International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-4 Number-2 Issue-15 June-2014

Deterministic Models for Microarray Gene Expression Profiles

V. Bhaskara Murthy[1], G. Pardha Saradhi Varma[2]

Abstract

Microarray Gene Profile studies can assess the global patterns of thousands of genes under different varying conditions. It provides important insights about the underlying genetic causes for diseases, ultimately allowing the development of modern chemical entities as medical-kit drug candidates. The informatics analysis and integration of microarray gene expression pattern are difficult for understanding or interpretation of gene array features. In this paper, we discuss the deterministic computational analysis of: the identification of differentially expressed genes using statistical methods, the discovery of gene clusters, and the classification of biological samples using standard clustering and classification approaches.

Keywords

Microarray gene expression profiles; Computational analysis; Clustering and classification

1. Introduction

Gene expression is the process by which a gene's coded information is converted into the structures as per the information present in and processing in the cell. Expressed genes include those that are transcribed into mRNA and then translated into protein, and those that are transcribed into RNA but not translated into protein. Not all genes are expressed, and gene expression involves the study of the expression level of genes in the cells under different conditions. Gene Expression Profiles using Microarrays is emerging key technology for understanding fundamental biology of gene function, development, and for discovering new classes of diseases and for understanding their molecular pharmacology.[1] Microarray technology allows expression levels of thousands of genes to be measured at the same time.

Manuscript received April 10, 2014.

V. Bhaskara Murthy, Associate Professor, Padmasri Dr. BVRICE,Vishnupur, Bhimavaram, W.G.Dt. A.P.

G. Pardha Saradhi Varma, Professor & Director PG Courses, Head, Department of IT, S.R.K.R. Engineering College, Chinamiram, Bhimavaram., W.G.Dt. A.P.

A microarray is typically a glass slide, on to which DNA molecules are attached at fixed spots. There may be tens of thousands of spots on an array, each containing a huge number of identical DNA molecules, of lengths from twenty to hundreds of nucleotides. Each of these molecules ideally should identify one gene or one exon in the genome. The spots are either printed on the microarrays by a robot, or synthesized by photolithography (as in computer chip productions), or by ink-jet printing.

Microarray studies often generate massive amounts of data, which are difficult to be examined by hand. Bioinformatics analysis and interpretation to extract genetic patterns from these data are therefore essential for gaining biological insights from experiments. The utility of computational analysis such as clustering, classification and feature selection is demonstrated by another recent study, where subtypes of diseases are successfully discovered without employing any prior biological knowledge. In this article, we describe computational methods for several common tasks in microarray studies: (1) identifying genes that experience significant changes in expression under different experimental conditions; (2) clustering of genes to identify groups of genes that are likely to be co-regulated or participating in related metabolic and regulatory pathways, (3) predicting and classifying experimental samples whether they belong to a particular type of tissue, disease or phenotype classes.

2. Identifying Differentially Expressed Genes

To identify genes differentially expressed under different conditions from cDNA microarray experiments, a heuristic approach frequently applied is to examine the ratio of fold increase/decrease of the expression levels of a gene. If the ratio is above a predefined cut-off threshold, these genes are declared to be differentially expressed, and are selected for further experimental validation. This approach is problematic, because the cut-off value is set rather arbitrarily, and it is difficult to assess the rate of false positives (unchanged genes declared differentially expressed) and rate of false negatives (missed differentially expressed genes). We discuss two

statistical methods [2, 3] that can be used in conjunction with permutation tests to identify differentially expressed genes. We begin with the lay-out of microarray data. Data from gene expression experiments can be organized as a matrix. Here, each row represents the hybridization results for a single gene across different conditions, and each column represents the measured expression levels of all genes for one condition. To draw statistical inference, it is essential to have replicated samples for each experimental condition. For identification of differentially expressed genes, we can test against the following null hypothesis: the mean expression levels xi of gene i under conditions 1 and 2 are the same. Here, we assume that there are r1 replicate samples for condition 1 and r2 replicate samples for condition 2. Simple yet more sophisticated and widely-used approach to the two class experiment is to use Student's t test to assess whether a gene is differentially expressed between biological conditions.

