

## Interactive Visualization and Analysis for Gene Expression Data

Chun Tang, Li Zhang and Aidong Zhang

Department of Computer Science and Engineering

The State University of New York at Buffalo

Buffalo, NY 14260

{chuntang, lizhang, azhang}@cse.buffalo.edu

### Abstract

New technology such as DNA microarray can be used to produce the expression levels of thousands of genes simultaneously. The raw microarray data are images which can be transformed into gene expression matrices where usually the rows represent genes, the columns represent various samples, and the number in each cell characterizes the expression level of the particular gene in a particular sample. Now the cDNA and genomic sequence projects are processing at such a rapid rate, more and more data become available to researchers who are working in the field of bioinformatics. New methods are needed to efficiently and effectively analyze and visualize the gene data.

A key step in the analysis of gene expression data is the detection of groups that manifest similar expression patterns and filter out the genes that are inferred to represent noise from the matrix according to samples distribution. In this paper, we present a visualization method which maps the samples'  $n$ -dimensional gene vectors into 2-dimensional points. This mapping is effective in keeping correlation coefficient similarity which is the most suitable similarity measure for analyzing the gene expression data. Our analysis method first removes noise genes from the gene expression matrix by sorting genes according to correlation coefficient measure and then adjusts the weight for each remaining gene. We have integrated our gene analysis algorithm into a visualization tool based on this mapping method. We can use this tool to monitor the analysis procedure, to adjust parameters dynamically, and to evaluate the result of each step. The experiments based on two groups of *multiple sclerosis* (MS) and treatment data demonstrate the effectiveness of this approach.

# 1 Introduction

DNA microarray technology can be used to measure expression levels for thousands of genes in a single experiment, across different conditions and over time [3, 16, 14, 24, 23, 27, 36, 20, 21, 8, 30]. To use the arrays, labeled cDNA is prepared from total messenger RNA (mRNA) of target cells or tissues, and is hybridized to the array. The amount of label bound is an approximate measure of the level of gene expression. Thus gene microarrays can give a simultaneous, semi-quantitative readout on the levels of expressions of thousands of genes. Just 4-6 such high-density “gene chips” could allow rapid scanning of the entire human library for genes which are induced or repressed under particular conditions. By preparing cDNA from cells or tissues at intervals following some stimulus, and exposing each to replicate microarrays, it is possible to determine the identity of genes responding to that stimulus, the time course of induction, and the degree of change.

Some methods have been developed using both standard cluster analysis and new innovative techniques to extract, analyze and visualize gene expression data generated from DNA microarrays. Gene expression matrix can be studied in two dimensions [2]: comparing expression profiles of genes by comparing rows in the expression matrix [22, 6, 26, 23, 25, 9, 33, 3] and comparing expression profiles of samples by comparing columns in the matrix [?, 10, 32]. In addition, both methods can be combined (provided that the data normalization [18, 38] allows it).

A key step in the analysis of gene expression data is the detection of groups that manifest similar expression patterns and filter out the genes that are inferred to represent noise from the matrix according to samples distribution. The corresponding algorithmic problem is to cluster multi-condition gene expression patterns. Data clustering [6] was used to identify patterns of gene expression in human mammary epithelial

cells growing in culture and in primary human breast tumors. DeRisi et al. [17] used a DNA array containing a complete set of yeast genes to study the diauxic shift time course. They selected small groups of genes with similar expression profiles and showed that these genes are functionally related and contain relevant transcription factor binding sites upstream of their open reading frames (ORFs). In [3], a clustering algorithm was introduced for analysis of gene expression data in which an appropriate stochastic error model on the input has been defined. Self-organizing maps [26, 19], a type of mathematical cluster analysis that is suited for recognizing and classifying features in complex, multidimensional data, was applied to organize the genes into biologically relevant clusters that suggest novel hypotheses about hematopoietic differentiation. In [12], the authors presented a strategy for the analysis of large-scale quantitative gene-expression measurement data from time-course experiments. The correlated patterns of gene expression from time series data suggests an order that conforms to a notion of shared pathways and control processes that can be experimentally verified. Brown et al. [23] applied a method based on the theory of support vector machines (SVMs). The method is considered as a supervised computer learning method because it exploits prior knowledge of gene function to identify unknown genes of similar function from expression data. They applied this algorithm on six functional classes of yeast gene expression matrices from 79 samples [22]. Alter et al. [25] used singular value decomposition in transforming genome-wide expression data from genes  $\times$  arrays space to reduced diagonalized “eigengenes”  $\times$  “eigenarrays” space to extract significant genes by normalizing and sorting the data. Hastie et al. [33] proposed a *tree harvesting* method for supervised learning from gene expression data. This technique starts with a hierarchical clustering of genes, then models the outcome variable as a sum of the average expression profiles of chosen clusters and their products. The method can discover genes that have strong effects on their own, and genes that interact with other genes.

On sample dimension, our task is to build a classifier which can predict the sample labels from the

expression profile. Golub et al. [32] applied neighborhood analysis to construct class predictors for samples, especially for leukemias. They were looking for genes whose expression data are best correlated with two known classes of leukemias, acute myeloid leukemia and acute lymphoblastic leukemia. They constructed a weighted vote classifier based on 50 genes (out of 6817) using 38 samples and applied it to a collection of 34 new samples. The classifier correctly predicted 29 of the 34 samples. In [10], the authors present a neural network model known as Simplified Fuzzy ARTMAP which can identify normal and diffuse large B-cell lymphoma (DLBCL) patients using DNA microarrays data generated by a previous study. Many traditional clustering algorithms such as the hierarchical [22, 15, 5] and K-means clustering algorithms [13, 28] have all been used for clustering expression profiles. Mathematical and statistical methods like Fourier and Bayesian analysis also have been used to discover profiles of cell cycle-dependent genes [31, 37, 1]. Our group has developed a maximum entropy approach to classifying gene array data sets [29]. We used part of pre-known classes of samples as training set and applied the maximum entropy model to generate an optimal pattern model which can be used to new samples.

