Classification and Clustering Methods for Gene Expression

Kim-Anh Do, Ph.D.
Department of Biostatistics
The University of Texas M.D. Anderson Cancer Center

Advanced Statistical Methods for the Analysis of Gene Expression and Proteomics

GSBS010103 and STAT675

Part of these lecture notes are adapted from Hilsenbeck's and Goldstein's lecture notes

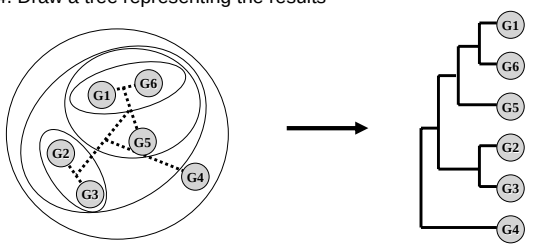
Tumor Classification Using Gene Expression Data

Three main types of statistical problems associated with the microarray data:

- **Identification of "marker" genes that characterize the different tumor classes (feature or variable selection).**
- **Identification of new/unknown tumor classes using gene expression profiles (unsupervised learning – clustering)**
- **Classification of sample into known classes (supervised learning – classification)**

Hierarchical Clustering

1. Calculate the distance between all genes. Find the smallest distance. If several pairs share the same similarity, use a predetermined rule to decide between alternatives
2. Fuse the two selected clusters to produce a new cluster that now contains at least two objects. Calculate the distance between the new cluster and all other clusters
3. Repeat steps 1 and 2 until only a single cluster remains
4. Draw a tree representing the results



Agglomerative Linkage Methods

Linkage methods are rules or metrics that return a value that can be used to determine which elements (clusters) should be linked.

Three linkage methods that are commonly used are:

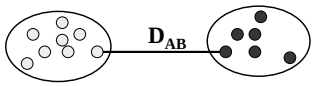
- Single Linkage
- Average Linkage
- Complete Linkage

Single Linkage

Cluster-to-cluster distance is defined as the *minimum distance* between members of one cluster and members of the another cluster. Single linkage tends to create 'elongated' clusters with individual genes chained onto clusters.

$$D_{AB} = \min (d(u_i, v_j))$$

where $u \in A$ and $v \in B$
for all $i = 1$ to N_A and $j = 1$ to N_B



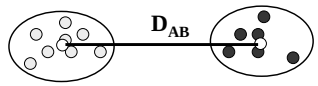
5

Average Linkage

Cluster-to-cluster distance is defined as the *average distance* between all members of one cluster and all members of another cluster. Average linkage has a slight tendency to produce clusters of similar variance.

$$D_{AB} = 1/(N_A N_B) \Sigma \Sigma (d(u_i, v_j))$$

where $u \in A$ and $v \in B$
for all $i = 1$ to N_A and $j = 1$ to N_B



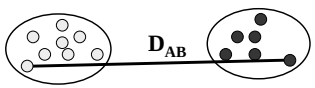
6

Complete Linkage

Cluster-to-cluster distance is defined as the *maximum distance* between members of one cluster and members of the another cluster. Complete linkage tends to create clusters of similar size and variability.

$$D_{AB} = \max (d(u_i, v_j))$$

where $u \in A$ and $v \in B$
for all $i = 1$ to N_A and $j = 1$ to N_B

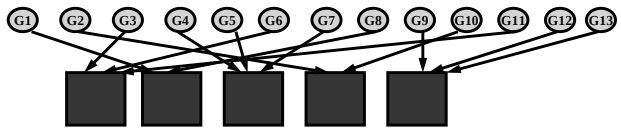


7

K-Means/Medians Clustering

1. Specify number of clusters, e.g., 5.

2. Randomly assign genes to clusters.

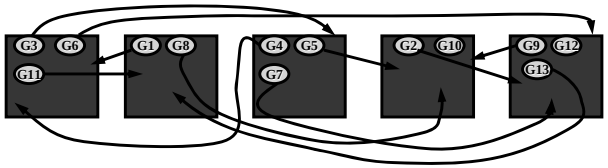


8

K-Means/Medians Clustering

K-Means is most useful when the user has an *a priori* specified number of clusters for the genes

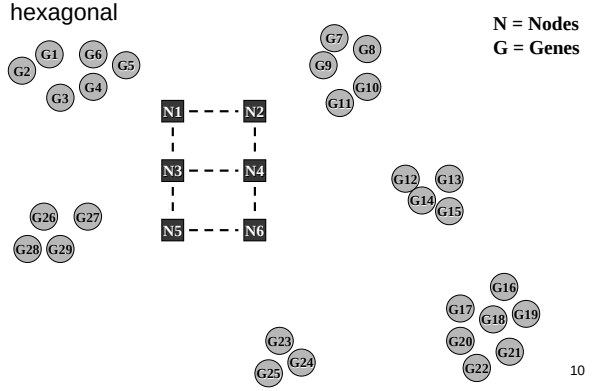
1. Specify number of clusters
2. Randomly assign genes to clusters
3. Calculate mean/median expression profile of each cluster
4. Shuffle genes among clusters such that each gene is now in the cluster whose mean expression profile is the closest to that gene's expression profile



5. Repeat steps 3 and 4 until genes cannot be shuffled around any more, OR a user-specified number of iterations has been reached.

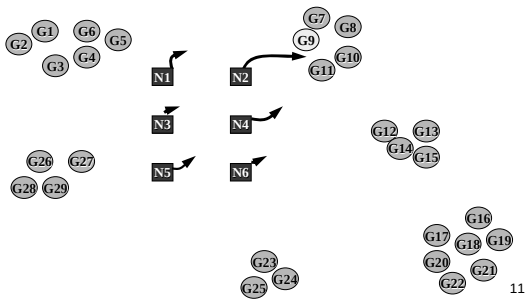
Self-organizing maps (SOMs)

1. Specify the number of nodes (clusters) desired, and also specify a 2-D geometry for the nodes, e.g., rectangular or hexagonal



SOMs

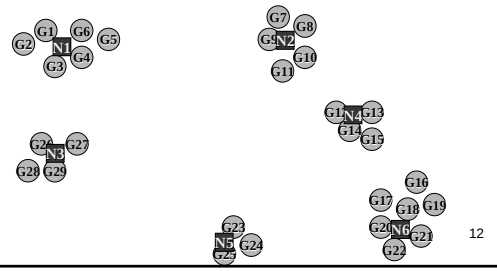
2. Choose a random gene, say, G9
3. Move the nodes in the direction of G9. The node closest to G9 (N2) is moved the most, and the other nodes are moved by smaller varying amounts. The farther away the node is from N2, the less it is moved.



SOMs

4. Repeat Steps 2 and 3 several thousand times; with each iteration, the amount that the nodes are allowed to move is decreased.

5. Finally, each node will "nestle" among a cluster of genes, and a gene will be considered to be in the cluster if its distance to the node in that cluster is less than its distance to any other node.



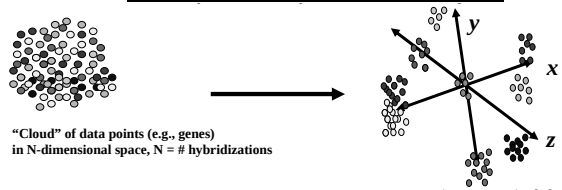
Principal Components Analysis

PCA is a dimension reduction technique

- Suppose we have measurements for each gene on multiple experiments
- Suppose some of the experiments are correlated.
- PCA will ignore the redundant experiments, and will take a weighted average of some of the experiments, thus possibly making the trends in the data more interpretable.
- The components can be thought of as axes in n-dimensional space, where n is the number of components. Each axis represents a different trend in the data.

13

Principal Components Analysis



"Cloud" of data points (e.g., genes) in N-dimensional space, N = # hybridizations

Data points summarized along 3 principal component axes.

Example:

x-axis could mean a continuum from over-to under-expression

y-axis could mean that "blue" genes are over-expressed in first five expts and under expressed in the remaining expts, while "brown" genes are under-expressed in the first five expts, and over-expressed in the remaining expts.

z-axis might represent different cyclic patterns, e.g., "red" genes might be over-expressed in odd-numbered expts and under-expressed in even-numbered ones, whereas the opposite is true for "purple" genes.

Interpretation of components is somewhat subjective.

14

Other developments

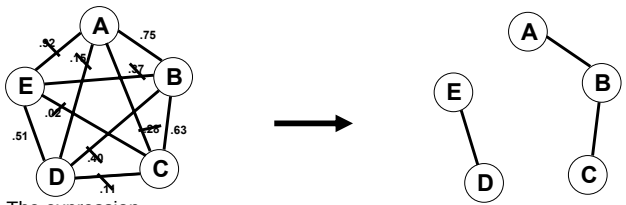
1. Supervised principal components
(Bair, Hastie, Paul, Tibshirani, JASA 2005)

Software:
http://www-stat.stanford.edu/~tibs/superpc/

2. Graphical networks, Bayesian networks, relevance networks,…

15

Relevance Networks



The expression pattern of each gene compared to that of every other gene

$T_{min} = 0.50$

$T_{max} = 0.90$

The remaining relationships between genes define the subnets

The ability of each gene to predict the expression of each other gene is assigned a correlation coefficient

Correlation coefficients outside the boundaries defined by the minimum and maximum thresholds are eliminated.

16

Methods

- **Discriminant Analysis**
  - **Linear and quadratic discriminant analysis**
  - **Diagonal LDA, QDA – ignores correlations between genes, reducing number of parameters to estimate**
  - **Weighted voting**
  - **Compound covariate predictor**
  - **Linear support vector machines (SVM)**
  - **Nearest (shrunken) centroids (PAM)**
- **Classification and regression trees**
- **K Nearest neighbors**
- **Others: Nonlinear SVM, Neural networks, etc**

Dudoit, S. and J. Fridlyand (2003). Classification in microarray experiments. Statistical Analysis of Gene Expression Microarray Data. T. P. Speed. New York, Chapman & Hall/CRC: 222.

Simon, R., L. M. McShane, et al. (2003). Class Prediction. Design and Analysis of DNA Microarray Investigations. New York, Springer: 199.

Classification

Y	Normal sample1	Normal sample2	Normal sample3	Cancer sample4	Cancer sample5	…	unknown =Y_new New sample
1	0.46	0.30	0.80	1.51	0.90	...	0.34
2	-0.10	0.49	0.24	0.06	0.46	...	0.43
3	0.15	0.74	0.04	0.10	0.20	...	-0.23
4	-0.45	-1.03	-0.79	-0.56	-0.32	...	-0.91
5	-0.06	1.06	1.35	1.09	-1.09	...	1.23
	X						X_new

- **Each object (e.g. arrays or columns)associated with a class label (or response) Y $\in$ {1, 2, …, K} and a feature vector (vector of predictor variables) of G measurements: X = (X_1, …, X_G)**
- **Aim: predict Y_new from** X_new**.**

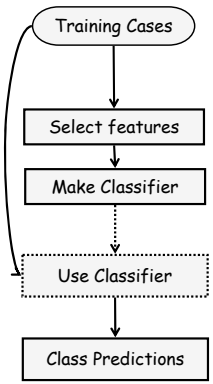
Classification and Prediction: Supervised Learning

- **Start with cases in known classes**
  - **Select "features" that characterize the classes**
  - **Make a rule that let you use the values of the "features" to assign a class (minimize the "cost" of misclassification)**
- **Apply the rule**
- **Estimate the accuracy of classification (XX% CI) and/or test to compare to an existing classifier**
- **Classification and prediction is an old area of statistics (LDA, Fisher, 1936), but still active area of research**
- **Traditionally, there are more cases than features**
- **Expression arrays, usually 100X more features**

Selecting a classification method

- **Best classifier is the one that makes the fewest mistakes (or least costly mistakes)**
- **How should we estimate the accuracy of the classifier?**

Resubstitution Estimate of Accuracy

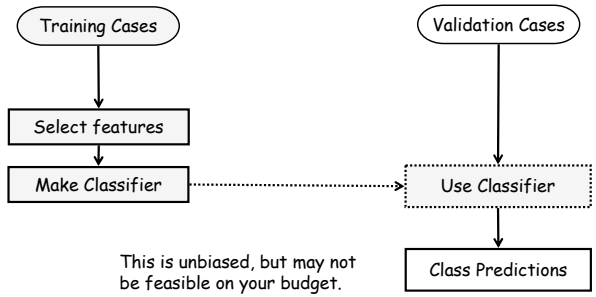


This is invariably biased, reporting a much more favorable accuracy then is true

Report the AMAZING result!!!

21

External Validation Estimate of Accuracy

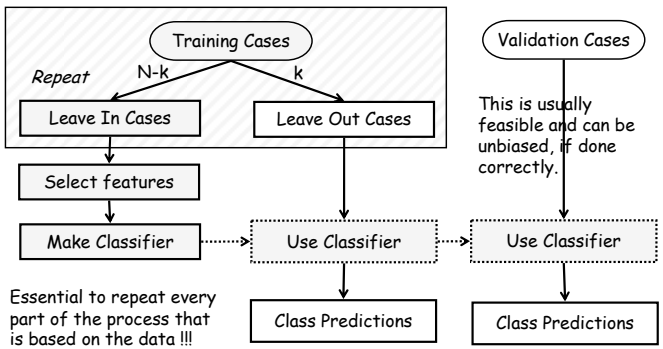


This is unbiased, but may not be feasible on your budget.

Report the less amazing but more generalizable result!!!
Inference - XX%CI excludes Null rate,
or explicit comparison to competing method

22

Cross-validation Estimate of Accuracy



This is usually feasible and can be unbiased, if done correctly.

Essential to repeat every part of the process that is based on the data !!!

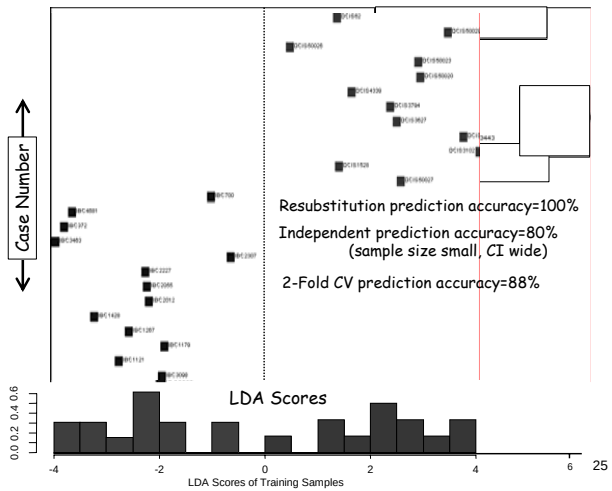
For Affymetrix data, can normalize left out cases against a fixed baseline, estimate expression using "probe sensitivities", then apply classifier.

23

Example of Accuracy Estimates

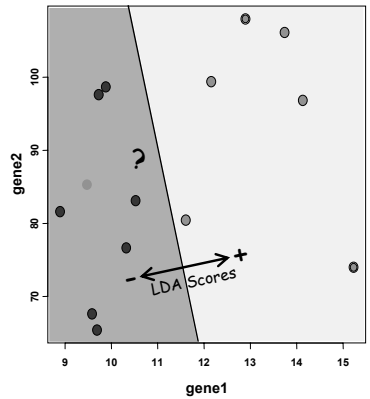
- **Select genes and create a classifier to distinguish between non-invasive dCIS and invasive breast cancer**
- **Sample:**
  - **HgU95av2, 12,558 genes**
  - **dCIS, n=25**
  - **IBC, n=25**
  - **Randomly select half in each group to be the training sample and half to be the test sample**
- **Train**
  - **Feature selection – use t-test to select 169 genes, α=0.001 (FDR~6%)**
  - **Classifier construction - Linear Discriminant Analysis**
- **Test – compute scores and predict class of test**

24



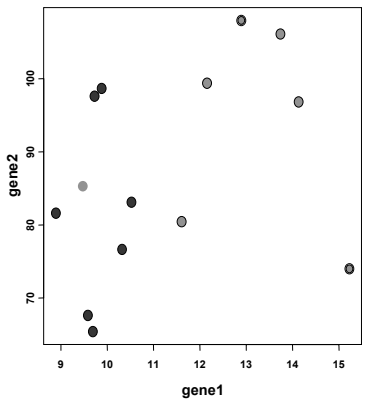
25

Discriminant Analysis



- **Discriminator is "linear" weighted combination of gene values**
- **Various distance metrics**
- **Many choices to make**
  - **Which features (genes) to use**
  - **Distance metric**
  - **Weights**
- **Choices must be made without reference to "leave out" cases**

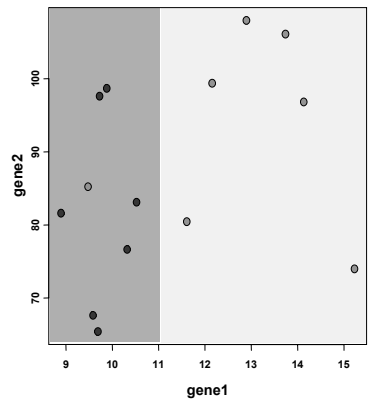
26

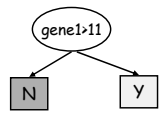


- Two groups of 7 cases each
- Two genes ($p<<0.001$)
- How can we use these two genes to construct a classifier to predict group?