2.1 t- test: The basis of this test is the t statistic, which is an assessment of signal to-noise ratio for the particular gene in question, comparing its expression measure for the two conditions under study. Student's t-test is a simple method for testing whether the distributions of two variables are identical. Provided that gene expression levels under two different experimental conditions have identical Gaussian distributions, the statistics

$$t_i = (\bar{x}_{2i} - \bar{x}_{1i}) / \sqrt{\left(\frac{r_1 S_{1i}^2 + r_2 S_{2i}^2}{(r_1 + r_2 - 2)} \left(\frac{1}{r_1} + \frac{1}{r_2}\right)\right)} \rightarrow (1)$$

follows a Student's t-distribution,[4]with r1+r2-2 degrees of freedom. Here, $\bar{x}_1, \bar{x}_2$ are the mean expression levels of gene i in the r1 replicated samples of condition 1 and r2 replicated samples of condition 2, respectively; S_{1i}^2 *and* S_{2i}^2 are the sample variances of gene i under these two conditions:

$$S_i^2 = \sum \frac{(x_i - \bar{x}_i)^2}{r} \rightarrow (2)$$

If ti exceeds the threshold value for a specific confidence level (e.g. 95%), the expression levels of gene i at conditions 1 and 2 will then be considered to be different. Although a large ti value indicates that the expression levels of gene i are different under conditions 1and 2, one cannot assume the distribution of gene expression level is Gaussian or the statistic t follows a t-distribution, and therefore, one cannot obtain direct estimates of statistical confidence intervals from standard tables of t-distributions. With multiplicative samples, permutation tests can be applied to assess the statistical significance of the observed t-statistic. We randomly divide the samples into group 1 with r1 samples, and group 2 with r2 samples. The statistic t can be calculated for this grouping. Altogether there are $\left(\frac{r1+r2}{r1}\right)$ such groupings, and when plausible, we can calculate the t-statistic, denoted as t*, for each of them. An alternative approach is to sample a few thousands of such groupings. The distribution of calculated t* values can provide an estimation of the p-value $Pi*$ for the observed value of t. If we let t to be the observed t t-statistic for gene i, $tk*$ the kth permuted sample, R to be the number of permuted samples, we have the estimated P-value for observing t:

$$P_i^* = 2 \times \frac{\min\left(\sum_{k=1}^{R} abs(t_k^* \geq t),\ \sum_{k=1}^{R} abs(t_k^* \leq t)\right)}{R} \rightarrow (3)$$

The following data came from a set of Affymetrix experiments Affymetrix MOE 430A Gene Chips done by Daniel Amador-Noguez. 24 different arrays were run looking at 2 different genotypes, wild type mice and Ames Dwarf mice. The results were normalized in dChip and a one-way ANOVA (t-test) was applied. Using different p-values of 0.001 and 0.0001(without multiple testing correction), the t-test generated a certain number of significant genes. Starting with 2 biological replicates for each treatment (in total 4 arrays), a t-test was run. Each data point represents how many statistically significant, differentially expressed genes were found per number of replicates used in the analysis.

In addition, different approaches were used in terms of the assumptions made about the variance across the samples for each gene. If you assume that the variance is equal between the two different samples across every gene, then you will get a larger number of significant genes, compared to assuming that the variance is not equal across the samples. Although you will get a larger number of differentially expressed genes from assuming the variance is equal, more than likely the safer bet statistically is to assume non-equal variance.

Tabe1: Affymetrix experimental Data

Number of differentially expressed genes vs. number of replicates (ANOVA)				
#of Replicates	Variance Not Equal		Variance Equal	
	p<0.001	p<0.0001	p<0.001	p<0.0001
2	3	1	32	4
3	74	10	160	27
4	292	61	441	135
5	456	140	618	228
6	791	296	956	420
7	1128	513	1294	628
8	1315	626	1469	727
9	1766	896	1895	990
10	1928	1016	2014	1121

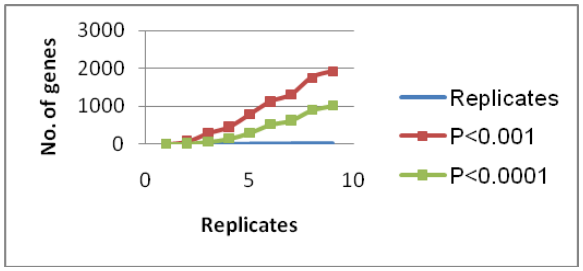

Figure 1: Number of Genes Vs Number of Replicates (Not-equal Variance)

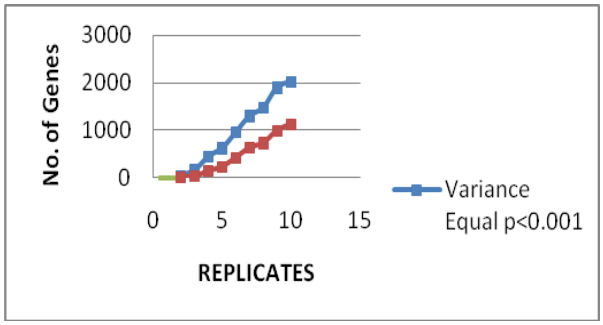

Figure 2: Number of Genes Vs Number of Replicates (Equal Variance)

Observation:
Table 1 and figure 1&2 clearly show that as you increase the number of replicates, your number of significant differentially expressed genes will also increase. Therefore, when planning microarray experiments, it is important to seriously consider the number of replicates for each treatment.