Sample clustering has been combined with gene clustering to identify which genes are the most important for sample clustering [34, 5]. Alon et al. [34] have applied a partitioning-based clustering algorithm to study 6500 genes of 40 tumor and 22 normal colon tissues for clustering both genes and samples. Getz et al. [11] present a method applied on colon cancer and leukemia data. By identifying relevant subsets of the data, they were able to discover partitions and correlations that were masked and hidden when the full dataset was used in the analysis. This method is called two-way clustering.

*Multiple sclerosis* (MS) is a chronic, relapsing, inflammatory disease. *Interferon- $\beta$*  ( $IFN - \beta$ ) has been the most important treatment for the MS disease for the last decade [35]. The DNA microarray technology makes it possible to study the expression levels of thousands of genes simultaneously. The gene expression

levels are measured by the intensity levels of the corresponding array spots. In this paper, we present a visualization method which maps the samples'  $n$ -dimension gene vectors into 2-dimensional points. This mapping is effective in keeping correlation coefficient similarity which is the most suitable similarity measure for analyzing gene expression data. Our gene expression data analysis method first removes noise genes from the expression matrix by sorting genes according to correlation coefficient measure and then adjusts the weight for each remaining gene. We have integrated our gene analysis algorithm into a visualization tool based on this mapping method. We can use this tool to monitor the analysis procedure, to adjust parameters, and to evaluate the result of each step. The experiments on the healthy control, MS and IFN-treated samples based on the data collected from the DNA microarray experiments demonstrate the effectiveness of this approach.

This paper is organized as follows. Section 2 introduces the visualization method. Section 3 describes the details of our gene data analyzing approach with the help of our visualization tool. Section 4 presents the experimental results. And finally, the conclusion is provided in Section 5.

## **2 Visualization Tool**

### **2.1 Mapping Method**

A typical gene expression matrix has thousands of rows (each row represents a gene) and several (usually less than 100) columns which represent samples related to a certain kind of disease or other condition. Each sample is marked with a label that points out which class it belongs to, such as control, patient and drug-treated. While we analyze these samples, we usually want to view the sample distribution during each step. How to visualize arbitrarily large dimensional data effectively and efficiently is still an open problem. The parallel coordinate system allows the visualization of multidimensional data by a simple two dimensional

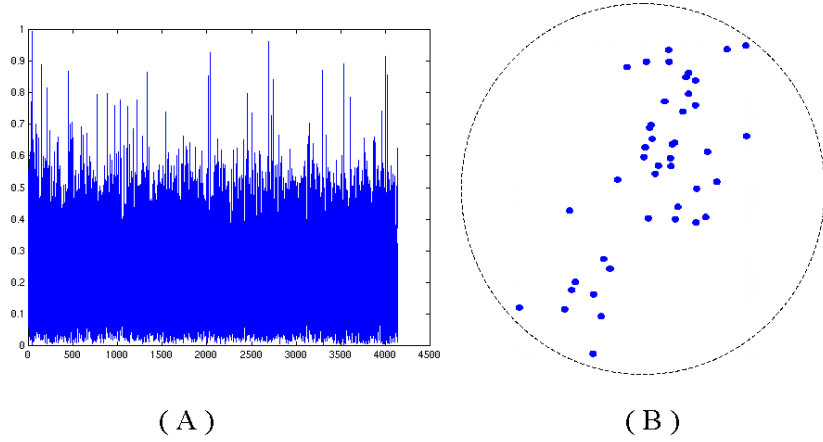


Figure 1: Distribution of our gene expression data of 44 samples, where each samples has 4132 genes. (A) Visualization using the parallel coordinate system, where the horizontal axis represents gene dimension and each multidimensional line represent a sample. The sample distribution is not clear. (B) Visualization using our tool, where each point represents a sample. The figure shows that after mapping original data into 2-dimensional space, sample distribution is clearly represented.

representation. But as the dimensions go higher, the displaying is less effective. Figure 1 shows an example of different visualization methods.

We use the idea of a linear mapping method that maps the multidimensional dataset to 2-dimensional space [7]. Let vector  $\vec{P}_g = (x_{g1}, x_{g2}, \dots, x_{gn})$  represent a data point in  $n$ -dimensional space, and total number of points in the space is  $m$ , denoted as  $P_1, P_2, \dots, P_m$ .

We use the Formula (1) to map  $\vec{P}_g$  into a 2-dimensional point  $\vec{P}_g^*$ :

$$\vec{P}_g^* = \sum_{i=1}^n (w_i * (4/n) * (x_{gi})) \vec{S}_i, \quad (1)$$

where  $w_i$  is an adjustable weight for each coordinate,  $n$  is vector length of the original space,  $4/n$  is a ratio to centralize the points, and  $\vec{S}_i$  ( $i = 1, 2, \dots, n$ ) are unit vectors which divide the center circle of the display screen equally (Figure 2 (B)).

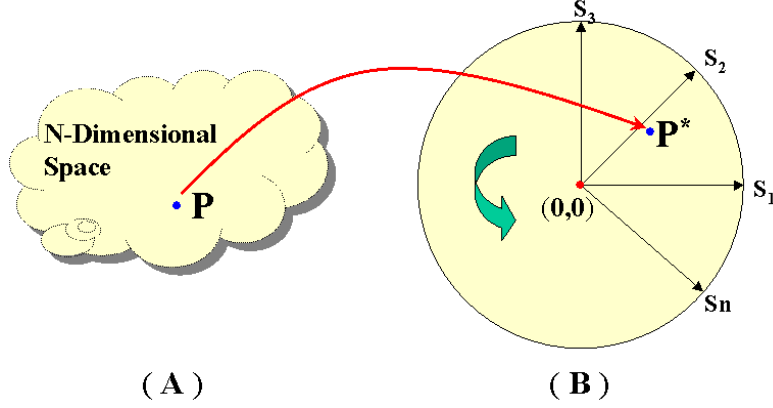


Figure 2: Mapping from  $n$ -dimensional space to 2-dimensional space. (A) A data point  $P$  in the original space. (B)  $P^*$  is the corresponding point using mapping function (Formula (1)) in the 2-dimensional displaying space. The red point marked as  $(0,0)$  is the center of the displaying screen,  $\vec{S}_i (i = 1, 2, \dots, n)$  are unit vectors which divided the unit circle of the display screen equally.

