

Detecting Stable Clusters Using Principal Component Analysis

Asa Ben-Hur and Isabelle Guyon

1 Introduction

Clustering is one of the most commonly used tools in the analysis of gene expression data (**1, 2**). The usage in grouping genes is based on the premise that co-expression is a result of co-regulation. It is thus a preliminary step in extracting gene networks and inference of gene function (**3, 4**). Clustering of experiments can be used to discover novel phenotypic aspects of cells and tissues (**3, 5, 6**), including sensitivity to drugs (**7**), and can also detect artifacts of experimental conditions (**8**). Clustering and its applications in biology are presented in greater detail in the chapter by Zhao and Karypis (see also (**9**)). While we focus on gene expression data in this chapter, the methodology presented here is applicable for other types of data as well.

Clustering is a form of unsupervised learning, i.e. no information on the class variable is assumed, and the objective is to find the “natural” groups in the data. However, most clustering algorithms generate a clustering even if the data has no inherent cluster structure, so external validation tools are required. Given a set of partitions of the data into an increasing number of clusters (e.g. by a hierarchical clustering algorithm, or k-means), such a validation tool will tell the user the number of clusters in the data (if any). Many methods have been proposed in the literature to address this problem (**10–15**). Recent studies have shown the advantages of sampling-based methods (**12, 14**). These methods are based on the idea that when a partition has captured the structure in the data, this partition should be stable with respect to perturbation of the data. Bittner *et al.* (**16**) used a similar approach to validate clusters representing gene expression of melanoma patients.

The emergence of cluster structure depends on several choices: data representation and normalization, the choice of a similarity measure and clustering algorithm. In this chapter we extend the stability-based validation of cluster structure, and propose stability as a figure of merit that is useful for comparing clustering solutions, thus helping in making these choices. We use this framework to demonstrate the ability of Principal Component Analysis (PCA) to extract features relevant to the cluster structure. We use stability as a tool for simultaneously choosing the number of principal components and the number of clusters; we compare the performance of different similarity measures and normalization schemes. The approach is demonstrated through a case study of yeast gene expression data from Eisen *et al.* (**1**). For yeast, a functional classification of a large number of genes is known, and we use this classification for validating the results produced by clustering. A method for comparing clustering solutions specifically applicable to gene expression data was introduced in (**17**). However, it cannot be used to choose the number of clusters, and is not directly applicable in choosing the number of principal components.

The results of clustering are easily corrupted by the addition of noise: even a few

noise variables can corrupt a clear cluster structure **(18)**. Several factors can hide a cluster structure in the context of gene expression data or other types of data: the cluster structure may be apparent in only a subset of the experiments or genes, or the data itself may be noisy. Thus clustering can benefit from a preprocessing step of feature/variable selection or from a filtering or de-noising step. In the gene expression case study presented in this chapter we find that using a few leading principal components enhances cluster structure. For a recent paper that discusses PCA in the context of clustering gene expression see also **(19)**.

PCA constructs a set of uncorrelated directions that are ordered by their variance **(20, 21)**. In many cases, directions with the most variance are the most relevant to the clustering. Our results indicate that removing features with low variance acts as a filter that results in a distance metric that provides a more robust clustering. PCA is also the basis for several variable selection techniques: variables that have a large component in low variance directions are discarded **(22, 23)**. This is also the basis of the “gene shaving” method **(24)** that builds a set of variables by iteratively discarding variables that are least correlated with the leading principal components. In the case of temporal gene expression data the principal components were found to have a biological meaning, with the first components having a common variability **(25–27)**. PCA is also useful as a visualization tool - it can provide a low dimensional summary of the data **(28)**, help detect outliers, and perform quality control **(20)**.

2 Principal Components

We begin by introducing some notation. Our object of study is an n by d gene expression matrix, X , giving the expression of n genes in d experiments. The gene expression matrix has a dual nature: one can cluster either genes or experiments; to express this duality we can refer to X as

$$X = \begin{pmatrix} - & \mathbf{g}_1 & - \\ & \vdots & \\ - & \mathbf{g}_n & - \end{pmatrix} \quad (1)$$

where $\mathbf{g}_i = (x_{i1}, \dots, x_{id})$ are the expression levels of gene i across all experiments, or as

$$X = \begin{pmatrix} | & & | \\ \mathbf{e}_1 & \cdots & \mathbf{e}_d \\ | & & | \end{pmatrix} \quad (2)$$

where $\mathbf{e}_j = (x_{1j}, \dots, x_{nj})'$ are the expression levels in experiment j across all genes. In this chapter we cluster genes, i.e. the n patterns \mathbf{g}_i . When clustering experiments, substitute \mathbf{e} and \mathbf{g} in what follows. We make a distinction between a *variable*, which is each one of the d variables that make up \mathbf{g}_i , and a *feature*, which denotes a combination of variables.

The principal components are q orthogonal directions that can be defined in several equivalent ways **(20, 21)**. They can be defined as the q leading eigenvectors of the covariance matrix of X . The eigenvalue associated with each vector is the variance in that

direction. Thus, PCA finds a set of directions that explain the most variance. For Gaussian data the principal components are the axes of any equiprobability ellipsoid. A low dimensional representation of a dataset is obtained by projecting the data on a small number of PCs (principal components). Readers interested in theoretical or algorithmic aspects of PCA should refer to textbooks devoted to the subject (**20, 21**).

The principal components can be defined as the q leading eigenvectors of the experiment-experiment covariance matrix:

$$Cov(X)_{ij} = \frac{1}{n}(\mathbf{e}_i - \langle \mathbf{e}_i \rangle)'(\mathbf{e}_j - \langle \mathbf{e}_j \rangle), \quad i, j = 1, \dots, d \quad (3)$$

where $\langle \mathbf{e}_j \rangle = \frac{1}{n} \sum_{i=1}^n x_{ij}(1, \dots, 1)$ is a d dimensional vector with the mean expression value for experiment j . Alternatively, the principal components can be defined through the correlation matrix:

$$Cor(X)_{ij} = \frac{1}{n} \frac{(\mathbf{e}_i - \langle \mathbf{e}_i \rangle)'(\mathbf{e}_j - \langle \mathbf{e}_j \rangle)}{\sigma(\mathbf{e}_i)\sigma(\mathbf{e}_j)}, \quad i, j = 1, \dots, d \quad (4)$$

where $\sigma(\mathbf{e}_i)$ is the vector of estimated standard deviation in experiment i . Equivalently, one can consider the principal components as the eigenvectors of the matrix XX' when applying first a normalization stage of *centering*:

$$\mathbf{e}_i \rightarrow \mathbf{e}_i - \langle \mathbf{e}_i \rangle \quad (5)$$

or *standardization*:

$$\mathbf{e}_i \rightarrow (\mathbf{e}_i - \langle \mathbf{e}_i \rangle) / \sigma(\mathbf{e}_i). \quad (6)$$

The first corresponds to PCA relative to the covariance matrix, and the second to PCA relative to the correlation matrix. To distinguish between the two, we denote them by centered PCA and standardized PCA, respectively. One can also consider PCs relative to the second moment matrix, i.e. without any normalization. In this case the first PC often represents the mean of the data, and the larger the mean, the larger this component relative to the others.

We note that for the case of two dimensional data the correlation matrix is of the form $(1, a; a, 1)$, which has *fixed* eigenvectors $(x, -x)$ and (x, x) ($x = \frac{\sqrt{2}}{2}$, for normalization), regardless of the value of a . Standardization can be viewed as putting constraints on the structure of the covariance matrix that in two dimensions fixes the PCs. In high dimensional data this is not an issue, but a low dimensional comparison of centered PCA and standardized PCA would be misleading. Standardization is often performed on data that contains incommensurate variables, i.e. variables that measure different quantities, and are incomparable unless they are made dimensionless, e.g. by standardization. In the case of commensurate variables it was observed that standardization can reduce the quality of a clustering (**29**). In microarray data all experiments measure the same quantity, namely mRNA concentration, but still, normalization across experiments might be necessary.

