lack of performance, we present a modification of PAM that maximizes over all potential medoids the average silhouette.

Given the number of clusters K, PAM maximizes over all potential medoids $\mathbf{M}$ the function

$$f(\mathbf{M}) = -\sum_j d_1(\mathbf{x}_j \ , \mathbf{M}). \tag{2}$$

Our first proposal is to replace $d_1(\mathbf{x}_j \ , \mathbf{M})$ in (2) by $S_j(\mathbf{M})$ defined in Equation (1) so that PAM maximizes the average silhouette of Kaufman and Rousseeuw over all potential medoids. A second proposal is to replace $d_1(\mathbf{x}_j \ , \mathbf{M})$ in (2) by $\tilde{S}_j(\mathbf{M}) = (\tilde{b}_j - a_j)/\max(a_j, \tilde{b}_j)$, where $\tilde{b}_j$ is the dissimilarity of the element $\mathbf{x}_j$ and the second-closest medoid M_k, that is $d(\mathbf{x}_j \ , M_k) = d_2(\mathbf{x}_j \ , \mathbf{M})$. Now, $\tilde{S}_j(\mathbf{M})$ is a so-called *medoid-based* silhouette which is a quick approximation of the silhouette and represents itself a validation criterion. Since the medoid-based silhouette for an element can be computed just as fast as the distance to its closest medoid, the algorithm for maximizing the criterion function $f(\cdot)$ is now just as fast as PAM. The medoid-based average silhouette is a less robust measure of clustering strength than average silhouette. We refer to the two partitioning around medoids algorithms as PAMSIL and PAMMEDSIL, respectively.

To maximize the two proposed functions $f(\mathbf{M})$ we use the steepest descent algorithm for functions defined on vectors of elements in a discrete set:
1) Let $m = 0$ and choose $\mathbf{M}^m$. Define $\mathbf{M}_{-i} = (M_1, \ldots, M_{i-1}, M_{i+1}, \ldots, M_K)$.
2) For every swap $(\mathbf{M}^m_{-i}, h)$ of one of the K current medoids i with element h calculate $f(\mathbf{M}^m_{-i}, h)$. Let i^* and h^* be the swap that gives the maximal value of $f(\cdot)$. Let $\mathbf{M}^{m+1} = (\mathbf{M}^m_{-i*}, h*)$.
3) Set $m = m + 1$ and repeat 2 and 3 until convergence. Denote the solution by $\mathbf{M}^*$.
In other words, one updates the vector of medoids with the single coordinate change

http://www.bepress.com/ucbbiostat/paper105

(swap of one medoid with a non-medoid) which maximizes the improvement for $f(\cdot)$ and one iterates until convergence. We note that the algorithm of Kaufman and Rousseeuw [1990] is actually applying this steepest descent algorithm, although it is not presented as such.

The medoids of PAM can be used as starting values for PAMMEDSIL and PAMSIL. Then, the medoids of PAMSIL will always correspond with an average silhouette which is at least as good as that of PAM. Similarly, the medoids of PAMMEDSIL will always correspond with an average medoid-based silhouette which is at least as good as that of PAM. Since these algorithms (including PAM) only converge to a local maxima, one might also choose to search a wider set of possible solutions by initializing with a number of starting values and choosing the final set of medoids that gives the best overall average silhouette. As with PAM, the average silhouette can be used to data-adaptively select the number of clusters K.