Table 2: List of Tasks – Methods (Algorithms) used for it

Task	Methods
Class Discovery	Hierarchical Clustering k-means Clustering Self-Organizing Maps Self-Organizing Trees Relevance Networks Force-directed layouts Principal Component Analysis
Class Comparison	t-test SAM Analysis of variance(ANOVA)
Class Prediction	k-nearest neighbors(KNN) Weighted Voting Artificial Neural Networks(ANN) Discriminant Analysis Classification and Regression Trees(CART) Support Vector Machines(SVM)

Table 2 provides list of tasks with the class identification, comparison, prediction and related deterministic models or algorithms that are applied to classes.

2.2 Wilcoxon test: Student's t-test is sensitive to extreme values.[2] It is often safer to use the nonparametric Wilcoxon test when there may be skewness or contamination in the gene expression data. In this test, we assume that xi is drawn from a symmetric distribution. We combine the r1 + r2 samples, and rank them in ascending order by their magnitude, and assign each sample the ranks 1, 2,. . ., r1 + r2. Next, we sum up the ranks of samples from condition 1, which will be our statistic w. To determine the significance of the P-value, the value of w can then be compared with the null model of the standard distribution of Wilcoxon rank sum values, which can be obtained by the moment generating function:

$$M(t) = \prod_{i=1}^{r_1+r_2}(e^{-it} + e^{it})/2 \rightarrow (4)$$ [12]

or more conveniently, it can be found in look-up tables in statistics[9]. With multiplicative samples, the permutation test again is more applicable to assess the statistical significance of the observed w statistic. With R permuted samples, we have the estimated P-value for observing w:

$$P_i^* = 2 \times \frac{\min((\sum_{k=1}^{R} abs(w_k \geq w),\ \sum_{k=1}^{R} abs(w_k \leq w))}{R} \rightarrow (5)$$

3. Pattern Discovery- Clustering Approach

A useful method to the analysis of microarray data is to use an unsupervised method to explore expression patterns of that exist in the data. Three of the most widely used methods are hierarchical clustering, k-means clustering, and self-organizing maps. Although each of these approaches will work with any dataset, in practice they often do not work well for large datasets where many of the genes do not vary between samples. It is useful to apply a statistical filter to the data to exclude genes which simply are not varying between experimental classes.[10] If there are no pre-determined classes in the data, a useful alternative is simply to eliminate those genes that have minimal variance across the collection of samples as those genes are not changing significantly in the dataset and are therefore the least likely to shed any light on subclasses that exist in the sample collection.

The quantitative expression levels of n genes under d conditions can be thought as n points in d-dimensional space. Clustering methods group points together those are close-by in the d dimensional space. Clustering has been shown to be very effective, in associating gene expression patterns with the ligand specificity of neurotransmitter receptors.

3.1 Distance and similarity measure

The ‘‘closeness’’ between genes becomes concrete once a distance measure or similarity measure is defined to quantitatively describe how similar or dissimilar the expression profiles of two genes are. For n genes in the microarray experiment, each pair (x, y) of the $(\frac{n}{2})$ pairs of genes can be assessed for their similarity in the expression levels under d condition. A widely used dissimilarity or distance measure is the Euclidean distance:

$$d_2(x,y) = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2} \rightarrow (6)$$

Another convenient measure is the correlation coefficients, which evaluates how correlated the expression levels of genes x and y under d different conditions:

$$R(x,y) = \sum_{i=1}^{d} \frac{(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{d}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{d}(y_i - \bar{y})^2}} \rightarrow (7)$$

The value 1-R(x, y) can also be used as a dissimilarity measure. When distances or correlations for all $(\frac{n}{2})$ pairs of genes are calculated, we obtain a n×n distance or similarity matrix, which can then be used for cluster analysis.[5,6]

3.2 Hierarchical clustering

Hierarchical clustering has become one of the most widely-used techniques for the analysis of gene expression data; it has the advantage that it is simple and the result can be easily visualized [8]. Initially, one starts with N clusters, where N is the number of genes (or samples) to be in the target dataset. Hierarchical clustering is an agglomerative approach in which single expression profiles are joined to form nodes, which are further joined until the process has been carried to completion, forming a single hierarchical tree. The algorithm proceeds in a straight forward manner:

1. Calculate the pair wise distance matrix for all of the genes to be clustered.
2. Search the distance matrix for the two most similar genes or clusters; initially each cluster consists of a single gene. This is the true first stage in the “clustering” process. If several pairs share the same degree of similarity, a predetermined rule is used to decide between alternatives.
3. The two selected clusters are merged to produce a new cluster that now contains at two or more objects.
4. The distances are calculated between this new cluster and all other clusters. There is no need to calculate all distances since only those involving the new cluster have changed.
5. Steps 2-4 are repeated until all objects are in one cluster.