The basic assumption in using PCA as a preprocessing before clustering is that directions of large variance are the result of structure in those directions. We begin with a toy

example that illustrates this, and later we will show results for gene expression data where this applies as well. Consider the data plotted in Figure 1 (see caption for details on its construction). There is clear cluster structure in the first and second variables. Centered PCA was applied. The first principal component captures the structure that is present in the first two variables. Figure 2 shows that this component is essentially a 45 degree rotation of the first two variables.

3 Clustering and hierarchical clustering

All clustering algorithms use a *similarity* or *dissimilarity* matrix, and group together patterns that are similar to each other. A similarity matrix gives a high score to “similar” patterns, with common examples being the Euclidean dot product or Pearson correlation. A dissimilarity matrix is a matrix whose entries reflect a “distance” between pairs of patterns, i.e. close patterns have a low dissimilarity. The input to the clustering algorithm is either the similarity/dissimilarity matrix or the data patterns themselves, and the elements of the matrix are computed as needed.

Clustering algorithms can be divided into two categories according to the type of *output* they produce:

- *Hierarchical* clustering algorithms – output a dendrogram, which is a tree representation of the data whose leaves are the input patterns and whose non-leaf nodes represent a hierarchy of groupings (see Figure 7). These come in two flavors: *agglomerative* and *divisive*. Agglomerative algorithms work bottom up, with each pattern in a separate cluster; clusters are then iteratively merged, according to some criterion. Divisive algorithms start from the whole data set in a single cluster and work top-down by iteratively dividing each cluster into two components until all clusters are singletons.
- *Partitional* algorithms: provide a partition of a dataset into a certain number of clusters. Partitional algorithms generally have input parameters that control the number of clusters produced.

A hierarchical clustering algorithm can be used to generate a partition, e.g. by cutting the dendrogram at some level to generate a partition into k clusters (see Figure 7 for an illustration). When doing so, we ignore singleton clusters, i.e. when cutting the dendrogram to generate k clusters, we look for k non-singleton clusters. We found it useful to impose an even higher threshold, to ignore very small clusters. This approach provides a unified way of considering hierarchical and partitional algorithms, making our methodology applicable to a generic clustering algorithm. We choose to use the average linkage variety of hierarchical clustering (**30, 31**), that has been used extensively in the analysis of gene expression data (**1, 2, 16**). In agglomerative hierarchical clustering algorithms the two nearest (or most similar) clusters are merged at each step. In average linkage clustering the distance between clusters is defined as the average distance between pairs of patterns that belong

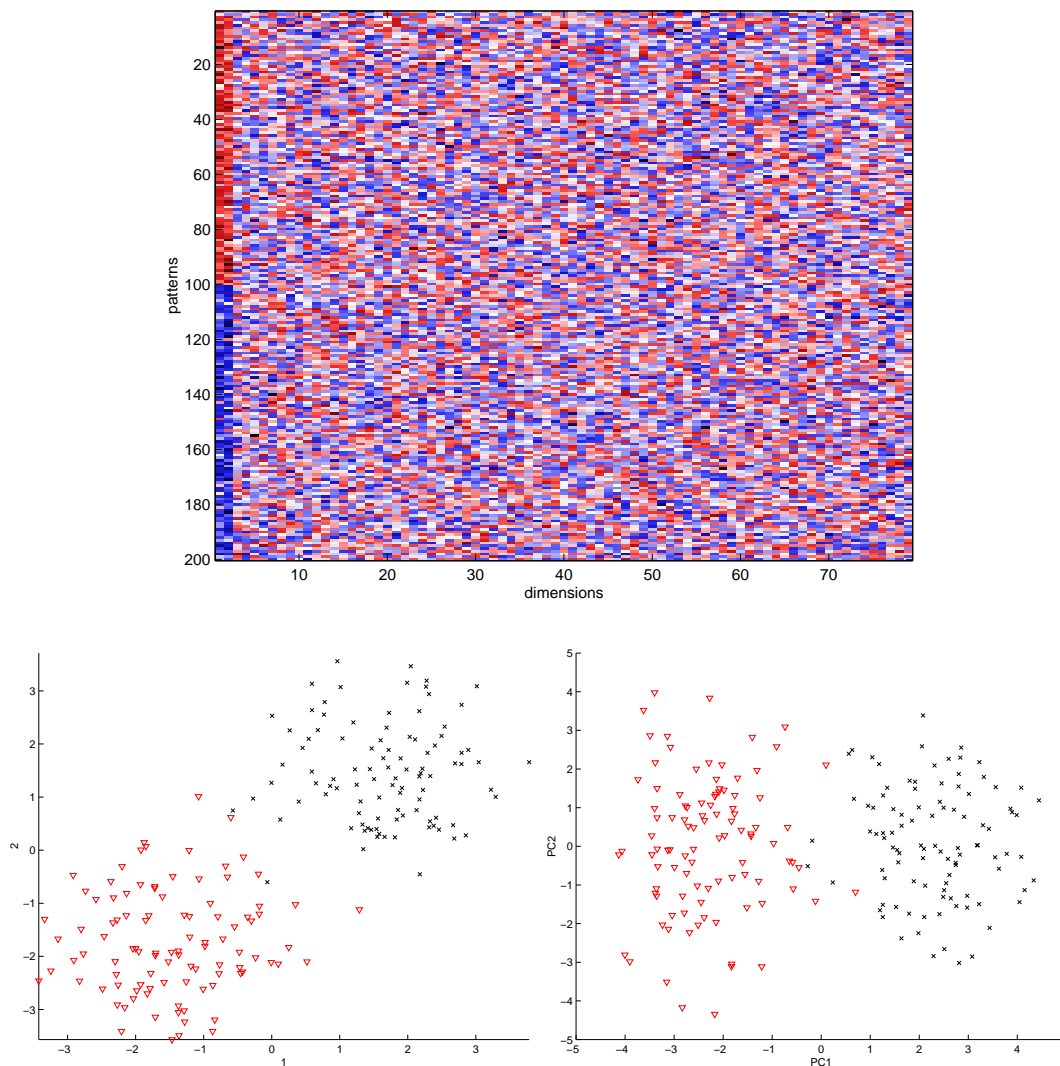


Figure 1: A synthetic data example. Data consists of 200 patterns with 79 dimensions (variables), with components that are standard Gaussian i.i.d. numbers. As can be seen in the representation of the data (top), we added an offset to the first 2 dimensions (a positive offset of 1.6 for the first 100 patterns and a negative offset of -1.6 for the last 100). This results in the two clusters apparent in the scatter plot of the patterns in the first two dimensions (bottom left). The direction that separates the two clusters is captured by the first principal component, as shown on the bottom right scatter plot of the first two principal components.

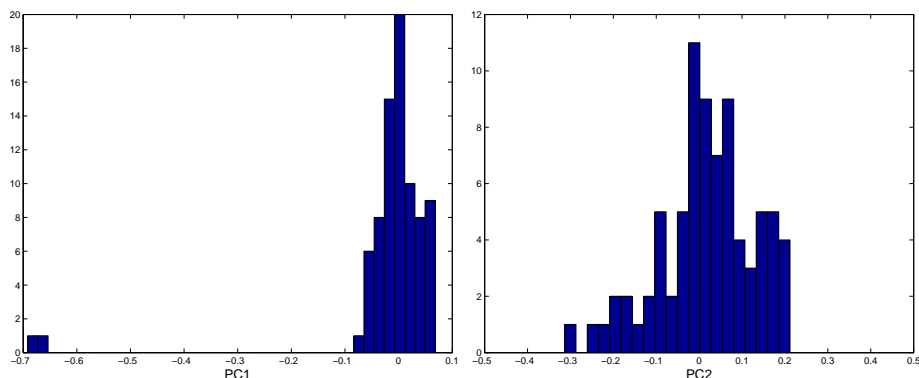


Figure 2: Histograms of the components of the first principal component (left) and the second principal component (right). The count near -0.7 is the value for the first and second variables, representing the 45 degree rotation seen in Figure 1.

to the two clusters; as clusters are merged the distance matrix is updated recursively, making average linkage and other hierarchical clustering algorithms efficient and useful for the large datasets produced in gene expression experiments.