2 Simulations

The dissimilarity matrix $\mathbf{D}$ for a set of genes based on a sample of observations (e.g.: patients, cell lines) can be viewed as an estimate of the true population dissimilarity matrix for these genes. Hence, the cluster labels produced by a given algorithm applied to $\mathbf{D}$ are an estimate of the true cluster labels, which we call the clustering parameter. For a given data generating distribution, this parameter depends on the algorithm so that two algorithms may or may not be estimating the same parameter. For each algorithm, it follows that as the number of samples n increases, the estimate of $\mathbf{D}$ will approach the true dissimilarity, and therefore the cluster labels will approach the true clustering parameter for that algorithm. The performance of clustering algorithms (or "clustering strength") can be measured by how close they are to their true clustering parameter. When two algorithms estimate the same parameter, their performances can be compared directly. The distance of an estimated clustering parameter (i.e.: observed clustering result) from the true clustering parameter can be measured by a validation criteria, such as average silhouette, or by visualization of the clustering result. When the true cluster labels are known, one can infer a correspondence between the observed and true clusters and compute the proportion of correctly clustered elements.

In order to investigate the clustering performance of PAMMEDSIL and PAMSIL, we have conducted simulations comparing these algorithms to the existing clustering routines PAM and KMEANS. We chose to use the Euclidean distance so that KMEANS (which allows only this distance metric in its usual implementation) could be compared to the other algorithms in the context where it performs best. We also repeated the simulations using the cosine-angle distance so that we could compare the algorithms in a more common context for gene expression data analysis. First, we fixed the number of clusters to be the correct number, so that we could directly compare the algorithms only on their ability to identify the correct clusters. Then, we investigated the ability of each algorithm to select the correct number of clusters K data-adaptively by maximizing average silhouette over a range of values for K. We also looked at how the relative performances of the algorithms varied with an increasing number of

Hosted by The Berkeley Electronic Press

samples n.

2.1 Data generation

Consider a sample of $n = 60$ relative gene expression profiles of dimension $p = 500$, corresponding to cancer patients. Suppose that in truth there are three groups of 20 patients corresponding with three distinct types of cancer, but this is unknown to the data analyst. To generate such data, we sampled three groups of 20 subjects from three multivariate normal distributions with diagonal covariance matrices, which differed only in their mean vector. All genes had common standard deviation log(1.6), which corresponds with a 0.75-quantile of all standard deviations in an actual data set. For the first subpopulation, the first 25 genes had $\mu_j = \log(3)$, genes 25-50 had $\mu_j = -\log(3)$, and the other 350 genes had mean zero. Then for the second subpopulation, genes 51-75 had $\mu_j = \log(3)$, genes 76-100 had $\mu_j = -\log(3)$ and the other 350 genes had mean zero. For the third subpopulation, genes 101-125 had $\mu_j = \log(3)$, genes 126-150 had $\mu_j = -\log(3)$ and the other 350 genes had mean zero. In other words, the cause of each of the three types of cancer is related to 50 genes of which 25 are suppressed (tumor-suppressor genes) and 25 are over-expressed (the onco-genes). All logs in the simulation are base 10.

We generated $B = 100$ such data sets. Note that this data generating distribution is such that a clustering routine is needed to identify the underlying structure. For example, when we simply ordered the genes by mean expression and made a picture of the corresponding Euclidean distance matrix, we saw no obvious pattern within the over-expressed genes and suppressed genes. We applied PAM, KMEANS, PAMMEDSIL and PAMSIL to each of the $B = 100$ data sets with the Euclidean distance metric and $K = 7$ clusters. We used the PAM medoids as starting values for PAMMEDSIL and PAMSIL.

2.2 Identifying the clustering parameter

Given this data generating distribution, each of the algorithms defines a clustering parameter. These parameters were identified by running the algorithms on the true Euclidean distance matrix, which is a function of the mean and covariance of the data generating distribution. In this case, all four algorithms have the same clustering parameter which is six clusters of 25 genes and one large cluster of 350 non-differentially expressed genes. The true cluster labels correspond with a true average silhouette of 0.27. Since the algorithms are estimating the same clustering parameter, the distance of average silhouette from this shared true average silhouette can be used as a measure of clustering performance. For example, if average silhouette for an algorithm is far from the true average silhouette then the cluster labels can not be correct. The converse is not necessarily true, so that when average silhouette is close to the true average silhouette, it is useful to also visualize the clustering result in order determine how close it is to the true seven clusters.

