Clustering Microarray Data

Andrea Vijverberg

Advisor: Professor Hardin

Pomona College

Department of Mathematics

Spring 2007

Contents

List of Tables

List of Figures

Abstract

Using microarray data, which gives thousands of genes' expression levels at once, we examine different clustering techniques and evaluation methods to effectively cluster the noisy data. The results of this study has implications for the field of biology. Genes with similar functions are grouped together, which gives insight into specific genes and their role in the cell. The cluster analysis employed uses different distance metrics, including Euclidean, Pearson's correlation, and percentage bend correlation, and we use the cluster methods of hierarchical, Partitioning Around Medoids (PAM), and Hierarchical Ordered Partitioning and Collapsing Hybrid (HOPACH). The evaluation methods involve silhouette width, the L method, and the bootstrap. We conclude with an experiment in improving the sample and the clustering output using the bootstrap method.

Chapter 1

Introduction

In order to measure genetic activity, scientists aim to discover the biological functions of specific genes. By clustering the expressed genes, which play an important role in the function of the cells, biologists can further understand specific genes and their role in the cell. The statistician clusters the genes by first using a measure of dissimilarity between the genes and then clustering the genes using a clustering algorithm. Afterwards, we test the clustering results to determine their stability.

In this project, we will use microarray data, which gives thousands of genes' levels of expression at once. Microarray data is very noisy, requiring a robust measure of dissimilarity. Different distance metrics will be briefly compared in their robustness. Several clustering methods will be employed, including hierarchical, partitioning, and a hybrid of the two. To estimate the strength of cluster membership, we apply the bootstrap. Bootstrap methods may be applied in several different ways, but the general method is as follows: create many resamples by sampling with replacement from the original sample and find the bootstrap distribution of the statistic from the resamples. In our case of clustering microarray data, we have one data set, and, by creating and clustering many resamples, we can determine the reproducibility of the clustering result and which genes cluster poorly. Instead of finding a bootstrap distribution, as in the general theory, we examine the variability of the cluster output.

The following section will give a more detailed description of microarrays and how the data are obtained. Chapter 3 will look at cluster analysis. The main mathematical background information, such as distance metrics and clustering theory, comes from Draghici [1], Johnson and Wichern [3], Kaufman and Rousseeuw [4], and Wilcox [10]. The specific clustering algorithms, Partitioning Around Medoids (PAM) and Hierarchical Ordered Partitioning and Collapsing Hybrid (HOPACH), were developed by Kaufman and Rousseeuw [4] and Van der Laan and Pollard [8], respectively. The description of the algorithms are followed by the theory on evaluation methods, drawing from the works of Salvador and Chan [7] for the L method and Van der Laan and Pollard [8] and [9] for the bootstrap criterion. Using Mooney and Duval [5] and Efron and Tibshirani [2],

I explain the theory on bootstrap techniques.

Chapter 4 will explain the data and Chapter 5 will use the clustering algorithms and tests on the data, giving graphs and a discussion of the results. Finally, I will conclude with a recap of the major results and possible further extensions on this project.

Chapter 2

Background

2.1 Microarrays

A microarray is a small chip that has thousands of single strands of genetic material tethered to it. Its creation begins with a process of denaturing a cell's mRNA into single strands and then attaching them to the chip. Each gene in the cell is represented by a dot on the chip, but not all possible genes are necessary for a particular cell to perform its function. For example, a blood cell only uses genes that aid in the functions of a blood cell which might be different from those a skin cell would use. We call the functional genes of the cell the expressed genes. Therefore, when a cell becomes diseased, say with cancer, certain expressed genes may have changed, causing the misbehavior of the cell. A microarray chip allows researchers to examine, all at once, the expressed genes of a cell in both healthy and unhealthy samples.

To analyze these gene functions, biologists take samples from cells known to be healthy or diseased and allow their single strands of DNA to attach to those on the microarray chip. This is done by gathering the mRNA of these known cells, which comes out of the cell nucleus during cell replication and gives the genetic code of the expressed genes. Using the mRNA to make cDNA and labelling the cDNA with a color of red or green, the expressed genes on the chip become labelled with the colors of the attaching cDNA, as the double helix reforms. The assignment of colors to the cDNA depends on its respective sample, experiment or control. Note that not all of the genes on the chip re-form the double helix with the cDNA because, as mentioned before, not all the genes on the chip are used in a cell's function.

The chip is scanned twice under a fluorescent light, once measuring the green color intensities and once measuring the red. Putting these scans together, each spot on the chip appears in one of four colors (Figure 6.1). A black spot means the gene is not expressed, i.e., no labelled cDNA attached to any genes in that spot. A yellow dot has genes of both samples expressed equally. A green spot has more of the green sample expressed, and similarly, the red spot has

more expression from the red sample.

Scanning the chip quantifies the fluorescence of each dot: more fluorescence means more of the sample stuck to the chip. Due to the double scan, each spot on the chip has a numerical value for red and green color intensity. Putting these numbers together, we obtain our data set which is of the log ratios of color intensities for each spot.

Chapter 3

Cluster Analysis

With the goal of assisting scientists in measuring genetic activity, many genes at a time, we would like to cluster the genes. Scientists can use the clustering results to learn how genes are co-regulated, giving them insight into certain genes' relation to each other and their functions. Since we do not have prior knowledge of which genes are related, we take on the "unsupervised learning" strategy of clustering, which is the ideal approach in cases where one has no preconceived knowledge of the groupings. Clustering is the process of grouping together similar entities. Any data set can be clustered, so the final goal is not the clustering result itself but the conclusions that can be drawn. With this in mind, we will apply different clustering techniques and compare the results.

Before we begin discussing clustering techniques, we must select a measure of similarity or dissimilarity between the genes. As one of our goals, we would like to compare robust distances with others to see which provides better clustering results. In particular, we will be using two non-robust distances, Euclidean distance and Pearson's correlation, and the robust distance of percentage bend correlation.

3.1 Distance Metrics

A measure of dissimilarity between two elements is their distance. A distance metric takes two points in the input space of the problem and calculates a positive number that contains information about how close the two points are to each other. It must take on three properties: symmetry, positivity, and triangle inequality. That is, for a distance metric d with two points x and y in an n-dimensional space $\Re^n$, the following must hold:

$$d(x, y) = d(y, x)$$

$$d(x, y) \geq 0$$

$$d(x, y) \leq d(x, z) + d(z, y)$$

where z is a third point.