27

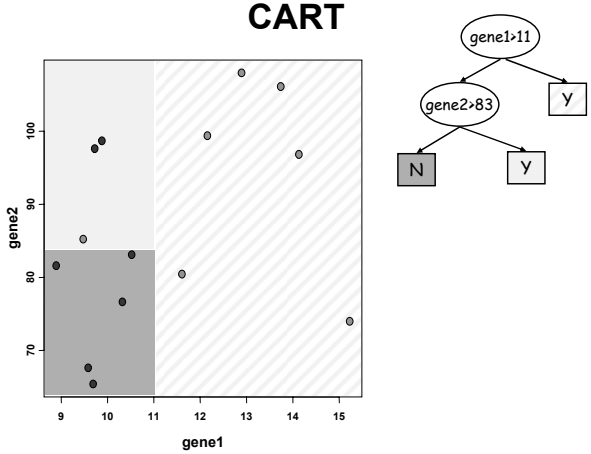
CART





28

CART



29

CART: Classification Tree

BINARY RECURSIVE PARTITIONING TREE

- **Binary**
  - **-- split parent node into two child nodes**
- **Recursive**
  - **-- each child node can be treated as parent node**
- **Partitioning**
  - **-- data set is partitioned into mutually exclusive subsets in each split**

-- L.Breiman, J.H. Friedman, R. Olshen, and C.J. Stone. Classification and regression trees. The Wadsworth statistics/probability series. Wadsworth International Group, 1984.

30

Classification Trees

- **Binary tree structured classifiers are constructed by repeated splits of subsets (nodes) of the measurement space X into two descendant subsets (starting with X itself)**
- **Each terminal subset is assigned a class label; the resulting partition of X corresponds to the classifier**
- **RPART in R or TREE in R**

31

Three Aspects of Tree Construction

- **Split Selection Rule**
- **Split-stopping Rule**
- **Class assignment Rule**
- **Different tree classifiers use different approaches to deal with these three issues, e.g. CART(Classification And Regression Trees)**

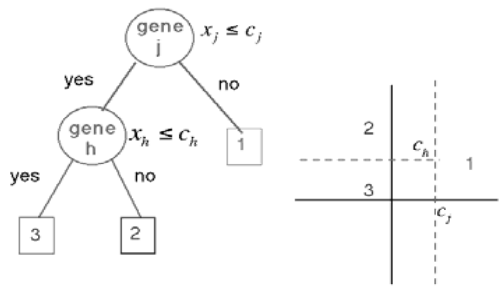
32

Three Rules (CART)

- Splitting: At each node, choose split maximizing decrease in impurity (e.g. Gini index, entropy, misclassification error).
- Split-stopping: Grow large tree, prune to obtain a sequence of subtrees, then use cross-validation to identify the subtree with lowest misclassification rate.
- Class assignment: For each terminal node, choose the class with the majority vote.

33

CART



Aggregating classifiers

- Breiman (1996, 1998) found that gains in accuracy could be obtained by aggregating predictors built from perturbed versions of the learning set; the multiple versions of the predictor are aggregated by weighted voting.
- Let $C(., L_b)$ denote the classifier built from the b-th perturbed learning set L_b, and let w_b denote the weight given to predictions made by this classifier. The predicted class for an observation x is given by

$$\text{argmax}_k \sum_b w_b I(C(x,L_b) = k)$$

-- L. Breiman. Bagging predictors. Machine Learning, 24:123-140, 1996.

-- L. Breiman. Out-of-bag eatimation. Technical report, Statistics Department, U.C. Berkeley, 1996

35

Aggregating Classifiers

- The key to improved accuracy is the possible instability of the prediction method, i.e., whether small changes in the learning set result in large changes in the predictor.
- Unstable predictors tend to benefit the most from aggregation.
  - Classification trees (e.g.CART) tend to be unstable.
  - Nearest neighbor classifier tend to be stable.

36

Bagging & Boosting

- **Two main methods for generating perturbed versions of the learning set.**
  - **Bagging.**
    -- *L. Breiman. Bagging predictors. Machine Learning, 24:123-140, 1996.*
  - **Boosting.**
    -- *Y.Freund and R.E.Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, 55:119-139, 1997.*

37

Bagging= Bootstrap aggregating

I. Nonparametric Bootstrap (BAG)

- **Nonparametric Bootstrap (standard bagging).**
  - **perturbed learning sets of the same size as the original learning set are formed by randomly selecting samples with replacement from the learning sets;**
- **Predictors are built for each perturbed dataset and aggregated by plurality voting plurality voting (w_b=1), i.e., the "winning" class is the one being predicted by the largest number of predictors.**

38

Bagging= Bootstrap aggregating

II. Parametric Bootstrap (MVN)

- **Parametric Bootstrap.**
  - **Perturbed learning sets are generated according to a mixture of multivariate normal (MVN) distributions.**
  - **The conditional densities for each class is a multivariate Gaussian (normal), i.e., $P(X|Y= k) \sim N(\mu_k, \Sigma_k)$, the sample mean vector and sample covariance matrix will be used to estimate the population mean vector and covariance matrix.**
  - **The class mixing probabilities are taken to be the class proportions in the actual learning set.**
  - **At least one observation be sampled from each class.**
- **Predictors are built for each perturbed dataset and aggregated by plurality voting (w_b=1).**

39

Bagging= Bootstrap aggregating

III. Convex pseudo-data (CPD)

- **Convex pseudo-data. One perturbed learning set are generated by repeating the following n times:**
  - **Select two samples (x,y) and (x', y') at random from the learning set L.**
  - **Select at random a number of v from the interval [0,d], 0<=d<=1, and let u=1-v.**
  - **The new sample is (x'', y'') where y''=y and x''=ux+vx'**
  - **Note that when d=0, CPD reduces to standard bagging.**
- **Predictors are built for each perturbed dataset and aggregated by plurality voting (w_b=1).**

40

Boosting

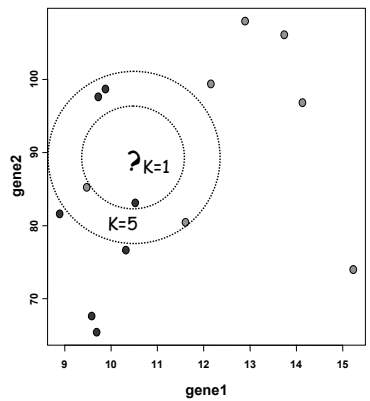
- **The perturbed learning sets are re-sampled adaptively so that the weights in the re-sampling are increased for those cases most often misclassified.**
- **The aggregation of predictors is done by weighted voting (w_b != 1).**

Boosting

- Learning set: L = ($\mathbf{X}_1$, Y_1), ..., ($\mathbf{X}_n$,Y_n)
- Re-sampling probabilities p={p_1,…, p_n}, initialized to be equal.
- The bth step of the boosting algorithm is:
  - Using the current re-sampling prob p, sample with replacement from L to get a perturbed learning set L_b.
  - Build a classifier C(., L_b) based on L_b.
  - Run the learning set L through the classifier C(., L_b) and let d_i=1 if the ith case is classified incorrectly and let d_i=0 otherwise.
  - Define $\varepsilon_b = \sum_i p_i d_i$ and $\beta_b = \frac{(1-\varepsilon_b)}{\varepsilon_b}$

    and update the re-sampling prob for the (b+1)st step by $p_i = \frac{p_i \beta_b^{d_i}}{\sum_j p_j \beta_b^{d_j}}$
- The weight for each classifier is $\varpi_b = \log(\beta_b)$

Nearest Neighbors



- **Majority or plurality voting**
- **Various distance metrics**
- **Often effective for small samples**
- **K is usually 1 or 3, up to 7**
- **Many choices to make**
  - **Features to use**
  - **Distance metric**
  - **K**
- **Choices must be made without reference to "leave out" cases**

Comparison of Methods

- **With modest sample sizes (<100), methods that DO NOT account for correlations among genes, interactions, or other structural complexity perform best**
- **Studies to date do not favor a single best classifier**
- **<u>Estimates of classifier accuracy should account for ALL "training decisions"</u>**

Dudoit, S., J. Fridlyand, et al. (2002). "Comparison of discrimination methods for the classification of tumors using gene expression data." J Am Statistical Assoc **97**(457): 77-87.

How bad could it be?

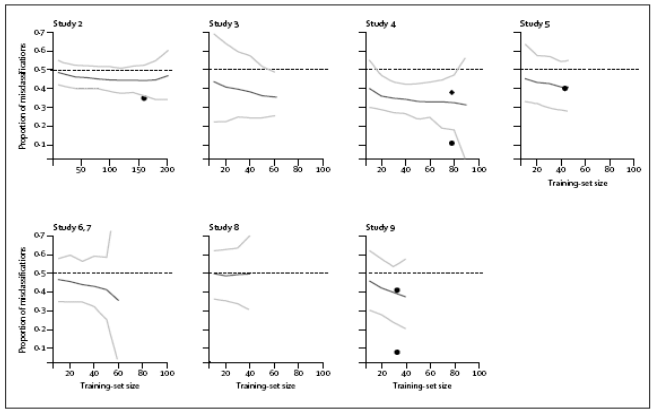


Figure 2: Proportion of misclassifications in validation sets as a function of corresponding training-set sizes in the seven studies[19]

S. Michiels, S. Koscielny, C. Hill, Lancet 365, 488-92 (Feb 5-11, 2005).

Comparison of classifiers

- Dudoit, Fridlyand, Speed (JASA, 2002)
- FLDA (Fisher Linear Discriminant Analysis)
- DLDA (Diagonal Linear Discriminant Analysis)
- DQDA (Diagonal Quantic Discriminant Analysis)
- NN (Nearest Neighbour)
- CART (Classification and Regression Tree)
- Bagging and boosting
  - Bagging (Non-parametric Bootstrap)
  - CPD (Convex Pseudo Data)
  - MVN (Parametric Bootstrap)
  - Boosting

-- *Dudoit, Fridlyand, Speed: "Comparison of discrimination methods for the classification of tumors using gene expression data", JASA, 2002*

46

Comparison study datasets

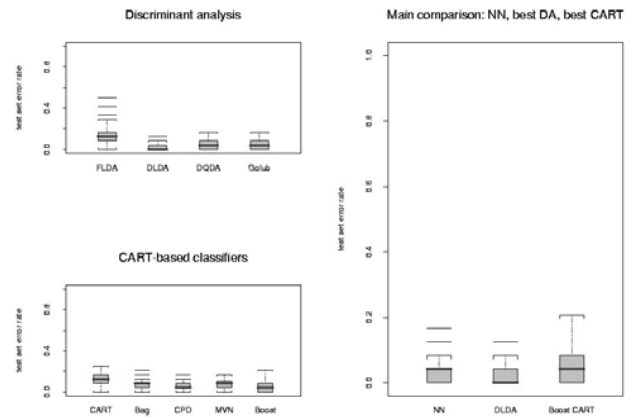
- **Leukemia – Golub et al. (1999)**
  - n = 72 samples, G = 3,571 genes
  - 3 classes (B-cell ALL, T-cell ALL, AML)
- **Lymphoma – Alizadeh et al. (2000)**
  - n = 81 samples, G = 4,682 genes
  - 3 classes (B-CLL, FL, DLBCL)
- **NCI 60 – Ross et al. (2000)**
  - N = 64 samples, p = 5,244 genes
  - 8 classes

47

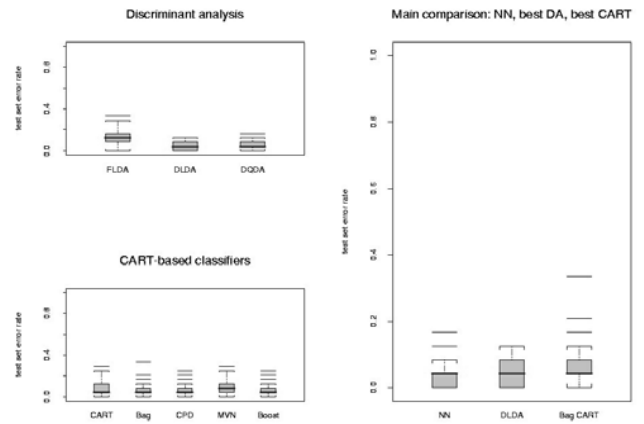
Procedure

- For each run (total 150 runs):
  - 2/3 of sample randomly selected as learning set (LS), rest 1/3 as testing set (TS).
  - The top p genes with the largest BSS/WSS are selected using the learning set.
    - p=50 for lymphoma dataset.
    - p=40 for leukemia dataset.
    - p=30 for NCI 60 dataset.
  - Predictors are constructed and error rated are obtained by applying the predictors to the testing set.

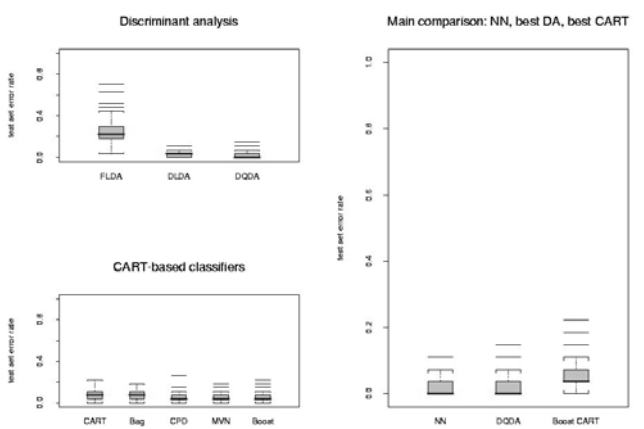
48



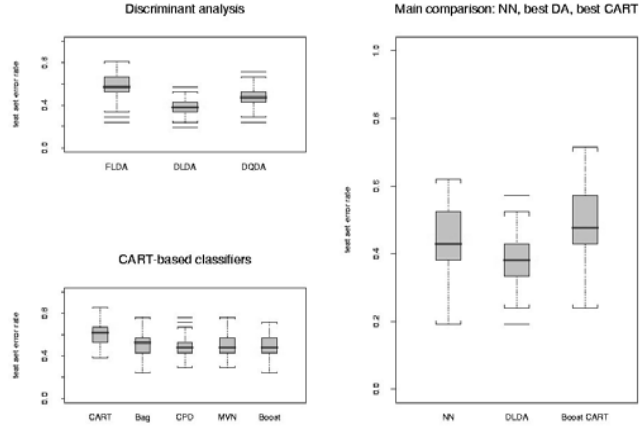
Leukemia data, 2 classes: Test set error rates;150 LS/TS runs 49



Leukemia data, 3 classes: Test set error rates;150 LS/TS runs 50



Lymphoma data, 3 classes: Test set error rates; N=150 LS/TS runs 51



NCI 60 data :Test set error rates;150 LS/TS runs 52

Results

- In the main comparison of Dudoit et al, NN and DLDA had the smallest error rates, FLDA had the highest
- For the lymphoma and leukemia datasets, increasing the number of genes to G=200 didn't greatly affect the performance of the various classifiers; there was an improvement for the NCI 60 dataset.
- More careful selection of a small number of genes (10) improved the performance of FLDA dramatically

53

Comparison study – Discussion (I)

- "Diagonal" LDA: ignoring correlation between genes helped here. Unlike classification trees and nearest neighbors, LDA is unable to take into account gene interactions
- Although nearest neighbors are simple and intuitive classifiers, their main limitation is that they give very little insight into mechanisms underlying the class distinctions

54

Comparison study – Discussion (II)

- Variable selection: A crude criterion such as BSS/WSS may not identify the genes that discriminate between all the classes and may not reveal interactions between genes
- With larger training sets, expect improvement in performance of aggregated classifiers

55

We shall present two methods for clustering microarray gene expression data:

- GENE SHAVING (Hastie $et\ al$, 2000):
  - Extracts coherent and typical small clusters of genes that vary greatly across samples.
  - Seeks different clusters where difference is measured by correlation of the cluster means.
  - Software development: **GeneClust**
- MIXTURE-MODEL BASED CLUSTERING (McLachlan $et\ al$ 2001):
  - Based on mixtures of t distributions and mixtures of factor analyzers
  - Three part process:
    * Selection of relevant genes;
    * Clustering of the selected genes;
    * Clustering of the tissues on the basis of the selected genes or gene clusters.
  - Software development: EMMIX-GENE

Clustering of Microarray Gene Expression Data

1. **Unsupervised:** Class discovery
   - Only expression data is available
   - Existing clustering techniques: hierarchical, K-means, self-organising maps, support vector machines, etc.
2. **Supervised:** Class prediction
   - A response measurement is available for each sample e.g. group membership, survival time
   - The problem is difficult because:
     - Many correlated genes relative to few samples
     - Want to describe how groups of genes work together to predict outcome, not "variable selection"
     - Existing methods such as tree-based classifiers, discriminant analysis, neural networks, etc. may not work well.

MOTIVATION FOR GENE SHAVING

1. CRITERION for "optimal" cluster of genes:

   Favor subsets of genes that are similar and show large variance across cell lines $\rightarrow$ Var (Cluster average)

2. GOAL: Seek a sequence of nested gene clusters S_k s.t. Var(cluster mean) is maximal over all clusters of size $\mathbf{k}$.

