Gene Expression Data Clustering and Visualization Based on a Binary Hierarchical Clustering Framework

Lap Keung SZETO[1], Alan Wee-Chung LIEW[1], Hong YAN[1,2] and Sy-sen TANG[1]

1 Department of Computer Engineering and Information Technology
City University of Hong Kong
83 Tat Chee Avenue, Kowloon, Hong Kong
2 School of Electrical and Information Engineer University of Sydney,
NSW 2006, Australia

E-mail: 50129934@student.cityu.edu.hk

Abstract

We describe the use of a binary hierarchical clustering (BHC) framework for clustering of gene expression data. The BHC algorithm involves two major steps. Firstly, the K-means algorithm is used to split the data into two classes. Secondly, the Fisher criterion is applied to the classes to assess whether the splitting is acceptable. The algorithm is applied to the sub-classes recursively and ends when all clusters cannot be split any further. BHC does not require the number of clusters to be known. It does not place any assumption about the number of samples in each cluster or the class distribution. The hierarchical framework naturally leads to a tree structure representation. We show that by arranging the BHC clustered gene expression data in a tree structure, we can easily visualize the cluster results. In addition, the tree structure display allows user judgement in finalizing the clustering result using prior biological knowledge.

Keywords: Gene expression data analysis, K-means clustering, Fisher linear discriminant, binary hierarchical clustering framework.

1 Introduction

In DNA Microarray technology, gene expression data can reveal many meaningful biological processes, for example, gene response to drug treatments, cancer diagnosis, etc. In gene expression data analysis, the data are arranged in a matrix form, where the rows correspond to genes and the columns correspond to the genes' responses under different experimental conditions. Hence, one can examine the expression profiles of different genes by comparing rows in the expression matrix, or study the responses of genes to different experimental conditions by examining the columns of the expression matrix. Gene expression data can be analysed in two ways: unsupervised and supervised analysis [10]. In supervised analysis, information about the structure/groupings of the objects is assumed known, or at least partially known. This prior knowledge is then used in the analysis process. However, in many situations, for example, for a new drug treatment, prior knowledge about the expression data is not available. In this case, the number of meaningful groupings, as well as their structures needs to be inferred directly from the given data in an unsupervised manner.

Cluster analysis is a powerful tool in the study of gene expression data. In unsupervised clustering, a metric is defined such that objects with similar properties are grouped together. Many clustering algorithms, for example, K-means, Self-Organizing Maps (SOM), Hierarchical clustering, Self-Organizing Tree Algorithm, Principal Component Analysis, and Multi-Dimensional Scaling, have all been applied to the study of high-dimension gene expression data. However, in most of these clustering algorithms, the number of clusters needs to be set by the user [2]. Even for nonpartition-based algorithm such as Hierarchical clustering, the number of classes is determined by cutting the tree structure at certain level, and the choice of this level is subjected to user's judgement and experience. Unfortunately, for many gene expression data, we usually have no idea about the number of clusters present in the data or their distributions and would prefer that to be estimated from the data themselves.

In this paper, we use the Binary Hierarchical (BHC) clustering algorithm for gene expression data analysis. Our algorithm is based on the idea of hierarchical subdivision of data as proposed in [12]. The BHC algorithm systematically subdivides the data into compact groups based on the Fisher criterion. In this way, the number of clusters is estimated automatically from the data itself, instead of been pre-specified by the user. This paper is divided into six sections. Section 2 describes the use of singular value decomposition (SVD) to filter expression data. Section 3 describes in detail the BHC framework for clustering the normalized gene expression data. Due to the high dimension of the expression data and the possibly very small number of genes in a particular class, singular matrix may be encountered during the Fisher linear discriminant computation. A method to handle this situation is proposed. Section 4 describes how the clustered gene expression data are used to build up a tree structure to visualize the relationship between groups. Experimental results about the performance of BHC on a set of real gene expression data is presented in Section 5, and finally, Section 6 draws the conclusions.

2 Singular Value Decomposition

Singular Value Decomposition (SVD) is used to reduce the data dimensionality and to filter the gene expression dataset. SVD can transform genome-wide expression data from gene × array space (a high dimension space) to a reduced, diagonalized "eigengenes × eigenarrays" space. It can be used to filter noise in expression data without eliminating genes and arrays from the dataset, and to interpolate missing values in the dataset [5, 6].