Typically, researchers use the Euclidean distance, whose numerical value stems from the Pythagorean Theorem. This distance, used for most practical purposes, is defined as

$$d_E(x,y) = \sqrt{(x_1-y_1)^2+(x_2-y_2)^2+...+(x_n-y_n)^2} = \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

The other most common distance metric used is the Pearson's correlation distance:

$$d_R(x,y) = 1 - r_{xy}$$

where r_{xy} is the Pearson's correlation coefficient of the vectors x and y:

$$r_{xy} = \frac{S_{xy}}{\sqrt{S_x}\sqrt{S_y}} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

where $\bar{x}$ and $\bar{y}$ are the sample means.

Since the Pearson's correlation coefficient r_{xy} varies only between -1 and 1, the distance will take values between 0 and 2. When a pair of elements are positively perfectly correlated, the correlation will be 1 and the distance will be zero. Similarly, a negatively perfectly correlated pair will have a distance of 2. Intuitively, smaller distances are assigned to pairs that change similarly. For example, the distance between an element and itself is zero since it is perfectly correlated with itself.

In general, the Pearson's correlation distance metric evaluates how genes move in relation to each other, whereas the Euclidean distance calculates the actual numerical difference in space between the vectors. Suppose there are three genes A, B, and C. In relation to gene A, gene B changes more similarly than gene C does but gene C is closer to gene A. Using Pearson's correlation distance metric, the distance between genes A and B is smaller than the distance between genes A and C. However, the Euclidean distance metric will give a smaller distance value for the pair A and C than for the pair A and B. The simple example shows the potential for a discrepancy in the final result that stems from the choice in distance metrics.

Another issue arises concerning the robustness of the Pearson's correlation distance metric. Well known in the statistical literature, Pearson's correlation lacks robustness because one point in one of the marginal distributions can have a large effect on r_{xy}. Therefore, we use a more robust distance metric, the percentage bend correlation. Given one point (x,y), we calculate the sample median M_x and M_y, which we use to calculate W_i, $W_i = \mid x_i - M_x \mid$. We rank the W_i values in ascending order: $W_{(1)} \leq W_{(2)} \leq ... \leq W_{(n)}$. Set $\hat{\omega}_x = W_{(m)}$.

Let i_1 be the number of x_i values for which $\frac{x_i - M_x}{\hat{\omega}_x} < -1$ and let i_2 be the number of x_i values such that $\frac{x_i - M_x}{\hat{\omega}_x} > 1$. Let $S_x = \sum_{i=i_1+1}^{n-i_2} x_{(i)}$. Similar calculations are done for the y variable. The percentage bend correlation is given by the following equation (Wilcox [10]):

$$r_{pb} = \frac{\sum A_i B_i}{\sqrt{\sum A_i^2 \sum B_i^2}}$$

where $A_i = \Psi(U_i)$ and $B_i = \Psi(V_i)$, $\Psi(x) = max[-1, min(1, x)]$, $U_i = (x_i - \hat{\phi}_x)/\hat{\omega}_x$ and $V_i = (y_i - \hat{\phi}_y)/\hat{\omega}_y$ and $\hat{\phi}_x = \frac{\hat{\omega}_x(i_2 - i_1) + S_x}{n - i_1 - i_2}$. The variables U_i and V_i represent a scaled distance from the center. Hence, the extreme outliers are set to a specific limit, so that they do not impact the correlation value as heavily. This adjustment makes the percentage bend correlation more robust than Pearson's correlation. For both the Pearson's correlation and the percentage bend correlation, the triangle inequality does not hold, but when discussing similarities, we often drop this requirement.

3.2 Clustering Techniques

Using the above distance metrics, we create a measure of similarity between genes, which is one minus the distance. Clustering results depend highly on the distance metric used. After calculating a distance matrix giving the distances of every element with respect to all the others, we employ a clustering technique. In this project, we focus on three methods: hierarchical clustering, partitioning around medoids (PAM), and hierarchical ordered partitioning and collapsing hybrid (HOPACH).

In clustering microarray data, hierarchical clustering has typically been used because the covariance structure of the data is not needed. The final result of hierarchical clustering gives a tree, called a dendrogram, showing individual patterns as leaves and the root as the convergence point of all the branches (see Figure 6.2). The tree can be obtained by either starting at the root and dividing the data into smaller groups, a divisive method, or by starting with individual data points and combining them, an agglomerative method. In general, the divisive method is less common due to its computationally intensive nature. To start, the divisive algorithm must consider $2^{n-1}-1$ possible divisions, which quickly becomes very time consuming. The agglomerative method only has $\frac{n(n-1)}{2}$ possible fusions to consider, which is quadratic but computationally feasible. This paper uses an agglomerative algorithm.

Another distinction within the hierarchical clustering method is the choice of linkage: single, complete, or average (Johnson and Wichern [3]). When two clusters must be joined, there are multiple ways to calculate the distance between the clusters, which results in different trees. Single linkage merges according to the nearest neighbors, i.e., the smallest distance or largest similarity between the two groups. For example, suppose we calculate the distance between

the cluster (UV), which includes the elements U and V, and the element W. Using single linkage, the distance is

$$d_{(UV)W} = \min\{d_{UW}, d_{VW}\}$$

where d_{UW} is the distance between elements U and W and d_{VW} is the distance between elements V and W. The single linkage measure of distance between clusters typically results in a portion of the data being grouped together in a string. Because the clustering may add one element at a time to the group, the single-linkage approach may result in a chain-like structure.

Complete linkage merges groups according to the maximum distance or farthest neighbor; it combines the groups with the smallest maximum distance of the elements of the two considered clusters. That is, complete linkage calculates the distance as

$$d_{(UV)W} = \max\{d_{UW}, d_{VW}\}$$

The average linkage method merges clusters according to the average distance between all the elements:

$$d_{(UV)W} = \frac{\sum_i \sum_k d_{ik}}{N_{(UV)} N_W}$$

where d_{ik} is the distance between element i in cluster (UV) and element k in W and $N_{(UV)}$ and N_W are the number of items in (UV) and W, respectively. The average linkage method of grouping items gives a hierarchical clustering output similar to complete-linkage.

In the partitioning around medoids (PAM) algorithm, k representative objects are selected so that the objects within each final cluster have a high degree of similarity while being as dissimilar to other clusters as possible. Hence, the objects within each cluster are "closer" to each other than they are to those outside of their cluster. The representative objects are called *medoids*, and after they are found, the remaining objects are assigned to the medoid (i.e., cluster) for which the dissimilarity is the smallest. The dissimilarity between objects can be found using any distance metric, allowing flexibility in the definition of "close".