Our initial setting is  $w_i = 0.5$  for all  $i = 1, 2, \dots, n$ , which means each coordinate of the original space contributes equally in the initial mapping. Under this setting, we can easily figure out that point  $(0,0,\dots,0)$  in the original  $n$ -dimensional space will be mapped to  $(0,0)$  which is the center of 2-dimensional displaying space based on mapping function (Formula (1)). In addition, a point which has the format of  $(a,a,\dots,a)$  will also be mapped to the center (Figure 3 (A)). Another property under the initial setting is keeping the correlation coefficient similarly of the original data vectors. Because the correlation coefficient has the advantage of depending only on shape but not on the absolute magnitude of the spatial vector, it is a better similarity measure than Euclidean distance [32, 4]. The formula of correlation coefficient between two vectors  $\vec{X}$  and  $\vec{Y}$  is:

$$co(\vec{X}, \vec{Y}) = \frac{n * (\sum_{i=1}^n x_i * y_i) - (\sum_{i=1}^n x_i) * (\sum_{i=1}^n y_i)}{\sqrt{[n * (\sum_{i=1}^n x_i^2) - (\sum_{i=1}^n x_i)^2][n * (\sum_{i=1}^n y_i^2) - (\sum_{i=1}^n y_i)^2]}}, \quad (2)$$

where

$$\vec{X} = (x_1, x_2, \dots, x_n),$$

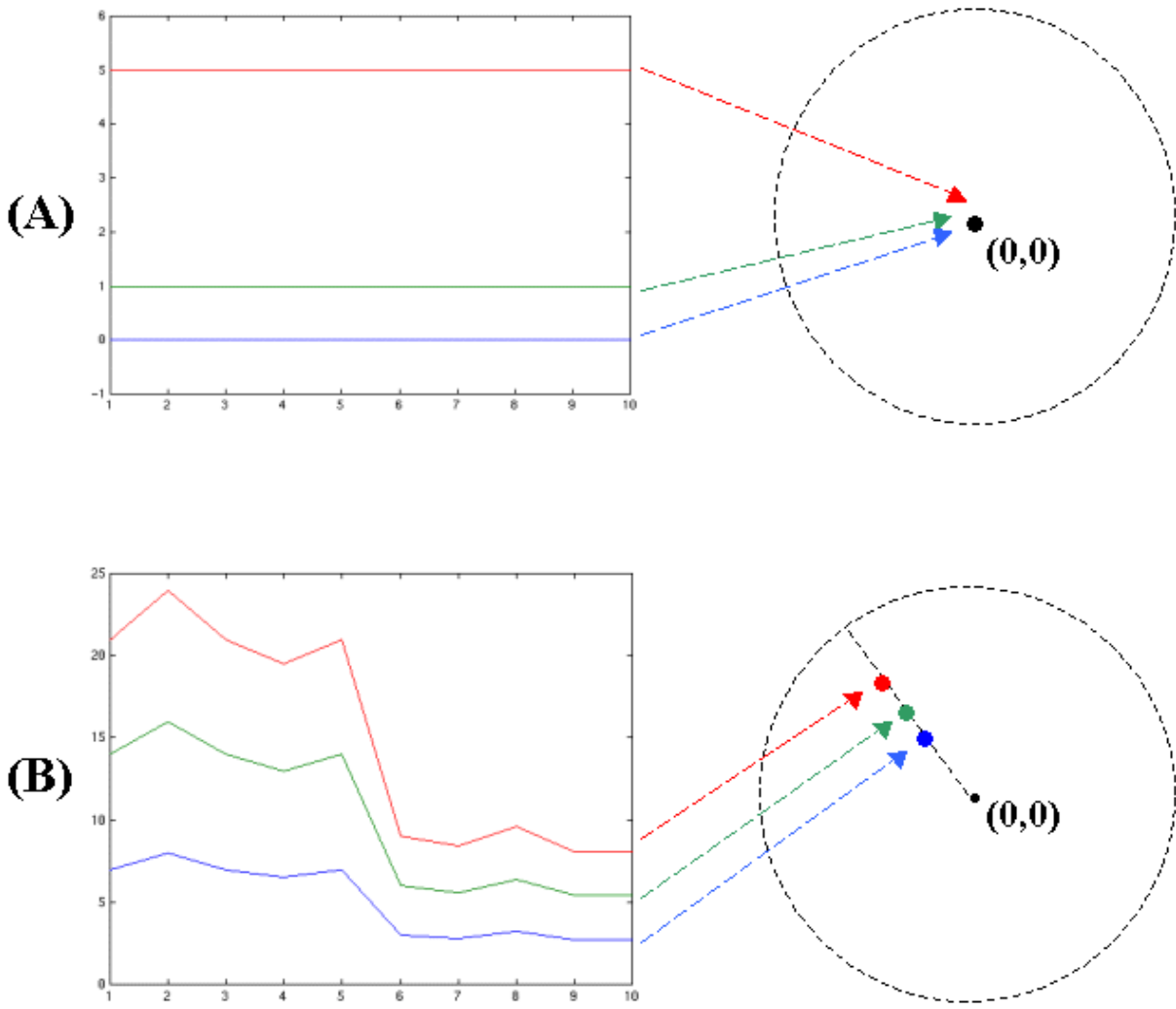


Figure 3: Some properties of our mapping function. (A) Shows every point whose coordinates in the original space can be represented as  $(a,a,\dots,a)$  will be mapped to the center of the 2-dimensional displaying space. (B) Shows points that have the same pattern, which means ratios of each pairs of coordinates in the original space are all equal, will be mapped onto a straight line across the center in the 2-dimensional displaying space.