3.1 Clustering stability

When using a clustering algorithm several issues must be considered **(10)**:

- The choice of a clustering algorithm.
- Choice of a normalization and similarity/dissimilarity measure.
- Which variables/features to cluster.
- Which patterns to cluster.
- How many clusters: A clustering algorithm provides as output either a partition into k clusters or a hierarchical grouping, and does not answer the question whether there is actually structure in the data, and if there is, what are the clusters that best describe it.

In this section we introduce a framework that helps in making these choices. We will use it to choose the number of leading PCs (a form of feature selection), and compare different types of normalization and similarity measures, simultaneously with the discovery of the cluster structure in the data.

The method we are about to describe is based on the following observation: when one looks at two sub-samples of a cloud of data patterns with a sampling ratio, f (fraction of patterns sampled) not much smaller than 1 (say $f > 0.5$), one usually observes the same general structure (see Figure 3). Thus it is reasonable to postulate that a partition into k

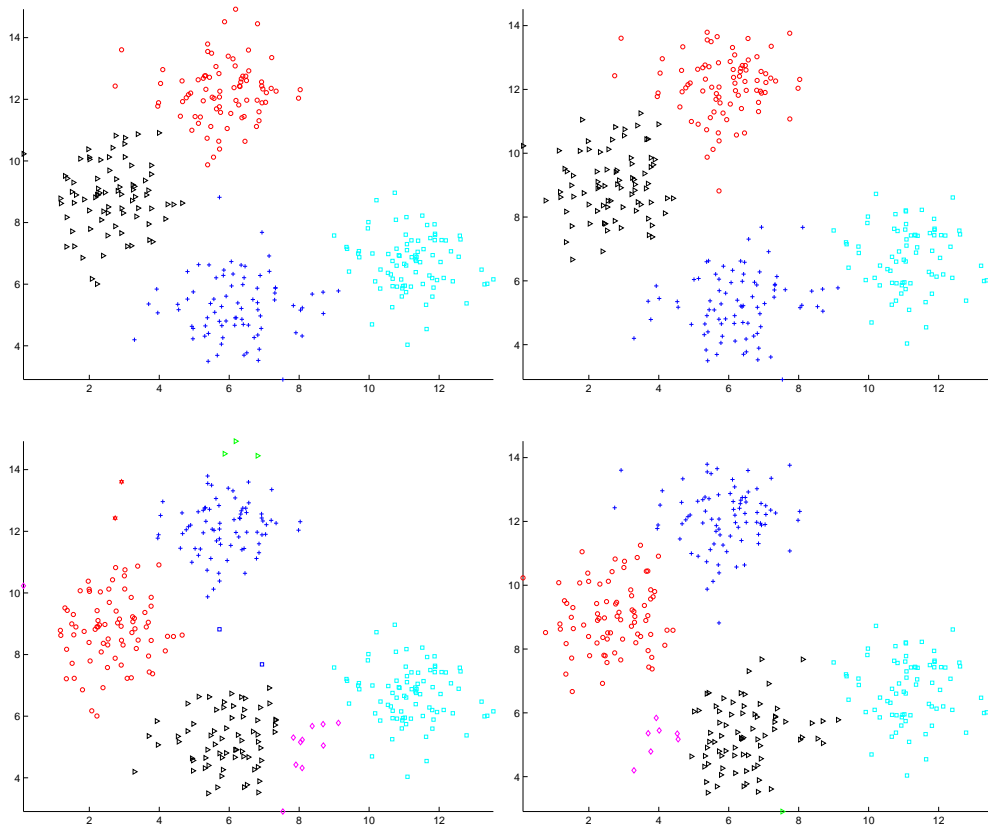


Figure 3: Two 320-pattern subsamples of a 400-pattern Gaussian mixture. Top: the two subsamples have essentially the same cluster structure, so clustering into 4 clusters yields similar results. Bottom: same subsamples; additional clusters can pop up in different locations due to different local substructure in each of the subsamples.

clusters has captured the structure in a dataset if partitions into k clusters obtained from running the clustering algorithm with different sub-samples are similar.

This idea is implemented as follows: the whole dataset is clustered (a reference clustering); a set of subsamples is generated and clustered as well. For increasing values of k the similarity between partitions of the reference clustering into k clusters and partitions of the subsamples are computed (see pseudo-code in Figure 4). When the structure in the data is well represented by k clusters the partition of the reference clustering will be highly similar to partitions of the subsampled data. At a higher value of k some of the clusters will become unstable, and a broad distribution of similarities will be observed (**12**). The stable clusters represent the statistically meaningful structure in the data. Lack of structure in the data can also be detected: In this case the transition to instability occurs between $k = 1$ (all partitions identical by definition), and $k = 2$.

The algorithm we have presented has two modular components (not mentioning the clustering algorithm itself):

- (1) A perturbation of the dataset (e.g. subsampling).
- (2) A measure of similarity between the perturbed clustering and a reference clustering (or alternatively, between pairs of perturbed clusterings).

Perturbing the data to probe for stability can be performed in several ways. One can subsample the patterns, as done here; when clustering experiments one can consider subsampling the genes instead: this is reasonable in view of the redundancy observed in gene expression – one typically observes many genes that are highly correlated with each other. Another alternative is to add noise to the data (**15**). In both cases the user has to decide on the magnitude of the perturbation – what fraction of the features or variables to subsample, or how much noise to add. In our experiments we found that subsampling worked well, and equivalent results were obtained for a wide range of subsampling fractions.

For the second component of the algorithm, a measure of similarity, one can choose one of the several similarity measures were proposed in the statistical literature. We introduce a similarity measure originally defined in (**15**), and then propose an additional one that provides more detailed information about the relationship between the two clusterings.

We define the following matrix representation of a partition:

$$C_{ij} = \begin{cases} 1 & \text{if } \mathbf{g}_i \text{ and } \mathbf{g}_j \text{ belong to the same cluster and } i \neq j, \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

Let two clusterings have matrix representations $C^{(1)}$ and $C^{(2)}$. The dot product

$$\langle C^{(1)}, C^{(2)} \rangle = \sum_{i,j} C_{ij}^{(1)} C_{ij}^{(2)} \quad (8)$$

counts the number of pairs of patterns clustered together in both clusterings and can also be interpreted as the number of edges common to the graphs represented by $C^{(1)}$ and $C^{(2)}$. The dot product satisfies the Cauchy-Schwartz inequality: $\langle C^{(1)}, C^{(2)} \rangle \leq \sqrt{\langle C^{(1)}, C^{(1)} \rangle \langle C^{(2)}, C^{(2)} \rangle}$, and thus can be normalized into a correlation or cosine similarity measure:

$$s(C^{(1)}, C^{(2)}) = \frac{\langle C^{(1)}, C^{(2)} \rangle}{\sqrt{\langle C^{(1)}, C^{(1)} \rangle \langle C^{(2)}, C^{(2)} \rangle}} \quad (9)$$

Input: A dataset X , k_{\max} : maximum number of clusters, $num_subsamples$: number of subsamples.