3. FIND a subset of rows of S_k that maximize between column variance.

   (a) Greedy Bottom-up strategy : simple but short-sighted for large clusters

   (b) SEEK a weighted average of genes with maximal variance.

Gene-shaving is an iterative algorithm based on the principal components or the singular value decomposition (SVD) of the data matrix, with a twist.

Gene-shaving allows a gene to belong to more than one cluster

PC Shaving Algorithm (Hastie *et al*, 2000)

1. Start with the entire expression matrix $\mathbf{A_{N \times p}}$, each row centered to have zero mean,
2. Compute the leading PC of the rows of $\mathbf{A}$, the *super gene*,
3. Shave off (discard) the proportion α (10%) of the rows having lowest correlation (inner–product) with the *super gene*,
4. Repeat steps 2 and 3 until only two genes remain. This produces a sequence of nested gene clusters $A = B_0 \supset B_1 \supset B_2 \supset \ldots \supset B_S$, where S denotes the number of total shaving iterations.
5. Estimate the optimal cluster size $\hat{k}$ using the Gap statistic, let C_{opt} denote the optimal cluster found.
6. Orthogonalize each row of $\mathbf{A}$ with respect to $\overline{C_{opt}}$, the column average of C_{opt}.
7. Repeat steps 1 – 5 above with A_{ortho}, to find the second optimal cluster. This process is continued until a maximum of *M* clusters are found, with *M* chosen *apriori*.

Choice of cluster size via the Gap statistic

1. Goal: Select clusters that simultaneously exhibit large variances between samples and high similarity between gene rows in the cluster.

2. ANOVA analogy: Percent variance explained

$$R^2(B_s) = 100\frac{V_B}{V_T} = \frac{V_B/V_W}{1 + V_B/V_W}$$

$$V_W = \frac{1}{p}\sum_{j=1}^{p}\left[\frac{1}{k_s}\sum_{i\in B_s}(a_{ij} - \bar{a}_{.j})^2\right], \tag{1}$$

$$V_B = \frac{1}{p}\sum_{j=1}^{p}(\bar{a}_{.j} - \bar{a}_{..})^2, \tag{2}$$

$$V_T = V_B + V_W = \frac{1}{k_s p}\sum_{i\in B_s}\sum_{j=1}^{p}(a_{ij} - \bar{a}_{..})^2. \tag{3}$$

   Large $R^2 \rightarrow$ tight clusters of coherent genes.

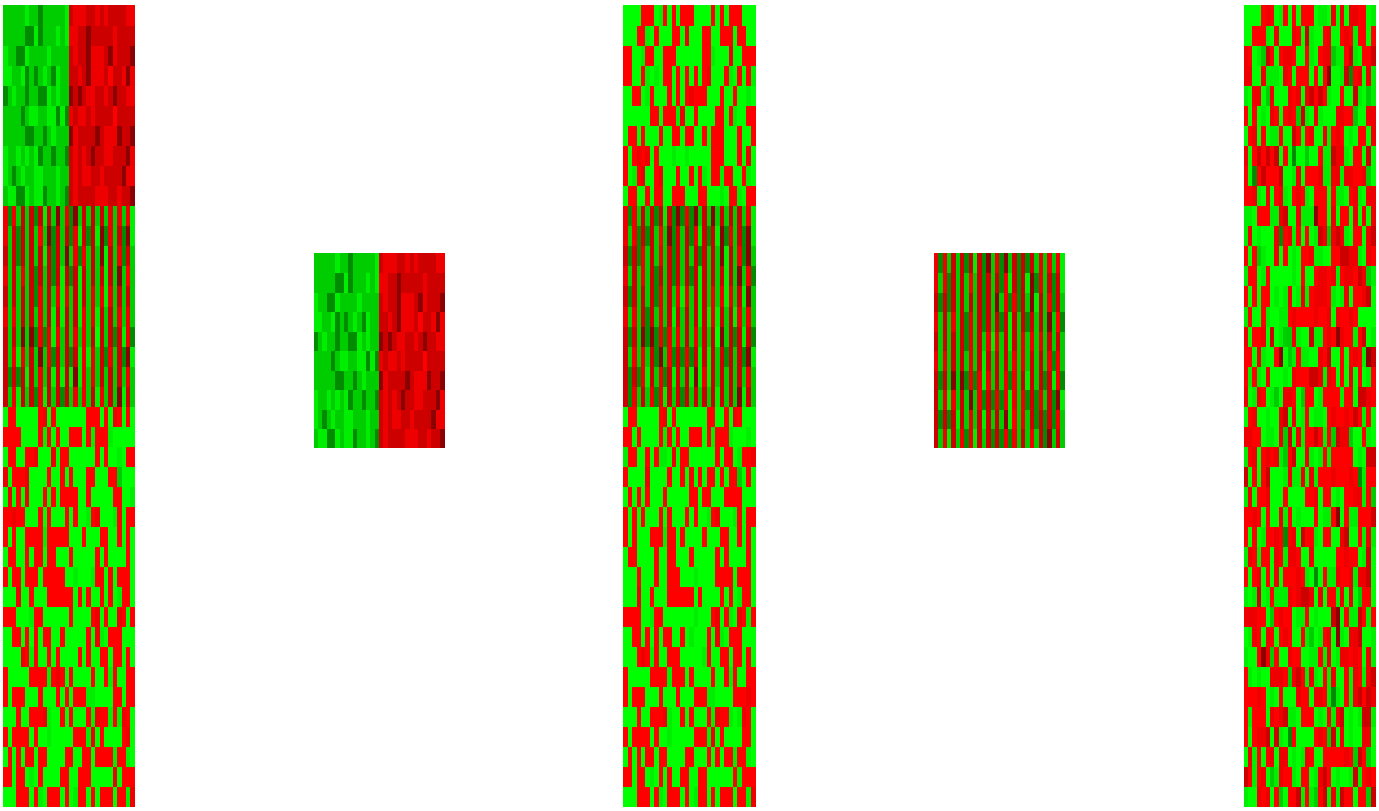
3. Gap function for cluster B_s of size k_s is defined as

$$Gap(k_s) = R^2_{k_s}(B_s) - \bar{R}^{2*}_{k_s}. \tag{4}$$

   k_{opt} maximizes the Gap statistic over all values of $k_s \in \{2, 3, \ldots, N\}$.

4. Plot Gap functions and percent variance curves.

Figure 3: Gene-shaving process for a simple example



Computational Efficiency

1. Repeated computation of the largest PC of $\mathbf{N}$ genes.
2. Singular value decomposition: $U\Lambda V'$ of A where
   - U is an $N \times N$ orthogonal matrix,
   - $\Lambda = diag(\sqrt{\lambda}_1, \ldots, \sqrt{\lambda}_r)$ where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r > \lambda_{r+1} = \cdots = \lambda_n = 0$, i
   - V is a $p \times p$ orthogonal matrix.
3. The columns of U are eigenvectors of AA', and the columns of V are the eigenvectors of $A'A$.
4. The eigenvector corresponding to the largest eigenvalue of AA' is the first column of U; its elements form the loadings corresponding to the first principal component of AA'.
5. Computational efficiency can be enhanced by deriving only the first column of U.

SUPERVISED GENE SHAVING

1. Let $C_k = a_{ij}(i = 1, \ldots, k; j = 1, \ldots p)$ with corresponding column average vector $\bar{a}$.
2. Aim: Find similarly expressing gene clusters that simultaneously have large variances but also discriminate well between the pre-determined classes.
3. Define Q^T to be a $g \times p$ matrix; each element q_{ij} in the i^{th} row of Q^T takes nonzero values of $\frac{1}{\sqrt{n_i}}$ if the j^{th} sample belongs to group G_i.
4. Define $A^\dagger = AQ$, an $N \times g$ matrix whose rows are standardized versions of the class means for each gene.
5. The cluster C_k of rows of the data matrix A with maximal *between-group variance* is also the cluster that maximizes the sum of squares of the mean of the rows of $A^\dagger$, a matrix with fewer columns than A.
6. Perform shaving algorithm on $A^\dagger$, but orthogonalize A.
7. General supervised gene shaving method is based on maximizing a weighted combination of the column means variance and the information measure

$$\max_{C_k}(1-\omega)Var(\bar{a}) + \omega\mathcal{J}_Y(\bar{a}) \qquad (5)$$

where $\omega \in [0, 1]$.

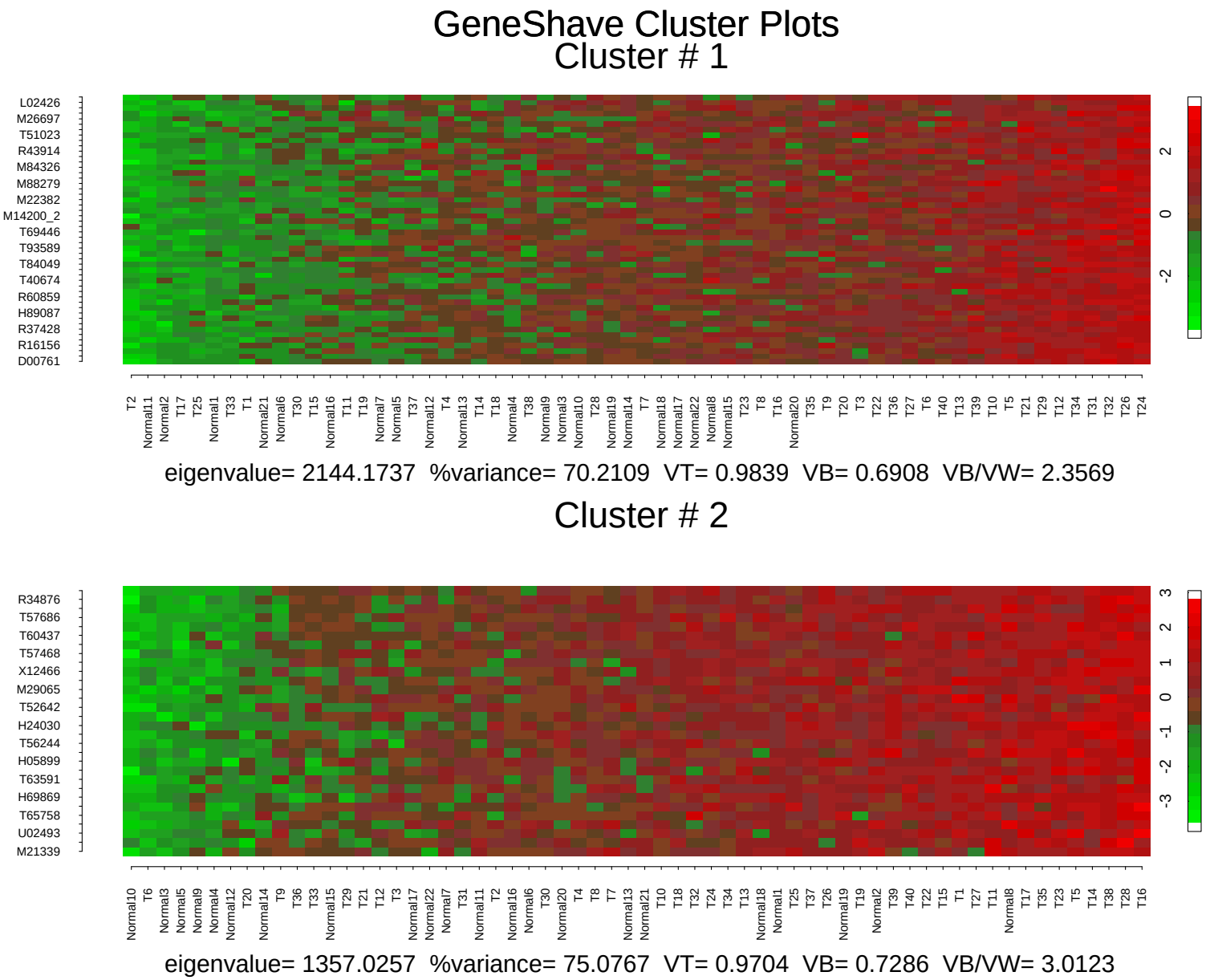


Figure 4: Heat maps of the first two unsupervised gene shaving clusters for the colon data, sorted by the column mean gene

Cluster 1: Specific genes highly expressed in tumors, 25 tumours grouped to the right.

Cluster 2: Pattern of high expression in tumor versus normal, orthogonal to CLuster 1.

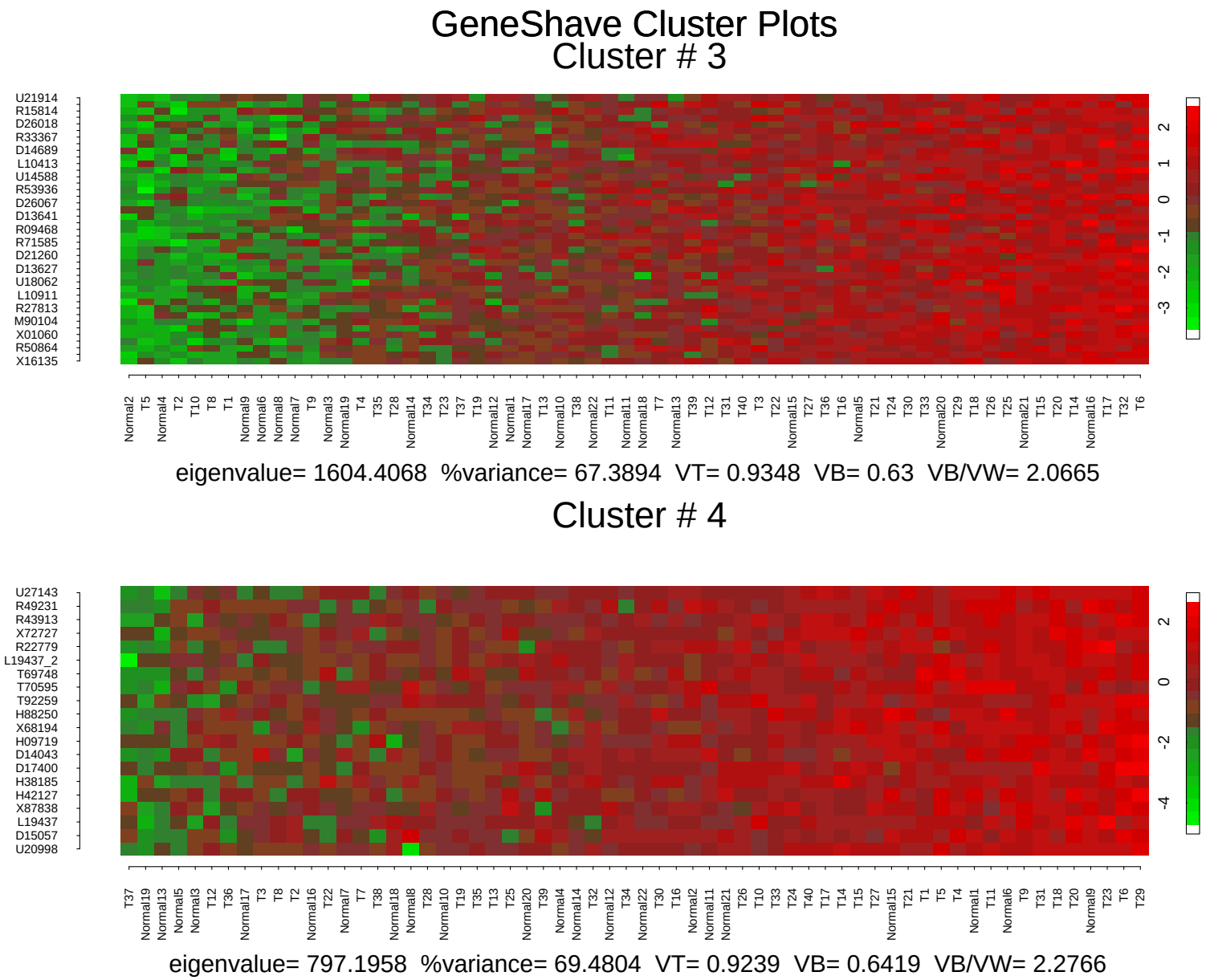
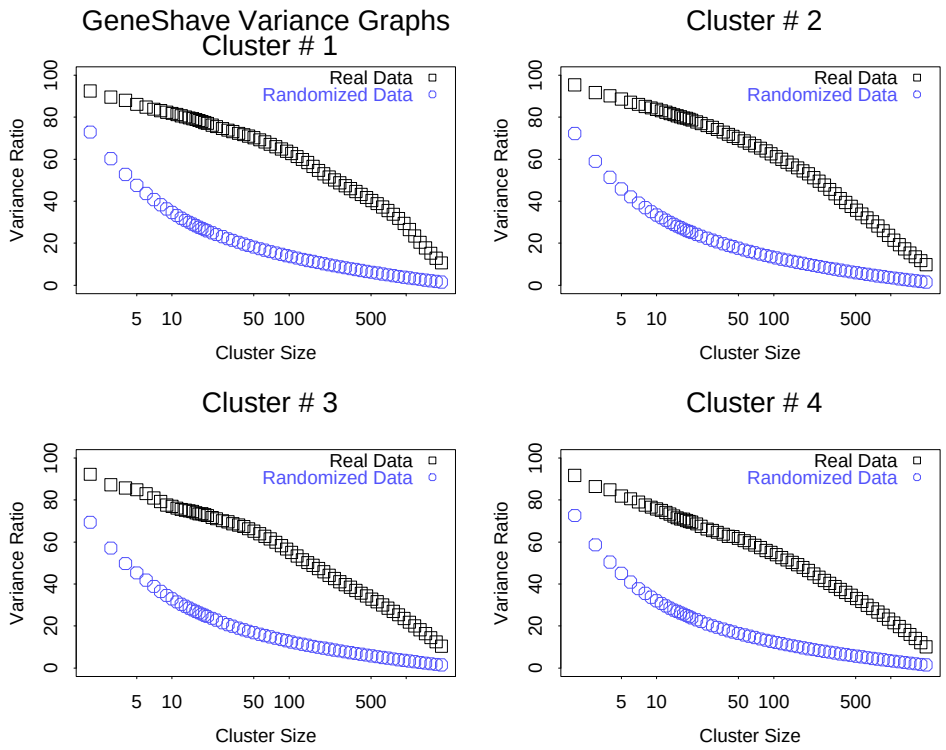


Figure 5: Heat maps of the third and fourth unsupervised gene shaving clusters for the colon data, sorted by the column mean gene

Cluster 3: Specific genes under expressed in the old protocol (first 11 patients) grouped to the left.