$$\vec{Y} = (y_1, y_2, \dots, y_n),$$

$$\sum_{i=1}^n x_i * y_i = \text{sum of the products of paired scalar},$$

$$\sum_{i=1}^n x_i = \text{sum of } x \text{ scalar},$$

$$\sum_{i=1}^n y_i = \text{sum of } y \text{ scalar},$$

$$\sum_{i=1}^n x_i^2 = \text{sum of squared } x \text{ scalar},$$

$$\sum_{i=1}^n y_i^2 = \text{sum of squared } y \text{ scalar}.$$

If  $\vec{X}$  and  $\vec{Y}$  have the same pattern, which means ratios of each pairs of coordinates of  $\vec{X}$  and  $\vec{Y}$  are all equal (Equation (3)), their correlation coefficient values will be 1. Using our mapping function, these two vectors will be mapped onto a straight line across the center in the 2-dimensional displaying space, and all other vectors which have the same pattern as  $\vec{X}$  and  $\vec{Y}$  will all be mapped onto that line (Figure 3 (B)), even if their Euclidean distances in the original space are very large.

$$\frac{x_1}{y_1} = \frac{x_2}{y_2} = \frac{x_3}{y_3} = \dots = \frac{x_n}{y_n}. \quad (3)$$

## 2.2 Parameter Adjustment

Our visualization tool allows the user to adjust the weight of each coordinate from  $-1$  to  $1$  to change data distribution in the displaying space, both manually and automatically. Mapping from a higher dimensional space to a lower dimensional space may not preserve all the properties of the dataset. By adjusting the coordinate weights of the dataset, data's original static state is changed into dynamic state which may be used to compensate the information loss from mapping. For example, two points  $(100, 100, \dots, 100)$  and  $(0, 0, \dots, 0)$  are far away in the original space, but by the initial setting, they are both mapped into the center

of the 2-dimensional displaying screen. Whenever any weight  $w_i$  in Formula (1) is changed, these two points will be separated. That is,  $(0, 0, \dots, 0)$  will be still at the center but  $(100, 100, \dots, 100)$  will no longer be mapped to the center. Figure 4 shows another example where the original dataset has 268 points (vectors) which can be divided into two clusters, marked as hollow red circles and filled blue circles. While mapping based on the initial setting, the cluster boundary is not clear enough: some parts of the two clusters are overlapped. After changing the weights of some coordinates, the data distribution is also changed, and the two clusters are separated from each other.

Since the original dimensions may be very high, such as several thousands, manually changing the weight for each dimension to find the best combination is impractical. That is the reason why our tool supports automatic changing of weights. The user only needs to set the adjustment direction (ascending or descending) and changing step for each dimension. The tool will perform an animation to show the data distribution while the weights are changed automatically. The user can obtain all the weights when the ideal distribution is reached.

### **3 Data Analyzing Approach**

Based on the gene expression matrix which has thousands of genes and several dozens of samples, our task is to build a classifier to predict the sample labels (classes) from the expression profile using the information already known from the experiment, such as diseased/normal attributes of the samples. A very common method is [32] first to reduce the number of genes in the gene dimension, which means to find important genes that are more related to such kind of idealized patterns; then to assign some weights from the first step to the remaining genes to construct a “class predictor”. How many genes are deleted is usually from experience. Researchers usually give a fixed number to every matrix, but the best number might be different