Output: $S(i, k)$ - a distribution of similarities between partitions into k clusters of a reference clustering and clustering of subsamples; $i = 1, \dots, num_subsamples$

Requires: $T = \text{cluster}(X)$: A hierarchical clustering algorithm

$L = \text{cut-tree}(T, k)$: produces a partition with k non-singleton clusters

$s(L_1, L_2)$: a similarity between two partitions

```
1:  $f = 0.8$ 
2:  $T = \text{cluster}(X)$  {the reference clustering}
3: for  $i = 1$  to  $num\_subsamples$  do
4:    $sub_i = \text{subsamp}(X, f)$  {sub-sample a fraction  $f$  of the data}
5:    $T_i = \text{cluster}(sub_i)$ 
6: end for
7: for  $k = 2$  to  $k_{\max}$  do
8:    $L_1 = \text{cut-tree}(T, k)$  {partition the reference clustering}
9:   for  $i = 1$  to  $maximum\_iterations$  do
10:     $L_2 = \text{cut-tree}(T_i, k)$ 
11:     $S(i, k) = s(L_2, L_1)$  computed only on the patterns of  $sub_i$ .
12:   end for
13: end for
```

Figure 4: Pseudo-code for producing a distribution of similarities. If the distribution of similarities is concentrated near its maximum value, then the corresponding reference partition is said to be stable. In the next sub-section it will be refined to assign a stability to individual clusters. Here it is presented for a hierarchical clustering algorithm, but it can be used with a generic clustering algorithm with minor changes.

Remark 3.1 The cluster labels produced by a clustering algorithm (“cluster 1”, “cluster 2” etc.) are arbitrary, and the similarity measure defined above is independent of actual cluster labels: it is defined through the relationship between pairs of patterns – whether patterns i and j belong to the same cluster, regardless of the label given to the cluster.

As mentioned above, the signal for the number of clusters can be defined by a transition from highly similar clustering solutions, to a wide distribution of similarities. In some cases the transition is not well defined: if a cluster breaks into two clusters, one large, and the other small, the measure of similarity presented above will still give a high similarity score to the clustering. As a consequence, the number of clusters will be over-estimated. To address this issue, and to provide stability scores to individual clusters, we present a new similarity score.

3.2 Associating clusters of two partitions

Here we represent a partition L by assigning a cluster label from 1 to k to each pattern. The similarity measure defined next is motivated by the success rate from supervised learning, which is the sum of the diagonal elements of the confusion matrix between two sets of labels L_1 and L_2 . The confusion matrix measures the size of the intersection between the clusters in two labelings:

$$M_{ij} = |L_1 = i \cap L_2 = j|, \quad (10)$$

where $|A|$ denotes the cardinality of the set A , and $L = i$ is the set of patterns in cluster i . A confusion matrix like

$$M = \begin{pmatrix} 47 & 2 \\ 1 & 48 \end{pmatrix},$$

represents clusterings that are very similar to each other. The similarity will be quantified by the sum of the diagonal elements. However, in clustering no knowledge about the clusters is assumed, so the labels $1, \dots, k$ are arbitrary, and any permutation of the labels represents the same clustering. So we might actually get a confusion matrix of the form

$$M = \begin{pmatrix} 1 & 48 \\ 47 & 2 \end{pmatrix},$$

that represents the relationship between equivalent representations of the same clustering. This confusion matrix can be “diagonalized” if we identify cluster 1 in the first labeling with cluster 2 in the second labeling and cluster 2 in the first labeling with cluster 1 in the second labeling. The similarity is then computed as the sum of the diagonal elements of the “diagonalized” confusion matrix. The diagonalization, essentially a permutation of the labels, will be chosen to maximize the similarity. We now formulate these ideas. Given two labelings L_1 and L_2 with k_1, k_2 labels, respectively, we assume $k_1 \leq k_2$. An *association* σ is defined as a one to one function $\sigma : \{1, \dots, k_1\} \mapsto \{1, \dots, k_2\}$. The unsupervised analog of the success rate is now defined:

$$s(L_1, L_2) = \max_{\sigma \text{ is an association}} \frac{1}{n} \sum_i M_{i\sigma(i)}. \quad (11)$$

Remark 3.2 The computation of the optimal association $\sigma(L_1, L_2)$ by brute-force enumeration is exponential in the number of clusters, since the number of possible one-to-one associations is exponential. To handle this, it was computed by a greedy heuristic: first, clusters from L_1 are associated with the clusters of L_2 with which they have the largest overlap. Conflicts are then resolved one by one: if two clusters are assigned to the same cluster, the one which has smaller overlap with the cluster is assigned to the cluster with which it has the next largest overlap. The process is iterated until there are no conflicting assignments are present. This way of conflict resolution guarantees the convergence of this process. For small overlap matrices where exact enumeration can be performed the results were checked to be identical. Even if from time to time the heuristic does not find the optimal solution, our results won't be affected, since we are interested in the statistical properties of the similarity.

Next, we use the optimal association to define concepts of stability for individual patterns and clusters. For a pattern i , two labelings L_1, L_2 , and an optimal association σ , define the *pattern-wise agreement* between L_1 and L_2 :

$$\delta_\sigma(i) = \begin{cases} 1 & \sigma(L_1(i)) = L_2(i), \\ 0 & \text{otherwise.} \end{cases} \quad (12)$$

Thus $\delta_\sigma(i) = 1$ iff pattern i is assigned to the same cluster in the two partitions relative to the association σ . We note that $s(L_1, L_2)$ can be equivalently expressed as $\frac{1}{n} \sum_i \delta_\sigma(i)$.

Now we define *pattern-wise stability* as the fraction of subsampled partitions where the subsampled labeling of pattern i agrees with that of the reference labeling, by averaging the pattern-wise agreement, equation (12):

$$n(i) = \frac{1}{N_i} \sum_{\text{subsamples}} \delta_\sigma(i), \quad (13)$$

where N_i is the number of sub-samples in which pattern i appears. The pattern-wise stability can indicate problem patterns that do not cluster well. Cluster stability is the average of the pattern-wise stability:

$$c(j) = \frac{1}{|L_1 = j|} \sum_{i \in (L_1=j)} n(i). \quad (14)$$

We note that cluster stability should not be interpreted as stability per se: suppose that a stable cluster splits into two clusters in an unstable way, and one cluster is larger than the other. In subsamples of the data, clusters will tend to be associated with the larger sub-cluster, with the result that the large cluster will have a higher cluster stability. Thus the stability of the smaller cluster is the one that reflects the instability of this split. This can be seen in Figure 7. Therefore we define the stability of a reference clustering into k clusters as:

$$S_k = \min_j c(j). \quad (15)$$

In computing S_k we ignore singletons or very small clusters. A dendrogram with stability measurements will be called a *stability annotated dendrogram* (see Figure 7).

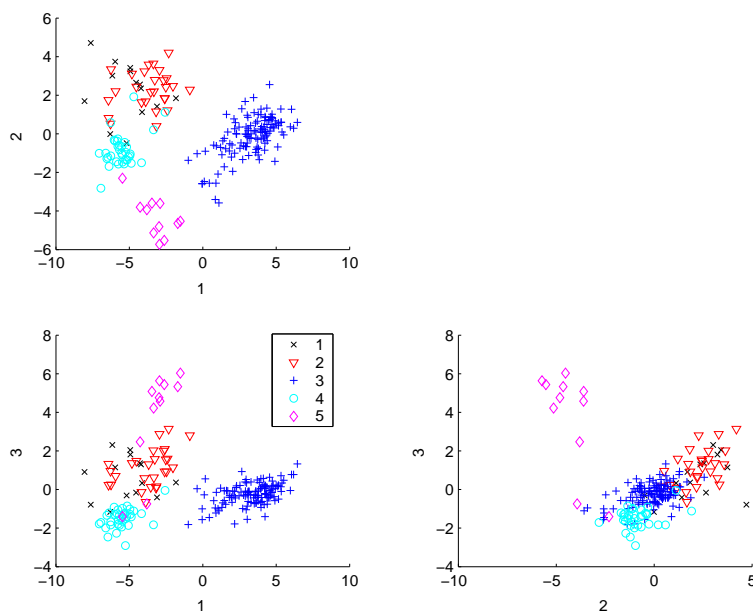


Figure 5: A scatter plot of the first three centered PCs of the yeast data. The symbols in the figure legend correspond to the 5 functional classes.