SVD is applied as a linear transformation of gene expression data. The expression data is an M (genes) × N (arrays) matrix A whose number of rows M is usually greater than the number of columns N. The matrix A can be SVD transformed to be the product of an M × N eigengene matrix U, an N × N diagonal matrix D with positive or zero elements, and the transpose of an N × N eigenarray matrix V, i.e.,

$$A = UDV^T \tag{1}$$

with

$$U^T U = I\,,\; V^T V = I\,,$$
$$D = dig(d_1 \geq d_2 \geq ....d_M \geq 0)$$

Although gene expression data are usually of very high dimension, most of the interesting variations are captured in the first few SVD components with large eigenvalues. By discarding SVD components with small eigenvalues, dimension reduction and suppression of noise in the data can be achieved. We chose to retain the first three elements from the diagonal matrix D and set all the other elements of D to zero. The SVD processed data are then use for BHC clustering.

3 The BHC Algorithm

The BHC algorithm provides a binary hierarchical framework to define a wholly unsupervised clustering solution. In a general unsupervised clustering problem, we need to consider three constraints:

(i) Knowledge of the class distribution is not known. In fact, each cluster may have different distribution.

(ii) Assumptions of the size or number of samples that belong to each class cannot be made.

(iii) Knowledge of the number of the clusters is not available

BHC is designed to handle the above constraints. The algorithm can be divided into two main steps (Fig.2).

Step 1: Use the K-means algorithm to cluster the expression data into two classes in order to obtain the class centroids and class members.

Step 2: Compute the Fisher criterion from the K-means class parameters. If it exceeds a set threshold, accept the binary splitting of the cluster in step 1; else do not subdivide the cluster.

3.1 Basic idea of BHC

The BHC algorithm uses the idea of a hierarchical binary division clustering framework. For example, let us consider a dataset, which is two-dimensional with three distinct classes (Fig.1). The algorithm starts by assuming that the dataset consists of one class. The first application of BHC generates two clusters A and BC. As the projection of class A and class BC have a large enough Fisher criterion on the A-BC discriminant line, the algorithm splits the original dataset into two clusters. Then, the BHC is applied onto each of the two clusters. The BC cluster will be separated into B and C clusters because its Fisher criterion is large. However, the Fisher criterion of cluster A is too low to allow further division, so it remains as a single cluster. Such hierarchical binary division process is repeated until all clusters have Fisher criterion too low for further splitting, i.e., the A, B, and C cluster cannot be further subdivided and the process is halted.

The BHC is non-parametric in nature, so assumptions about the class distributions and the number of clusters *NC* are not needed. The only parameter required is the threshold for the Fisher criterion. Although one may argue that the Fisher criterion plays the role of *NC* since it eventually affects the number of clusters in the dataset, the former has a clear physical interpretation relating to the spatial structure of the cluster in consideration, while the later is difficult to be determined from the given dataset without any prior knowledge of the data.

In the clustering process, there is a range of acceptable Fisher criterion indicating that a cluster should be split. For dataset containing fairly well separated clusters, the final number of clusters is relatively insensitive to the threshold setting, and this threshold can be estimated with some experimentation.

3.2 K-means clustering

In BHC, the K-means algorithm is used to cluster the data into two clusters. Let C_j be the j^{th} cluster, which is a disjoint subset of the gene expression data, such that $\bigcup_j C_j$ gives the original dataset. The K-means algorithm iterates to minimize the following squared error function,

$$E = \sum_{j=1}^{2} \sum_{x \in C_j} \left| x - w_j \right|^2 \tag{2}$$

The K-means algorithm minimizes *E* by alternatingly updating the membership assignment of each data points and the class centroids. During K-means clustering, each data point is assigned to the class that corresponds to the nearest centroid. Then, the centroids are updated by taking the arithmetic means of the members in each class,

$$w_j = \frac{1}{|C_j|} \sum_{x \in C_j} x \tag{3}$$

where $|C_j|$ denotes the cardinality of the class.

The iteration stops when *E* does not change significantly or when the cluster membership no longer changes.

3.3 Fisher linear discriminant

The K-means clustering algorithm produces two subclasses, with the mean of each subclass given by its centroid. Then, the Fisher linear discriminant is used to compute the Fisher criterion of these two subclasses. The Fisher criterion is used to decide whether splitting of the two subclasses is allowed.