The PAM algorithm is as follows (Kaufman and Rousseeuw [4]):

- Choose the number of clusters k.
- BUILD
  - Select the first medoid where the sum of the dissimilarities to all other elements is as small as possible.
  - To select the remaining $k-1$ medoids:
    1. Consider an element i which has not yet been selected
    2. Consider another nonselected element j

3. Calculate the difference between the dissimilarity D_j of element j with the most similar previously selected medoid and the element's dissimilarity $d(j,i)$ with element i
4. If the difference from step 3 is positive, element j will benefit if element i is selected as a medoid. Hence, element j positively influences the decision to select element i, and we calculate

$$C_{ji} = max(D_j - d(j,i), 0).$$

5. Calculate the total gain obtained by selecting element i:

$$\sum_j C_{ji}$$

6. Choose as the next medoid the not yet selected element i which maximizes the total gain:

$$\max_i \sum_j C_{ji}$$

– Continue steps 1-6 until k medoids have been found.

- SWAP

– To calculate the effect of a swap between medoid i and nonselected element h on the value of the clustering:

1. Consider a nonselected object j and calculate its contribution C_{jih} to the swap:
   (a) If j is further from both i and h than from another representative object, C_{jih} is zero.
   (b) If j is not further from i than from any other medoid ($d(j,i) = D_j$), two situations must be considered:
      i. j is closer to element h than to the second closest medoid $d(j,h) < E_j$ where E_j is the dissimilarity between j and the second most similar representative object. In this case, the contribution of element j to the swap between objects i and h is

$$C_{jih} = d(j,h) - D_j$$

      ii. j is at least as distant from h than from the second closest medoid $d(j,h) \geq E_j$. In this case, the contribution of object j to the swap is

$$C_{jih} = E_j - D_j$$

   (c) j is further from medoid i than from at least one of the other representative objects but closer to element h than to any medoid. In this case, the contribution of j to the swap is

$$C_{jih} = d(j,h) - D_j$$

2. Calculate the total results of a swap by adding the contributions C_{jih}:

$$T_{ih} = \sum_j C_{jih}$$

– To decide whether to carry out a swap:

1. Select the pair (i, h) which

$$\min_{i,h} T_{ih}$$

2. If the minimum T_{ih} is negative, the swap is carried out and the algorithm returns to step 1. If the minimum T_{ih} is positive or 0, the value of the objective cannot be decreased by carrying out a swap and the algorithm stops.

Since the number of clusters produced by the PAM algorithm is user-specified, we first run a loop using PAM for many different numbers of clusters. The information returned includes average silhouette width, which quantifies the stability of the clusters. Upon maximization of average silhouette width, we find the optimal number of clusters and run PAM again with this number, returning the best clustering result.

As the name suggests, our second clustering algorithm, Hierarchical Ordered Partitioning and Collapsing Hybrid (HOPACH), combines features of both partitioning and agglomerative clustering methods. The algorithm begins by applying PAM to identify medoids. Next, the ordering scheme preserves the tree structure. The collapsing step examines whether clusters should be combined or not by calculating the average silhouette for both cases. When the average silhouette improves for the level, the collapse step gives the labels of one cluster to the other, preserving the tree structure. Repeating these partitioning, ordering, and collapsing steps at each level until each cluster contains no more than three elements, the final level is an ordered list of the elements. Though the HOPACH partitions the genes, the final ordered list allows us to make a dendrogram, unavailable with PAM, but the tree is not necessarily hierarchical, unlike the dendrogram produced by the hierarchical clustering. When plotting the results, a heat map shows the clusters by reordering the rows and columns of the distance matrix according to the final level of the tree, with red representing small distances and white representing large distances. Along the diagonal we see groups of red, where clusters are assigned. Details of the HOPACH algorithm follow (Van der Laan and Pollard [8]):

- Consider a $p \times p$ dissimilarity matrix D. Begin with all p of the elements in one cluster.
- **Partition:**

1. Select k, the number of clusters ("child clusters") in the next level for each node ("parent") by maximizing average silhouette over the range

$2, ..., K$, where K is a user-supplied maximum number of children per parent node.

2. Apply PAM to the elements in each parent cluster in any level to create child clusters for the next level.

- **Order:** To produce a meaningful final ordered list of elements

  1. Consider a set of k child clusters with medoids $M_1, ..., M_k$.
  2. Define a distance between clusters (distance between the medoids).
  3. To order the k_1 clusters in the first level of the tree:
     (a) If $k_1 = 2$, then the ordering does not matter.
     (b) If $k_1 > 2$, build a hierarchical tree from the k_1 medoids by applying HOPACH-PAM to the medoids.
  4. For all the other levels of the tree, define their neighboring cluster as the cluster to the right of their parent (in the previous level) and denote its medoid M_N, N for neighbor.
     (a) Order the k clusters left to right from largest to smallest distance to the neighboring cluster which allows preservation of the tree structure by keeping the children of each parent together and ordering them.
  5. Order the elements within each of the clusters by either
     (a) their distance with respect to the medoid of that cluster so that the badly clustered elements end up at the right edge of these clusters
     OR
     (b) their distance with respect to the medoid of the neighboring cluster.

- **Collapse:** If by collapsing the average silhouette for the whole level increases, a collapse takes place where the labels of one cluster are given to another, an arbitrary way of grouping the joined clusters as one. The relabeling allows preservation of the tree structure.

  1. Collapse until there is no pair of clusters for which collapsing improves average silhouette for the whole level.
  2. Choose a new medoid for the merged cluster.

- Repeat the partition, order, and collapse steps in each level of the tree and stop when each cluster contains no more than 3 elements.

- The final level is an ordered list of the elements.

As PAM uses average silhouette width to find the optimal number of clusters, HOPACH relies on mean split silhouette (MSS), which is a measure of cluster stability given two or more divisions in each cluster. Pollard and

van der Laan note that the "average silhouette tends to be a global criteria in the sense that it is not necessarily maximized at the level of the tree which we would select visually but rather usually higher up in the tree" (2003: 284). As a result, they created MSS. MSS is applied similarly to average silhouette by running down the tree until no improvements can be made, stopping at the level with the significant number of clusters.