Cluster 4: Pattern found not coinciding with any known external classification.

(a)



(b)

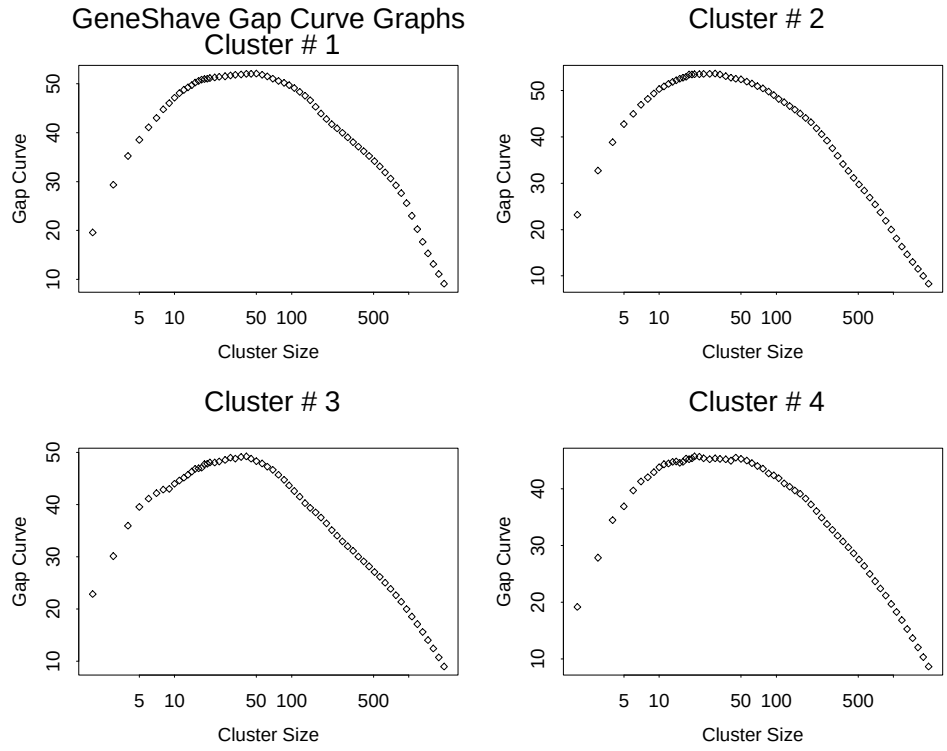


Figure 6: **(a)** Variance plots for the original and randomized data. The percent variance explained by each cluster, both for the original data, and for an average over twenty rando mized versions. **(b)** Gap estimates of cluster size. The Gap curve corresponds to the d ifference between the pair of variance curves.

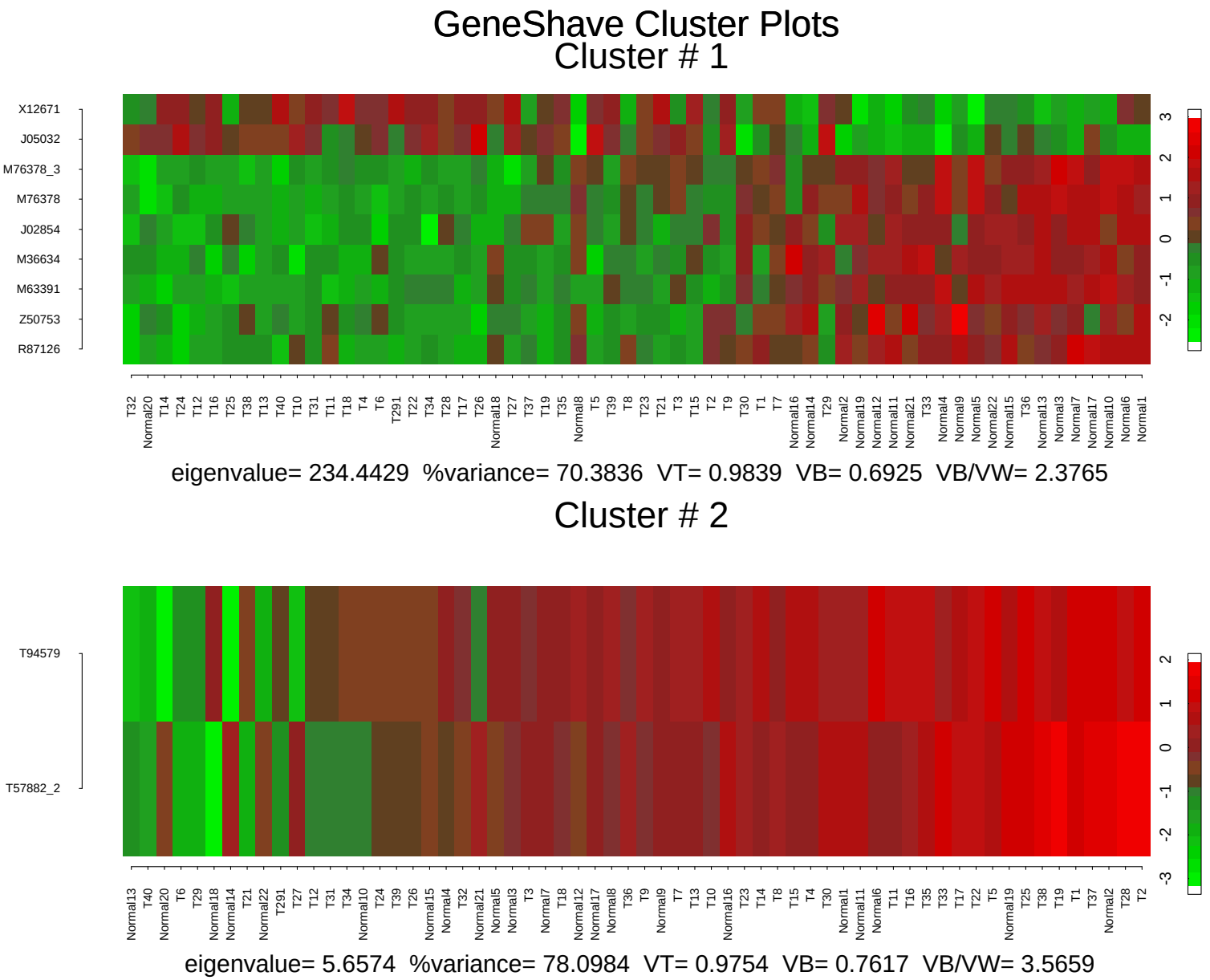


Figure 7: Heat maps of the first two gene shaving clusters for the colon data with full supervision; the samples are sorted by the column mean gene.

CLUSTER 1: Classifies normals versus tumours with an error rate of 6, comparable to other methods.

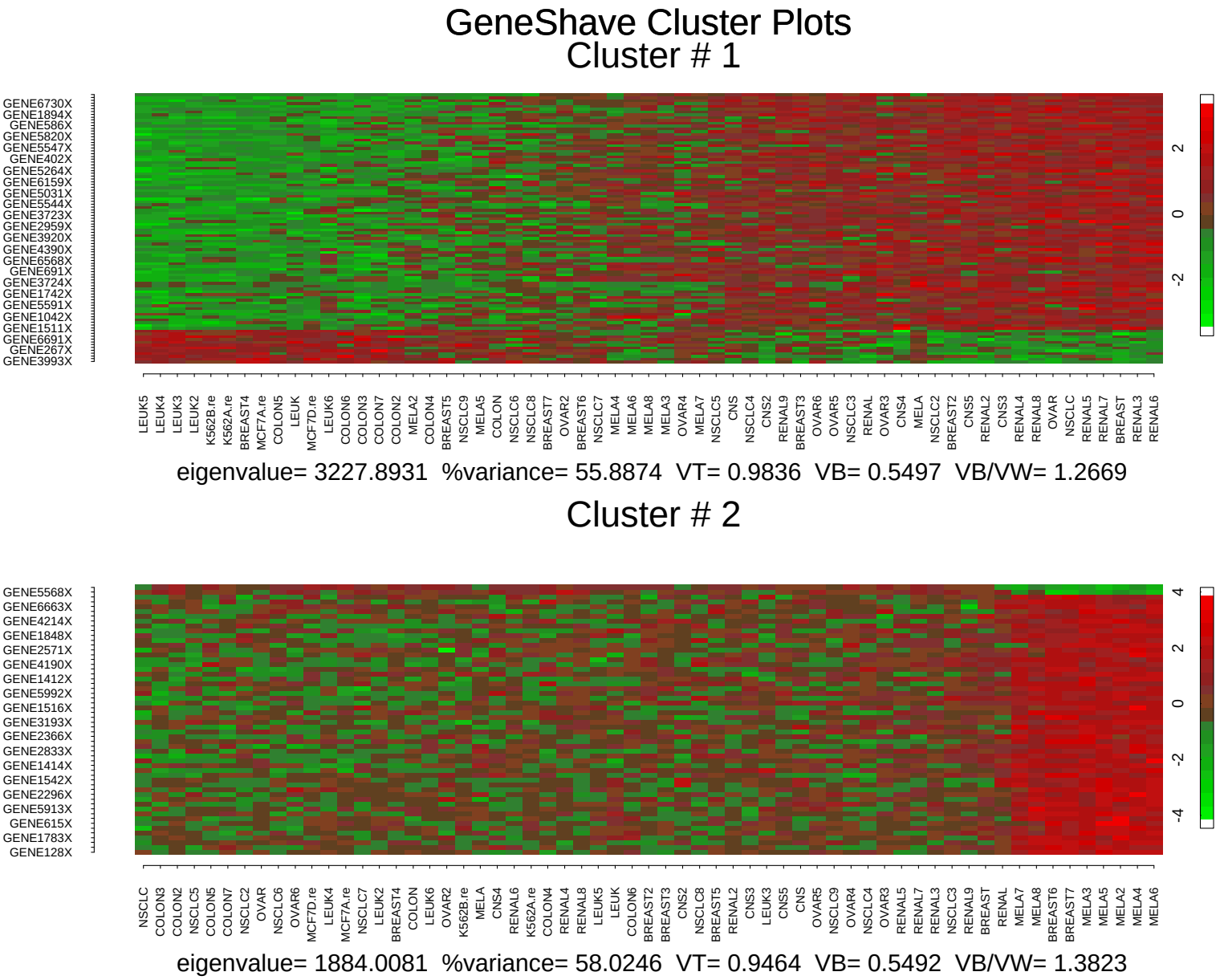


Figure 8: Heat maps of the first two unsupervised gene shaving clusters for the NCI60 data; the samples are sorted by the column mean gene.

CLUSTER 1 (96 genes): Leukemia to left, renal to right.

CLUSTER 2 (56 genes): Specific genes that underexpress or overexpress for melanoma and several breast tissues.

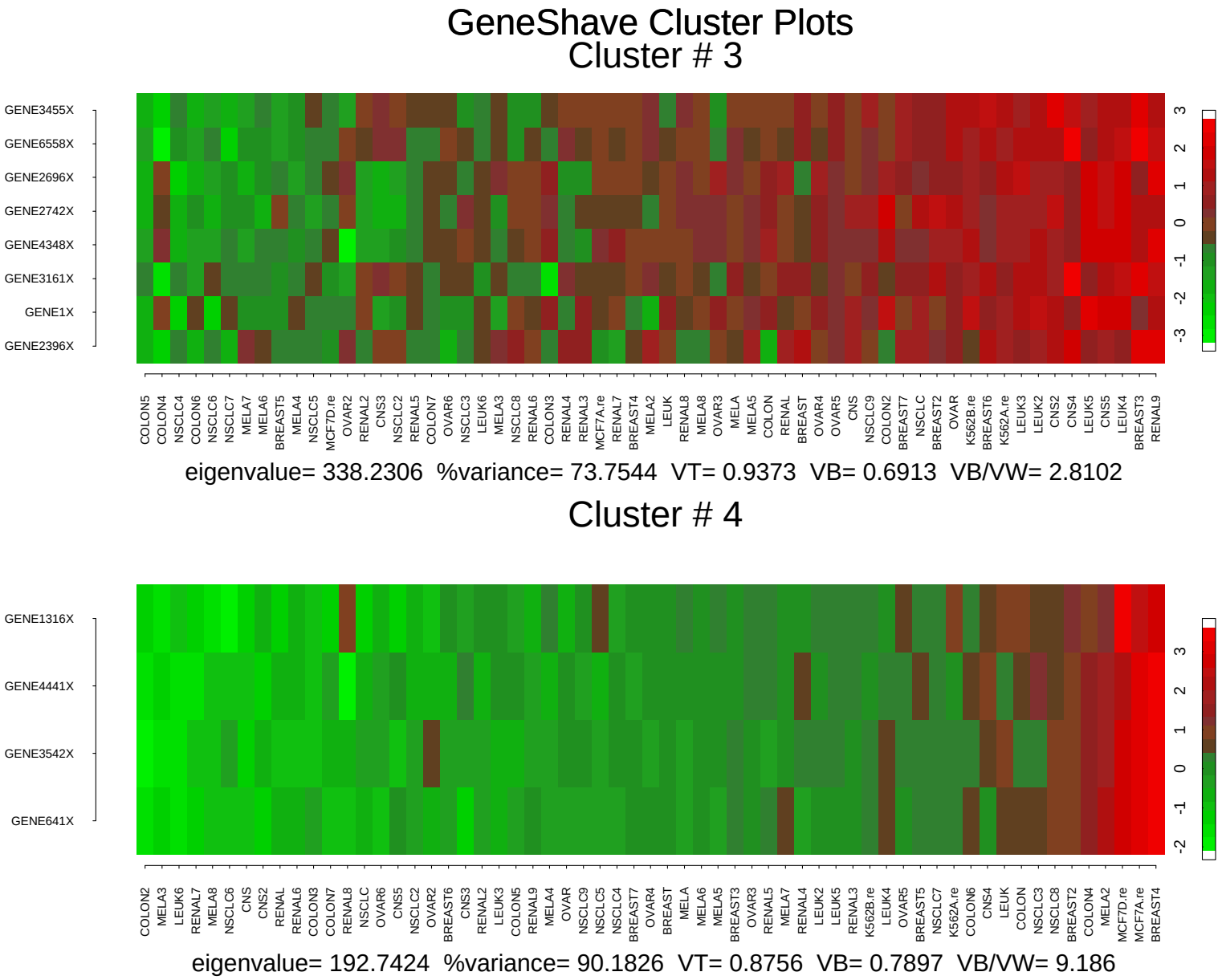


Figure 9: Heat maps of the third and fourth unsupervised gene shaving clusters for the NCI60 data; the samples are sorted by the column mean gene.

CLUSTER 3 (8 genes): Highly expressed genes for most CNS and leukemia.

CLUSTER2 (4 genes): Overexpressed genes for breast cell lines and breast tissues.

SOFTWARE IMPLEMENTATION

1. **GeneShaving**: for both unsupervised and supervised analyses, in SPlus and R. The source code is available from the StatLib S-archive collection at

   http://lib.stat.cmu.edu/S/

2. **Geneclust** has a graphical user interface (GUI) written in JAVA.

   JAVA GUI invokes the back-end statistical analysis process. This is an S-PLUS (or R) application with which the GUI communicates using a pseudo-terminal. The computationally intensive gene shaving algorithm is implemented using C, and is dynamically loaded into S-PLUS (or R) to perform the analysis. After the clusters have been extracted, the S-PLUS (or R) application presents the analysis results graphically.