The Fisher criterion is defined as

$$\tau(\omega) = \tau = \frac{\omega^T S_b \omega}{\omega^T S_w \omega} \tag{4}$$

where S_b is the between-class scatter matrix, and S_w is the within-class scatter matrix defined by

$$S_b = (m_1 - m_2)(m_1 - m_2)^T \tag{5}$$

$$S_w = S_1 + S_2 \tag{6}$$

where $S_i = \frac{1}{N_i - 1} \sum_{x \in C_i} (x - m_i)(x - m_i)^T$

The class means m_1, m_2 are given by the cluster centroids w_1, w_2 . The Fisher criterion measures the weighted (by the intra-class scattering) separation between the two classes. The larger the Fisher criterion, the larger the weighted distance between the two clusters. The optimal discriminant vector ω maximizing (4) is given by

$$\omega = S_w^{-1}(m_1 - m_2) \tag{7}$$

3.4 Singular matrix problem

In Fisher linear discriminant analysis, when the number of genes in a cluster is less than the feature dimension *n*, the within-class scatter matrix S_w would be singular. This situation often happens in BHC clustering of gene expression data because of the high dimensionality of the data and the fact that the number of genes with similar expression profiles is usually small after several levels of subdivision. When S_w is singular, the fisher criterion and the discriminant vector cannot be computed in (7).

We use the PCA+LDA method to solve this problem [7-9]. The principal component analysis (PCA) is used to reduce the dimension of the data before the application of LDA.

We define $S_t = S_b + S_w$ and m = rank (S_t) . Let $\beta_1, \beta_2, ..., \beta_m$ be the *m* orthonormal eigenvectors of S_t corresponding to the nonzero eigenvalues of S_t . Let *Y* be a vector in the PCA transformed space R^m . Then the corresponding isomorphic mapping between *X* in the original *n* dimensional space and *Y* is

$$X = PY, \quad \text{where } P = (\beta_1, \beta_2, ..., \beta_m)$$

The Fisher criterion function can be expressed as

$$J(X) = \frac{X^T S_b X}{X^T S_t X} = \frac{Y^T (P^T S_b P) Y}{Y^T (P^T S_t P) Y} = \frac{Y^T \tilde{S}_b Y}{Y^T \tilde{S}_t Y} = \tilde{J}(Y) \tag{7}$$

Let

$$\tilde{S}_b = P^T S_b P$$

$$\tilde{S}_w = P^T S_w P$$

$$\tilde{S}_t = P^T S_t P \tag{8}$$

be the between class, within class and total scatter matrix in R^m , respectively. In the PCA transformed space R^m , split the within-class scatter matrix $\tilde{S}_w$ into two spaces: the null space $\tilde{\Phi}_w^{\perp} = span\{\gamma_{q+1}, \ldots, \gamma_m\}$, and its orthogonal complement $\tilde{\Phi}_w = span\{\gamma_1, \ldots, \gamma_q\}$, where $\gamma_1, \ldots, \gamma_m$ are orthonormal eigenvectors of $\tilde{S}_w$. The first *q* eigenvectors correspond to positive eigenvalues. The new fisher criterion is computed in the two subspaces as follows. For any nonzero vector Y in the $\tilde{\Phi}_w^{\perp}$ subspace, the within-class scatter is at a minimum since $Y^T \tilde{S}_w Y = 0$ and the between-class scatter $Y^T \tilde{S}_b Y > 0$. So we can just replace the original fisher criterion by $\tilde{J}(Y) = \tilde{J}_b = Y^T \tilde{S}_b Y$. For any nonzero vector Y in the $\tilde{\Phi}_w$ subspace, $Y^T \tilde{S}_w Y > 0$, and the fisher criterion remains unchanged as $\tilde{J}(Y)$.

The detail steps of the PCA+LDA algorithm is given by:

Step 1. Compute $\tilde{S}_b$, $\tilde{S}_w$ and $\tilde{S}_t$ from the original scatter matrices S_b , S_w and S_t respectively, using (8).

Step 2. Compute the orthonormal eigenvectors $\gamma_1, \ldots, \gamma_m$ of the within-class scatter matrix $\tilde{S}_w$. Suppose the first *q* ones are corresponding to positive eigenvalues.

Step 3. Let $P_1 = (\gamma_{q+1}, \ldots, \gamma_m)$ and $\overline{S}_b = P_1^T \tilde{S}_b P_1$, compute the orthonormal eigenvectors $Z_1, \ldots, Z_l$ of $\overline{S}_b$. Then, the optimal discriminant vectors derived from $\tilde{\Phi}_w^{\perp}$ are $Y_j = P_1 Z_j$, $j = 1, \ldots, l$. Generally, $l = c - 1$, where *c* is the number of classes.

Step 4. Let $P_2 = (\gamma_1, \ldots, \gamma_q)$ and $\hat{S}_b = P_2^T \tilde{S}_b P_2$, $\hat{S}_t = P_2^T \tilde{S}_t P_2$, compute the *d* - *l* generalized eigenvectors $Z_{l+1}, \ldots, Z_d$ of $\hat{S}_b$ and $\hat{S}_t$ corresponding to the first *d* –

l largest eigenvalues. Then, the optimal discriminant vectors derived from $\widetilde{\Phi}_w$ are $Y_j = P_2 Z_j$, $j = l+1,\ldots,d$.