3.3 Evaluation Methods

Our evaluation of the clustering output can be classified into two categories: determination of the optimal number of clusters and determination of the variability of the clustering output. There are several methods to find the optimal number of clusters, including maximizing average silhouette width, minimizing mean split silhouette, and applying the L-Method to different evaluation metrics. When evaluating the variability of cluster outputs, one may look at how individual elements would cluster in different samples. That is, by "creating" more samples and examining their clustering results, we can calculate the frequency an element is clustered in relation to its original cluster assignment. The described method is known as bootstrapping.

3.3.1 Maximizing Average Silhouette Width

The silhouette of a gene measures how well matched it is to the other objects in its own cluster versus how well matched it would be if it were moved to another cluster. For gene j, let a_j be the average dissimilarity of gene j with the other elements of its cluster and let b_j be the minimum average dissimilarity of gene j with members of other clusters. The silhouette of gene j is given by

$$S_j = \begin{cases} 1 - \frac{a_j}{b_j} & \text{if } a_j < b_j \\ 0 & \text{if } a_j = b_j \\ \frac{b_j}{a_j} - 1 & \text{if } a_j > b_j \end{cases}$$

which can be rewritten as

$$S_j = \frac{b_j - a_j}{\max(a_j, b_j)} \tag{3.1}$$

If the dissimilarity within the cluster is much smaller than the dissimilarity between clusters, we call gene j "well classified" and the value of the silhouette is close to one. When the dissimilarities are nearly equal, it is unclear which group gene j should be placed and the silhouette is 0. In the last case, where gene j lies closer to another cluster than its own, it is "misclassified" and the silhouette is close to -1. In all cases, the possible values of the silhouette range from -1 to 1.

Since silhouettes only require a clustered sample and a set of dissimilarities, the average silhouette width can be found using any clustering method and

any distance metric, allowing the maximization of average silhouette width to be an evaluation technique. The average silhouette width is found by calculating the silhouette for all objects. Taking the average of the silhouettes of the objects in a cluster gives the average silhouette width for that cluster, and the average silhouette for the entire sample can be similarly calculated using the silhouettes of all the objects. The choice to maximize the average silhouettes arises from the range of values mentioned before. When all the elements are best classified, the silhouettes should all be close to 1, and the average silhouette width will be high. Therefore, the highest average silhouette width gives the strongest clustering structure, resulting in an output which gives the optimal number of clusters.

3.3.2 Minimizing Mean Split Silhouette

Finding the mean split silhouette (MSS) expands on the silhouette theory, as described above. Beginning with a clustering result of k clusters, each cluster is split into two or more clusters with the same clustering technique as used to find the initial k clusters. After this split, each gene has a new silhouette (see equation 3.1), which is computed relative to only those genes with which it shares a parent. The mean of the silhouettes for each parent cluster i is the split silhouette, SS_i, which measures the cluster's homogeneity. Finally, finding the mean of these split silhouettes over the k clusters gives us the mean split silhouette.

A low split silhouette means the cluster was homogeneous and should not have been split. Therefore, when minimizing the mean split silhouette, the optimal structure will be given as k clusters.

3.3.3 L Method

Another method of determining the number of clusters is to find the knee of a curve describing an evaluation method. In looking at an evaluation graph, we would roughly be finding the point of maximum curvature. There are several methods to find the knee: the largest magnitude difference between two points, the largest ratio difference between two points, the data point with the largest second derivative, to name a few. The L method, created by Salvador and Chan [7], finds the knee by fitting a pair of straight lines that most closely fit the curve and locating the boundary or point of intersection. This method locates the knee globally, since it considers all points. Local trends and outliers do not hinder locating the true knee, whereas some of the other methods may be led astray by these possibilities. The L method can also find knees at sharp jumps in the curve.

The evaluation graph used by the L method has the number of clusters on the horizontal axis. The evaluation function on the vertical axis may vary. What we have used so far, and what will be discussed in this paper, comes from PAM and HOPACH results. Running PAM with the range of numbers of clusters, the program returns several useful variables: the maximum distance

from the medoid, the average dissimilarity of the medoid to other elements in the cluster, and the maximum overall distance. These variables return a value for each cluster, and we take a weighted average according to cluster sizes, which is our evaluation metric on the vertical axis. The points on the graph represent the weighted averages of the variable for different numbers of clusters. Fitting two straight lines to the curve by minimizing the mean squared error of the best-fit lines, the knee occurs at the intersection of the two lines. Each line must have at least two points, and the optimal number of clusters is the right most point on the first line. That is, if the first line ranges from two to c clusters and the second line is from $c+1$ to k clusters where k is the upper bound number of clusters, then the optimal number of clusters is c.

3.3.4 Bootstrap Method

The previous evaluation methods considered the optimal number of clusters, comparing this number to that given in the clustering output. Another way of evaluating cluster output is to consider whether individual data elements would be re-clustered into the same group. With only one data set, applying the same clustering technique only results in identical output. As a result, we employ the bootstrap method, which resamples from the original sample with replacement. So, in a sample with 100 elements, we would "create" another sample of 100 by randomly choosing one element, putting it back, and randomly selecting another element from the original sample, including the one previously chosen. Repeating the resampling process at least 1000 times gives a close approximation to the original. When trying to estimate a parameter with one sample, the bootstrap method allows estimation of the parameter along with the variability of the estimate from many resamples. Typically, bootstrap resampling creates very little bias, where most of the bias comes from that of the sample representing the population.

Applying the bootstrap theory to clustering results, van der Laan and Pollard [8] run a nonparametric bootstrap, resampling from the empirical distribution which puts mass $\frac{1}{n}$ on each of the original observations. To estimate the variance of the clustering outputs, they find a correspondence between the original clusters and the bootstrap resample clusters by fixing the medoids from the original clustering sample. Given the type of data we are working with, where we have n objects with p variables measured on each object, the bootstrap method resamples by choosing the variables with replacement. The resample will include every object with a new combination of variables. Since the variables may change in each resample, an element may not necessarily be assigned to its original cluster. If the variable resampled played an influential role in the original medoid decision, then without that variable, the element may be closer to another medoid, in which case it is assigned to that corresponding cluster. With the clustering results from the resamples, we can conclude whether an original cluster is stable or not based on how consistently its elements are placed into the same cluster. The bootstrap technique allows a closer inspection of the strength of the clustering result through an examination of

individual genes cluster placements in slightly different samples.