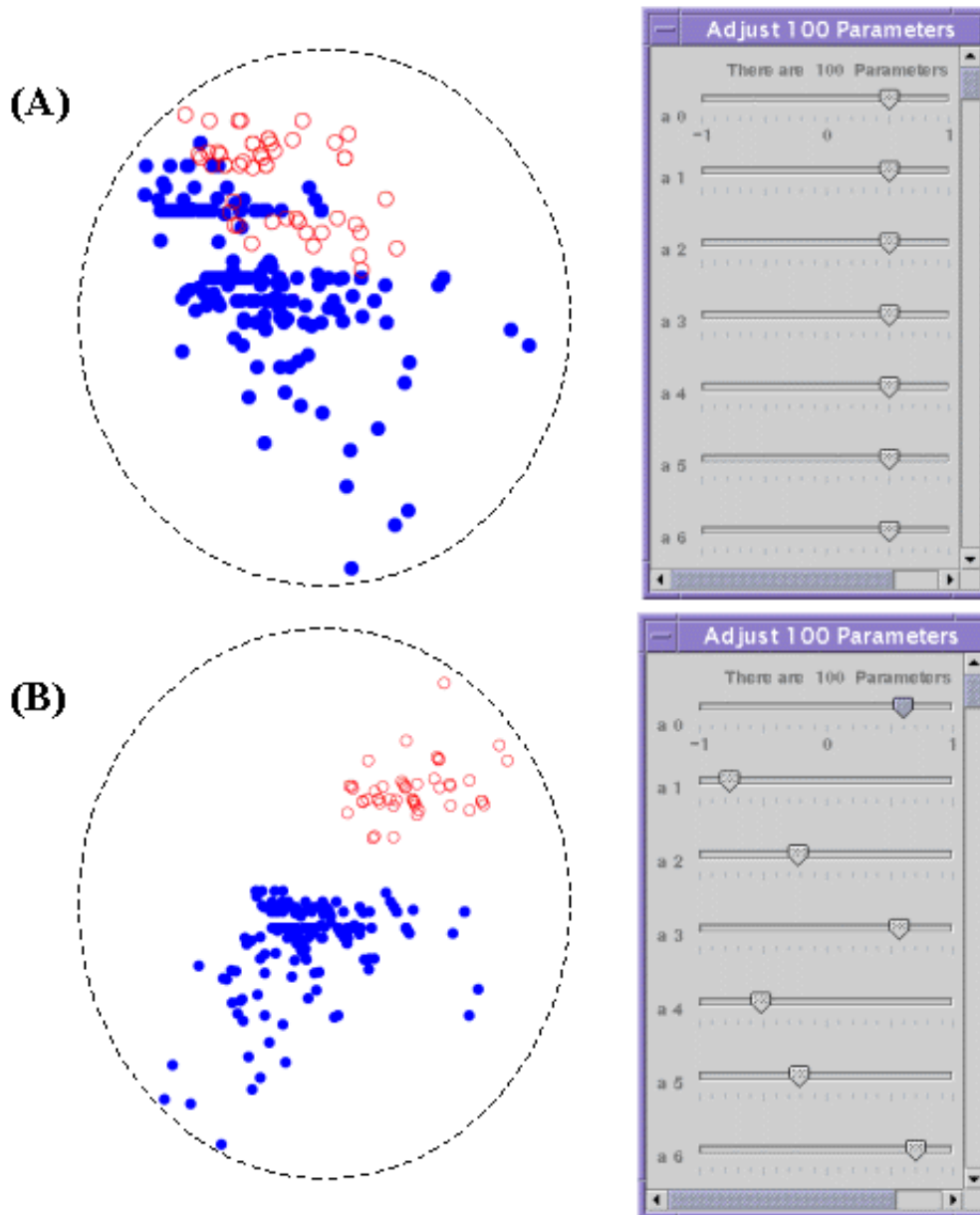


Figure 4: Effect of weights adjustment. Left circles show the data distribution in displaying space, right side slides show the weight adjustment environment. (A) Shows mapping result of a dataset using the initial setting. The dataset include two clusters. We marked as hollow red circles and filled blue circles. (B) After changing the weights of some coordinates, the data distribution is also changed, and two clusters are separated from each other.

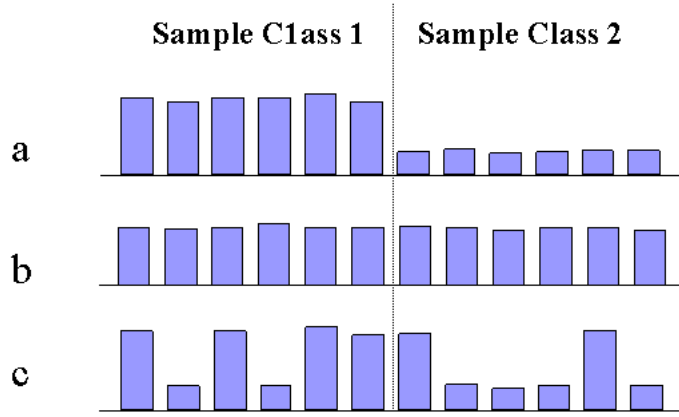


Figure 5: Different kinds of gene patterns. Assume that samples have two classes. If we want to construct a “class predictor” from the gene matrix, genes have pattern like (a) give the correct information which are called “important gene”. Pattern (b) means useless genes. Pattern (c) is noise in the dataset. So our task is to select (a), remove (b) and (c) from the set of genes.

for each dataset. How to choose the best weight is another open problem.

### 3.1 Normalization

In the gene expression matrix, different genes have different ranges of intensity values. The intensity values alone may not have significant meaning, but the relative changing levels are more intrinsic. So we should first normalize the original gene intensity values into relative changing levels. Our general formula is

$$CL_{gi} = (x_{gi} - \mu_g * t) / (\mu_g * t), \quad \text{where } \mu_g = \sum_{i \in S} x_{gi} / |S|, \quad (4)$$

where  $CL_{gi}$  denotes changing level for gene  $g$  of sample  $i$ ,  $x_{gi}$  represents the original intensity value for gene  $g$  of sample  $i$ ,  $t$  is a parameter, and  $\mu_g$  is the mean of the intensity values for gene  $g$  for all samples  $S$ .

### 3.2 Selecting Important Genes

Notice that among thousands of genes, not all of them have the same contribution in distinguishing the classes. Actually, some genes have little contribution or just represent noise from the matrix (Figure 5). We need to remove those genes.

Assuming there are  $n$  genes and  $m$  samples, where each gene vector (after normalization) is denoted as

$$X_g = (x_{g1}, x_{g2}, \dots, x_{gk}, x_{g(k+1)}, \dots, x_{gm}), \quad \text{where } g = 1, 2, \dots, n \text{ for each gene.} \quad (5)$$

Without losing generality, we assume the first  $k$  samples belong to one class while the remaining samples belong to another class. The ideal gene which is highly correlated with the samples distribution should match the pattern  $\vec{P} = (1, 1, \dots, 1, 0, \dots, 0)$  (first  $k$  number is “1” followed by  $(m-k)$  “0”) or  $\vec{Q} = (0, 0, \dots, 0, 1, \dots, 1)$  (first  $k$  number is “0” followed by  $(m-k)$  “1”). We calculate correlation coefficient (Formula (2)) between each gene vector and the pre-defined stable pattern  $\vec{P}$ , then sort genes using these correlation coefficients by a descending sequence which exactly matches the ascending sequence if sorting by correlation coefficients with another pattern  $\vec{Q}$ . We know a certain number of genes from the top and the bottom of this sequence should be chosen as the “important genes”, but the number is usually decided from the experience. Also the best number for different datasets is different. Using our tool, we can conduct flexible judgment. First, we map the whole dataset into 2-dimensional displaying space. We then remove “unimportant” genes from the middle of the sequence one by one. When the boundary of the clusters of the samples could be clearly separated, we stop this procedure. The remaining genes are chosen as “important” genes (Figure 6). The whole procedure is integrated into our visualization tool. Because our mapping function has the property of preserving correlation coefficient similarity, samples with similar patterns will be mapped close to each other.