There are a number of variants of hierarchical Clustering that reflect different approaches to calculate distances between the newly defined clusters and the other genes or clusters:

3.2.1. Single linkage clustering uses the shortest distance between one cluster and any other,

3.2.2. Complete linkage clustering takes the largest distance between any two clusters, and

3.2.3. Average linkage clustering uses the average distance between two clusters.

A1: Algorithm Hierarchical Clustering
repeat
find two clusters Ci and Cj
where d(Ci, Cj) = minr ≠ s d(Cr,Cs).
merge Ci, Cj into a single cluster Cq.

replace clusters Ci, Cj with Cq.
update distance matrix of new clusters.
until all genes lie in the same cluster.

In order to update the distance matrix when two clusters Ci, Cj are merged into a new cluster Cq, the key question is how to define the distance between the new cluster Cq and all other existing clusters.

In the single linkage approach, the distance of Cq to another existing cluster Cs is calculated as:

$$d(C_q, C_s) = \min\left(d(C_i, C_s), d(C_j, C_s)\right) \rightarrow (8)$$

Where d is the distance or dissimilarity measure used. In the complete linkage approach, the distance is calculated as:

$$d(C_q, C_s) = \max\left(d(C_i, C_s), d(C_j, C_s)\right) \rightarrow (9)$$

In the weighted pair group method average (WPGMA) approach:

$$d(C_q, C_s) = \frac{\left(d(C_i, C_s) + d(C_j, C_s)\right)}{2} \rightarrow (10)$$

In the unweighted pair group method average (UPGMA) approach:

$$d(C_q, C_s) = a_i * d(C_i, C_s) + a_j * (C_j, C_s) \rightarrow (11)$$

Where $a_i = \frac{|Ci|}{|Ci| + |Cj|}$ and $a_j = \frac{|Cj|}{|Ci| + |Cj|}$

Typically, the relationship between samples is represented using a Dendrogram, where branches in the tree are built based on the connections determined between clusters as the algorithm progresses. In order to visualize the relationships between samples, the dendrogram is used to rearrange the rows (or columns as appropriate) in the expression matrix to visualize patterns in the dataset. Hierarchical clustering is often misused to partition data into some number of clusters without the application of any objective criterion. Fortunately, there are a number of approaches that can be used to identify subgroups in the clustering dendrograms.

One method is to simply use the distances calculated in building the clusters as a measure of the connectivity of the individual clusters. As one moves up the dendrogram from the individual elements, the distance between clusters increases. Consequently, as one increases the distance threshold, the effective number of clusters decreases. An alternative approach is to use bootstrapping or jack-knifing techniques to measure the stability of relationships in the dendrogram, using this stability as a measure of the number of clusters represented. In bootstrapping, [11] there are a number of approaches that can be used, but the simplest is to use sampling of the dataset with replacement, each time calculating a new hierarchical clustering dendrogram and simply counting how often each branch in the dendrogram is recovered; a percentage cutoff on the dendrogram sets the number of clusters. In making a bootstrap estimate for gene cluster stability, it is appropriate to resample the collection of biological samples while in estimating the number of clusters in the biological samples, one bootstraps the gene expression vectors. Jack-knifing is similar, but instead of resampling, the appropriate vectors are sequentially left out as new dendrograms are calculated until all vectors have been considered. Once again, the stability of each cluster is estimated based on how often a given relationship in the dendrogram is recovered. One potential problem with many hierarchical clustering methods is that, as clusters grow in size, the expression vector that represents the cluster when calculating distance may no longer represent any of the genes within the cluster. Consequently, as clustering progresses, the actual expression patterns of the genes themselves become less relevant. Furthermore, if a bad assignment is made early in the process, it cannot be corrected. An alternative, which can avoid these artifacts, is to use a divisive clustering approach, such as k-means, to partition data (either genes or samples) into groups having similar expression patterns.

3.3 K-means Clustering

If there is advance knowledge regarding the number of clusters that should be represented in the data, k-means clustering is a good alternative to hierarchical methods. In K-means, objects are partitioned into a fixed number (k) of clusters such that the clusters are internally similar but externally dissimilar. No dendrograms are produced, but one could use hierarchical techniques on each of the data partitions after they are constructed. The process involved in k-means clustering is conceptually simple, but can be computationally intensive:

1. All initial objects are randomly assigned to one of k clusters (where k is specified by the user).
2. An average expression vector is then calculated for each cluster and this is used to compute the distances between clusters.
3. Using an iterative method, objects are moved between clusters and intra and inter-cluster distances are measured with each move. Objects are allowed to

remain in the new cluster only if they are closer to it than to their previous cluster.
4. Following each move, the expression vectors for each cluster are recalculated.
5. The shuffling proceeds until moving any more objects would make the clusters more variable, increasing intra-cluster distances and decreasing inter-cluster dissimilarity.