COBRA
Collection of Biostatistics
Research Archive

http://www.bepress.com/ucbbiostat/paper105

2.3 Average silhouette

Table 1 shows the mean and standard error of average silhouette across the $B = 100$ data sets for each algorithm. According to this criteria, both PAMMEDSIL and PAMSIL performed better than the existing algorithms since they were less variable and produced higher average silhouettes. PAMSIL did better than PAMMEDSIL, however, since it actually performs the maximization of average silhouette whereas PAMMEDSIL uses a fast-to-compute proxy (medoid-based silhouette). Nonetheless, PAMMEDSIL produced average silhouettes which were on average more than twice as large as PAM. Both KMEANS and PAM were quite variable, sometimes producing average silhouettes as high as PAMMEDSIL and PAMSIL, but also many average silhouettes below 0.1. On average, KMEANS performed better than PAM in terms of maximizing average silhouette, but it was almost twice as variable.

Routine	Mean (SE) Average Silhouette
KMEANS	0.16 (0.080)
PAM	0.094 (0.045)
PAMMEDSIL	0.20 (0.022)
PAMSIL	0.24 (0.0036)
TRUE VALUE	0.27

Table 1: Average silhouettes from Euclidean distance simulation with $K = 7$ clusters. The mean and standard error across the $B = 100$ simulated data sets is reported for each clustering algorithm along with the shared true value.

2.4 Visualization

In order to further investigate how well each algorithm is able to identify the seven clusters and the degree to which average silhouette measures this success, we plotted the reordered distance matrix for each data set according to the cluster labels from each algorithm (so that genes clustered together would appear consecutively, but the clusters were not ordered). The performance of the algorithms varied greatly. KMEANS often split one or more of the six small clusters or combined (parts of) two of these together, sometimes with mean zero genes also. This result may be a consequence of the lack of robustness of cluster means. PAM did not tend to split or combine the six small clusters, but often split the mean zero genes into two clusters, combining about half of them with one of the six small clusters. This makes sense since PAM minimizes the sum of the distances to the closest medoid and splitting the mean zero genes reduces the distance to the closest medoid for many genes. PAMMEDSIL tended to put some genes from several of the six small clusters into the mean zero cluster, creating one large and six (sometimes very) small clusters. PAMSIL had by far the best performance, only occasionally clustering one or two genes incorrectly. The typical errors for PAM, KMEANS and PAMMEDSIL are illustrated in Figure 1, which shows the reordered distance matrices for each algorithm applied to a randomly selected data set.

COBRA
Collection of Biostatistics
Research Archive

Hosted by The Berkeley Electronic Press

All of the algorithms identified the clusters perfectly at least once, and when any clustering result was at least nearly perfect the average silhouette was always near the true value (0.27). The erroneous clustering results of PAM always resulted in an average silhouette less than 0.10. This makes sense since splitting the mean zero genes creates two clusters whose neighbors are very close, making the silhouettes of the genes in these clusters very small. Due to the fact that splitting small clusters has little effect on average silhouette, some of the erroneous clustering results for KMEANS and PAMMEDSIL had relatively high average silhouettes (> 0.15). In fact, for some data sets PAM produced a clustering result with an average silhouette less than that of KMEANS and PAMMEDSIL, but the visualization of the clusters indicated that PAM was in fact doing a better job at finding the true clusters. By applying PAMSIL to the PAM result in these cases, average silhouette and the visualized result are improved dramatically as illustrated in Figure 1. When PAM occasionally finds a (near) perfect clustering, PAMSIL will converge quickly to the nearby maximum and – by definition – never produces a worse result in terms of average silhouette. In other words, the PAM medoids seem to be good starting values for PAMSIL, even when they correspond with an erroneous clustering result.