Chapter 4

Data

The microarray data used in this thesis represent the gene expression of aging yeast cells. There are approximately 5500 genes with 28 observations per gene. Each observation represents a yeast sample of a particular generation (that is, a particular number of cell divisions). To find the dissimilarity between two particular genes, the distance is found between the 28 observations of those two genes, and we mainly use Pearson's correlation and percentage bend correlation. So, we calculate the correlation between the 28 observations of two genes to obtain their distance.

One goal of the project is to explore the different clustering and testing methods. In order to begin with a knowledge of the truth, clustered samples are pseudo-simulated. Applying the clustering and testing techniques to the pseudo-simulated sample, we have a preconceived notion of what the output should be. Our pseudo-simulated sample consists of three very distinct clusters found by choosing one gene at random and selecting the twenty genes most correlated with that gene, giving us our first cluster. Then, taking the least correlated gene to the first one randomly selected and finding its twenty most correlated genes, we have a second cluster. Two lists ranking the remaining genes are created with the furthest gene first on the list. The first list will rank the remaining genes according to their correlations with the first chosen gene, and the second similarly ranks according to the correlation with the second selected gene. The third gene that is chosen is the gene that ranks the highest on both lists. After that gene's selection, the twenty genes most correlated with the chosen gene form the third cluster. More clusters can be created following the same pattern, but for simplicity, three clusters are created here. We create two pseudo-simulated samples using the two distance metrics of Pearson's correlation and percentage bend correlation.

With such distinct clusters, the different distance metrics may easily distinguish the clusters, giving us little or no insight into their qualities. However, the clarity of results with the different clustering algorithms allow easier interpretation of the plots, and we know that the testing methods should find three as the optimal number of clusters. Since we know what the desired output is,

any unexpected results are more significant.

We use a different sample for the bootstrap testing method. Since the bootstrap method finds the variability of clustering output, we need a sample whose "correct" clustering may be ambiguous. Microarray data is very noisy, so we keep the noise present in this sample and we randomly select a subset of size 500 genes from the entire sample. The choice of taking a subset of 500 genes stems from the demanding computation time of larger samples sizes.

Using the bootstrap technique, we would like to find a method of improving the cluster output, in which case a representation of the original data is necessary. To explain the bootstrap theory applied to our microarray data, the 28 observations of the genes are resampled with replacement to create the new sample. The 500 genes of the sample are all present in each resample, but the observations included may change. In order to determine whether our method of improving the cluster output indeed works, we apply our method to ten different samples, each of size 500.

Chapter 5

Results and Discussion

5.1 Clustering

We begin with using the two pseudo-simulated samples. Using the distance metrics of Pearson's correlation and percentage bend correlation, we obtain a total of four distance matrices; for each sample, two distance matrices are calculated with the different distance metrics. The clustering algorithms are run for each distance matrix, and the remainder of the section contains a selection of the results produced.

The hierarchical clustering in all cases of single, complete, or average linkage typically found the three main clusters for all four distance matrices. Figure 6.2 shows the dendrogram using average linkage of the Pearson's correlation distance metric applied to the pseudo-simulated sample formed with the Pearson's correlation metric. The three main branches split at a much greater height than the individual elements divide from each other. This leads us to conclude, from Figure 6.2, that the clusters are quite separated and tight.

Using the same pseudo-simulated sample but now with the percentage bend correlation distance metric, the hierarchical clustering output in Figure 6.3 shows three distinct branches that also split quite high, but the clusters appear to have a more chain-like grouping of the individual "leaves" than in Figure 6.2, with the Pearson's correlation distance metric.

With our second clustering technique, PAM, and our other pseudo-simulated sample formed using percentage bend correlation distance, we can visually see the distinct, tight clusters of the percentage bend sample using the same distance metric. Shown in Figure 6.4, PAM groups nineteen elements in each cluster. When using the Pearson's correlation distance on the same percentage bend correlation psuedo-simulated sample, PAM groups two clusters of twenty elements and one group of seventeen. Visually, the groups are larger and less dense, as shown with less shading (Figure 6.5).

Using HOPACH on the Pearson's correlation pseudo-simulated sample with the Pearson's correlation distance metric, the mean split silhouette is min-

imized at twenty-four clusters. However, as seen in Table 6.1, many "clusters" are merely singletons. Looking at the heat map (Figure 6.6), we still see three main clusters from the three darker squares along the diagonal. Using percentage bend distance metric on this sample, HOPACH gives twelve clusters, with two having twenty elements each. This seems to be a better result than that using the Pearson's correlation distance metric since it correctly finds two clusters, while splitting up the last one into a few more groups. On the other pseudo-simulated sample of percentage bend correlation, HOPACH gave fifteen and twenty-two clusters when using the distance metrics of percentage bend correlation and Pearson's correlation, respectively. Both results on this sample give multiple singletons.

5.2 Evaluation Methods

5.2.1 Optimal Number of Clusters

In looking at the plots of average silhouette, the average silhouette width is maximized for three clusters using both distance metrics on both samples. Figure 6.7 shows average silhouette width of the percentage bend correlation sample with percentage bend correlation distance metric. The average silhouette plot behaves as expected, given our data. One interesting observation arises from the plots of the Pearson's correlation sample. For both distance metrics, the average silhouettes, ranging from two to twenty clusters, form three sections (Figure 6.8). A possible explanation for this behavior is the appearance of small clusters within a cluster, where we could possibly split the three distinct clusters further. However, when we begin dividing the clusters into too many smaller clusters, the tight groupings are separated and the b_j variable of the silhouette equation becomes very small, giving a low silhouette. Having such odd patterns on the average silhouette plot depends on the particular data set and whether there are groups of points very close together or whether they are all relatively equidistant from each other.

The MSS plots using Pearson's correlation distance metric on the percentage bend correlation sample give surprising results, in that the optimal number of clusters is neither the correct number, three, nor the number of clusters used by HOPACH. (See Figure 6.9). Using percentage bend correlation distance, the MSS plots find three clusters as the minimum for both samples. Perhaps, the greater robustness of the percentage bend correlation distance metric allows the MSS to be minimized at the correct number.