Step 5. Let $Y_j = P_1 Z_j$ ($j = 1,\ldots,l$) and $Y_j = P_2 Z_j$ ($j = l+1,\ldots,d$) act as projection axes to form the feature extractor $\Phi = (Y_1,...,Y_l,Y_{l+1},...,Y_d)$.

4 Cluster Visualization

The binary hierarchical framework naturally leads to a tree structure representation. We build a tree structure display of the BHC clustering results similar to the hierarchical clustering. The tree structure can show the relationship between each cluster, the adjacency between different clusters, as well as the variation within each cluster. Since similar clusters are adjacent to each other in the display, the user can easily decide whether two adjacent clusters should be merged based on his/her knowledge about the biological process involved and the similarity of the visual pattern in the display. The tree structure display thus allows user judgement in finalizing the clustering result using additional biological knowledge, in a manner similar to that in hierarchical clustering.

5 Experimental Results

The gene expression dataset for evaluating the performance of the BHC algorithm comes from Spellman et al., (1998) (http://cellcycle-www.stanford.edu) [11]. It contains expression profiles for 6220 genes under different experimental conditions, for example, cdc15, and cdc28. Data with missing values are filtered out. The BHC algorithm is applied to each dataset from each experiment and the result is displayed using the tree structure visualization.

In the alpha experiment, the gene expression dataset contains 4490 genes with 18 sample points each. BHC is applied with the threshold value set to 0.65. Twenty-six clusters are found (Fig. 3).

In the cdc15 experiment, the gene expression dataset contains 4382 genes with 24 sample points each. BHC is applied with the threshold value set to 0.671. Thirty-one clusters are found (Fig. 4).

In the elu experiment, the gene expression dataset contains 5236 genes with 13 sample points each. BHC is applied with the threshold value set to 0.678. Thirty-two clusters are found (Fig. 5).

In the cdc28 experiment, the gene expression dataset contains 5236 genes with 13 sample points each. BHC is applied with the threshold value set to 0.689. Eleven clusters are found (Fig. 6).

We see that genes with similar expression profiles are clustered successfully by the BHC algorithm into the same group. The clustering results are similar to those obtained by Spellman [11]. By visualizing the results using the tree structure representation, biologists can interpret the clustering results easily and search the tree structure for meaningful clusters or groups of closely related clusters [1,11].

6 Conclusions

In this paper, we described the unsupervised clustering of gene expression data using the BHC algorithm. BHC uses a binary hierarchical framework to cluster the data. It involves using the K-means algorithm to split the data into two classes, and then verify whether the split is acceptable using the Fisher criterion. The main advantages of the BHC clustering algorithm are: (1) The number of clusters can be estimated from the data directly using a binary hierarchical framework; (2) No constraint about the number of samples in each cluster is required, and (3) No prior assumption about the class distribution is needed. We have described how the singular matrix problem in the Fisher linear discriminant analysis, which is common for gene expression data, could be handled. The binary hierarchical framework naturally leads to a tree structure representation. By visualizing the clustering results using a tree structure, the relationship between each cluster, the adjacency between different clusters, as well as the variation within each cluster can be observed easily. The tree structure display thus allows user judgement in finalizing the clustering result using additional biological knowledge, in a manner similar to that in hierarchical clustering. Our experimental results show that the proposed clustering algorithm can group the expression data into visually distinct clusters.

Acknowledgment

This work is supported by a CityU SRG grant (7001183) and an interdisciplinary research grant (9010003).

References

[1]M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein (1998), Cluster analysis and display of genome-wide expression patterns *Proc. Natl Acad. Sci USA*, Vol. 95, December 1998, pp.14863-14868.

[2] N. S. Halter, A. Maritan, M. Cieplak, N. V. Fedoroff, and J. R. Bamavar(2000): Dynamic modelling of gene expression data. Department of Physic and Center for materials physics 104 Davey, and Department of Biology and the Life Sciences Consortium, 519 Wartik Laboratory, Penndylvania State University, University Park., PA 16802.