These findings offer support for the PAMSIL algorithm and suggest that a sensible strategy would be to first apply PAM and then PAMSIL. When we applied PAMSIL with random starting values, its performance was still good. Hence, it is also possible to simply apply PAMSIL without PAM, although the time saved by skipping the PAM step is lost in the extra time it takes for PAMSIL to converge from random starting values.

2.5 Choosing the number of clusters

The number of clusters K can be given or it can be data-adaptively selected, for example, by maximizing the average silhouette as recommended by Kaufman and Rousseeuw. For each data set (with $p = 500$ genes and $n = 60$ samples), we applied each algorithm with Euclidean distance and $K = 2, \ldots, 10$ clusters. PAMMEDSIL and PAMSIL were applied with the PAM medoids as starting values. We observed that KMEANS and PAMMEDSIL produced clustering results with maximum average silhouette frequently at $K = 6$ and PAM produced clustering results with average silhouettes increasing in K up to $K = 8$ or 9, so that choosing the number of clusters by maximizing average silhouette (as suggested by Kaufman and Rousseeuw) would usually not result in the correct number of clusters ($K = 7$) with KMEANS, PAMMEDSIL or PAM. The maximum PAMSIL average silhouette, however, was indeed always at $K = 7$ clusters. In other words, PAM often found a local maximum at $K = 7$ with an average silhouette lower than both its own average silhouette for $K = 8$ or 9 and the global maximum at $K = 7$ found correctly by PAMSIL.

COBRA
A BEPRESS REPOSITORY
Collection of Biostatistics
Research Archive

2.6 Asymptotic consistency.

Since the clustering algorithms estimate the same clustering parameter, their differing performance was the consequence of their having different efficiencies for a fixed,

http://www.bepress.com/ucbbiostat/paper105

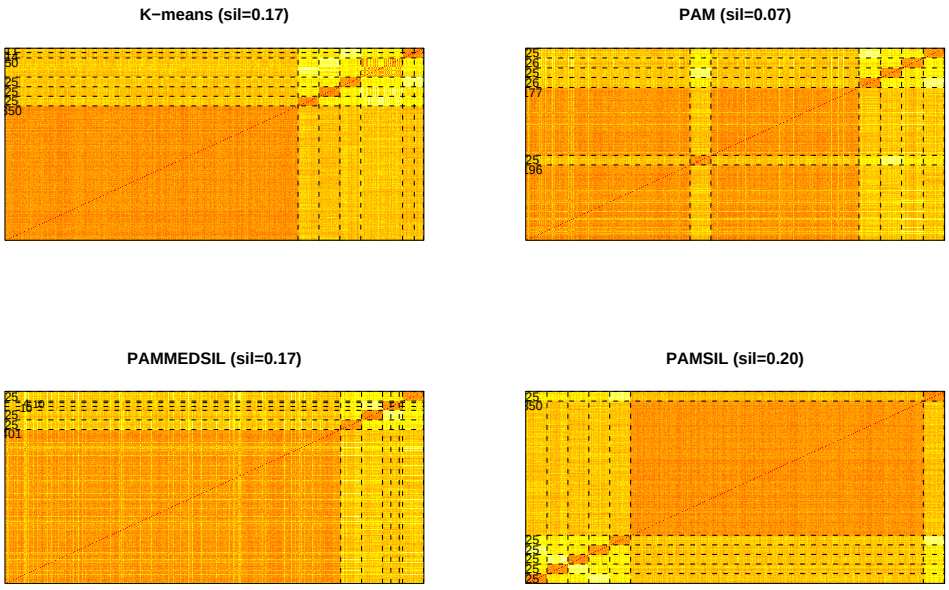


Figure 1: Reordered Euclidean distance matrices for each algorithm applied to a randomly selected data set with $K = 7$ clusters. Each point is a pairwise distance between two genes. Red corresponds with smallest and yellow with largest distance. Genes are reordered so that those in the same cluster appear consecutively. The ordering of the clusters is arbitrary. The dotted lines indicate the cluster boundaries. The numbers on the right side give the cluster sizes.