4 Experiments on gene expression data

In this section we use the yeast DNA microarray data of Eisen *et al.* (**1**) as a case study. Functional annotations were used to choose the 5 functional classes that were most learnable by SVMs (**32**), and noted by Eisen *et al.* to cluster well (**1**). We looked at the genes that belong uniquely to these 5 functional classes. This gave a dataset with 208 genes and 79 variables (experiments) in the following classes:

- (1) Tricarboxylic acid cycle (TCA) (14 genes)
- (2) Respiration (27 genes)
- (3) Cytoplasmatic ribosomal proteins (121 genes)
- (4) proteasomes (35 genes)
- (5) Histones (11 genes).

4.1 Clustering centered PCA variables

A scatter plot of the first three centered PCs is shown in Figure 5. Clear cluster structure that corresponds to the functional classes is apparent in the plot. Classes 1 and 2 however, are overlapping. For comparison we show a scatter plot of the next three PCs (Figure 6), where visual inspection shows no structure. This supports the premise that cluster structure should be apparent in the leading principal components. To further support this, we analyze results of clustering the data in the original variables, and in a few leading principal components. We use the average linkage hierarchical clustering algorithm (**30**) with a Euclidean distance.

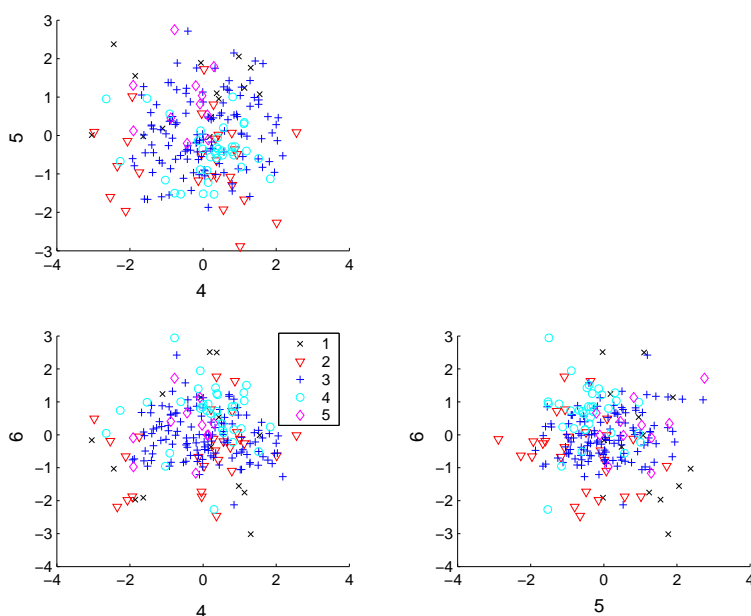


Figure 6: A scatter plot of PCs 4-6 for the yeast data.

When using the first three centered PCs, the functional classes are recovered well by clustering with the exception of classes 1 and 2 (TCA and respiration) that cannot be distinguished. The high similarity between the expression patterns of these two classes were also noted by Brown *et al.* (32). This is seen in the confusion matrix between the functional class labels and the clustering labels for $k = 4$:

		MIPS classification				
classes		1	2	3	4	5
clustering	1	11	27	0	2	0
	2	0	0	121	0	0
	3	2	0	0	33	3
	4	0	0	0	0	8

As already pointed out, when we say that we partition the data into 4 clusters we mean four non-singleton clusters (or more generally, clusters larger than some threshold). This allows us to ignore outliers, making comparisons of partitions into k clusters more meaningful, since an outlier will not appear in all the subsamples clustered. And indeed, in the dendrogram Figure 7 for $k = 4$, we find one singleton cluster for a total of 5 clusters.

The choice of $k = 4$ is justified by the stability method: The top dendrogram in Figure 7 shows a stability annotated dendrogram for the PCA data. The numbers at each node indicate the cluster stability of the corresponding cluster averaged over all the levels at which the cluster appears. All the functional classes appear in clusters with cluster stability above 0.96. The ribosomal cluster then splits into two clusters, one of them having stability value of 0.62, indicating that this split is unstable; the stability annotated dendrogram (Figure 7)

and the plot of the minimum cluster stability, S_k (Figure 9) show that the functional classes correspond to the stable clusters in the data.

When using all the variables (or equivalently, all PCs), the functional classes are recovered at $k = 5$, but not as well:

		MIPS classification				
		1	2	3	4	5
clustering	1	6	22	0	2	0
	2	3	0	0	0	0
	3	0	0	121	0	0
	4	4	5	0	33	2
	5	0	0	0	0	9

The same confusion matrix was observed when using 20 leading PCs or more. However, clustering into $k = 5$ clusters is not justified by the stability criterion: the split into the proteasome and TCA/respiration clusters is highly unstable (see bold numbering in the bottom stability annotated dendrogram in Figure 7). Only a partition into 3 clusters is stable.

Inspecting the dendrograms also reveals an important property of clustering of the leading PCs: the cluster structure is more apparent in the PCA dendrogram, using $q = 3$ components. Using all the variables, the distances between nearest neighbors and the distances between clusters are comparable, whereas for $q = 3$ nearest neighbor distances are very small compared to the distances between clusters, making the clusters more well defined, and consequently, more stable.

Using the comparison with the “true” labels, it was clear that the 3-PC data provided more detailed stable structure (3 vs. 4 clusters). “True” labels are not always available, so we would like a method for telling which of two partitions is more “refined”. We define a *refinement score*, also defined using the confusion matrix:

$$r(L_1, L_2) = \frac{1}{n} \sum_i \max_j M_{ij}. \quad (16)$$

The motivation for this score is illustrated with the help of Figure 8. Both blue clusters have a big overlap with the large red clusters and $r(blue, red) = \frac{1}{16}(7 + 7)$, that is close to 1, in agreement with our intuition that the blue clustering is basically a division of the big red cluster into two clusters. On the other hand $r(red, blue) = \frac{1}{16}(2 + 7)$, with a much lower score. It is straightforward to verify that $1/2 \leq r(L_1, L_2) \leq 1$, and thus $r(red, blue)$ is close to its lower bound. To make the relationship with the previous score more clear, we note that it can be defined as $s(L_1, L_2)$, where the association σ is not constrained to be one-to-one, so that each cluster is associated with the cluster with which it has the maximum overlap. If L_1 is obtained by splitting one of the clusters of L_2 , then $r(L_1, L_2) = 1$, and $r(L_2, L_1) = 1 - \text{fraction of patterns in smaller cluster of the two clusters that have split}$. The refinement score is interesting when it comes to comparing clusterings obtained from different algorithms, different subsets of features or different dendrograms obtained