   Available versions: **Solaris, Linux**

   Future version: **Windows NT**

   http://odin.mdacc.tmc.edu/~kim/geneclust

Selection of relevant genes by EMMIX-GENE (McLachlan et al 2001)

- Due to the possible presence of atypically large expression values for a particular tissue in the microarray data, it is better to use mixtures of t components as opposed to mixtures of normal components.
- When assessing the relevance of a gene, we examine $-2\log\lambda$ where λ is the likelihood ratio statistic for testing $g = 1$ versus $g = 2$ components in the mixture model.
- Although the t mixture model may provide robust estimates of the underlying distribution, it does not provide a robust assessment of the number of clusters in the data.
- If

$$-2\log\lambda > b_1 \tag{6}$$

then the gene is taken to be relevant provided that

$$s_{\min} \geq b_2, \tag{7}$$

where $s_{\min}$ is the minimum size of the two clusters implied by the two-component t mixture model and b_2 is a specified threshold.

- If (6) holds but (7) does not for a given gene, then the three-component t mixture model is fitted to the tissue samples on this gene, and the value of $-2\log\lambda$ calculated for the test of $g = 2$ versus $g = 3$;

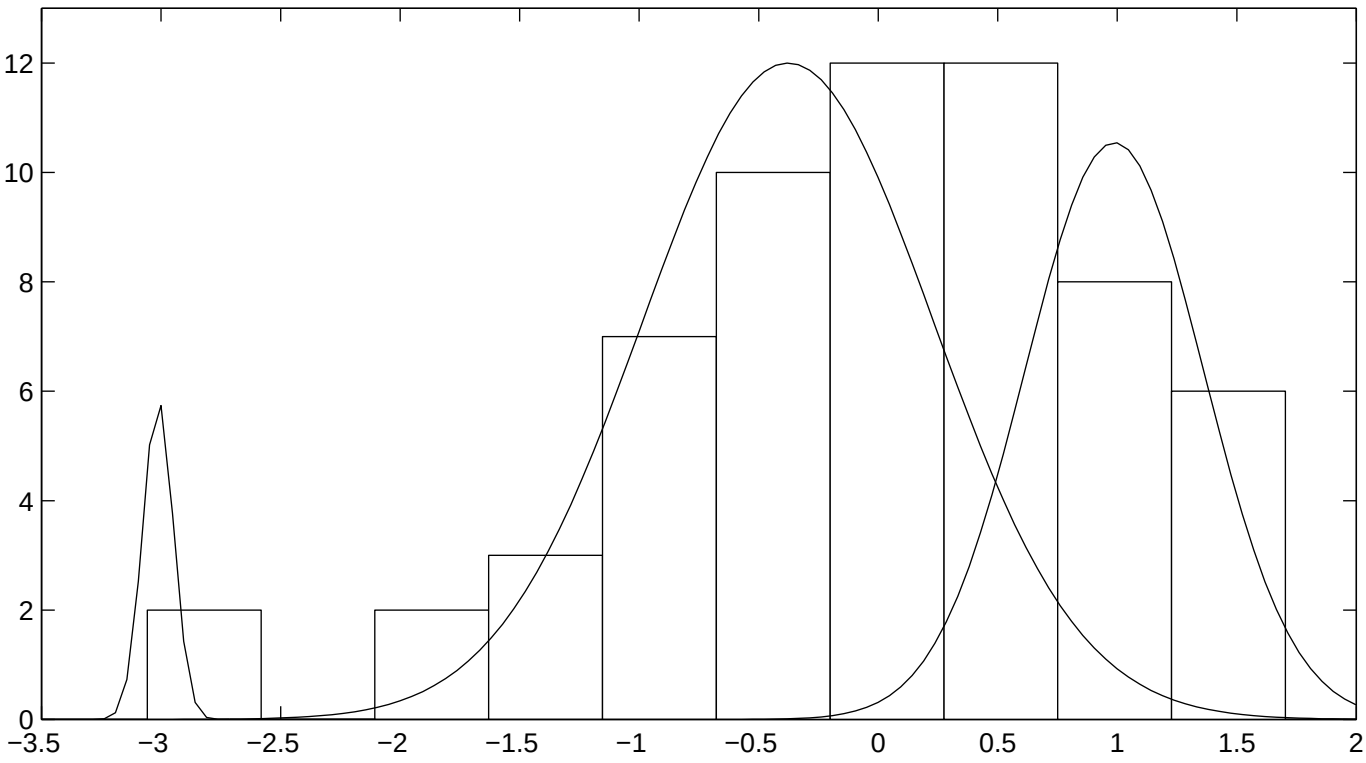


Figure 10: Histogram of Gene 474 (T70046) with Mixture of $g = 3$ Fitted t Components

- If (6) holds for this value of $-2\log\lambda$, the gene is selected as being relevant. Although the null distribution of $-2\log\lambda$ for $g = 2$ versus $g = 3$ is not the same as for $g = 1$ versus $g = 2$ components, it would appear to be reasonable here to use the same threshold (6).

Clustering of genes by EMMIX-GENE

- The genes are clustered into a user-specified number (N_0) of clusters by fitting a mixture of $g = N_0$ normal distributions with covariance matrices restricted to being equal to a multiple of the $p \times p$ identity matrix.
- The clusters of genes are ranked in terms of the likelihood ratio statistic calculated on the basis of the fitted mean of a cluster over the tissues for the test of a single versus two t-distributions.
- New software adaptation of EMMIX available at

  http://www.maths.uq.edu.au/~gjm/emmix/emmix.

Clustering of tissues by EMMIX-GENE

- The tissues are clustered by fitting mixtures of factor analyzers to the genes. Factor analysis can be used for dimensionality reduction by modeling $\mathbf{X}_j$ as

$$\mathbf{X}_j = \mu + \mathbf{B}\mathbf{U}_j + \mathbf{e}_j \quad (j = 1, \ldots, n), \tag{8}$$

  where $\mathbf{U}_j$ is a q-dimensional ($q < p$) vector of latent factors and $\mathbf{B}$ is a $p \times q$ matrix of factor loadings.

- The $\mathbf{U}_j$ are assumed to be .d.) as $N(\mathbf{O}, \mathbf{I}_q)$, independently of the errors $\mathbf{e}_j$, which are assumed to be i.i.d. $N(\mathbf{O}, \mathbf{D})$, where

$$\mathbf{D} = \text{diag}\,(\sigma_1^2, \ldots, \sigma_p^2),$$

  and where $\mathbf{I}_q$ denotes the $q \times q$ identity matrix.

- Thus, conditional on the u_j, the $\mathbf{X}_j$ are independently distributed as $N(\mu + \mathbf{B}\,\mathbf{u}_j, \mathbf{D})$.

- Unconditionally, the $\mathbf{X}_j$ are i.i.d. according to a normal distribution with mean $\mathbf{mu}$ and covariance matrix

$$\Sigma = \mathbf{B}\mathbf{B}^T + \mathbf{D}. \tag{9}$$

- At each iteration the inversion of the current value of the $p \times p$ matrix $(\mathbf{B}\mathbf{B}^T + \mathbf{D})$ can be undertaken using only inverses of $q \times q$ matrices

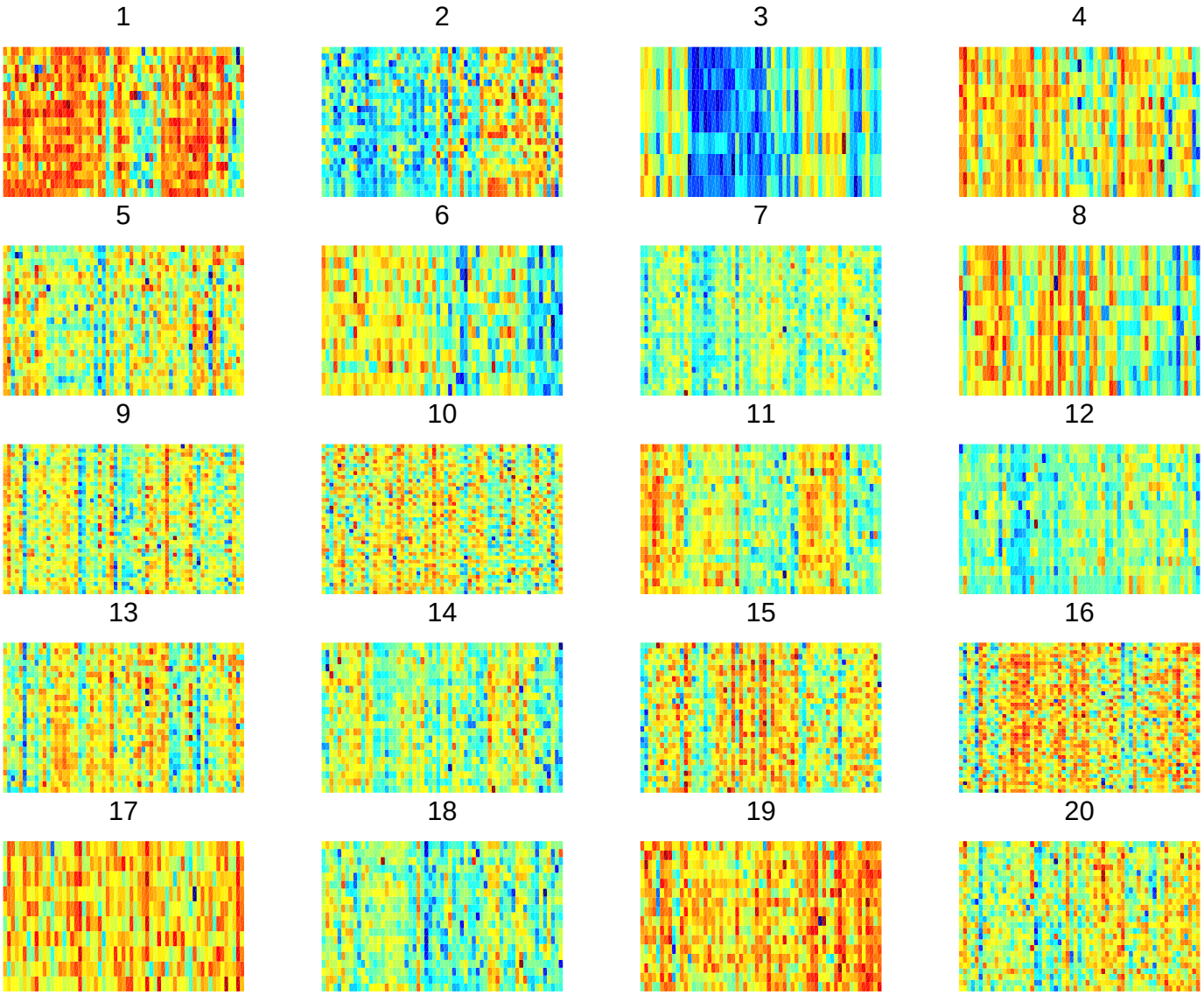


Figure 11: Heat maps of the twenty EMMIX-GENE clusters for the colon data

SIMULATIONS

- **Binary model:**
  Generated data with N=200 genes , p=20 samples.

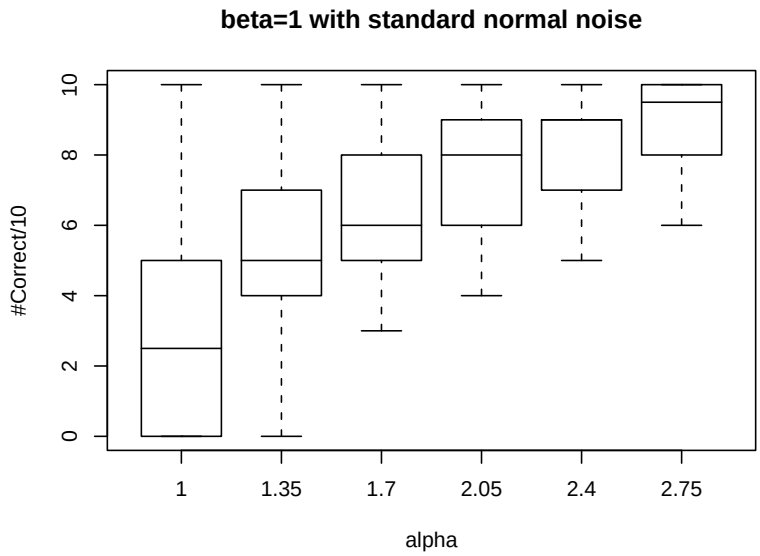
$$a_{ij} = s_{ij} + e_{ij}$$

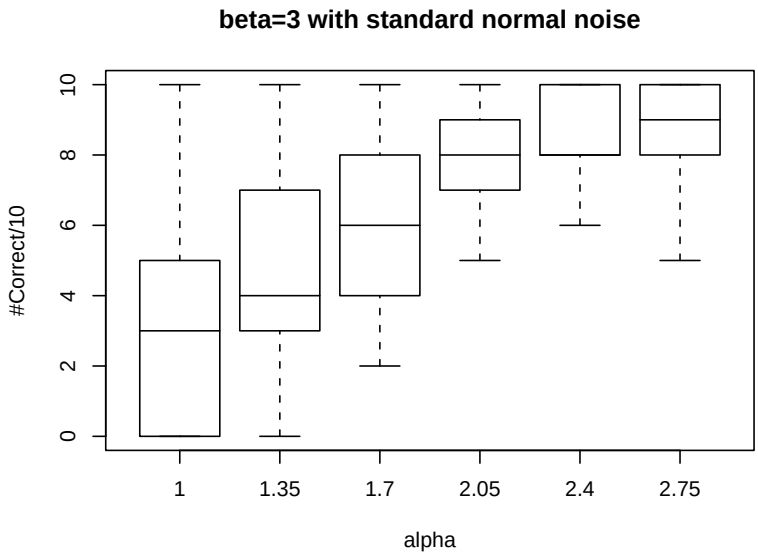
  where $e_{ij} \sim N(0,1)$;

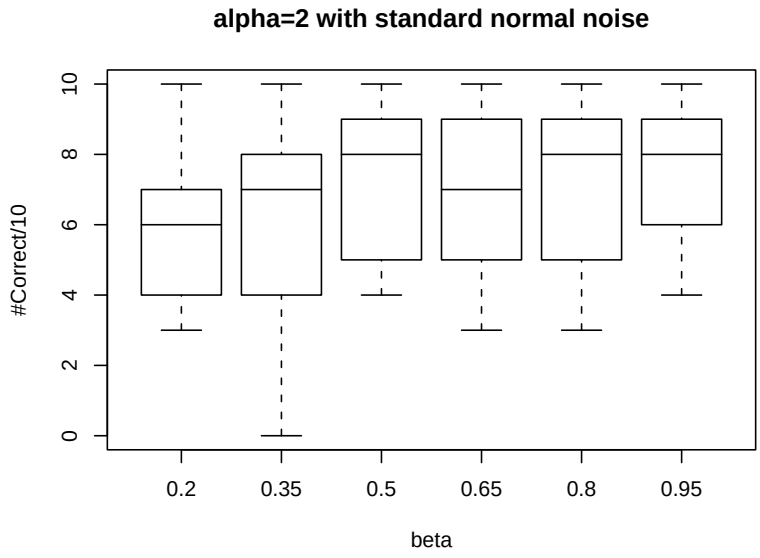
  for $1 \leq i \leq 10$ $\quad s_{ij} = 2\alpha(j/20)^{\beta} - \alpha(1 \leq j \leq 20)$

  for $i > 10$ $\quad s_{ij} = 0.$

- **Numerous Plaid models**
- Assess performance by number of correct genes (out of 10) identified in the first cluster