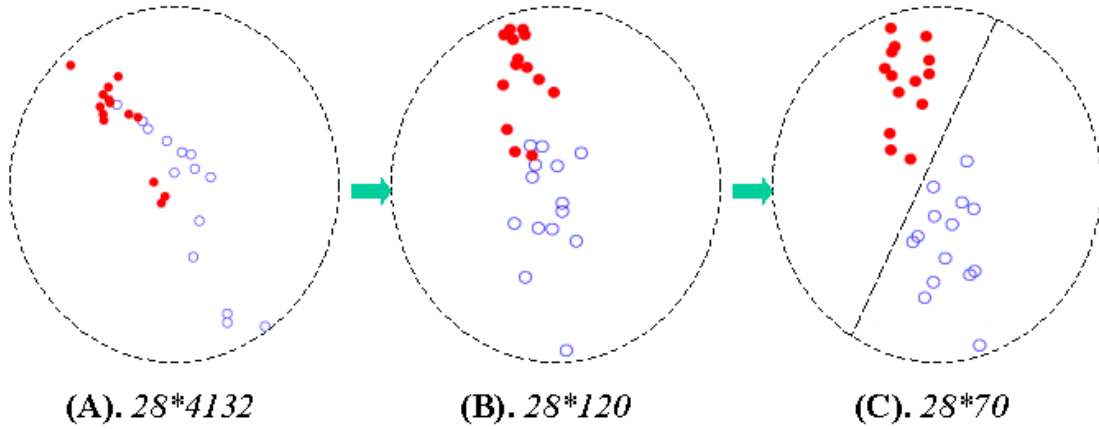


Figure 6: Procedure of removing “unimportant” genes based on a gene expression data of 28 samples which belong to two clusters. (A) Shows original distribution of 28 samples mapping from 4132 genes vectors. (B) Samples distribution reduced to 120 genes. (C) Samples distribution reduced to 70 genes. There is a clear boundary between two clusters.

### 3.3 Weight Adjustment

After selecting a subset of important genes, the next step is to build a class predictor based on these genes.

Usually this predictor has the format of:

$$(w_1 g_1, w_2 g_2, \dots, w_k g_k), \quad (6)$$

where  $g_i$  is the value of the selected gene of the sample to classify and  $w_i$  is weight of the selected gene.

Now the problem is how to decide the weight of each gene. We can directly use the correlation coefficient with the pattern  $\vec{P} = (1, 1, \dots, 1, 0, \dots, 0)$  or  $\vec{Q} = (0, 0, \dots, 0, 1, \dots, 1)$  as the weight, or get from the parameter adjustment function of our tool. By using our tool, we can manually adjust the weights of each gene as illustrated in Figure 4, or let the tool to perform automatic adjustment. For automatic adjustment, we present a measure that evaluates the quality of the distribution of two clusters, and use this measure to decide when to stop the adjustment procedure.

First we define the *center* of each sample cluster as  $\vec{CS}$  using Formula (7):

$$\vec{CS} = (cs_1, cs_2, \dots, cs_n), \quad \text{where } cs_i = \frac{\sum_{s \in S} x_{si}}{n}. \quad (7)$$

Then for each sample cluster, we define a value  $G(S)$  to measure the gather degree of all the points in this cluster using Formula (8):

$$G(S) = \sqrt{\frac{1}{m-1} \sum_{s \in S} | \vec{s} - \vec{CS} |^2} = \sqrt{\frac{1}{m-1} \sum_{s \in S} \sum_{i=1}^n (s_i - cs_i)^2}, \quad (8)$$

where  $m = |S|$  is the number of samples in this cluster.

This measure is similar to the standard deviation in one dimensional space. If the distribution of points in sample cluster  $S$  is sparse,  $G(S)$  will be large, otherwise it is small. If we have two clusters of samples denoted as  $S_1$  and  $S_2$ , we hope each cluster gather together as well as the distance between the two clusters is as large as possible. So we present another measure to describe the quality of the cluster distribution as Formula (9):

$$c = \frac{G(S_1) + G(S_2)}{|CS_1 - CS_2|}, \quad (9)$$

where  $CS_1$  is the center of  $S_1$ ,  $CS_2$  is the center of  $S_2$ , and  $|CS_1 - CS_2|$  is the distance between  $CS_1$  and  $CS_2$ .

We hope the distance between the two clusters can be as large as possible and the sparse degree can be as low as possible. So the best distribution is reached when  $c$  value is the smallest. We apply this measure on the 2-dimensional displaying space to evaluate the samples distribution while adjusting weights. Notice that to try the combinations of all the weights is impractical because of exponential time complexity. What we can do is when the points distribution reaches a local lowest value of  $c$ , we stop the adjust procedure and set combination of the weights.

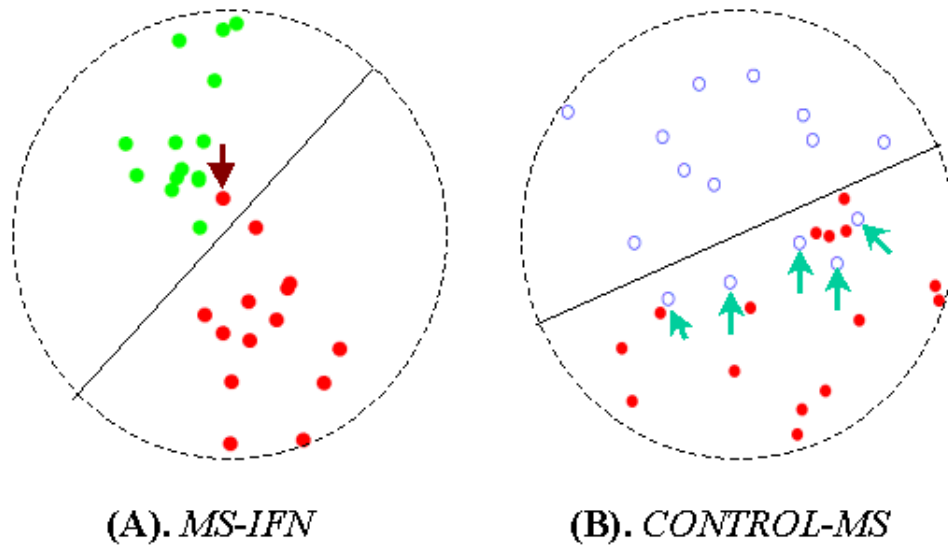


Figure 7: Experiment result for MS\_IFN group and CONTROL\_MS group. Red circles denote samples belong to MS group, green circles represent samples in IFN group while blue circles mean CONTROL samples. Samples pointed by arrows are wrongly classified. Straight lines across the big circle denote “class predictor” which are built using the cross-validation method.