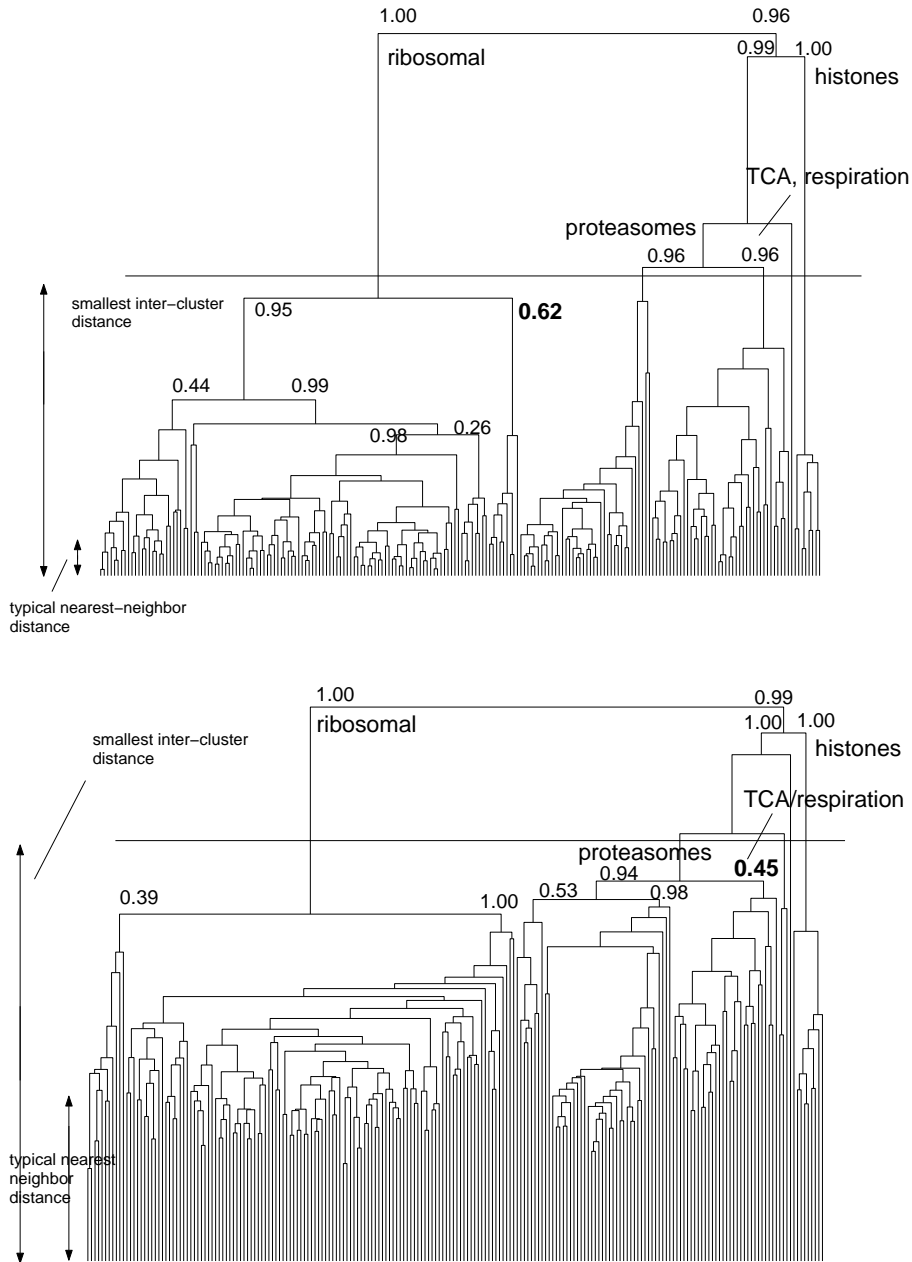


Figure 7: Stability annotated dendrograms for the yeast data. Numbers represent the cluster stability of a node. Cluster stability is averaged over all the levels in the hierarchy in which a cluster appears. The horizontal line represents the cutoff suggested by cluster stability; in boldface find the stability of the corresponding unstable split. Top: Data composed of three leading centered PCA variables. The vertical axis gives inter-cluster distances. The nodes that correspond to the functional classes are indicated. Bottom: Clustering of all 79 centered variables.

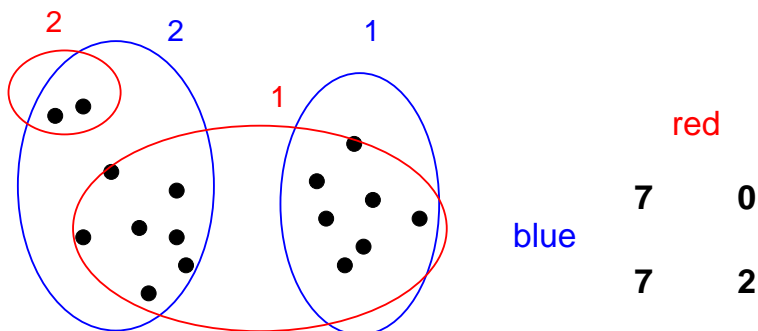


Figure 8: Two clusterings of a data set, represented by red and blue, with the associated confusion matrix.

with any kind of parameter change. Given two stable partitions L_1 and L_2 , one can then determine which of the two partitions is more refined according to which of $r(L_1, L_2)$ or $r(L_2, L_1)$ is larger. Merely counting the number of clusters to estimate refinement can be misleading since given two partitions with an identical number of clusters, one may be more refined than the other (as exemplified in Figure 8). It is even possible that a partition with a smaller number of clusters will be more refined than a partition with a larger number of clusters.

4.2 Stability-based choice of the number of PCs

The results of the previous subsection indicated that three principal components gave a more stable clustering than that obtained using all the variables. Next we will determine the best number of principal components. In the choice of principal components we will restrict ourselves to choosing the number of leading components, rather than choosing the best components, not necessarily by order of variance. We still consider centered-PCA data.

The stability of the clustering as measured by the minimum cluster stability, S_k , is plotted for a varying number of principal components in Figure 9. Partitions into up to 3 clusters were stable regardless of the number of principal components, as evidenced by S_k being close to 1. Partitions into 4 clusters were most stable for 3 or 4 PCs, and slightly less so for 7 PCs, with S_k still close to 1. For a higher number of PCs the stability of 4 clusters becomes lower (between 0.4 and 0.8). Moreover, the stable structure observed in 3-7 components at $k = 4$, is only observed at $k = 5$ for a higher number of PCs (and is unstable). The cluster structure is less stable with two principal components than in 3-7. The scatter plot of the principal components shows that the third PC still contains relevant structure; this is in agreement with the instability for 2 components, which are not sufficient.

To conclude, a small number of leading principal components produced clustering solutions that were more stable, i.e. significant, and also agreed better with the known classification.

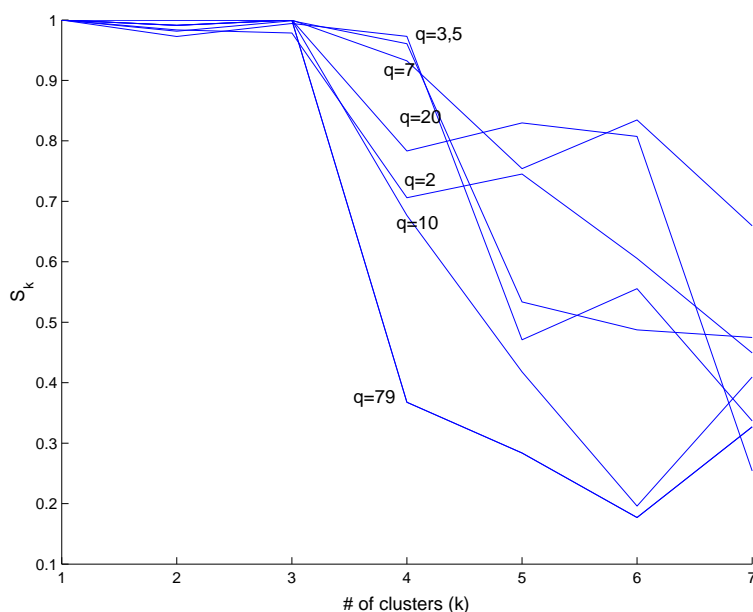


Figure 9: For each k , the average of the minimum cluster stability, S_k (cf. equation 15), is estimated for a varying number of principal components q .