Using the different variables PAM returns, we have several evaluation graphs for the L method. For both samples, using the percentage bend correlation or Pearson's correlation distance metrics, the L method always results with three as the optimal number of clusters. Two examples can be seen in Figures 6.10 and 6.11. When changing the distance metric to Euclidean, we see the L method gives very different results (Figure 6.12). Using this non-robust distance metric on these samples will produce very different clusters, in this case

with many more clusters. We can conclude that, due to the structure of the samples, the Euclidean distance metric should not be used. Since the samples were created using correlation, it is likely that the Euclidean distances of the elements within the clusters are quite large, relative to their correlation. Hence, the two correlation distance metrics may not have difficulty finding the three clusters, whereas the Euclidean distance fails.

5.2.2 Bootstrap

Now looking at the stability of the clustering results, the bootstrap program calculates the proportion of times each element is assigned to the different clusters based on the 1000 resamples. With each resample, the original medoids remain the centers of the clusters and the remaining genes are clustered. The cluster assignment for each gene is noted for the calculation of the proportion of times the gene is put in that specific cluster. Combining the results of the 1000 resamples, the proportions of each gene into each cluster is found and displayed on the plot seen in Figure 6.13.

The proportions for each gene's placement in the different clusters are found following horizontal lines on the plot. The genes are ordered along the vertical axis, based on their original cluster assignment. The colors represent the different clusters, so we see solid regions of colors. Starting from the bottom, each new region begins with a single-colored line, as each cluster is introduced. The bootstrap method used here keeps the same medoids for each resample, and thus they are always assigned to the original cluster, represented by the single-colored line.

Between each solid-colored region we see lines with only a small proportion matching the original cluster color. That is, some genes between the main clusters are not re-assigned to their original cluster the majority of the time. To improve our clustering results, we remove these genes from our sample and re-cluster. Our criteria for gene removal includes two conditions: the highest proportion the gene is assigned to for any cluster must be larger than 0.5 and the gene must be reassigned to the original cluster the greatest proportion of times. Visually, the idea of the removal should provide a bootstrap plot with fewer lines between the solid color portions. An example of a bootstrap plot with the sample without the removed genes can be seen in Figure 6.14. Comparing Figures 6.13 and 6.14, we do indeed see a reduction in the lines between the solid color regions.

5.3 Further Results

In our experiment of trying to find an improved distance metric and clustering technique for microarray data, we use ten different samples of 500 randomly selected genes and compare them using the distance metrics of Pearson's correlation and percentage bend correlation. For each sample, we remove the genes that do not cluster well originally and recluster the "improved" sam-

ple. Hence, for each sample, we run the clustering algorithm four times; using the two distance metrics, we have an original and an "improved" samples. One measure of comparison that we calculate is the percentage of times the genes are placed in the original cluster the greatest proportion of times. For example, suppose that 75 genes of a 100 gene sample are assigned to their original cluster the highest proportion of times using the bootstrap method. Then, we compare the 75% of this sample with the percentage of the "improved" sample.

The comparison of the original sample with the "improved" sample can be found in Tables 6.2 and 6.3. Ideally, the percentages for the improved sample using percentage bend correlation should be the highest. However, we find these to be the lowest. The original sample with the Pearson's correlation metric gave the highest proportions of genes correctly assigned. These results are summarized visually in the boxplots shown in Figure 6.15.

A problem that arises that likely causes the low percentages with the percentage bend correlation metric is the drastic increase in clusters created with the improved sample. Given the same number of genes to cluster, a grouping with less clusters will have greater dissimilarity between the clusters than an output with more groups. With many more clusters, it is difficult for each gene to be reassigned to its original cluster if the clusters are more similar. Since the numbers of clusters for the original and improved samples are not equal, our measure of comparison may need revision.

Another problem that could explain the unexpected results is our method of removing genes to improve our sample. The purpose of removing the genes is to cluster the genes that are consistently assigned to certain clusters, giving a stable clustering result, and then replacing the removed genes, placing them in clusters already formed by the consistent genes. The goal is to find a method to improve the stability of the clustering output, and our method of removing genes by the criteria above does not fulfill our goal, according to the measure of comparison used.

Chapter 6

Conclusion

The work done in the project includes an examination of the different clustering techniques and the evaluation methods using different distance metrics and an experiment in improving the sample and clustering output using the bootstrap method. Most comparisons were done using the similar distance metrics of Pearson's correlation and percentage bend correlation. In specific cases, such as the minimization of MSS and the HOPACH results on the Pearson's correlation sample, the percentage bend correlation performs better than Pearson's correlation. The result provides the output we hoped to obtain, as we hypothesized that using the robust distance metric of percentage bend correlation would give more correct clusterings than when using Pearsons correlation, a non-robust distance metric.

The experiment in improving the clustering results, given the noisy data and using the bootstrap method, did not produce satisfactory results. With the intention of increasing the percentages of genes correctly clustered, we found that the percentages typically decreased. The robust distance metric of percentage bend correlation also performed worse than the Pearson's correlation metric. The poor performance may be due to our limitations in specifying the number of clusters HOPACH produces, where we cannot require HOPACH to group the data into the same number of clusters for both the original and the "improved" samples. However, it is possible that our method of improving the clustering just does not work.

A further extension of this project includes applying the adjusted Rand index to different clustering results, particularly focusing on the two distance metrics. With the pseudo-simulated samples, one could use adjusted Rand to compare the clustering results to the true clusters, the original clusters chosen in creating the samples (Yeung and Ruzzo [12]). Also, a topic for further investigation involves the evaluation method of MSS. The conflict arising from our results is why HOPACH does not cluster according to the smallest MSS, though HOPACH clusters according to where MSS is minimized. Lastly, the method of improving the sample requires more work, as our current results are undesired.

Bibliography

[1] Draghici, Sorin. *Data Analysis Tools for DNA Microarrays*, Chapman & Hall/CRC, London, 2003.

[2] Efron, Bradley and Robert J. Tibshirani. *An Introduction to the Bootstrap*, Chapman & Hall, New York, 1993.

[3] Johnson, Richard A. and Dean W. Wichern. *Applied Multivariate Statistical Analysis*, Prentice Hall, New Jersey, 2002.

[4] Kaufman, Leonard and Peter J. Rousseeuw. *Finding Groups in Data, And Introduction to Cluster Analysis*, Wiley-Interscience Publication, New York, 1990.

[5] Mooney, Christopher Z. and Robert D. Duval. *Bootstrapping: A Nonparametric Approach to Statistical Inference*, Sage Publications, London, 1993.

[6] Pollard, Katherine S. and Mark J. van der Laan. "Cluster Analysis of Genomic Data with Applications in R" *U.C. Berkeley Division of Biostatistics Working Paper Series*. Working Paper 167, 2005.