[3] F. De Smet, J. Mathys, K. Marchal, G. Thijs, B. De Moor and Y. Moreau(2002): Adaptive quality-based clustering of gene expression profiles. *Bioinformatics,* Vol 18, no. 5, 2002, pp. 735-746

[4] A. k. Jain, R. C. Dubes (1948): *Algorithms for Clustering Data*, Prentice Hall

[5] O. Alter, P. O. Brown, and D. Botstein (2001): Processing and modelling genome-wide expression data using singular value decomposition. Departments of Genetics and Biochemistry, Stanford University, Stanford, CA 94305

[6] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, Trevor Hastie, R. Tibshirani, D. Botsttein and R. B. Altman (2001): Missing value estimation methods for DNA microarrays. *Bioinformatics,* Vol 17, no.6, 2001, pp. 520-525.

[7] Y. Jain, and Y. jingyu (2001):An Optimal FLD Algorithm for Facial Feature Extraction. Department of Computer Science, Nanjing University of Science & Technology, Nanjing 210094, People's Republic of China

[8] L. F. Chen, H. Y. Mark Liao, M. T. Ko, Ja-Chen Lin, Gwo-Jong Yu (2000): A new LDA-based face recognition system which can solve the small sample size problem. *Pattern Recognition*, vol. 33, (2000) pp. 1713-1726

[9] J. Yang, and J. Y. Yang (2002): Why can LDA be performed in PCA transformed space? *Pattern Recognition,* PR 1673, pp: 1— 4

[10] A. Brazma, J. Vilo (2000): Minireview Gene expression data analysis. European Molecular Biology Laboratory, Outstation Hinxton – the European Bioinformatics institute, Cambridge CB10 ISD UK.

[11] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, Kirk Anders, Mi. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher (1998): Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast Saccharomyces cerevisiae by Microarray Hybridization, *Molecular Biology of the Cell,* Vol. 9, December 1998, pp.3273-3297.

[12] D. A. Clausi (2002), K-means Iterative Fisher (KIF) unsupervised clustering algorithm applied to image texture segmentation. Pattern Recognition, Vol. 35, pp. 1959-1972.