Our observation on the power of PCA to produce a stable classification was seen to hold on datasets in other domains as well. Typically, when the number of clusters was higher, more principal components were required to capture the cluster structure. Other clustering algorithms that use the Euclidean distance were also seen to benefit from a preprocessing using PCA (33).

4.3 Clustering standardized PCs

In this subsection we compare clustering results on centered and standardized PCs. We limit ourselves to the comparison of $q = 3$ and $q = 79$ PCs. A scatter plot of the first three standardized PCs is shown (Figure 10). These show less cluster structure than in the centered PCs (Figure 5). The stability validation tool indicates the existence of 4 clusters, with the following confusion matrix:

		MIPS classification				
		1	2	3	4	5
clustering	1	9	27	0	35	2
	2	5	0	0	0	0
	3	0	0	121	0	0
	4	0	0	0	0	9

In this case classes 1, 2, and 4 cannot be distinguished by clustering. A similar confusion matrix is obtained when clustering the standardized variables without applying PCA.

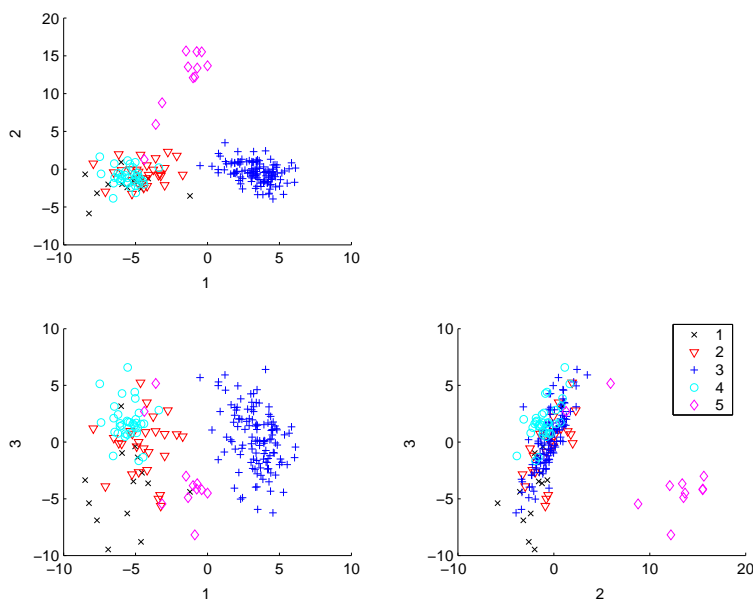


Figure 10: Scatter plot of the first three standardized PCs.

We conclude that the degradation in the recovery of the functional classes should be attributed to normalization, rather than to the use of PCA. Other authors have also found that standardization can deteriorate the quality of clustering (29, 34).

Let $L_{centered}$ and $L_{standardized}$ be the stable partitions into 4 and 3 clusters respectively of the centered and standardized PCA data using 3 PCs. The refinement scores for these partitions were found to be: $r(L_{centered}, L_{standardized}) = 0.8125$ and $r(L_{standardized}, L_{centered}) = 0.995$, showing that the centered clustering contains more detailed stable structure. Thus the known cluster labels are not necessary to arrive at this conclusion.

4.4 Clustering using the Pearson correlation

Here we report results using the Pearson correlation similarity measure (the gene-gene correlation matrix). Clustering using the Pearson correlation as a similarity measure recovered the functional classes in a stable way without the use of PCA (see a stability annotated dendrogram in Figure 11). It seems that the Pearson correlation is less susceptible to noise than the Euclidean distance: the Pearson correlation clustering is more stable than the Euclidean clustering that uses all the variables; also compare the nearest neighbor distances that are much larger in the Euclidean case. This may be the result of the Pearson correlation similarity being a sum of terms that are either positive or negative, resulting in some of the noise canceling out; in the case of the Euclidean distance, no cancellation can occur since all terms are positive. The Pearson correlation did not benefit from the use of PCA.

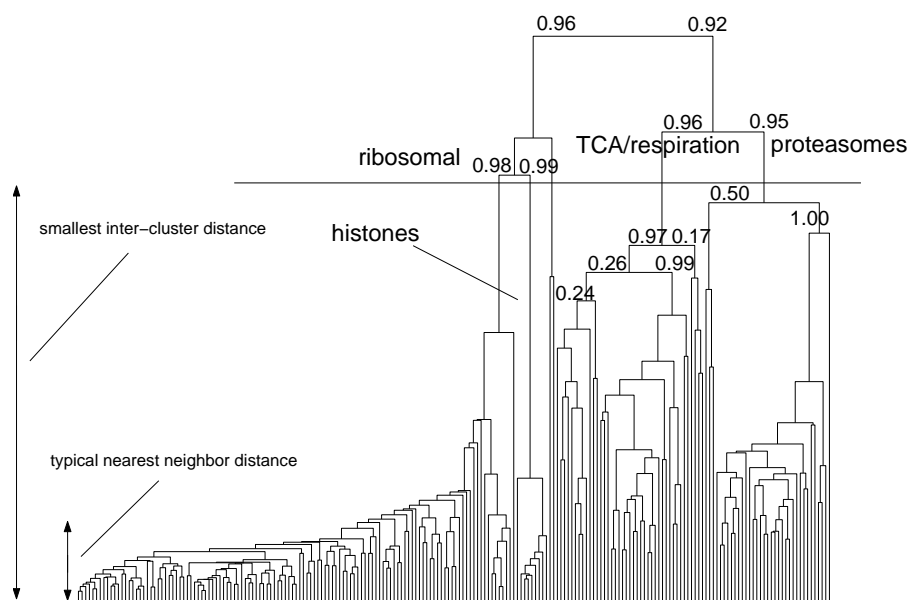


Figure 11: Stability annotated dendrogram for the yeast data with the Pearson correlation similarity measure. The vertical axis is 1-correlation. The functional classes are stable; their sub-clusters are unstable.

5 Other methods for choosing PCs

In a recent paper it was claimed that PCA does not generally improve the quality of clustering of gene expression data (19). We suspect that was a result of the use of standardization as a normalization, that in our analysis reduced the quality of the clustering, rather than the use of PCA. The authors' criterion for choosing components was external: comparison with known labels, and thus cannot be used in general. We use a criterion that does not require external validation. However, running it is relatively time consuming since it requires running the clustering algorithm a large number of times to estimate the stability (a value of 100 was used here). Therefore we restricted the feature selection to the choice of the number of leading principal components.

When using PCA to approximate a data matrix, the fraction of the total variance in the leading PCs is used as a criterion for choosing how many of them to use (20). In the yeast data analyzed in this chapter the first three PCs contain only 31% of the total variance in the data. Yet, 3-5 PCs out of 79 provide the most stable clustering that also agrees best with the known labels. Thus, the total variance is a poor way of choosing the number of PCs when the objective is clustering. On the contrary, we do not wish to reconstruct the matrix, only the essential features that are responsible for cluster structure.

6 Conclusions

In this chapter we propose a novel methodology for evaluating the merit of clustering solutions. It is based on the premise that well defined cluster structure should be stable under perturbation of the data. We introduced notions of stability at the level of a partition, cluster, and pattern, that allow us, in particular, to generate stability-annotated dendrograms. Using stability as a figure of merit, along with a measure of cluster refinement, allows us to compare stable solutions run using different normalizations, similarity measures, clustering algorithms, or input features.