## 4 Experimental Results

The experiments are based on two data sets: the MS\_IFN group and the CONTROL\_MS group. The MS\_IFN group contains 28 samples (14 MS samples and 14 IFN samples) while the CONTROL\_MS group contains 30 samples (15 control samples and 15 MS samples). Each sample has 4132 genes.

During the data processing procedure, by sorting genes using the correlation coefficients of Formula (2), we select 70 genes (Figure 6 (C)) for MS\_IFN group and 58 genes for CONTROL\_MS group. We then adjust weights for the genes in these two groups using automatic adjustment function in our tool.

We use K-means clustering method to evaluate the “weighted class predictor” which we get from the analyzing procedure. Here we choose the cross-validation method [32] to evaluate each group. In each

group, choose a sample, use the remaining samples of this group to select important genes, and get class predictor. Then predict the class of the withheld sample. The process is repeated for each sample, and the cumulative error rate is calculated. For MS\_I group, samples in the IFN group were all predicted correctly but one sample in the MS group was incorrectly classified. For the CONTROL\_MS group, samples in the MS group were all predicted correctly, but five samples in the CONTROL group were wrongly classified. Figure 7 shows the evaluation result of our “weighted class predictor” for these two groups using the cross-validation method.

## 5 Conclusion

In this paper, we presented a visualization method which maps the samples’  $n$ -dimensional gene vectors into 2-dimensional points. This mapping is effective in keeping correlation coefficient similarity which is the most suitable similarity measure for analyzing gene expression data. Our analysis method first removes noise genes from the gene expression matrix by sorting genes according to correlation coefficient measure, and then adjusts the weight for each remaining gene. We also presented a measure to judge the quality of the cluster distribution. We have integrated our gene analysis algorithm into a visualization tool based on the mapping method. We can use this tool to monitor the analysis procedure, to adjust parameters dynamically, and to evaluate the result of each step of adjustment. Our approach takes the advantage of data analysis and dynamic visualization methods to reveal correlated patterns of gene expression data. In particular, we used the above approach to distinguish the healthy control, MS, IFN-treated samples based on the data collected from DNA microarray experiments. From our experiments, we demonstrated that this approach is a promising approach to be used for analysis and visualization of gene array data sets.

## References

- [1] A. Ben-Dor, N. Friedman, and Z. Yakhini. Class discovery in gene expression data. In *Proc. Fifth Annual Inter. Conf. on Computational Molecular Biology (RECOMB 2001)*, 2001.
- [2] Alvis Brazma and Jaak Vilo. Minireview: Gene expression data analysis. *Federation of European Biochemical societies*, 480:17–24, June 2000.
- [3] Amir Ben-Dor, Ron Shamir and Zohar Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3/4):281–297, 1999.
- [4] Anna Jorgensen. Clustering excipient near infrared spectra using different chemometric methods. Technical report, Dept. of Pharmacy, University of Helsinki, 2000.
- [5] Ash A. Alizadeh, Michael B. Eisen, R. Eric Davis, Chi Ma, Izidore S. Lossos, Adreas RosenWald, Jennifer C. Boldrick, Hajeer Sabet, Truc Tran, Xin Yu, John I. Powell, Liming Yang, Gerald E. Marti et al. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, Vol.403:503–511, February 2000.
- [6] Charles M. Perou, Stefanie S. Jeffrey, Matt Van De Rijn, Christia A. Rees, Michael B. Eisen, Douglas T. Ross, Alexander Pergamenschikov, Cheryl F. Williams, Shirley X. Zhu, Jeffrey C. F. Lee, Deval Lashkari, Dari Shalon, Pat rick O. Brown, and David Bostein. Distinctive gene expression patterns in human mammary epithelial cells and breast cancers. *Proc. Natl. Acad. Sci. USA*, Vol. 96(16):9212–9217, August 1999.

- [7] D. Bhadra and A. Garg. An interactive visual framework for detecting clusters of a multidimensional dataset. Technical Report 2001-03, Dept. of Computer Science and Engineering, University at Buffalo, NY., 2001.
- [8] D. Shalon, S.J. Smith, P.O. Brown. A DNA microarray system for analyzing complex DNA samples using two-color fluorescent probe hybridization. *Genome Research*, 6:639–645, 1996.
- [9] Elisabetta Manduchi, Gregory R. Grant, Steven E. McKenzie, G. Christian Overton, Saul Surrey and Christian J. Stoeckert Jr. Generation of patterns form gene expression data by assigning confidence to differentially expressed genes. *Bioinformatics*, Vol. 16(8):685–698, 2000.
- [10] Francisco Azuaje Department. Making genome expression data meaningful: Prediction and discovery of classes of cancer through a connectionist learning approach, 2000.
- [11] Gad Getz, Erel Levine and Eytan Domany. Coupled two-way clustering analysis of gene microarray data. *Proc. Natl. Acad. Sci. USA*, Vol. 97(22):12079–12084, October 2000.
- [12] G.S. Michaels, D.B. Carr, M. Askenazi, S. Fuhrman, X. Wen and R. Somogyi. Cluster Analysis and data visualization of large-scale expression data. In *Pac Symposium of Biocomputing*, volume 3, pages 42–53, 1998.
- [13] Hartigan J.A. *Clustering Algorithm*. John Wiley and Sons, New York., 1975.
- [14] J. DeRisi, L. Penland, P.O. Brown, M.L. Bittner, P.S. Meltzer, M. Ray, Y. Chen, Y.A. Su, J.M. Trent. Use of a cDNA microarray to analyse gene expression patterns in human cancer. *Nature Genetics*, 14:457–460, 1996.