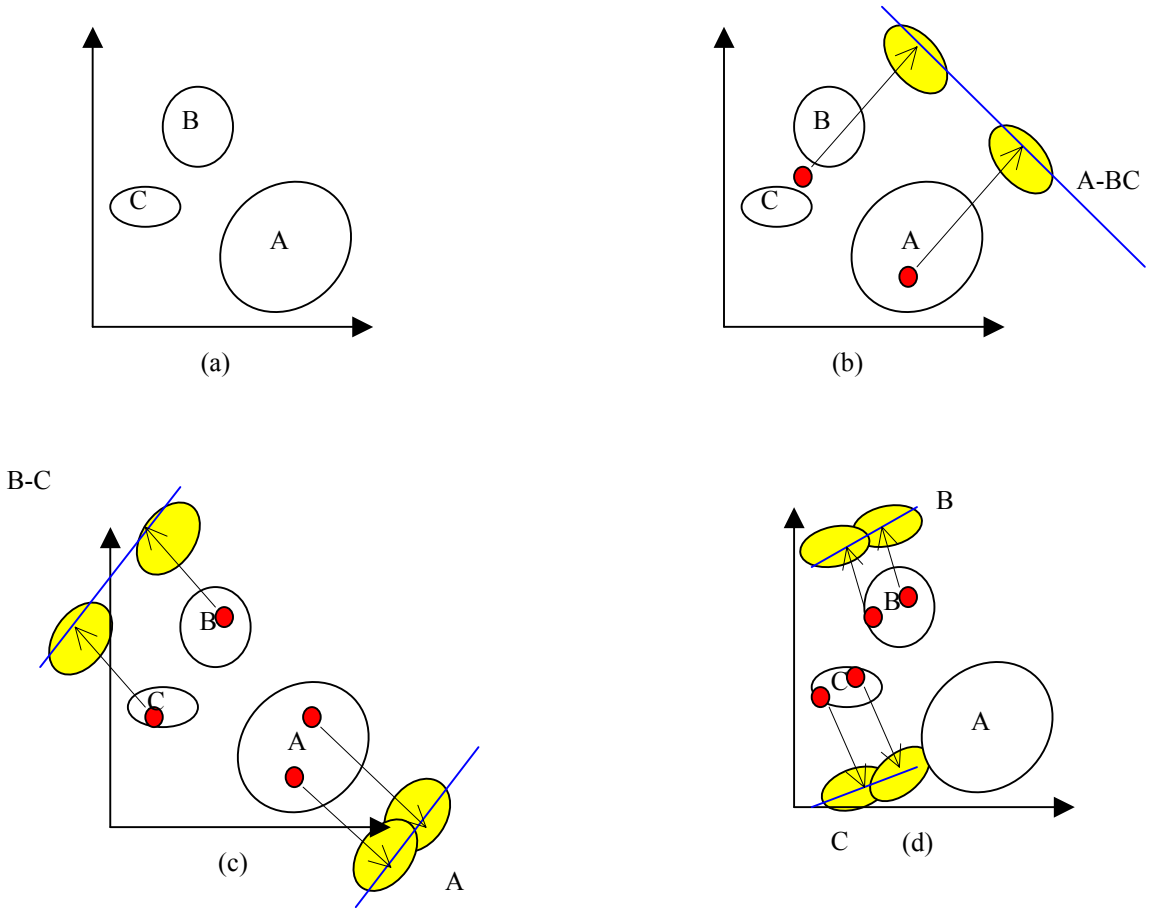


Fig. 1. Binary hierarchical clustering algorithm. (a) Original gene expression data treated as one class. (b) Split the class into two clusters, A and BC. (c) Cluster A cannot be split, but cluster BC is split into two clusters, B and C. (d) Both cluster B and C cannot be split any more, so we have three clusters A, B, and C.