We demonstrated this methodology on the task of choosing the number of principal components and normalization to be used to cluster a yeast gene expression dataset. It was shown that PCA improves the extraction of cluster structure, with respect to stability, refinement, and coincidence with known “ground truth” labels. This means that, in this data set, the cluster structure is present in directions of largest variance (or best data reconstruction). Beyond this simple example, our methodology can be used for many “model selection” problems in clustering, including selecting the clustering algorithm itself, the parameters of the algorithm, the variables and patterns to cluster, the similarity, normalization or other preprocessing, and the number of clusters. It allows us not only to compare clustering solutions, but also to detect presence or absence of structure in data. In our analysis we used a hierarchical clustering algorithm, but any other clustering algorithm can be used. The versatility and universality of our methodology, combined with its simplicity, may appeal to practitioners in Bioinformatics and other fields. Further work include laying the theoretical foundations of the methodology and further testing of the ideas in other domains.

Acknowledgements

The authors thank André Elisseeff and Bronwyn Eisenberg for their helpful comments on this manuscript.

References

1. Eisen, M., Spellman, P., Brown, P. and Botstein, D. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci USA*, **95**, 14863–14868.
2. Quackenbush, J. (2001) Computational analysis of microarray data. *Nature Reviews Genetics*, **2**, 418–427.
3. Ross, D., Scherf, U., Eisen, M., Perou, C., Rees, C., Spellman, P., Iyer, V., Jeffrey, S., de Rijn, M. V., Waltham, M., Pergamenschikov, A., Lee, J., Lashkari, D., Shalon, D., Myers, T., Weinstein, J., Botstein, D. and Brown, P. (2000) Systematic variation in gene expression patterns in human cancer cell lines. *Nature genetics*, **24**, 227–235.
4. D’haeseleer, P., Liang, S. and Somogyi, R. (2000) Genetic network inference: from co-expression clustering to reverse engineering. *Bioinformatics*, **16**, 707–726.

5. Alon, U., Barkai, N., Notterman, D., Gish, G., Ybarra, S., Mack, D. and Levine, A. J. (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *PNAS*, **96**, 6745–6750.
6. Alizadeh, A.A. *et al.* (2000) Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, **403**, 503–511.
7. Scherf, U., Ross, D. T., Waltham, M., Smith, L. H., Lee, J. K., Tanabe, L., Kohn, K. W., Reinhold, W. C., Myers, T. G., Andrews, D. T., Scudiero, D. A., Eisen, M. B., Sausville, E. A., Pommier, Y., Botstein, D., Brown, P. O. and Weinstein, J. N. (2000) A gene expression database for the molecular pharmacology of cancer. *nature genetics*, **24**, 236–244.
8. Getz, G., Levine, E. and Domany, E. (2000) Coupled two-way clustering analysis of gene microarray data. *Proc. Natl. Acad. Sci USA*, **94**, 12079–12084.
9. Shamir, R. and Sharan, R. (2001) Algorithmic approaches to clustering gene expression data. in Jiang, T., Smith, T., Xu, Y. and Zhang, M., eds., *Current Topics in Computational Biology*, MIT Press.
10. Milligan, G. (1996) Clustering validation: results and implications for applied analysis, in *Clustering and Classification* (Arabie, P., Hubert, L. and Soete, G. D., eds.), World Scientific, River Edge, NJ. 341–374.
11. Tibshirani, R., Walther, G. and Hastie, T. (2001) Estimating the number of clusters in a dataset via the gap statistic. *Journal of the Royal Statistical Society B*, **63**, 411–423.
12. Ben-Hur, A., Elisseeff, A. and Guyon, I. (2002) A stability based method for discovering structure in clustered data, in *Pacific Symposium on Biocomputing* (Altman, R., Dunker, A., Hunter, L., Lauderdale, K. and Klein, T., eds.), World Scientific, 6–17.
13. Levine, E. and Domany, E. (2001) Resampling method for unsupervised estimation of cluster validity. *Neural Computation*, **13**, 2573–2593.
14. Fridlyand, J. (2001) *Resampling methods for variable selection and classification: applications to genomics*. Ph.D. thesis, UC Berkeley.
15. Fowlkes, E. and Mallows, C. (1983) A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, **78**, 553–584.
16. Bittner, M., Meltzer, P., Chen, Y., Jiang, Y., Seftor, E., Hendrix, M., Radmacher, M., Simon, R., Yakhini, Z., Ben-Dor, A., Dougherty, E., Wang, E., Marincola, F., Gooden, C., Lueders, J., Glatfelter, A., Pollock, P., Gillanders, E., Leja, D., Dietrich, K., Berens, C., Alberts, D., Sondak, V., Hayward, N. and Trent, J. (2000) Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature*, **406**, 536–540.

17. Yeung, K., Haynor, D. and Ruzzo, W. (2001) Validating clustering for gene expression data. *Bioinformatics*, **17**, 309–318.
18. Milligan, G. (1980) An examination of the effect of six types of error perturbation on fifteen clustering algorithms. *Psychometrika*, **45**, 325–342.
19. Yeung, K. and Ruzzo, W. (2001) An empirical study of principal component analysis for clustering gene expression data. *Bioinformatics*, **17**, 763–774.
20. Jackson, J. (1991) *A user's guide to principal components*. John Wiley & Sons.
21. Jolliffe, I. (1986) *Principal component analysis*. Springer-Verlag.
22. Jolliffe, I. (1972) Discarding variables in principal component analysis I: Artificial data. *Applied Statistics*, **21**, 160–173.
23. Jolliffe, I. (1972) Discarding variables in principal component analysis II: Real data. *Applied Statistics*, **21**, 160–173.
24. Hastie, T., Tibshirani, R., Eisen, M., Alizadeh, A., Levy, R., Staudt, L., Chan, W., Botstein, D. and Brown, P. (2000) “gene shaving” as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology*, **1**, 1–21.
25. Alter, O., Brown, P. and Botstein, D. (2000) Singular value decomposition for genome-wide expression data processing and modeling. *Proc. Natl. Acad. Sci USA*, **97**, 10101–10106.
26. Raychaudhuri, S., Stuart, J. and Altman, R. (2000) Principal components analysis to summarize microarray experiments: Application to sporulation time series, in *Pacific Symposium on Biocomputing* 5, 452–463.
27. Hilsenbeck, S., Friedrichs, W., Schiff, R., O'Connell, P., Hansen, R., Osborne, C. and Fuqua, S. (1999) Statistical analysis of array expression data as applied to the problem of tamoxifen resistance. *Journal of the National Cancer Institute*, **91**, 453–459.
28. Wen, X., Fuhrman, S., Michaels, G., Carr, D., Smith, S., Barker, J. and Somogyi, R. (1998) Large-scale temporal gene expression mapping of central nervous system development. *Proc. Natl. Acad. Sci USA*, **95**, 334–339.
29. Milligan, G. and Cooper, M. (1988) A study of variable standardization. *Journal of classification*, **5**, 181–204.
30. Jain, A. and Dubes, R. (1988) *Algorithms for clustering data*. Prentice Hall, Englewood Cliffs, NJ.
31. Kaufman, L. and Rousseeuw, P. (1990) *Finding groups in data*. Wiley Interscience, John Wiley & Sons.

32. Brown, M., Grundy, W., Lin, D., Cristianini, N., Sugnet, C., Ares, M. and Hausler, D. (2000) Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc. Natl. Acad. Sci. USA*, **97**, 262–267. URL <http://www.cse.ucsc.edu/research/compbio/genex>.
33. Ben-Hur, A., Horn, D., Siegelmann, H. and Vapnik, V. (2001) Support vector clustering. *Journal of Machine Learning Research*, **2**, 125–137.
34. Anderberg, M. (1983) *Cluster analysis for applications*. Academic Press, New York.