Some implementations of k-means clustering allow not only the number of clusters to be specified, but also seed cases for each cluster. This has the potential to allow one to use prior knowledge of the system helps to define the cluster output, such as a typical profile for a few key genes known to distinguish classes. Of course, the "means" in K-means refers to the use of a mean expression vector for each emerging cluster. As one might imagine, there are variations that also use other measures, such as K-medians clustering.

It has the advantage that no strict phylogenetic relationship is enforced on every gene, as is in hierarchical clustering, which can be problematic because there is no absolute ancestral relationship in expression patterns. In this method, genes are classified as belonging to one of the k clusters. Cluster membership is determined by calculating the centers a_1, a_2, a_3 . ., .$a_k \in \mathbb{R}$ for each gene cluster, and assigning each gene i according to its expression profile xi to the cluster with the closest centroid. The goal is to find empirically optimal cluster centers a_1, a_2, a_3 . ., .a_k such that the empirical error

$$E = \frac{1}{n}\sum_{i=1}^{n} \min_{1 \le j \le k} \| x_i - a_j \|^2 \rightarrow (12)$$

is minimized. This is achieved through an iterative approach:
Algorithm k-Means Clustering
i: = 0
Assign k initial centers $a_1^{(0)}$,. . ., $a_k^{(0)}$ arbitrarily;
Repeat
cluster genes x_1,. . .,x_n to k clusters
for x_j, $j \in$ a[1,. . .,n]
if $\|x_j - a_m\|^2$ $\| x_j - a_l \|^2$ l m
Assign x_j to the m-th cluster
update cluster centers
$a_m^{(i+1)} = \sum_{j:xj} \in C_m^{(i)} X_j / |C_m^{(i)}|$
i: = i+1
Until no changes in the cluster centers.

3.4 Self Organizing Maps

A self-organizing map (SOM) is a neural network-based divisive clustering approach. A SOM assigns genes to a series of partitions based on the similarity of their expression vectors to reference vectors that are defined for each partition. It is the process of defining these reference vectors that distinguishes SOMs from k-means clustering.[20] Prior to initiating the analysis, the user defines a geometric configuration for the partitions, typically a two-dimensional rectangular or hexagonal grid. Random vectors are generated for each partition, but before genes can be assigned to partitions, the vectors are first "trained" using an iterative process that continues until convergence so that the data are most effectively separated:[16]
1. Random vectors are constructed and assigned to each partition.
2. A gene is picked at random and, using a selected distance metric, the reference vector that is closest to the gene is identified.
3. The reference vector is then adjusted so that it is more similar to the randomly picked gene. The reference vectors that are nearby on the two dimensional grid are also adjusted so that they too are more similar to the randomly selected gene.
4. Steps 2 and 3 are iterated several thousand times, decreasing the amount by which the reference vectors are adjusted and increasing the stringency used to define closeness in each step. As the process continues, the reference vectors converge to fixed values.
5. Finally, the genes are mapped to the relevant partitions depending on the reference vector to which they are most similar.
In choosing the geometric configuration for the clusters,[21] the user is, effectively, specifying the number of partitions into which the data are to be divided. As with k-means clustering, the user has to rely on some other sources of information, such as principal component analysis (PCA), to determine the number of clusters that best represents the available data.

1. Initialization – Choose random values for the initial weight vectors wj
2. Sampling – Draw a sample training input vector x from the input space.
3. Matching – Find the winning neuron I(x) that has weight vector closest to the input vector, i.e. the minimum value of $d_j(x) = \sum_{i=1}^{D} (x_i - w_{ji})^2$
4. Updating – Apply the weight update equation
$\Delta w_{ji} = \eta(t) T_{j, I(x)}(t)(x_i - w_j)$

Where $T_{j, I(x)}(t)$ is a Gaussian neighbourhood and $\eta(t)$ is the learning rate.
5. Continuation – keep returning to step 2 until the feature map stops changing.