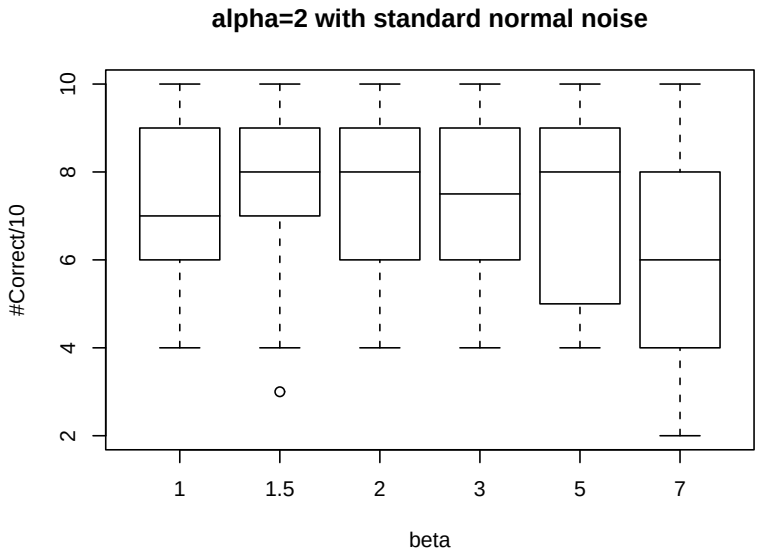


Figure 12: Simulation results: varying alpha and beta

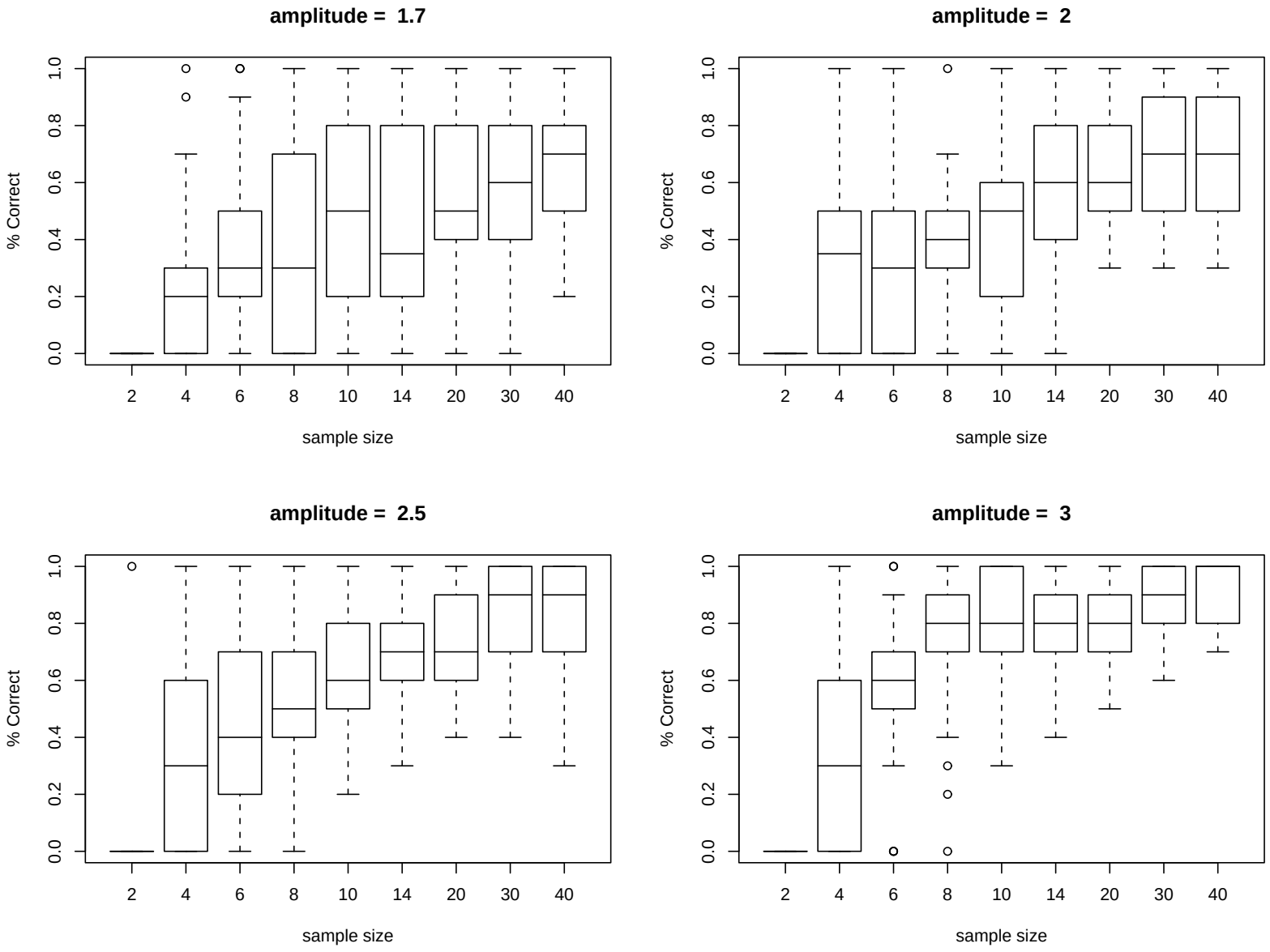


Figure 13: Simulation results for sample size

SIMULATION RESULTS

- β fixed, better clustering results with larger $\alpha(> 2.0)$
- $\alpha = 2.0$ fixed, clustering results do not seem to exhibit a clear pattern with varying β
- Gene shaving favors balanced samples
- Assuming balanced sample size, if the true cluster size is n, then require at least n samples and n tumors to achieve $> 50\%$ clustering power. Power increases with amplitude of signal.

15

GeneClust

Kim-Anh Do, Bradley Broom, Sijin Wen

Abstract

Two-way clustering techniques—such as hierarchical clustering, K-means clustering, tree-structured vector quantization, self-organizing map, and principal components analysis—have been used to organize genes into groups or "clusters" with similar behavior across relevant tissue samples or cell lines. However, these procedures seek a single global re-ordering of the samples or cell lines for all genes, and although they are effective in uncovering gross global structure, they are much less effective when applied to more complex clustering patterns; for example, where there are overlapping gene clusters. This chapter describes *gene shaving* (Hastie et al., 2000), a simple but effective method for identifying subsets of genes with coherent expression patterns and large variation across samples or conditions. After summarizing the gene shaving methodology, we describe two software packages implementing the method: a small package written in S (useable in either S-PLUS or R), and a considerably faster, mixed-language implementation with a graphical user interface intended for more applied use. The package can perform unsupervised, fully supervised, or partially supervised gene shaving, and the user is able to specify various parameters pertinent to the algorithm. The package outputs graphical representations of the extracted clusters (as colored heat maps) and diagnostic statistics. We then demonstrate how the latter tool can be used to analyze two published data sets (the Alon colon data and the NCI60 data).

15.1 Introduction

Two-way clustering techniques have been explored by many researchers in the field of bioinformatics to organize genes into groups or "clusters" with similar behavior across relevant tissue samples or cell lines. Such methods include hierarchical clustering, K-means clustering, tree-structured vector

quantization, self-organizing map, and principal components analysis, to name a few. These procedures seek global organization of genes and samples and are effective in uncovering gross global structure by seeking a single re-ordering of the samples or cell lines for all genes. However, the power of these methods are challenged by more complex clustering patterns. For example, distinct gene groups may cluster the samples in different ways; or there may exist overlapping gene clusters where the genes in cluster C_1 may suggest discrimination between cancer groups G_1 and G_2, say, while the genes in cluster C_2, possibly including some genes in C_1, may suggest a different way to distinguish between cancer groups G_3 and G_4.

Gene shaving (Hastie et al., 2000) is a simple but ingenious method that was proposed with the aim to resolve these issues. The development of the gene shaving methodology was motivated by the research goal to identify distinct sets of genes whose variation in expression could be related to a biological property of the tissue samples.

15.2 Methods

Let $X = x_{ji}$ be a row-centered $J \times I$ matrix of real-valued measurements representing the gene expression matrix, assuming no missing values. The rows are genes, the columns are tissue samples or cell lines, and x_{ji} is the measured (log) expression, relative to a baseline.

15.2.1 Algorithm

Gene-shaving is an iterative algorithm based on the principal components or the singular value decomposition (SVD) of the data matrix. It starts with the entire micro-array gene expression matrix X and seeks a function F of the genes in the direction of maximal variation across the tissue samples. The general algorithm for gene shaving may be described in the following steps:

- *Step 1:* Calculate the simplest form of the function F as a normalized linear combination of the genes weighted by its largest principal component loadings, referred to as the *super gene*. The genes may be sorted according to the principal component weights.
- *Step 2:* A fraction α of the genes having lowest correlation (essentially the absolute inner product) with the *super gene* are then *shaved* off (discarded) from the original data matrix.
- *Step 3:* The process of calculating the leading principal component and shaving off some genes is iterated on the reduced data matrix until only two genes remain. This iterative top-down process produces

a sequence of nested gene blocks of sizes ranging from the full set of J genes down to the final block consisting of just two genes.

- *Step 4:* Select the optimal cluster size based on a quality measure. In particular, to select the optimal number of genes in the cluster we use a *Gap* function, which is based on the between and within variances of the gene blocks computed from the raw data matrix and its permutation. The number of permutations should be specified *a priori*.

- *Step 5:* Remove the effect of genes in the optimal cluster, C_1 say, from the original matrix X. By computing the average gene or the vector of column averages for C_1, denoted by $\bar{C}_1$, we can remove the component that is correlated with this average. This is equivalent to regressing each row of X on the average gene row $\bar{C}_1$, and replacing the former with the regression residuals. Such a process is referred to as *orthogonalization* (Hastie et al., 2000), from which a modified data matrix X_{ortho} is produced.

- *Step 6:* With X_{ortho}, the whole process is repeated of calculating the leading principal component, producing another nested sequence of shaved gene blocks, applying the Gap statistic to obtain the next optimal cluster C_2, and orthogonalizing the current data matrix. This sequence of operations is iterated until M gene clusters $C_1, \ldots, C_M$ are found. The number of clusters to find, M, should be specified *a priori*.

- *Step 7:* For visual inspection, display graphically the derived M gene shaving clusters and corresponding variance and Gap plots.

To allow for negatively correlated genes to be included in a cluster, the average gene is actually a *signed mean gene*, that is if a gene row has a negative principal component weight, then the signs of the expression values are flipped before the average is calculated. The shaving process requires repeated computation of the largest principal component of a particular data matrix X or its subset (after at least one step of shaving). This process is easily implemented using the singular value decomposition $U\Lambda V'$ of X where U is an $J \times I$ matrix with orthonormal columns, $\Lambda = diag(\sqrt{\lambda}_1, \ldots, \sqrt{\lambda}_I)$ where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r > \lambda_{r+1} = \cdots = \lambda_I = 0$, and V is an $I \times I$ orthogonal matrix. The columns of U are eigenvectors of XX', and the columns of V are the eigenvectors of $X'X$. Since the eigenvector corresponding to the largest eigenvalue of XX' is the first column of U; its elements form the loadings corresponding to the first principal component of XX'. At each step of the shaving process, it is not required to compute the complete SVD of the data matrix. Computational efficiency can be enhanced by deriving only the first column of U.

15.2.2 Choice of cluster size via the Gap statistic

One important goal for any clustering technique is the ability to assess whether the extracted cluster is real, that is, we should be able to distinguish real patterns from random small clusters. Tibshirani and colleagues investigated a somewhat similar problem: estimation of the false discovery rate in the context of detecting differential gene expression in DNA microarrays (Tusher et al., 2001; Storey and Tibshirani, 2003). In the context of gene shaving, the *Gap statistic* was devised for selecting a reasonable cluster size from the sequence of nested gene blocks (Hastie et al., 2000). Further theoretical and simulation results (Tibshirani et al., 2001) showed that the Gap statistic usually outperforms other methods proposed in the literature in its ability to estimate the actual number of clusters or groups in a set of data. The Gap test is an adaptation of the usual permutation test based on randomization and an appropriate definition of a quality measure, or test statistic, of each cluster. In other words, under the null hypothesis that the rows and columns of the data matrix are independent, the permuted version of X, obtained by permuting the elements within each row of X, should "look" the same as the original X. Thus, if we obtain many permutations X^* of X and compute any test statistic $T(X^*)$ repeatedly, we can obtain the *null distribution* of T. If the independence (or no structure) hypothesis were true, we would expect T to behave like what we see in the repeated permutations. For our purposes here, the definition of T should reflect our ultimate goal of selecting clusters that simultaneously exhibit large variances between samples and high similarity between gene rows in the cluster. Established methods for a two-way analysis of variance can be adapted to our situation where the choice of a cluster size is governed by some function of the *between-to-within* variance ratio. After s shavings, let B_s denote the resulting gene block of k rows with elements x_{ji}. Define the *percent variance explained* as

$$R^2(B_s) = 100 \times \frac{V_B}{V_B + V_W} = 100 \times \frac{V_B/V_W}{1 + V_B/V_W}$$

where V_W, V_B and V_T denote the within, between, and total variances for the gene block B_s. The range of R^2 is over the interval $[0, 1]$ where values close to 0 imply no clustering evidence, while values closer to one imply tight clusters of similarly expressing genes. R^2 for a gene block of size k_s

can be explicitly computed by the following formulae

$$V_W = \frac{1}{I}\sum_{i=1}^{I}\left[\frac{1}{k_s}\sum_{j\in B_s}(x_{ji}-\bar{x}_{.i})^2\right], \tag{15.1}$$

$$V_B = \frac{1}{I}\sum_{i=1}^{I}(\bar{x}_{.i}-\bar{x}_{..})^2, \tag{15.2}$$

$$V_T = V_B + V_W = \frac{1}{k_s I}\sum_{j\in B_s}\sum_{i=1}^{I}(x_{ji}-\bar{x}_{..})^2. \tag{15.3}$$

The *Gap function* for the gene block B_s of size k_s is defined as

$$Gap(k_s) = R^2_{k_s}(B_s) - \bar{R}^{2*}_{k_s}. \tag{15.4}$$

where $\bar{R}^{2*}_{k_s}$ denote the averaged estimated value of the percent variance explained by blocks of size k_s, computed from $NumPerm$ permutations.

The optimal cluster size is the value k_{opt} that maximizes the Gap statistic over all values of $k_s \in \{2, 3, \ldots, N\}$. Implementation of the Gap statistic criterion is enhanced by plotting the percent variance curve $R^2_{k_s}(B_s)$ for the observed data matrix, versus the corresponding averaged curve $\bar{R}^{2*}_{k_s}$ for the collection of permuted data matrices, as a function of $k_s \in \{2, \ldots, N\}$. Alternatively, one can also include a plot of the computed values of $Gap(k_s)$ against $k_s \in \{2, \ldots, N\}$. Since the optimal cluster size usually assumes a small integer, these plots are more meaningful if depicted on the log scale.

The problem of estimating the number of clusters and finding the optimal cluster size in a dataset is a difficult one since there is no clear definition of a "cluster". Simulation studies have demonstrated that the Gap estimate is good for identifying well separated clusters (Hastie et al., 2000). However, when data are not clearly separated into groups, different people might have different opinions about the number of distinct clusters. In practice with DNA microarray data, the Gap curve plots often exhibit some flatness near the maximum, or may have more than one peak that are almost equivalent. A practicing biologist often wishes to explore further those genes next in line to the ones in the cluster corresponding to the estimated maximum Gap value. Therefore, graphing the Gap statistic with respect to increasing cluster size is important. Further to this aim of exploratory analysis, a more relaxed or flexible method for cluster size selection is to choose a cluster with a larger size than the optimal cluster but still maintaining a Gap statistic within a small percentage of the maximal Gap statistic, say 5%. In our implementations, we refer to this parameter as the *Gap Tolerance*, where a Gap Tolerance of 0 refers to cluster sizes corresponding to the maximum Gap estimate.

15.2.3 Supervised gene shaving for class discrimination

Either fully or partially supervised shaving with the ultimate aim of class discrimination can be carried out if the column (sample) groupings is known. Briefly, supervised shaving maximizes a weighted combination of column variance and an information measure that depends on the nature of the auxiliary information. Let B_k denote a generic cluster of k genes where $B_k = x_{ji}(j = 1, \ldots, k; I = 1, \ldots I)$ with corresponding column average vector $\bar{x}_{B_k}$. Suppose there are c classes or categories where $c < I$. Let $Y = (y_1, \ldots, y_I)$ denote a vector of integers indicating class membership for each sample. For example, if the data samples represent normals and tumors then Y_j can take a value of 1 if the sample came from a normal tissue, or a value of 2 if it was extracted from a tumor sample. Let class averages be denoted by $\bar{x}_{class} = (\bar{x}_{normals}, \bar{x}_{tumors})$ which represent the average expression of all genes in the gene block B_k summed over the class of normal or tumor tissues respectively. Thus a measure of class discrimination can be represented as $\mathrm{Var}(\bar{x}_{class})$. Under principal component shaving, it can be proven that

$$\mathrm{Var}(\bar{x}_{B_k}) = \mathbf{w}(X^{\dagger} X^{\dagger T})\mathbf{w},$$

where $\mathbf{w} = (w_1, \ldots, w_J)$ represent the principal component loadings associated with each gene in the original data matrix, and $X^{\dagger}$ is a i $J \times c$ matrix whose rows are standardized versions of the class means for each gene. Define an information measure $\mathcal{J}_Y(\bar{x}_{B_k})$ to be an appropriate quadratic function of the column averages $\bar{x}_{B_k}$, then the general supervised gene shaving method is based on maximizing a weighted combination of the column means variance and the information measure

$$(1 - \omega)Var(\bar{x}_{B_k}) + \omega \mathcal{J}_Y(\bar{x}_{B_k}) \quad \omega \in [0, 1], \tag{15.5}$$

over all possible clusters of sizes ranging from 2 to J. Full supervision is equivalent to $\omega = 1$; while partial supervision is indicated by values of ω between 0 and 1. The equation (15.5) presents a compromise between supervised and unsupervised clustering. In the context of class discrimination, an indicator vector with the length of samples is needed to specify the external classification of the samples; the information measure is then taken as the sum of squared differences between the class averages. For a more detailed discussion of the methodology development and extension in the context of general supervision, see Hastie et al. (2000).

15.3 Software

15.3.1 The GeneShaving package

We have implemented the gene shaving method, for both unsupervised and supervised analyses, in an S-language package we call **GeneShav-**

ing. The source code is available from the StatLib S-archive collection at `http://lib.stat.cmu.edu/S/`.