relatively small number of samples ($n = 60$). In order to investigate the asymptotic consistency, we repeated the simulation for $n = 1200$ samples. We again used Euclidean distance and $K = 7$ clusters. With this increased number of samples, PAM, PAMMEDSIL and PAMSIL all consistently produced the correct clustering result, which corresponds also with applying these clustering routines to the true distance matrix defined by the data generating distribution. KMEANS, however, only produced the correct clustering result about half of the time. We believe this result follows from the dependence of KMEANS on its starting values (determined randomly), so that for some starting values the algorithm converges to the correct result and for others it converges to an incorrect local maximum. In fact, this sensitivity to starting values can be seen by repeatedly applying KMEANS to the same data set and noting that the clustering result and corresponding average silhouette vary a great deal. When we increased the number of samples to $n = 3600$, KMEANS still showed this sensitivity to starting values. When the means of the PAM clusters were used to derive starting values for KMEANS, KMEANS then converged consistently to the same parameter as the other algorithms.

Collection of Biostatistics
Research Archive

Hosted by The Berkeley Electronic Press

2.7 Finding very small clusters

The simulation results above gave us some insight into the different performance of PAM versus PAMSIL. Although we found that both algorithms were estimating the same parameter in this simulation, we noted that for a fixed sample size ($n = 60$) PAM tended to be more robust (e.g.: preferring to split the mean zero genes than to identify small clusters), while PAMSIL was more efficient (e.g.: easily identifying small clusters). In this example, the small clusters were intended to be meaningful and so we liked the way PAMSIL performed. But, a clustering routine, such as PAMSIL, that can pick up very small clusters will also sometimes pick up outliers and call them a cluster. In the case where the outlier cluster is just noise, we would not like the way PAMSIL performed.

In order to explore the efficiency of PAMSIL, we repeated the simulation with one of the mean zero genes shifted to have mean log(5). This gene could be a random outlier in some contexts or a meaningful but *very* small cluster in others. We used $n = 1200$ patients in order to see the asymptotic performance of the algorithms. The results correspond with looking at the true clustering parameter for each algorithm.

First we set $K = 7$. All of the algorithms except PAMMEDSIL put the mean log(5) gene into one of the smaller clusters, while PAMMEDSIL clustered it alone. This result indicates that PAMMEDSIL may be even more aggressive at finding small clusters than PAMSIL, which also makes sense in light of the tendency of PAMMEDSIL to split the small clusters in the original simulation. Next we set $K = 8$, which would be the correct number of clusters if the mean log(5) gene was a true cluster. PAM and KMEANS put the mean log(5) gene into one of the smaller clusters and tended to split the mean zero genes. PAMSIL and PAMMEDSIL put the mean log(5) gene in its own cluster, identified the six small clusters, and put all of the mean zero genes together. The average silhouettes for the PAMSIL and PAMMEDSIL results were much higher than those for PAM and KMEANS. PAM and PAMSIL are not estimating the same parameter now. If this gene cluster of size one was meaningful, we would prefer PAMSIL, but if the gene were truly an outlier or we were interested in a more robust description of the global structure we would prefer PAM.

2.8 Cosine-angle distance

We repeated the simulations with the cosine-angle distance metric, which is commonly employed in gene expression clustering analysis. Because the cosine-angle distance equals the Euclidean distance after standardization by the Euclidean norm, it follows that the mean zero genes are no longer close to each other and do not form a cluster. They each belong weakly to one of the six other clusters and can be referred to as "noisy" genes. We often find genes like these in real data analyses and have noted that existing clustering algorithms have trouble identifying the true underlying clustering pattern in the presence of much noise.