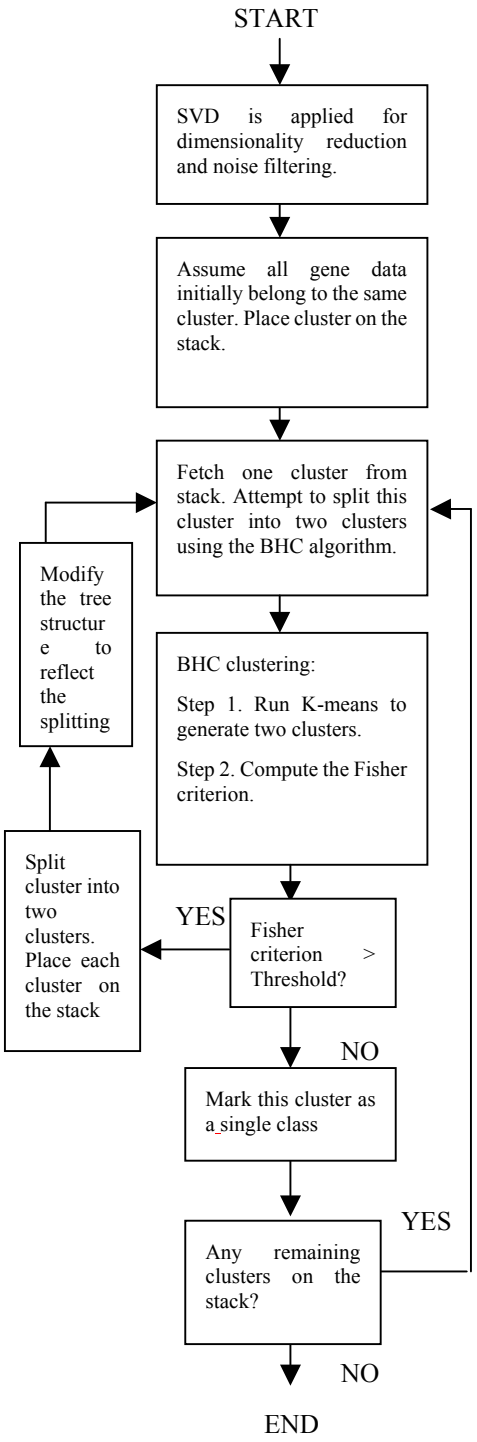
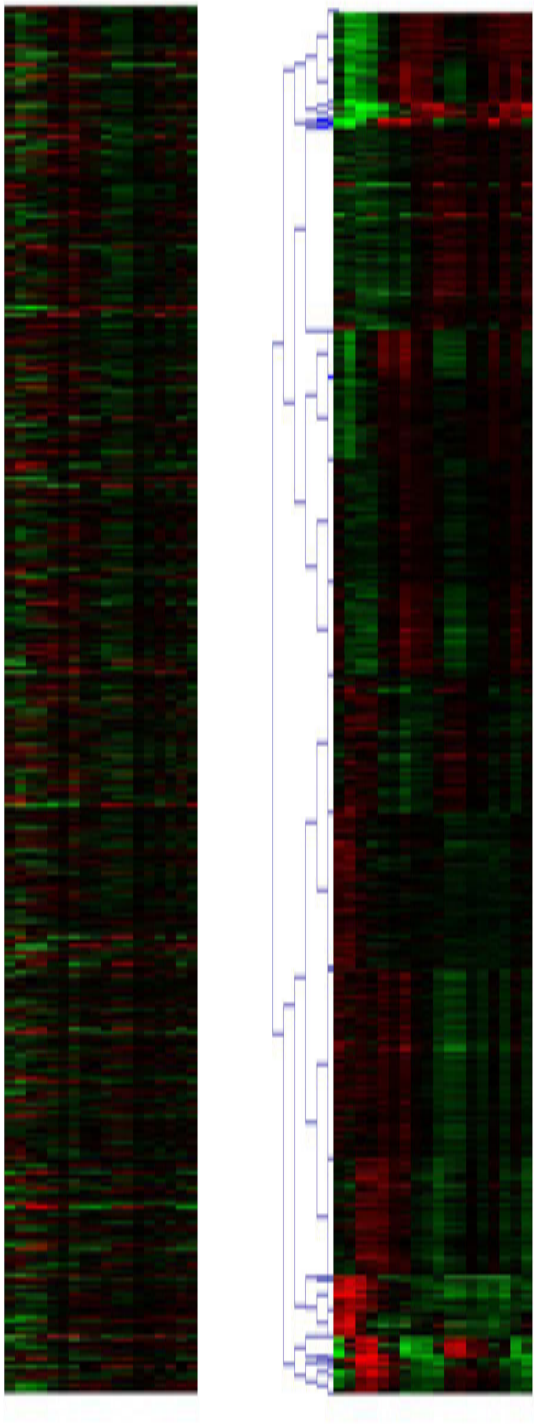
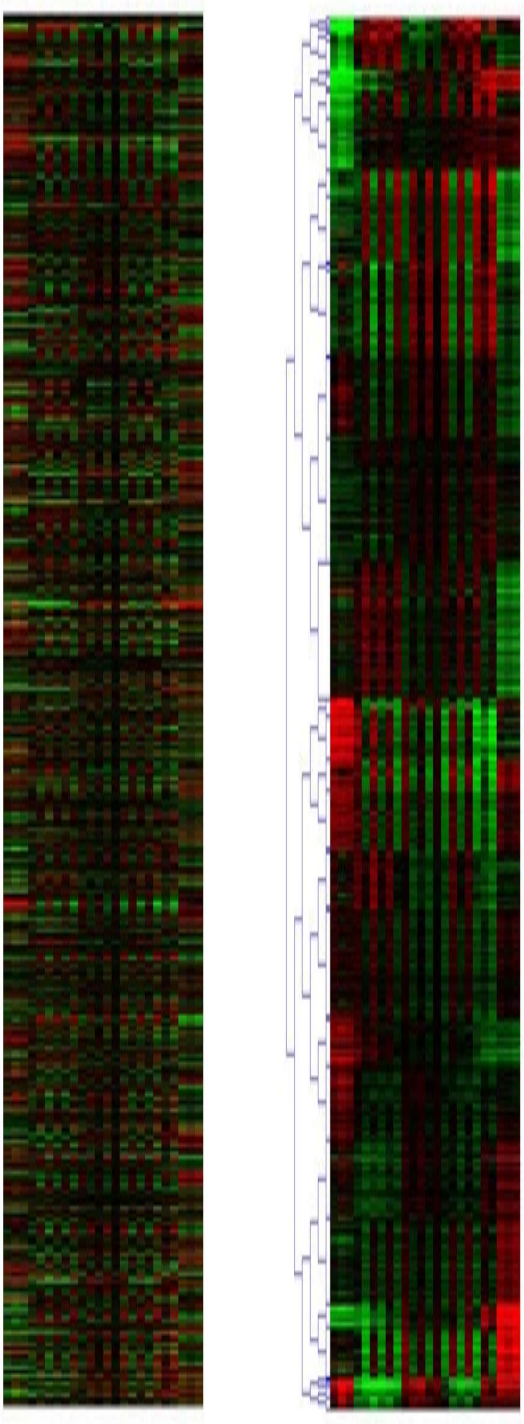


Fig.2. Flow chart of BHC clustering.