To illustrate the gene-shaving methodology, and in particular the use of the **GeneShaving** package, we present below a simple example involving simulated data. We generated a data matrix X of J genes and I samples. Each element x_{ji} consists of a simple signal s_{ji} that can take values $-H, 0, +H$, and noise $\epsilon_{ji} \sim N(0,1)$. Specifically, for K an even integer less than J let

$$x_{ji} = s_{ji} + \epsilon_{ji}; \tag{15.6}$$

$$\text{for } 1 \le j \le K/2: \quad s_{ji} = \begin{cases} -H & \text{if} \quad i \le I/2 \\ +H & \text{if} \quad i > I/2 \end{cases}$$

$$\text{for } K/2 < j \le K: \quad s_{ji} = \begin{cases} -H & \text{if} \quad i \text{ odd} \\ +H & \text{if} \quad i \text{ even} \end{cases}$$

$$\text{for } j > K: \quad s_{ji} = 0.$$

For example, suppose that a data frame *demo.dat* is generated by taking $H = 2.5$, $J = 100$, $K = 40$ and $I = 40$.

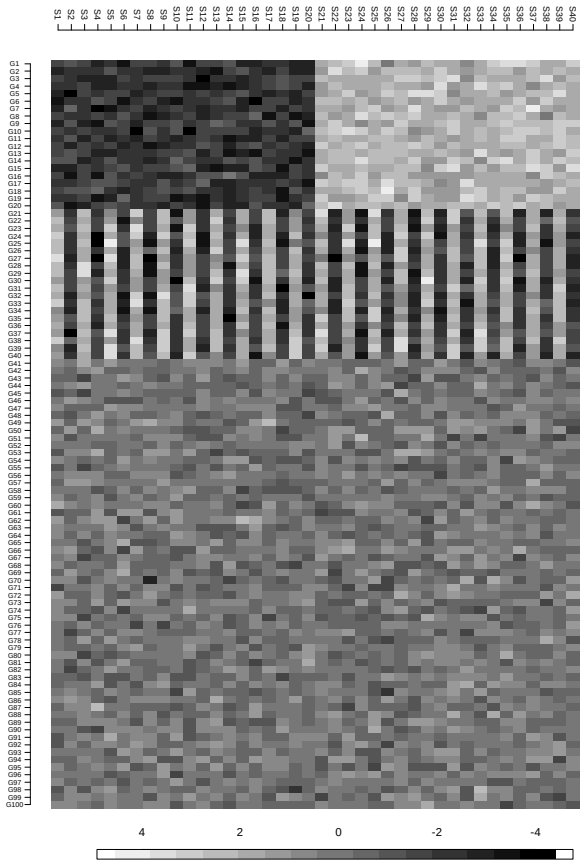
```
> dim(demo.dat)
[1] 100  40
# 100 genes (rows) and 40 samples (columns)
> demo.dat[(1:5), ]
# part of the data frame (first 5 genes)

          S1        S2        S3        S4        S5
G1 -2.207234 -1.480847 -2.177457 -4.260801 -3.407388
G2 -3.793671 -2.979609 -3.069576 -3.474391 -1.611161
G3 -2.108358 -3.166314 -3.811151 -4.053201 -3.869151
G4 -3.112067 -2.992520 -2.024178 -3.646037 -4.109120
G5 -4.371155 -5.390998 -1.957976 -3.270669 -2.245929

          S36      S37      S38      S39      S40
       5.052145 4.323419 2.703080 2.548090 4.489734
       1.713629 3.588597 2.443339 3.032010 2.852629
. . .  2.743546 3.445167 4.670663 2.045603 2.986980
       2.964064 1.041393 2.827109 3.781293 2.153697
       2.241520 3.447944 3.595235 4.946946 2.183231
```

Its heat map on the gray scale, Figure 15.1, was produced by the *image()* function in S-PLUS.

The S-PLUS function *shave()* implements *unsupervised gene shaving*. It has three parameters:

- **X**, the gene expression micro-array data. It is a data frame with gene names (rows) and sample names (columns).

Figure 15.1. Image plot of the simulated data



- **NumClusts**, the number of clusters to extract.
- **NumPerm**, the number of permutations used to select the optimal cluster size by the *Gap* function.

For instance, to extract two gene clusters using 5 permutations, the inputs of *shave()* are

1. the S-PLUS data frame *demo.dat*,
2. the number of clusters (i.e., 2), and
3. the number of permutations (i.e., 5).

The *shave()* function returns the gene labels belonging to each extracted cluster.

```
> shave(demo.dat,2,5)
## this is the output of gene labels for cluster #1:
 [1] "G1"  "G2"  "G3"  "G4"  "G5"  "G6"  "G7"
 [8] "G8"  "G9"  "G10" "G11" "G12" "G13" "G14"
[15] "G15" "G16" "G17" "G18" "G19" "G20"
## this is the output of gene labels  for cluster #2:
 [1] "G21" "G22" "G23" "G24" "G25" "G26" "G27"
 [8] "G28" "G29" "G30" "G31" "G32" "G33" "G34"
[15] "G35" "G36" "G37" "G38" "G39" "G40"
```

The *shave()* function also outputs, for each extracted cluster, three plots:

- the percent variance explained by the actual and permuted data,
- a gap plot,
- a cluster plot (heat map) of the genes.

Figure 15.2 shows the plots output by *shave()* for the two clusters extracted from the simulated data.

Suppose that instead of unsupervised shaving we wish to perform a supervised shaving procedure. Perhaps the 40 samples (columns) in *demo.dat* are classified into two groups, with the first 20 samples in the first group and the last 20 samples in the second group. Thus an indicator vector can be constructed and is coded 1 for the first 20 samples (columns) and coded 2 for the last 20 samples (columns):

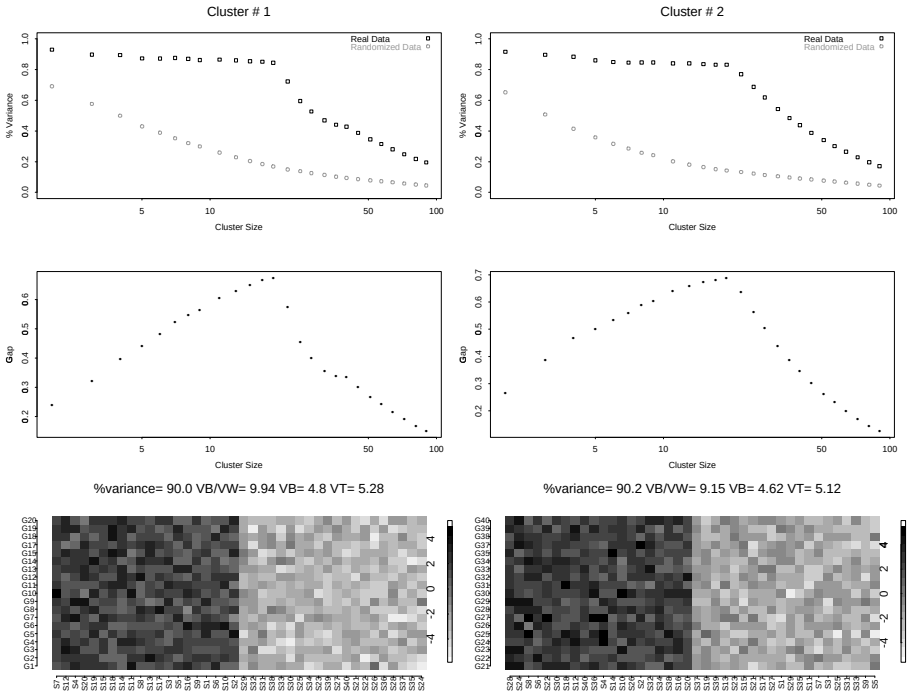
```
>clf <- c(rep(1,20),rep(2,20)) #indicator vector
> clf
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[11] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

Since there are only two classes of samples, we should only try to find a single cluster. In the general scenario of more $g > 2$ classes, the number of clusters to consider can be greater than g to cover both the main effects and the possible interactions between the different classes.

In the **GeneShaving** package, the S-PLUS function *super.part()* implements supervised gene shaving. The outputs for *super.part()* are the same as those in *shave()*; however, the inputs are slightly different:

- **X**: gene expression micro-array data. It is a data frame in S-PLUS with gene names (rows) and sample names (columns).
- **clf**: an indicator vector to indicate the sample classification.
- **lam**: supervision parameter - fully supervised shaving is carried out if *lam = 1*; partially supervised shaving is carried out if lam takes a value between 0 and 1.

Figure 15.2. Gap plots, variance plots, and the derived clusters from applying unsupervised gene shaving to the simulated data.

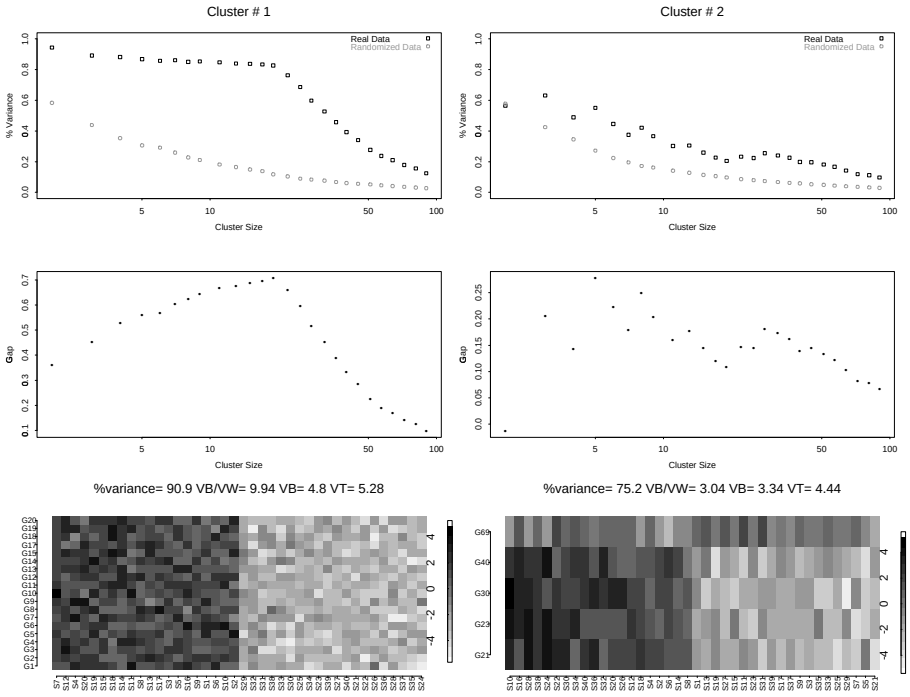


- **NumClusts**: the number of clusters to extract.
- **NumPerm**: the number of permutations for selecting the optimal cluster by the Gap function.

```
## fully supervised shaving with 5 permutations
## 2 clusters to extract

> super.part(demo.dat,clf,1,2,5)