We applied each clustering algorithm to $B = 100$ samples with $K = 6$ clusters. For the PAM, PAMMEDSIL and PAMSIL algorithms we used the cosine-angle distance. The results of these cosine-angle distance simulations were similar to those reported

http://www.bepress.com/ucbbiostat/paper105

above for Euclidean distance except that, as expected, KMEANS performed very poorly since it is not able to cluster with respect to cosine-angle distance. PAM often finds the correct clustering pattern and when it fails to do so, PAMSIL is able to improve upon its performance. PAMMEDSIL again does not perform well. It was also observed that the PAM family of algorithms has the property that the medoids are a good representation of the true clustering patterns even in the presence of noise.

3 Discussion

We have presented a new partitioning around medoids clustering algorithm, PAMSIL, which seeks the partitioning of a data set into the K clusters which maximize average silhouette. This algorithm represents a change in the objective function used in PAM by Kaufman and Rousseeuw from the sum of distances of each element to the nearest medoid to the average silhouette. This change means that the impact of each element on the choice of medoids is based not only on how well it belongs to its own cluster, but also on how well it belongs to the next closest cluster. Since Kaufman and Rousseeuw propose to use average silhouette to evaluate the strength of different clustering results, we thought it would be useful to have an algorithm which in fact maximizes this criteria. We also proposed PAMMEDSIL as a fast-to-compute proxy for PAMSIL, but we can not generally recommend its use.

In the first simulation, the PAM family of clustering algorithms all consistently estimated the correct clustering result. The better performance of PAMSIL relative to the other algorithms at $n = 60$ was a consequence of its greater efficiency. We investigated this issue further in a simulation with the addition of a single gene with very different expression. In this case, PAM estimated a different parameter from PAMSIL. PAM represented a more robust clustering parameter in which the unique gene was added to one of the small clusters and the mean zero genes were split if more clusters were requested. PAMSIL, in contrast, clustered the unique gene alone, illustrating its capability at finding small clusters. In many biological contexts this ability is advantageous, but in the case where such small clusters are simply noise, the greater efficiency of PAMSIL (relative to the more robust PAM) would be a penalty.

The contrast between PAM and PAMSIL is intuitive when we consider the criteria that each algorithm maximizes. Minimization of the sum of distances to the closest medoid in PAM (i) results in splits of large clusters with many elements that will thereby be closer to their medoid and (ii) is not sensitive to a small number of outlying elements that will not contribute much to the sum of distances. Maximization of the average silhouette in PAMSIL (i) results in not splitting large clusters with many elements that would thereby be very close to their neighboring cluster relative to their own and (ii) is sensitive to any elements which are very far from any other cluster since these will contribute large silhouettes to the sum.

In light of these findings, we propose a class of clustering algorithms of the form $f(\mathbf{M}) = \sum_j (\alpha S_j(\mathbf{M}) - (1-\alpha) d_1(\mathbf{x}_j \ , \mathbf{M}))$, for $\alpha \in [0,1]$. When $\alpha = 1$ this is PAMSIL, and when $\alpha = 0$ this is PAM. For intermediate values of α, the clustering algorithm will have performance in between the two extremes of robustness and efficiency. Notice

Hosted by The Berkeley Electronic Press

that determination of α is not a model selection problem, but rather a decision on the part of the researcher about what kind of clusters to seek. In practice, one might choose to experiment with different values of α, and for each compute average silhouette and also visualize the corresponding clustering result in order to select a procedure with the appropriate performance for the given context. One could also apply the bootstrap to the clustering results from a range of values of α, as proposed in van der Laan and Bryan [2001], and then choose the most reproducible procedure.

References

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

J. Fridlyand. Ph.d thesis. Department of Statistics, UC Berkeley, 2001.

L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York, 1990.

M.J. van der Laan and J.F. Bryan. Gene expression analysis with the parametric bootstrap. *Biostatistics*, 2:1–17, 2001.

COBRA
A BEPRESS REPOSITORY
Collection of Biostatistics
Research Archive