4. Classifying biological samples predictors and classifiers

As mentioned earlier, some microarray experiments do not focus on identifying function, but rather on finding genes that can be used to group samples into biologically or clinically relevant classes and supervised approaches to data analysis are particularly useful for these studies. One typically begins with a priori knowledge of the groups represented in the data, although any hypothesis along these lines can be further explored using clustering techniques and other information. With those groups, one then asks whether there are genes that can be used to separate the relevant classes. For two groups of samples, a t-test or unpaired two-class Significance Analysis of Microarrays SAM are useful tools while, for a larger number of classes, ANOVA or multi-class SAM are appropriate. Having identified a set of genes that show significant differences, one then builds a classification algorithm that can be used to assign a new sample to one of the classes.[11]
There are a wide range of algorithms that have been used for classification, including weighted voting, artificial neural networks,[17] discriminant analysis, classification and regression trees, support vector machines, k-nearest neighbours, and a host of others. Essentially, each of these uses an original set of samples – a training set – to develop a rule that takes a new test sample from a test set and uses its expression vector sample, trimmed to a previously identified set of classification genes, to place this test sample into the context of the original sample set, thus identifying its class.
In many ways, KNN is the simplest approach to doing classification. First, one must assemble a collection of expression vectors for our samples and assign the samples to various experimental classes. We will refer to these samples, about which we have prior knowledge, as our training set. Next, genes are selected that separate the various classes using an appropriate statistical test to identify good classification candidate genes, thus reducing the size of the sample classification vectors.[18] This represents a first-pass collection of classification genes. The next step is to identify and eliminate samples that appear to be outliers. These may be important because they possibly represent new subclasses in our original sample classification set; alternatively, they may just represent poor-quality data. The outlying samples are identified by applying a correlation filter to the reduced sample expression vectors, as follows:
1. The Pearson correlation coefficient (r) is computed between a given vector and each member of the training set; the maximum r identified is called the rmax for that vector. The vector is randomized a user specified number of times. Each time, an rmax is calculated using the randomized vector (called r*max), just as in Step 1.
2. The fraction of times r*max exceeds rmax over all randomizations is used to calculate a p-value for that vector.
3. If the p-value for a vector is less than a user-specified threshold (meaning it is well-correlated with other samples), that vector is retained for further analysis. Otherwise, it is discarded.
Steps 1-3 are repeated for every sample vector in the set.
At this point, the training set has led to the generation of a collection of sample vectors that represent prior knowledge regarding the biological classes represented in the data. The next step in the analysis involves assigning new samples from the test set to classes, based on their expression vectors. For each sample in the test set, its expression vector is reduced to include only those genes previously identified as being significant for classification. The distance between this reduced expression vector and the reduced expression vectors is then computed for each and every sample in the training set. As the name KNN implies, some number k of nearest neighbours is chosen from the training set – those k vectors that have the smallest distances from the test sample. The new test vector is then assigned to the class most highly represented in its k nearest neighbours. If there is a tie, the new sample remains unclassified.

KNN is one of the simplest nonlinear classifiers that have found practical use in many applications. To classify a biological sample j of unknown phenotype, we calculate its distance based on its expression profile yj to all of the dt training set samples, where classifications are known. We then look for the k nearest neighbour samples to the dt training set samples. The class for each of the k nearest neighbours is then identified, and the unknown sample is assigned to the class where the majority of the k neighbours belong.

4.1 Classifiers based on Gaussian distribution

We begin with a simple parametric model for describing microarray data. Gaussian distribution is a convenient model for studying a wide variety of physical processes. The probability density function of a univariate Gaussian distribution takes the following familiar form:

$$p(x) = \left(\frac{1}{\sqrt{2\pi\sigma}}\right) exp^{\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)} \rightarrow (13)$$

where is the mean and 2 is the variance. Its generalization is the multivariate Gaussian distribution[12,14] $\mathcal{N}(\mu,\Sigma)$ where $\mu \in \mathbb{R}^d$ is the mean vector and ∑ is the covariance matrix:

$$\Sigma = E[(x-\mu)(x-\mu)^T \rightarrow (14)$$

and its probability density function is:

$$p(x) = \frac{1}{\sqrt{2\,\pi}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) \rightarrow (15)$$

$x \in \mathbb{R}^d$

When classifying biological samples [13] where each sample belongs to exactly one of the P classes, we can evaluate the probability that sample j belongs to a specific class K:

P (y_j, K) =P (y_j | K)π(K) →(16) here, $y_j \in \mathbb{R}$ is the vector representing the global expression profile of all n genes from sample j, i.e. it is a column vector in the n X d data matrix. π(K) is the prior probability that any given sample belongs to class K1, and P(yj | K) is the conditional probability of observing yj from a sample of class K. Assume that the pdf of P(yj | K) is a Gaussian distribution $\mathcal{N}(\mu_K,\sum_K)$, several classifiers can be developed with different additional assumptions [7,8].

4.1.1 Quadratic classifier. If we can assess the joint probability P(y_j,K) for the global expression profile of all n genes in condition j for every class K P, we can simply classify sample j into the class with the highest probability P(y_j,K). Technically, it is more convenient to work with the log transformed discriminant function g_k:

$$g_K = \ln[P(y_j,K)\pi(K)] \rightarrow (17)$$

When P(y_j, K) follows a Gaussian distribution,

$$g_K = -\frac{1}{2}(X-\mu)\sum_K^{-1}(X-\mu) + \ln P(K) + \text{Const.} \rightarrow (18)$$

The first term on the right hand side is quadratic. Standard techniques can be applied to calculate this term, for example, by using Moore–Penrose pseudo inverse.