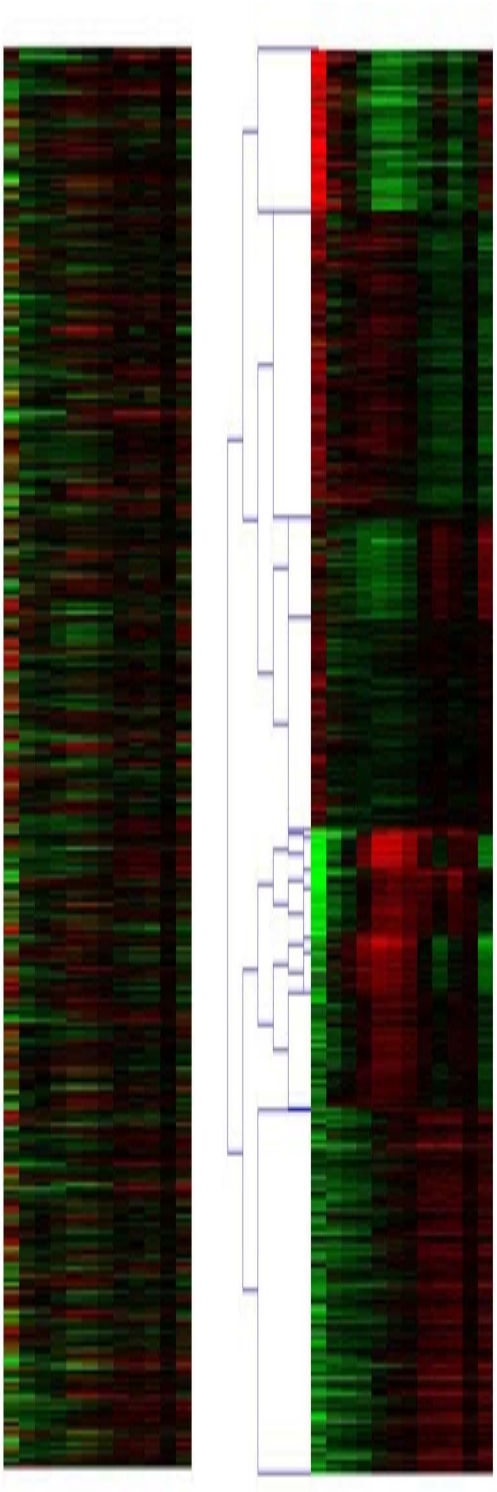
(a) (b)

Fig. 3. The alpha experiment dataset (a) Original gene expression data. (b) Expression data after BHC clustering.



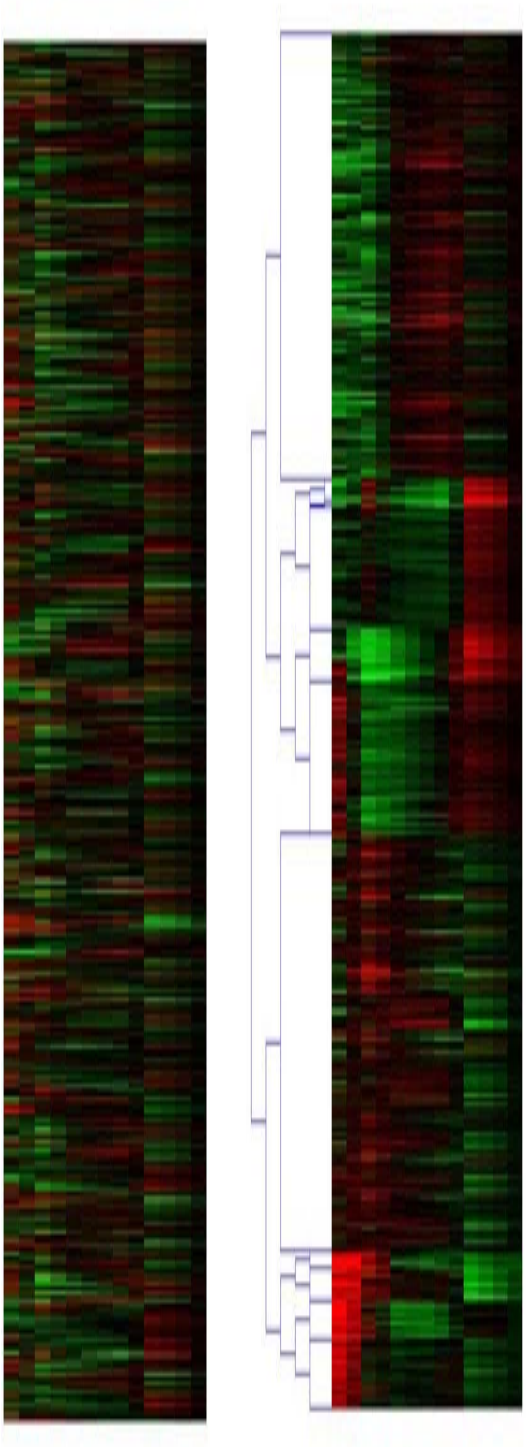
(a) (b)

Fig. 4. The cdc15 experiment dataset (a) Original gene expression data. (b) Expression data after BHC clustering.



(a) (b)

Fig. 5. The elu experiment dataset (a) Original gene expression data. (b) Expression data after BHC clustering.



(a) (b)

Fig. 6. The cdc28 experiment dataset (a) Original gene expression data. (b) Expression data after BHC clustering.