## this is the output of gene labels for cluster #1:
 [1] "G1"  "G2"  "G3"  "G4"  "G5"  "G6"  "G7"  "G8"
 [9] "G9" "G10"  "G11" "G12" "G13" "G14" "G15" "G16"
[17] "G17" "G18" "G19" "G20"
## this is the output of gene labels  for cluster #2:
 [1] "G21" "G23" "G30" "G40" "G69"
```

The outputs including gap plots, variance plots, and heat maps of the derived clusters under full supervision are shown in Figure 15.3. Observe that the first cluster gives a perfect discrimination of the samples (samples

Figure 15.3. Gap plots, variance plots, and the derived clusters from applying fully supervised gene shaving to the simulated data.



S1-S20 versus samples S21-S40). The second cluster is orthogonal to the first derived cluster and groups the even numbered samples together versus the odd numbered samples. Since full supervision was employed, the whole process was dominated by the external classification pattern (captured by cluster 1), and only the strongest genes for the alternative clustering pattern emerged in cluster 2, suggesting that further exploration is required here, such as a partially supervised analysis.

15.3.2 GeneClust: a faster implementation of gene shaving

The **GeneShaving** package described above has two significant limitations that prevent its routine use in clinical data analyses:

- it is far too slow, and
- it is only usable by S programmers.

We have developed the **GeneClust** software package to address these issues.

GeneClust has a graphical user interface (GUI) written in JAVA. Figure 15.4 shows the initial GUI interface. The GUI allows users to

- Perform a simple one-way hierarchical clustering by genes or by samples;
- Perform unsupervised, fully supervised, or partially supervised gene shaving; if the latter is chosen the user can also select the amount of partial supervision;
- Specify the number of clusters to extract, the percent to shave for each iteration, the number of permutations used to calculate the Gap statistic, and the level of Gap tolerance.

GeneClust may be used either to analyze a real data set by selecting the **Raw Data** mode, or to investigate the performance of gene shaving for simulated data sets that are variations of the model (15.6), by selecting the **Demo** mode.

When the user starts the gene shaving procedure, the GUI invokes the back-end statistical analysis process. Because this is an S-PLUS (or R) application, with which the GUI communicates using a pseudo-terminal, either S-PLUS or R must be installed (see Chapter 1). The computationally intensive gene shaving algorithm is implemented using C, and is dynamically loaded into S-PLUS (or R) to perform the analysis. After the clusters have been extracted, the S-PLUS (or R) application presents the analysis results graphically.

The **Geneclust** software has been implemented for the Solaris and Linux operating systems, and for the S-PLUS and R statistical programming environments. For exact details and continuous updates, check the website:

`http://odin.mdacc.tmc.edu/~kim/geneclust`.

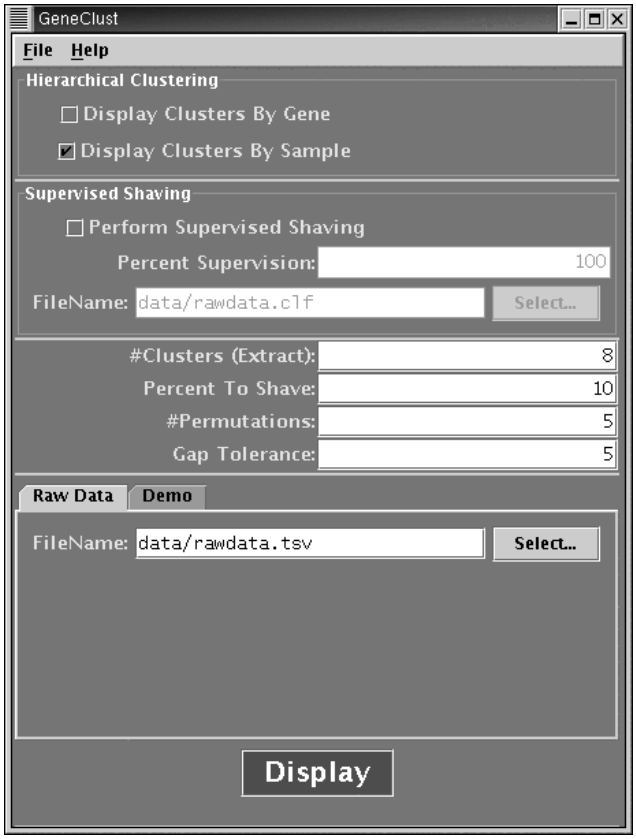
15.4 Applications

We demonstrate how **GeneClust** can be employed in the analyses of two real published data sets: a colon data set (Alon et al., 1999), and the human tumor NCI60 data set (Ross et al., 2000; Scherf et al., 2000).

15.4.1 The Alon colon data set

Alon and colleagues used Affymetrix oligonucleotide arrays to monitor absolute measurements on expressions of over 6,500 human gene expressions in 40 tumor and 22 normal colon tissue samples (Alon et al., 1999). These samples were taken from 40 different patients so that 22 patients supplied both a tumor and normal tissue sample. They focussed on the 2,000 genes with highest minimal intensity across the samples. There was a change in

Figure 15.4. A screen dump of the **GeneClust** GUI template



the protocol during the conduct of the microarray experiments (Getz et al., 2000). Tumor and normal tissue samples were taken from the first 11 patients using a poly detector, while the remaining tumor and normal tissue samples were taken from the remaining 29 patients using total extraction of RNA.

Before performing any clustering of this set, we processed the data by taking the (natural) logarithm of each expression level in X. Then each column of this matrix was standardized to have mean zero and unit standard deviation. Finally, each row of the consequent matrix was standardized to have mean zero and unit standard deviation.

We then selected a 446-gene reduced Alon data set, as described in Mclachlan *et al.* (2002), by selecting a subset of relevant genes based on the likelihood ratio statistic $-2\log\lambda$ calculated from fitting a single t distribution versus a mixture of two t components. The relevance of each of the J genes was assessed by fitting one- and two-component t mixture models to

the expression data over the I tissues for each gene considered individually. If $-2\log\lambda$ is greater than a specified threshold b_1,

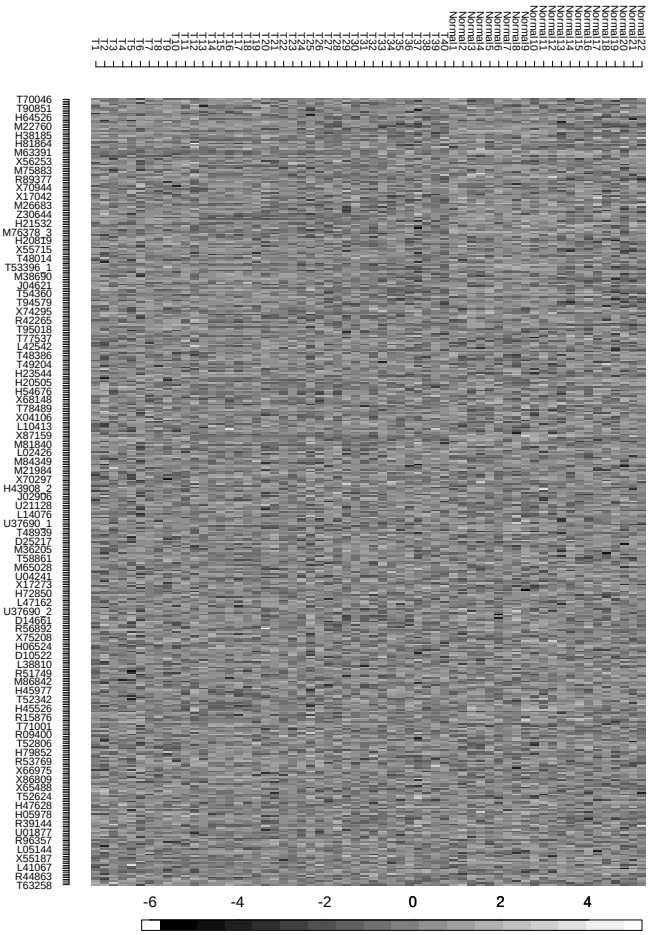
$$-2\log\lambda > b_1 \tag{15.7}$$

then the gene is taken to be relevant provided that

$$s_{\min} \geq b_2, \tag{15.8}$$

where $s_{\min}$ is the minimum size of the two clusters implied by the two-component t mixture model and b_2 is a specified threshold. Following McLachlan, we chose $b_1 = 8$ and $b_2 = 8$, thus retaining the reduced Alon microarray data matrix X that has $J = 446$ rows and $I = 62$

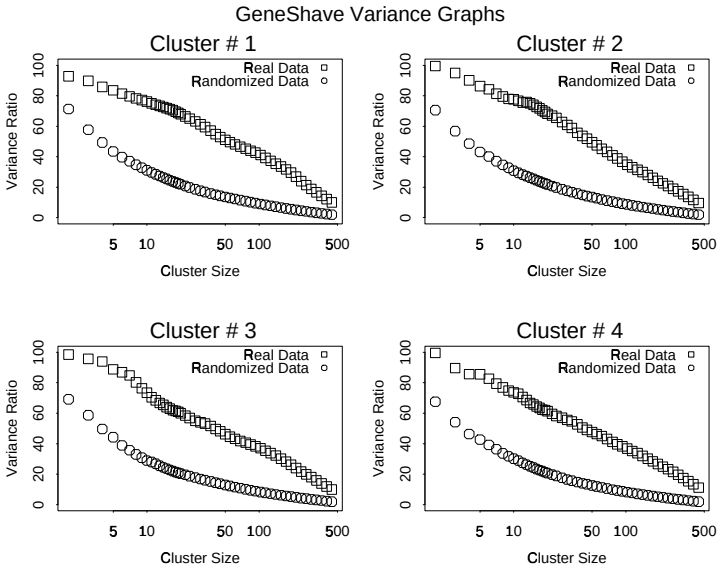
Figure 15.5. Heat map of the reduced Alon colon expression array of 446 genes



columns. We have rearranged the data consecutively so that the first 40 columns correspond to tumor samples followed by 22 columns of normal tissues. Further, the first eleven tumor columns are matched with the first eleven normal columns by patient. Figure 15.5 displays the observed gene expression matrix X of 446 genes.

We investigated the performance of unsupervised gene shaving when applied to the reduced Alon data set of 446 genes. We applied **GeneClust** with a Gap Tolerance of 5% which allows us to pick larger cluster sizes than those dictated by the Maximum Gap criterion. The first four orthogonal unsupervised clusters along with their corresponding variance and Gap plots are presented in Figures 15.6 and 15.7. The columns in the heat maps are sorted according to the column averages. Estimated eigenvalues, R^2 and between to within variance ratios are also printed beneath the heat map for each cluster. The first cluster corresponds to a relatively small between-to-within variance ratio of 1.72 and does not seem to reflect any pattern resembling the normal versus tumor or change in paradigm external classification. It can be seen, however, that cluster 2 (19 genes) consists of two negatively correlated subclusters that capture the normal versus tumor structure quite well showing genes that are either under expressed or over expressed for most of the normal tissues (clustered to the right). In contrast, the change in paradigm structure is reflected in cluster 3 (7 genes) where most of the normal and tumor tissues corresponding to the first 11 patients are clustered to the right. A number of muscle-specific genes have been identified as being characteristic of normal colon samples (Ben-Dor et al., 2000). We note that two of these genes (J02854, T60155) along with two suspected smooth-muscle genes (M63391, X74295) are included in the second cluster. Clusters 2 and 3 also correspond to high R^2 estimates of around 70% or more and large values of V_B/V_W (2.3 and 5.6 respectively). Inspection of the variance and Gap curves can also be informative. The variance curves are well separated for all four clusters indicating that the clusters found are not entirely random. Except for cluster 3 with a clear maximum at a cluster size of 9, the Gap curves for the other clusters exhibit some flatness to the right of the maximum, suggesting that it is worth exploring additional genes in slightly larger cluster sizes than those presented here.

(a)



(b)

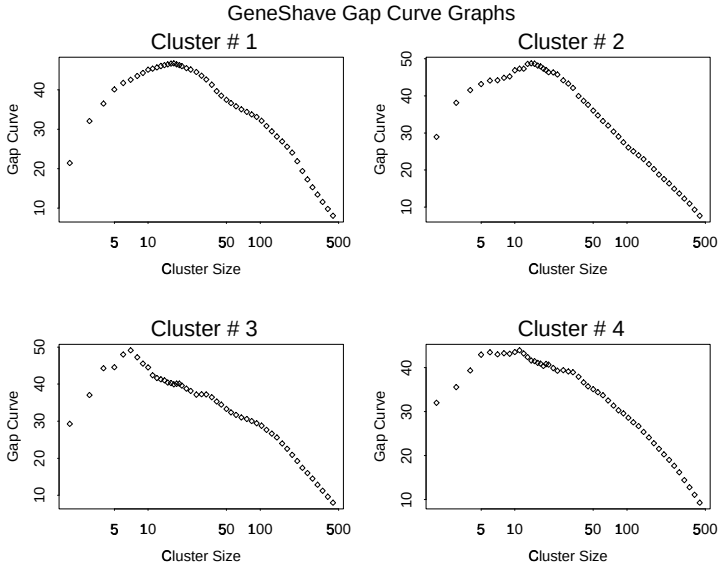


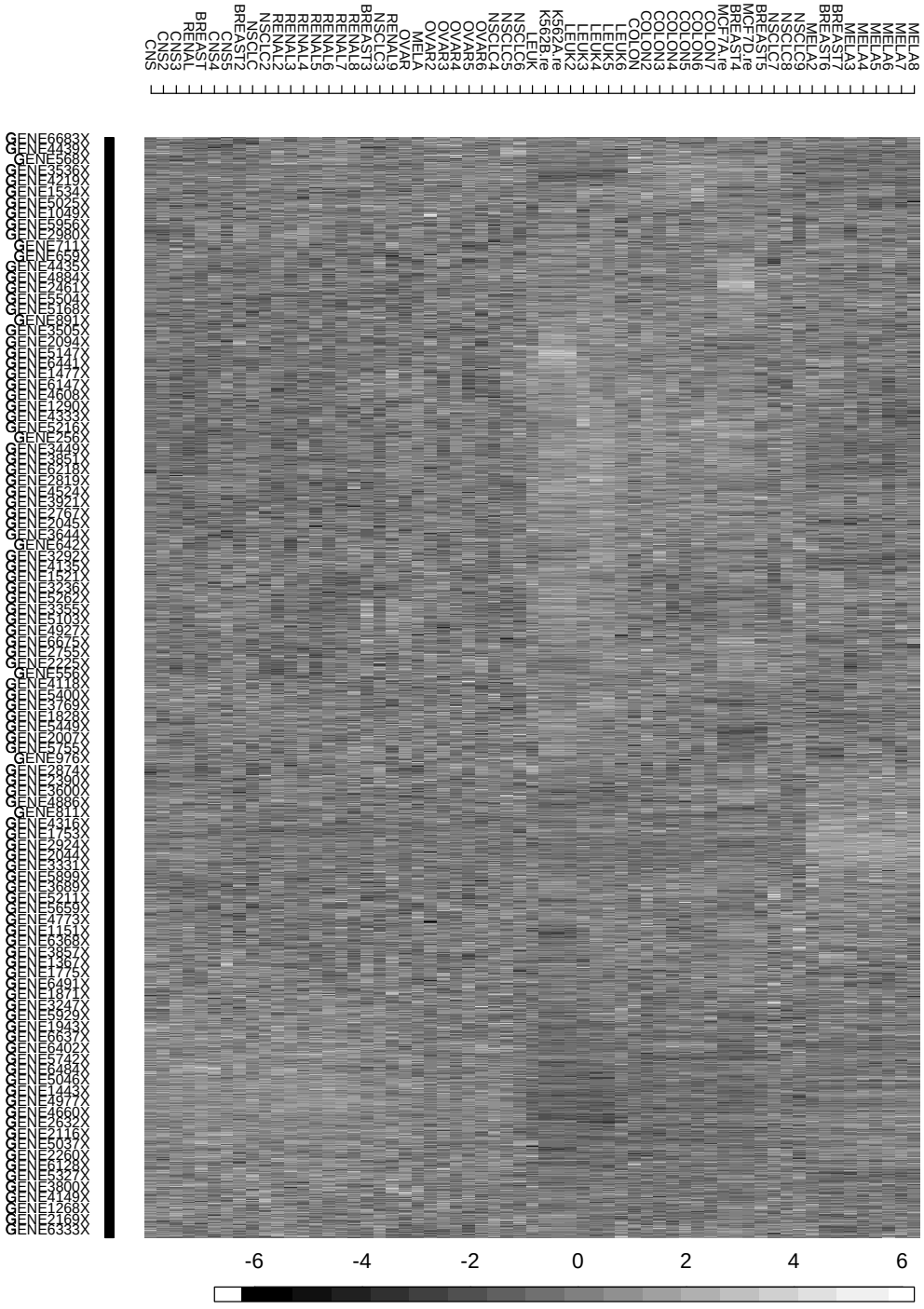
Figure 15.7. *Alon data set.* **(a)** Variance plots for the original and randomized data. The percent variance explained by each cluster, both for the original data, and for an average over twenty randomized versions. **(b)** Gap estimates of cluster size. The Gap curve corresponds to the difference between the pair of variance curves.

15.4.2 The NCI60 data set:

The second data set we analyzed is the cDNA microarray gene expression data collected by the National Cancer Institute's Developmental Therapeutics Program (DTP) (Ross et al., 2000). The gene expression represents 60 cancer cell lines (the NCI60) and contains 9,703 spotted cDNA sequences. The cell lines are derived from tumors with different sites of origin: 7 breast, 5 central nervous system (CNS), 7 colon, 6 leukemia, 8 melanoma, 9 non-small-cell-lung carcinoma (NSCLC), 6 ovarian, 2 prostate, 9 renal, and one unknown. The fluorescent cDNA targets were prepared from an mRNA sample using red dye Cy5, while the reference sample used green dye Cy3. All hybridizations were prepared by pooling equal mixtures of mRNA from 12 of the cell lines. Independent microarray experiments using a leukemia cell line (K562) and a breast cancer cell line (MCF7) were each grown in independent cultures to investigate the reproducibility of the entire experiment. As in Dudoit et al. (2001), we performed some pre-processing steps including: (i) from the 60 tumor cell lines, we excluded the two prostate cancer cell line observations and the unknown cell line observation, the remaining are labeled BREAST-BREAST7, CNS-CNS5, COLON-COLON7, LEUK-LEUK6, MELA-MELA8, NSCLC-NSCLC9, and OVAR-OVAR6, RENAL-RENAL9; (ii) retained four of the independent cell line experiments (labeled K562A, K562B), (iii) screened out genes with more than 2 missing data points; (iv) for genes with two or fewer missing data points, imputation was performed by replacing the missing entry with the average of the corresponding entries from 5 nearest neighboring genes (in terms of correlation). Our final matrix X thus consists of 5244 genes (rows) and 61 samples (columns) where x_{ji} denotes the natural logarithm of the red/green fluorescence ratio for gene j in sample i. Figure 15.8 displays the observed gene expression matrix X.

A fully supervised gene shaving analysis of the NCI data set was performed. The first four supervised clusters obtained from running **GeneClust** with a 5% Gap Tolerance based on 5 permutations are presented in Figure 15.9. Cluster 1 (86 genes) exhibits two subclusters that group most of the leukemia samples to the left while the majority of the renal tissue samples cluster at the rightmost of the heat map. Cluster 2 (33 genes), orthogonal to cluster 1, found a different set of genes that specifically over express or under express for all melanoma tissues and several breast tissues. Five genes that specifically over express for colon and NSCLC cancers are displayed in cluster 3. The fourth cluster consists of only three genes that seem to express highly for the two breast cell lines and several breast tissues.

Figure 15.8. The NCI60 human tumor expression array



15.5 Discussion

We have presented a brief description of the methodological underpinnings and implementation details of our software **GeneClust** for the clustering of microarray gene expression data by the gene shaving approach. We illustrated the simplicity of usage of our software via some simple tutorial exercises based on simulated data, as well as its application to two real published data sets. Often in practice, scientific investigators also wish to evaluate how much of the dimensionality of the gene expression is captured by the first few clusters derived from any specific method used. For gene shaving, the expression profile for each gene in the original complete data set can be explained as a linear combination of the super genes from each gene-shaved cluster. Alternatively, one can also compute the relative sum squares total for each cluster relative to that of the complete dataset, as long as the number of overlapping genes between clusters is small. The percent variance explained by the first K clusters from gene shaving can be compared with those obtained from other methods, such as a full principal component analysis, by appropriate plots (Hastie et al., 2000). In the group discrimination context, the error rates can also be compared between methods. The gene shaving method finds clusters of genes in which the gene expression varies greatly over the tissue samples while maintaining a high level of coherence between the gene expression profiles. The extracted clusters are orthogonal to each other and are of varying sizes, so that once a specific structure is captured in one cluster, the same structure will no longer be captured in subsequent clusters. However, overlapping genes are allowed between clusters if such genes induce different groupings of the columns (tissue samples). Gene shaving is also flexible in allowing the user to apply any amount of supervision required during the data analysis process. **GeneClust** is implemented with this flexibility in mind and allows interaction with the user to choose the Gap Tolerance level as well as the amount of supervision. **GeneClust** is at least a hundred-fold faster than the **GeneShaving** package implemented fully in S, and hence is recommended for real-time interactive analyses of large microarray data.

Acknowledgments

We would like to thank the programmers, Paul Roebuck and Rumiana Nikolova, for programming assistance and the development and maintenance of the **GeneClust** website. Keith Baggerly provided many insightful discussions. This work was partially funded by the National Institute of Health via the University of Texas SPORE in Prostate Cancer (CA90270) and the Early Detection Research Network grant (CA99007).

Bibliography

Alon U, Barkai N, Notterman DA, Gish K, Ybarra S, Mack D, Levine AJ (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. In: *Proceedings of the National Academy of Sciences*, volume 96, 6745–6750

Ben-Dor A, Bruhn L, Friedman N, Nachman I, Schummer M, et al. (2000). Tissue classification with gene expression profiles. *Journal Computational Biology* 7:559–584

Dudoit S, Fridlyand J, Speed TP (2001). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association* To appear

Getz G, Levine E, Domany E (2000). Coupled two–way clustering analysis of gene microarray data. *Cell Biology* 97:12079–12084

Hastie T, Tibshirani R, Eisen MB, Alizadeh A, Levy R, Staudt L, Chan W, Botstein D, Brown P (2000). 'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology* 1:research0003.1–0003.21

McLachlan GJ, Bean RW, Peel D (2002). EMMIX-GENE: A mixture-model based approach to the clustering of microarray expression data. *Bioinformatics* To appear

Ross DT, Scherf U, Eisen MB, Perou CM, Reese C, Spellman P, Iyer V, Jeffrey SS, de Rijn MV, Waltham M, Pergamenschikov A, Lee JCF, Lashkari D, Shalon D, Myers TG, Weinstein JN, Botstein D, Brown PO (2000). Systematic variation in gene expression patterns in human cancer cell lines. *Nature Genetics* 24:227–235

Scherf U, Ross DT, Waltham M, Smith LH, Lee JK, Tanabe L, Kohn KW, Reinhold WC, Myers TG, Andrews DT, Scudiero DA, Eisen MB, Sausville EA, Pommier Y, Botstein D, Brown PO, Weinstein JN (2000). A gene expression database for the molecular pharmacology of cancer. *Nature Genetics* 24:236–244

Storey J, Tibshirani R (2003). SAM thresholding and false discovery rates for detecting differential gene expression in DNA microarrays. In: *The Analysis of Gene Expression Data: Methods and Software*, chapter 12. New York: Springer

Tibshirani R, Walther G, Hastie T (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society B* 63:411–423

Tusher VG, Tibshirani R, Chu G (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Sciences* 98(9):5116–5121