4.1.2 Linear classifier. When all classes have the same covariance matrix, i.e. $\sum_i = \sum$, the discriminant function is:

$$g_K = -\frac{1}{2}(X-\mu)^T \sum^{-1}(X-\mu_k) + \ln P(K) + \text{Const.} \rightarrow (19)$$

Here, the quadratic term becomes the same for all classes, and the boundary between classes becomes linear.

4.1.3 Diagonal classifier. When the covariance matrices for all classes are the same, and when the expression levels of all genes are uncorrelated, the covariance matrices are all diagonal K= diag(σ_1^2,…. (σ_n^2). The discriminant function in this case is simple:

$$g_k = -\frac{1}{2}\sum_{i=1}^{n}(x_i-\mu_{ik})^2/\sigma_i^2) \rightarrow (20)$$

The quadratic classifier is the most sophisticated among the three, and it is provable that it is the optimal classifier for Gaussian distributions. Quadratic classifier involves estimating mean vectors μ_1, μ_2 and covariance matrices $\sum_1$, $\sum_2$ altogether n(n + 3)/2 parameters. Estimating these parameters with high accuracy is necessary for constructing good discriminant rule, because the calculation of the inverse matrices $\sum_1^{-1}$, $\sum_2^{-1}$ are often ill-conditioned. Estimating the high-dimensional covariance matrices requires a large amount of data. In contrast, simpler classifiers, such as the diagonal linear classifier, require only the estimation of order O(n) number of parameters.

The quadratic classifier is the most sophisticated among the three, and it is provable that it is the optimal classifier for Gaussian distributions. Quadratic classifier involves estimating mean vectors μ_1, μ_2 and covariance matrices $\sum_1$, $\sum_2$ altogether n(n + 3)/2 parameters. Estimating these parameters with high accuracy is necessary for constructing good discriminant rule, because the calculation of the inverse matrices $\sum_1^{-1}$, $\sum_2^{-1}$ are often ill-conditioned. Estimating the high-dimensional covariance matrices requires a large amount of data. In contrast, simpler classifiers, such as the diagonal linear classifier, require only the estimation of order O(n) number of parameters.

Maximize $L(\alpha)=\sum_i\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j . x_i x_j$

with constraints $\sum_i\alpha_i y_i=0$ and $\alpha_i \geq 0$ →(21)

Because this is an optimization problem on convex set, the solution found is automatically guaranteed to be the global solution. This offers an important advantage not shared by other classifiers such as the

neural network, where one often encounters the problem of local optimum in the training phase.
Because a large data set is involved, solving the quadratic programming efficiently is crucial in developing an effective SVM classifier. Recent development in subset selection methods such as Sequential Minimal Optimization (SMO) allows practical implementation of SVM for solving classification problems involving very large data sets such as gene array data.

Another major bioinformatics challenge of microarray analysis is the global integration of microarray studies [19] of different tissues and cell lines under various different conditions from different investigators. Yet another challenge is to integrate microarray expression profiles with other bioinformatics analyses, for examples, the detection of membrane proteins as potential markers, the discovery of previously unknown biological roles by combining expression studies and the detection of sequence/structure function motifs, as well as integration with pharmacological studies. Ultimately, the integration of gene expression under various conditions with the analysis of multiple bioinformatics tools will help to tease out various components of regulatory and metabolic genetic networks of cells.

During the past few years, there have been many discussions in the literature on "noise" in microarray says: disparate results arising from the use of different platforms, questions regarding the validity of microarray results, and the need to validate the findings.[15] If one closely examines the underlying issues, it is clear that microarrays are no different than any other approach to assaying levels of gene expression each method has its own biases and limitations. What is underlying all of these issues is trying to understand what can be done with the data that emerges. Although there are no absolute answers, there are some overarching generalizations that can be made that will help guide the follow-on experiments. First, whether one is trying to identify genes that can be used for sample classification, what microarray assays generally give us are lists of genes that can be significantly correlated with some classes in our experiments. These should be treated not as truths, but as hypotheses that can be tested. Second, statistical significance is fine, but biological significance is better. Statistics provides very powerful tools for identifying candidate genes, for prioritizing them in the lack of any other evidence, and in helping to resolve features in the data.

5. Conclusion

In this paper we are used both Statistical and Computer Science methods to represent use Micro array Profiles in a deterministic way. Using this as a basis one can apply for it any disorder or disease data to get clear understanding of microarray profiles. It can be extended to incorporate many more methods to estimate and analyse in a specified time to take an early action on the genes that cause a particular disorder that causes the disease.