[7] Salvador, Stan and Philip Chan. "Determining the Number of Clusters/Segments in Hierarchical Clustering/Segmentation Algorithms," *ictai*, 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'04), 2004: pp.576-584.

[8] Van der Laan, Mark J. and Katherine S. Pollard. "A New Algorithm for Hybrid Hierarchical Clustering with Visualization and the Bootstrap" *Journal of Statistical Planning and Inference*, 117 (2003): 275-303.

[9] Van der Laan, Mark J., Katherine S. Pollard, and Jennifer Bryan. "A New Partitioning Around Medoids Algorithm" *U.C. Berkeley Division of Biostatistics Working Paper Series*. Working Paper 105, 2002.

[10] Wilcox, Rand. *Introduction to Robust Estimation and Hypothesis Testing*, Elsevier, Amsterdam, 2004.

[11] Wilcox, Rand. "The Percentage Bend Correlation Coefficient" *Psychometrica*Volume 59, Issue 4 (December 1994):601-616.

[12] Yeung, Ka Yee and Walter L. Ruzzo. "Details of the Adjusted Rand index and Clustering algorithms Supplement to the paper 'An empirical study on Principal Component Analysis for clustering gene expression data" To appear in *Bioinformatics* (May 2003).

Table 6.1: HOPACH clustering on Pearson's correlation sample using Pearson's correlation distance metric

Cluster Labels	110	120	130	200	310	320	330	340	350	410
Number of Clusters	1	1	3	3	2	1	3	1	1	1
Cluster Labels	420	500	610	620	810	820	830	840	850	860
Number of Clusters	3	8	8	4	1	2	1	2	1	3
Cluster Labels	871	873	874	880						
Number of Clusters	7	1	1	1						

Table 6.2: Comparison of Original and "Improved" Samples using Pearson's Correlation

Sample Number	1	2	3	4	5	6	7	8	9	10
Original Sample										
Percentage of correct assignment	.77	.738	.722	.796	.676	.73	.806	.792	.622	.818
Clusters Created	9	5	5	7	12	6	7	6	5	8
"Improved" Sample										
Percentage of correct assignment	.68	.63	.632	.766	.604	.532	.72	.786	558	.532
Clusters Created	7	7	7	7	21	8	11	5	7	9

Table 6.3: Comparison of Original and "Improved" Samples using Percentage Bend Correlation

Sample Number	1	2	3	4	5	6	7	8	9	10
Original Sample										
Percentage of correct assignment	.68	.792	.634	.774	.506	.558	.768	.57	.628	.73
Clusters Created	8	6	13	5	17	15	5	5	6	3
"Improved" Sample										
Percentage of correct assignment	.49	.604	.508	.638	.31	.334	.7	.334	.496	.484
Clusters Created	35	10	13	14	16	16	7	16	12	27

Figure 6.1: A Microarray Chip

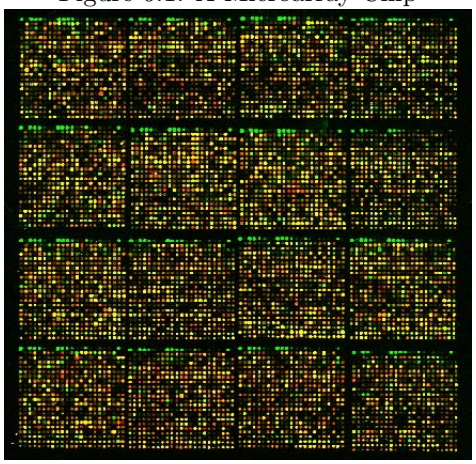

Figure 6.2: Hierarchical Clustering Dendrogram of Pearson's Correlation Sample using Pearson's Correlation Distance Metric

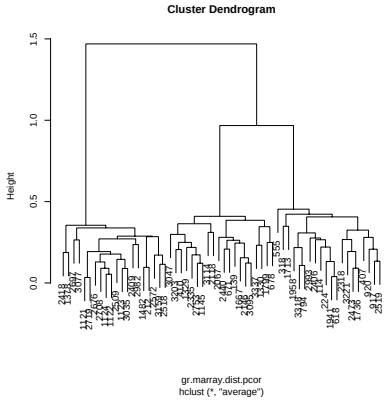

Figure 6.3: Hierarchical Clustering Dendrogram of Pearson's Correlation Sample using Percentage Bend Correlation Distance Metric

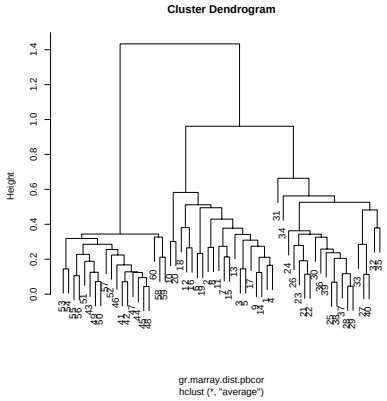

Figure 6.4: PAM Partitioning of Percentage Bend Correlation Sample using Percentage Bend Correlation Distance Metric

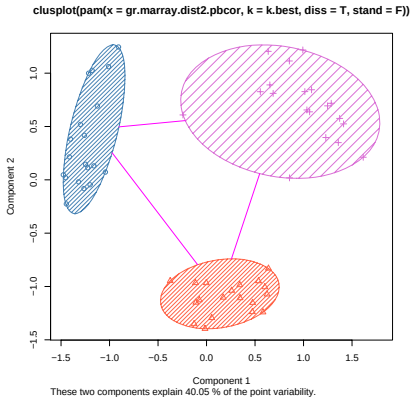

Figure 6.5: PAM Partitioning of Percentage Bend Correlation Sample using Pearson's Correlation Distance Metric

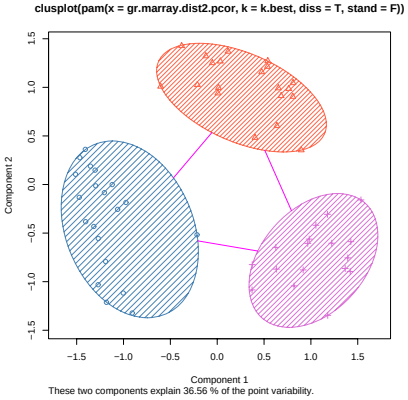

Figure 6.6: HOPACH Heat Map of Pearson's Correlation Sample using Pearson's Correlation Distance Metric