- [15] Javier Herrero, Alfonso Valencia, and Joaquin Dopazo. A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics*, 17:126–136, 2001.
- [16] J.J. Chen, R. Wu, P.C. Yang, J.Y. Huang, Y.P. Sher, M.H. Han, W.C. Kao, P.J. Lee, T.F. Chiu, F. Chang, Y.W. Chu, C.W. Wu, K. Peck. Profiling expression patterns and isolating differentially expressed genes by cDNA microarray system with colorimetry detection. *Genomics*, 51:313–324, 1998.
- [17] J.L. DeRisi, V.R. Iyer and P.O. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, pages 680–686, 1997.
- [18] Johannes Schuchhardt, Dieter Beule, Arif Malik, Eryc Wolski, Holger Eickhoff, Hans Lehrach and Hanspeter Herzel. Normalization strategies for cDNA microarrays. *Nucleic Acids Research*, Vol. 28(10), 2000.
- [19] T. Kohonen. *Self-Organization and Associative Memory*. Spring-Verlag, Berlin, 1984.
- [20] M. Schena, D. Shalon, R.W. Davis, P.O. Brown. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*, 270:467–470, 1995.
- [21] Mark Schena, Dari Shalon, Renu Heller, Andrew Chai, Patrick O. Brown, and Ronald W. Davis. Parallel human genome analysis: Microarray-based expression monitoring of 1000 genes. *Proc. Natl. Acad. Sci. USA*, Vol. 93(20):10614–10619, October 1996.
- [22] Michael B. Eisen, Paul T. Spellman, Patrick O. Brown and David Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, Vol. 95:14863–14868, 1998.

- [23] Michael P. S. Brown, William Noble Grundy, David Lin, Nello Cristianini, Charles Sugnet, Terrence S. Furey, Manuel Ares and Jr.David Haussler. Knowledge-based analysis of microarray gene expression data using support vector machines. *Proc. Natl. Acad. Sci.*, 97(1):262–267, January 2000.
- [24] O. Ermolaeva, M. Rastogi, K.D. Pruitt, G.D. Schuler, M.L. Bittner, Y. Chen, R. Simon, P. Meltzer, J.M. Trent, M.S. Boguski. Data management and analysis for gene expression arrays. *Nature Genetics*, 20:19–23, 1998.
- [25] Orly Alter, Patrick O. Brown and David Bostein. Singular value decomposition for genome-wide expression data processing and modeling. *Proc. Natl. Acad. Sci. USA*, Vol. 97(18):10101–10106, August 2000.
- [26] Pablo Tamayo, Donna Solni,m Jill Mesirov, Qing Zhu, Sutisak Kitareewan, Ethan Dmitrovsky, Eric S. Lander and Todd R. Golub. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. USA*, Vol. 96(6):2907–2912, March 1999.
- [27] R.A. Heller, M. Schena, A. Chai, D. Shalon, T. Bedilion, J. Gilmore, D.E. Woolley, R.W. Davis. Discovery and analysis of inflammatory disease-related genes using cDNA microarrays. *Proc. Natl. Acad. Sci. USA*, 94:2150–2155, 1997.
- [28] S. Tavazoie, D. Hughes, M.J. Campbell, R.J. Cho and G.M. Church. Systematic determination of genetic network architecture. *Nature Genet*, pages 281–285, 1999.
- [29] Shumei Jiang, Chun Tang, Li Zhang and Aidong Zhang , Murali Ramanathan. A maximum entropy approach to classifying gene array data sets. In *Proc. of Workshop on Data mining for genomics, First SIAM International Conference on Data Mining*, 2001.

- [30] S.M. Welford, J. Gregg, E. Chen, D. Garrison, P.H. Sorensen, C.T. Denny, S.F. Nelson. Detection of differentially expressed genes in primary tumor tissues using representational differences analysis coupled to microarray hybridization. *Nucleic Acids Research*, 26:3059–3065, 1998.
- [31] Spellman P.T., Sherlock G., Zhang M.Q., Iyer V.R., Anders K., Eisen M.B., Brown P.O., Botstein D., Futcher B. . Exploring the metabolic and genetic control of gene expression on a genomic scale. *Mol. Biol. Cell*, page 3273, 1998.
- [32] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gassenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, D.D. Bloomfield and E.S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, Vol. 286(15):531–537, October 1999.
- [33] Trevor Hastie, Robert Tibshirani, David Boststein and Patrick Brown. Supervised harvesting of expression trees. *Genome Biology*, Vol. 2(1):0003.1–0003.12, January 2001.
- [34] U. Alon, N. Barkai, D.A. Notterman, K.Gish, S. Ybarra, D. Mack and A.J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide array. *Proc. Natl. Acad. Sci. USA*, Vol. 96(12):6745–6750, June 1999.
- [35] V. Yong, S. Chabot, Q. Stuve and G. Williams. Interferon beta in the treatment of multiple sclerosis: mechanisms of action. *Neurology*, 51:682–689, 1998.
- [36] V.R. Iyer, M.B. Eisen, D.T. Ross, G. Schuler, T. Moore, J.C.F. Lee, J.M. Trent, L.M. Staudt, Jr. J. Hudson, M.S. Boguski, D. Lashkari, D. Shalon, D. Botstein, P.O. Brown. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283:83–87, 1999.

- [37] Y Barash and N Friedman. Context-specific bayesian clustering for gene expression data. *Bioinformatics, RECOM01*, 2001.
- [38] Yang Y.H., Dudoit S., Luu P. and Speed T. P. Normalization for cDNA Microarray Data. In *Proceedings of SPIE BiOS 2001*, San Jose, California, January 2001.