Acknowledgement

This paper provides information for the use of expression profiles as biomarkers to predict disease prognosis and advanced future applications are sure to follow. We are still eagerly awaiting the outcome of this study.

References

[1] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek,J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A.Calgiuri, C.D. Bloomfield, E.S. Lander, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 286 (1999) 531–537.

[2] M.-L.T. Lee, F.C. Kuo, G.A. Whitmore, J. Sklar, Importance of replication in microarray gene expression studies: statistical methods and evidence from repetitive cDNA hybridizations, Proc. Natl. Acad. Sci. U. S. A. 97 (2000) 9834–9839.

[3] Hoffmann R, Seidl T, Dugas M. Profound effect of normalization on detection of differentially expressed genes in oligonucleotide microarray data analysis. Genome Biol 2002;3(7),RESEARCH0033.

[4] R.V. Hogg, A.T. Craig, Introduction to Mathematical Statistics, Prentice-Hall, Upper Saddle River, New Jersey, 1995.

[5] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E.Dmitrovsky, E.S. Lander, T.R. Golub, Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation, Proc. Natl. Acad. Sci. U. S. A. 96 (1999) 2907– 2912.

[6] J.A. Hartigan, Clustering Algorithms, Wiley, 1975.

[7] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis,Wiley, 1973.

[8] P.J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, J. Comput. Appl. Math. 20 (1987) 53–65.
[9] W.N. Venables, B.D. Ripley, Modern Applied Statistics with S-Plus, Springer, 1999.
[10] K. Zhang, H. Zhao, Assessing reliability of gene clusters From gene expression data, Funct. Integr. Genomics 1 (2000) 156–173.
[11] M.K. Kerr, G.A. Churchill, Bootstrapping cluster analysis: assessing the reliability of conclusions from microarray experiments, Proc. Natl. Acad. Sci. U. S. A. 98 (2001) 8961–8965.
[12] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.
[13] C. Hu, J. Liang, Classification of biologically active compounds by support vector machines, J. Chem. Inf. Comput.Sci., Chemometrics and Intelligent Laboratory Systems 95 (2009) 188–198.
[14] J.S. Liu, C.E. Lawrence, Bayesian inference on biopolymer models, Bioinformatics 15 (1999) 38–52.
[15] J.S. Liu, Monte Carlo Strategies in Scientific Computing, Springer-Verlag, New York, 2001.
[16] Toronen P, Kolehmainen M, Wong G, Castren E.Analysis of gene expression data using self organizing maps. FEBS Lett 1999; 451(2): 142-6.
[17] Herrero J, Valencia A, Dopazo J. A hierarchical unsupervised growing neural network for clustering gene expression patterns. Bioinformatics 2001;17(2): 126-36.
[18] Antoniadis A, Lambert-Lacroix S, Leblanc F. Effective dimension reduction methods for tumor classification using gene expression data. Bioinformatics 2003; 19(5): 563-70.
[19] Doniger SW, Salomonis N, Dahlquist KD, Vranizan K, Lawlor SC, Conkli, BR. MAPPFinder: using Gene Ontology and GenMAPP to create a global gene-expression profile from microarray data. Genome Biol 2003; 4(1): R7.
[20] Michaels GS, Carr DB, Askenazi M, Fuhrman S, Wen X, Somogy, R. Cluster analysis and data visualization of large-scale gene expression data.Pac Symp Biocomput 1998;: 42-53.
[21] M.B. Eisen, P.T. Spellman, P.O. Brown, D. Botstein, Cluster analysis and display of genome-wide expression patterns, Proc. Natl. Acad. Sci. U. S. A. 95 (1998) 14863–14868.

V. Bhaskara Muthry was born in Akividu, West Godavari Dt., Andhra Pradesh in 1968. He received B.Sc(MPC), PGDCA and M.Sc.(CS) from Banaras Hindu University in 1992. He has a teaching experience of 22 years and presently working as Assoc. Professor in Padmasri Dr. B.V.Raju Institute of Computer Education, Vishnupur, W.G.Dt.,A.P. He has publications in international journals IJERT, IJARCS. Springer. He is pursuing his Ph.D. from Acharya Nagarjuna University, Guntur. A.P.

Dr. G. Pardha Saradhi Varma has obtained UG B.tech(CSE), M.Tech(CSE) and PhD CST in Distinction. He has teaching experience of 22 years and 8 years in research. He has professional memberships with CSI and ISTE. He has 62 publications in national journals and 88 publications in International journals. He is an author 8 text books. Presently he is Professor & Director PG Courses, Head, Department of IT, S.R.K.R. Engineering College, Chinamiram, Bhimavaram., W.G.Dt. A.P.