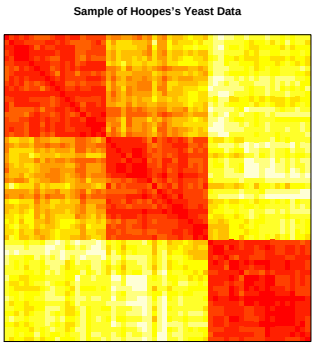

Figure 6.7: Average Silhouette of Percentage Bend Correlation Sample using Percentage Bend Correlation Distance Metric

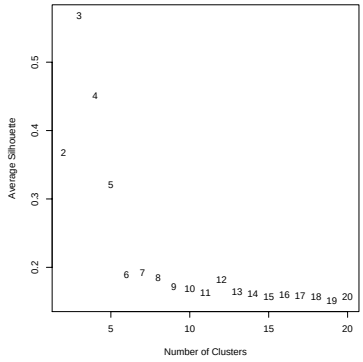

Figure 6.8: Average Silhouette of Pearson's Correlation Sample using Pearson's Correlation Distance Metric

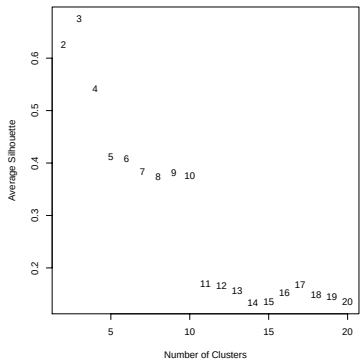

Figure 6.9: Mean Split Silhouette of Percentage Bend Correlation Sample using Pearson's Correlation Distance Metric

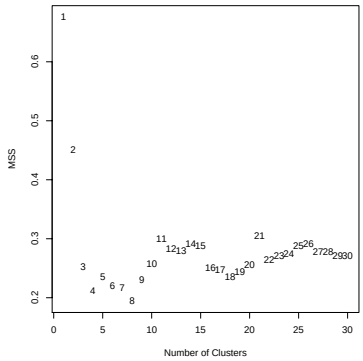

Figure 6.10: L Method on PAM Results of Percentage Bend Correlation Sample using Percentage Bend Correlation Distance Metric

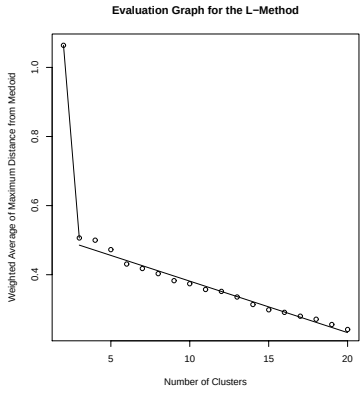

Figure 6.11: L Method on PAM Results of Pearson's Correlation Sample using Percentage Bend Correlation Distance Metric

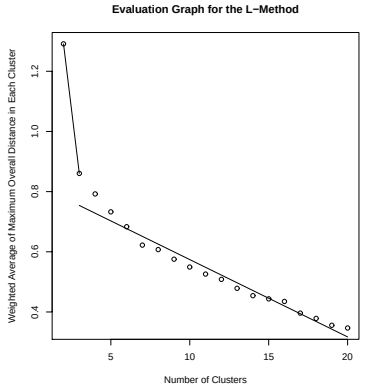

Figure 6.12: L Method on PAM Results of Percentage Bend Correlation Sample using Euclidean Distance Metric

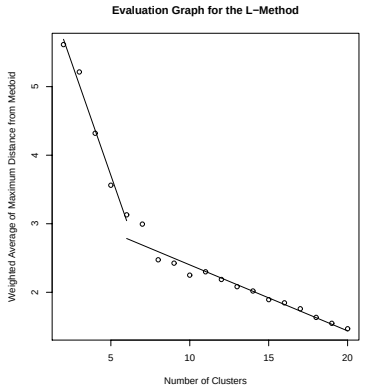

Figure 6.13: Original Sample Bootstrap Plot

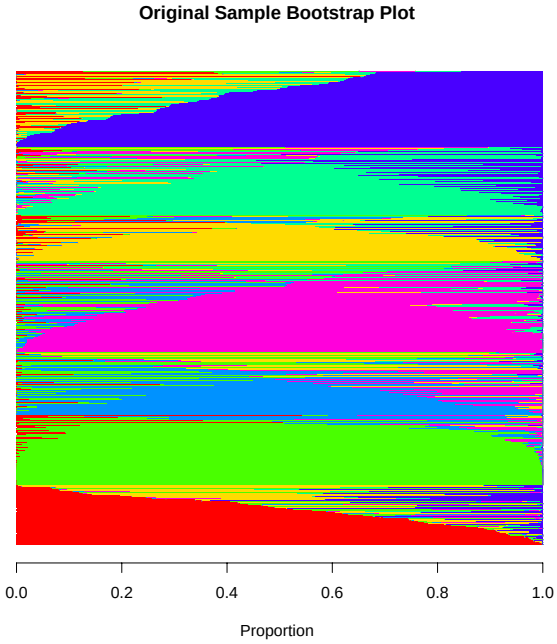

Figure 6.14: Improved Sample Bootstrap Plot

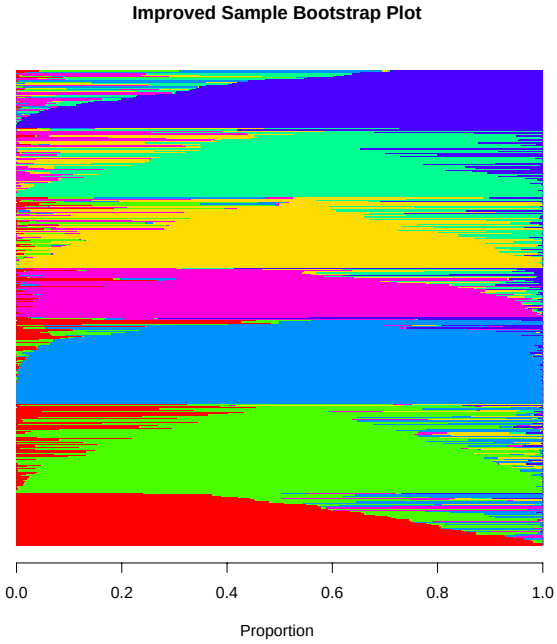

Figure 6.15: Boxplot of Proportions of Genes in the Correct Cluster